

# 中山大学数据科学与计算机学院本科生实验报告

## ( 2017 年秋季学期 )

课程名称：手机应用平台开发

任课教师：刘宁

年级	15 级	专业 ( 方向 )	软件工程 ( 计应 )
学号	15331421	姓名	郑佳锋
电话	13719325472	Email	1773643139@qq.com
开始日期	10.14	完成日期	10.15

### 一、 实验题目

Lab5

### 二、 实现内容

1. 用 RecyclerView 实现主界面。先在 xml 布局文件中声明：

```
1. <android.support.v7.widget.RecyclerView
2.     android:id="@+id/id_recyclerview"
3.     android:divider="#ffff0000"
4.     android:dividerHeight="10dp"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent" />
```

在 MainActivity 中 设置 Adapter ， 为了 实现 动画 效果 ， 调用 ScaleInAnimationAdapter 类。

```
1. final HomeAdapter myAdapter = new HomeAdapter();
2. ScaleInAnimationAdapter animationAdapter = new ScaleInAnimationAdapter(myAdapter);
3. animationAdapter.setDuration(1000);
4. final RecyclerView myRecView = (RecyclerView) findViewById(R.id.id_recyclerview);
5. myRecView.setLayoutManager(new LinearLayoutManager(this));
6. myRecView.setAdapter(animationAdapter);
7. myRecView.setItemAnimator(new OvershootInLeftAnimator());
```

再来看自定义的类 HomeAdapter，该类继承自 RecyclerView 的 Adapter 类，以自定义的类中类 MyViewHolder 作为参数。MyViewHolder 继承自 RecyclerView 的 ViewHolder 类，该类包含两个成员，即主界面每个条目中的 Button 和 TextView。

HomeAdapter 中实现了 5 个方法：

(1)public MyViewHolder onCreateViewHolder(ViewGroup parent, int viewType) ;

调用 inflate 方法获取将要呈现的布局。

(2)public void setOnItemClickListener(OnItemClickListener mOnItemClickListener) ;

RecyclerView 没有点击和长按事件，需要自己定义。OnItemClickListener 也是自定义的接口，包含 onItemClick 和 onItemLongClick 方法。

(3)public void onBindViewHolder(final MyViewHolder holder, int position) ;

该方法将数据绑定到布局中的对应元素。同时处理点击和长按事件，将位置信息作为参数传递给 onItemClick 和 onItemLongClick 方法。

(4)public void removeData(int position) ;

长按删除某个条目，通知数据移除相关信息。

(5)public int getItemCount() ;

返回条目的数量。

HomeAdapter 的完整定义如下：

```
1. class HomeAdapter extends RecyclerView.Adapter<HomeAdapter.MyViewHolder>
2. {
3.
4.     @Override
5.     public MyViewHolder onCreateViewHolder(ViewGroup parent, int viewType)
6.     {
7.         MyViewHolder holder = new MyViewHolder(LayoutInflater.from(
8.             MainActivity.this).inflate(R.layout.item_home, parent,
9.             false));
10.        return holder;
11.    }
12.
13.    private OnItemClickListener mOnItemClickListener;
14.
15.    public void setOnItemClickListener(OnItemClickListener mOnItemClickListener)
16.    {
17.        this.mOnItemClickListener = mOnItemClickListener;
18.    }
19.
20.    @Override
21.    public void onBindViewHolder(final MyViewHolder holder, int position)
22.    {
23.        holder.tv.setText(mDatas.get(position));
24.        holder.btn.setText(mDatas.get(position).substring(0,1));
25.        if (mOnItemClickListener != null)
26.        {
27.            holder.itemView.setOnClickListener(new View.OnClickListener()
28.            {
```

```

29.         @Override
30.         public void onClick(View v)
31.         {
32.             int pos = holder.getLayoutPosition();
33.             mOnItemClickListener.onItemClick(holder.itemView, pos);
34.         }
35.     });
36.
37.     holder.itemView.setOnLongClickListener(new View.OnLongClickListener()
38.     {
39.         @Override
40.         public boolean onLongClick(View v)
41.         {
42.             int pos = holder.getLayoutPosition();
43.             mOnItemClickListener.onItemLongClick(holder.itemView, pos);
44.             return false;
45.         }
46.     });
47. }
48. }
49.
50. public void removeData(int position) {
51.     mDatas.remove(position);
52.     notifyItemRemoved(position);
53. }
54.
55. @Override
56. public int getItemCount()
57. {
58.     return mDatas.size();
59. }
60.
61. class MyViewHolder extends RecyclerView.ViewHolder
62. {
63.
64.     TextView tv;
65.     Button btn;
66.
67.     public MyViewHolder(View view)
68.     {
69.         super(view);
70.         tv = (TextView) view.findViewById(R.id.id_num);
71.         btn = (Button) view.findViewById(R.id.button);
72.     }
73. }
74. }

```

自定义的 OnItemClickListener 接口：

```

1. interface OnItemClickListener {
2.     void onItemClick(View view, int position);
3.     void onItemLongClick(View view , int position);
4. }

```

在 RecyclerView 中重写点击和长按事件，点击打开新页面，使用 Intent 类，将商品名传递给新的页面；长按删除条目信息，弹出 toast 信息。

```

1. myAdapter.setOnItemClickListener(new OnItemClickListener()
2. {
3.

```

```

4.         @Override
5.         public void onItemClick(View view, int position)
6.         {
7.             Intent intent = new Intent(MainActivity.this, NewActivity.class);
8.             String extra = mDatas.get(position);
9.             intent.putExtra("passname", extra);
10.            MainActivity.this.startActivity(intent);
11.        }
12.
13.        @Override
14.        public void onItemLongClick(View view, int position)
15.        {
16.            Toast.makeText(MainActivity.this, "移除第" + position + "个商品",
17.                Toast.LENGTH_SHORT).show();
18.            myAdapter.removeData(position);
19.        }
20.    });

```

RecyclerView 的布局，使用 FrameLayout 布局实现：

```

1. <?xml version="1.0" encoding="utf-8"?>
2. <FrameLayout
3.     android:layout_height="wrap_content"
4.     android:layout_width="match_parent"
5.     xmlns:android="http://schemas.android.com/apk/res/android">
6.
7.     />
8.     <Button
9.         android:id="@+id/button"
10.        android:layout_width="60dp"
11.        android:layout_height="60dp"
12.        android:background="@drawable/shape"
13.        android:text="E"
14.        android:textSize="35dp"
15.        android:textColor="@color/buttontext"
16.        android:layout_marginLeft="20dp"
17.        android:layout_marginTop="53dp"
18.    />
19.
20.     <TextView
21.         android:id="@+id/id_num"
22.         android:layout_width="match_parent"
23.         android:layout_height="50dp"
24.         android:gravity="left"
25.         android:text="1"
26.         android:textColor="@color/black"
27.         android:textSize="27dp"
28.         android:layout_marginLeft="90dp"
29.         android:layout_marginTop="66dp"
30.    />
31.
32. </FrameLayout>

```

RecyclerView 的数据，使用 initData 进行初始化：

```

1. private void initData(){
2.     mDatas = new ArrayList<String>();
3.     mDatas.add("Enchated Forest");
4.     mDatas.add("Arla Milk");
5.     mDatas.add("Devondale Milk");
6.     mDatas.add("Kindle Oasis");
7.     mDatas.add("waitrose 早餐麦片");

```

```

8.      mDatas.add("Mcvitie's 饼干");
9.      mDatas.add("Ferrero Rocher");
10.     mDatas.add("Maltesers");
11.     mDatas.add("Lindt");
12.     mDatas.add("Borggreve");
13. }

```

至此，RecyclerView 基本完工。

2.设置悬浮按钮，点击一下转到购物车界面，再点击一下回到主界面，按钮背景会跟着改变。

Xml：

```

1. <FrameLayout
2.     android:layout_width="match_parent"
3.     android:layout_height="match_parent">
4.     <android.support.design.widget.FloatingActionButton
5.         android:id="@+id/floatButton"
6.         android:layout_width="wrap_content"
7.         android:layout_height="wrap_content"
8.         android:layout_gravity="bottom|end"
9.         android:layout_margin="45dp"
10.        android:elevation="20dp"
11.        android:src="@mipmap/shoplist"
12.        android:background="#FFF"
13.        app:backgroundTint="#FFFFFF" />
14. </FrameLayout>

```

在 MainActivity 中设置点击事件，tag 变量是 0 或者 1，用来记录当前是哪个界面：

```

1. final FloatingActionButton floatBtn = (FloatingActionButton) findViewById(R.id.floatButt
   on);
2. floatBtn.setOnClickListener(new View.OnClickListener() {
3.     @Override
4.     public void onClick(View v) {
5.         if (tag == 0) {
6.             myRecyclerView.setVisibility(View.INVISIBLE);
7.             shopList.setVisibility(View.VISIBLE);
8.             floatBtn.setImageResource(R.mipmap.mainpage);
9.             tag = 1;
10.        }
11.        else if (tag == 1){
12.            shopList.setVisibility(View.INVISIBLE);
13.            myRecyclerView.setVisibility(View.VISIBLE);
14.            floatBtn.setImageResource(R.mipmap.shoplist);
15.            tag = 0;
16.        }
17.    }
18. });

```

从上面代码可以看出，点击悬浮按钮后并没有条状到新的 activity 中。也就是说主界面的 RecyclerView 和购物车的 ListView 处于同一个布局文件中。所谓的“跳转”只是设置它们的可见性而已。下面开始实现购物车的 ListView。

3.购物车页面布局，包括一个 Button 和两个 TextView，用 LinearLayout 布局实现：

```
1. <LinearLayout
2.     android:layout_width="match_parent"
3.     android:layout_height="match_parent"
4.     android:descendantFocusability="blocksDescendants"
5.     xmlns:android="http://schemas.android.com/apk/res/android">
6.
7.     <Button
8.         android:id="@+id/shopButton"
9.         android:layout_width="40dp"
10.        android:layout_height="40dp"
11.        android:background="@drawable/shape"
12.        android:text="*"
13.        android:textSize="20dp"
14.        android:textColor="@color/buttontext"
15.        android:layout_marginLeft="20dp"
16.        android:layout_marginTop="22dp"
17.    />
18.    <TextView
19.        android:id="@+id/goods"
20.        android:layout_width="wrap_content"
21.        android:layout_height="wrap_content"
22.        android:padding="15dp"
23.        android:textSize="15sp"
24.        android:textColor="#000000"
25.        android:layout_marginTop="15dp"
26.        android:layout_marginBottom="15dp"
27.        android:text="购物车购物车购物车"
28.    />
29.    <TextView
30.        android:id="@+id/price"
31.        android:layout_weight="1"
32.        android:layout_width="0dp"
33.        android:layout_height="wrap_content"
34.        android:gravity="end"
35.        android:padding="15dp"
36.        android:textSize="15sp"
37.        android:textColor="#000000"
38.        android:layout_marginTop="15dp"
39.        android:layout_marginBottom="15dp"
40.        android:text="价格"
41.    />
42. </LinearLayout>
```

实现 ListView，先声明，注意和 RecyclerView 的声明在一块：

```
1. <ListView
2.     android:id="@+id/shoppinglist"
3.     android:layout_width="match_parent"
4.     android:layout_height="wrap_content"
5.     android:clickable="true"
6.     xmlns:android="http://schemas.android.com/apk/res/android">
7.
```

## 8. </ListView>

在 MainActivity 中实现 ListView，先初始化数据，购物车页面第一栏比较特殊，它的 Button 中符号为 “\*” ,而其他条目是名称的第一个字母，所以需要特殊处理：

```
1. final ListView shopList = (ListView) findViewById(R.id.shoppinglist);
2. String[] goods = new String[]{"购物车"};
3. String[] price = new String[]{"价格"};
4. for (int i = 0; i < goods.length; i++) {
5.     Map<String, String> temp = new LinkedHashMap<String, String>();
6.     temp.put("goods", goods[i]);
7.     temp.put("price", price[i]);
8.     if (goods[i] == "购物车"){
9.         temp.put("shopButton", "*");
10.    }
11.    else{
12.        temp.put("shopButton", goods[i].substring(0, 1));
13.    }
14.    data.add(temp);
15. }
```

使用 SimpleAdapter：

```
1. simpleAdapter = new SimpleAdapter(MainActivity.this, data, R.layout.item_shop,
2.     new String[]{"goods", "price", "shopButton"}, new int[]{R.id.goods, R.id.pri
3.     ce, R.id.shopButton});
4. shopList.setAdapter(simpleAdapter);
```

设置点击事件：同样将点击位置的商品名称作为 Intent 的参数传递给下一个界面，注意界面的第一栏同样需要特殊处理，点击后没有响应；设置长按事件：弹出移除商品的对话框，点击取消返回原界面，点击确定删除位置的商品，并通知数据队列更新数据：

```
1. shopList.setOnItemClickListener(new AdapterView.OnItemClickListener() {
2.     @Override
3.     public void onItemClick(AdapterView<?> parent, View view, int position, long id)
4.     {
5.         if (position == 0){}
6.         else {
7.             Intent intent = new Intent(MainActivity.this, NewActivity.class);
8.             Map<String, String> extra = data.get(position);
9.             intent.putExtra("passname", extra.get("goods"));
10.            MainActivity.this.startActivity(intent);
11.        }
12.    });
13. shopList.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener(){
14.     @Override
15.     public boolean onItemLongClick(AdapterView<?> arg0, View arg1,
16.         final int arg2, long arg3) {
17.         AlertDialog.Builder dialogbuilder = new AlertDialog.Builder(MainActivity.thi
18.         s);
19.         dialogbuilder.setTitle("上传头像");
20.         String temp = data.get(arg2).get("goods");
```

```

20.         dialogbuilder.setMessage("从购物车移除" + temp + "?");
21.         dialogbuilder.setNegativeButton("取消", click1);
22.         dialogbuilder.setPositiveButton("确定",
23.             new DialogInterface.OnClickListener() {
24.                 @Override
25.                 public void onClick(DialogInterface dialogInterface, int i) {
26.                     data.remove(arg2);
27.                     simpleAdapter.notifyDataSetChanged();
28.                 }
29.             });
30.         AlertDialog dialog = dialogbuilder.create();
31.         dialog.show();
32.         return true;
33.     }
34.     private DialogInterface.OnClickListener click1 = new DialogInterface.OnClickListener() {
35.         @Override
36.         public void onClick(DialogInterface arg0, int arg1) {
37.             arg0.cancel();
38.         }
39.     };
40. });

```

至此，主界面基本完工。下面开始新页面的设计。

4.新页面总体使用约束布局实现。顶部图片占三分之一使用引导线实现，底部更多信息同样使用 ListView 实现。布局文件如下：

```

1. <android.support.constraint.ConstraintLayout
2.     android:layout_width="match_parent"
3.     android:layout_height="match_parent"
4.     android:background="#FFF"
5.     xmlns:android="http://schemas.android.com/apk/res/android"
6.     xmlns:app="http://schemas.android.com/apk/res-auto">
7.     <android.support.constraint.Guideline
8.         android:id="@+id/guideline1"
9.         android:layout_width="match_parent"
10.        android:layout_height="match_parent"
11.        android:orientation="horizontal"
12.        app:layout_constraintGuide_percent="0.333333"
13.    />
14.    <android.support.constraint.Guideline
15.        android:id="@+id/guideline2"
16.        android:layout_width="match_parent"
17.        android:layout_height="match_parent"
18.        android:orientation="vertical"
19.        app:layout_constraintGuide_percent="0.25"
20.    />
21.    <android.support.constraint.Guideline
22.        android:id="@+id/guideline3"
23.        android:layout_width="match_parent"
24.        android:layout_height="match_parent"
25.        android:orientation="vertical"
26.        app:layout_constraintGuide_percent="0.75"
27.    />
28.    <android.support.constraint.Guideline
29.        android:id="@+id/guideline4"
30.        android:layout_width="match_parent"
31.        android:layout_height="match_parent"
32.        android:orientation="horizontal"
33.        app:layout_constraintGuide_percent="0.46"
34.    />

```



```

35.     <ImageView
36.         android:id="@+id/leftBlank"
37.         android:layout_width="wrap_content"
38.         android:layout_height="0dp"
39.         android:adjustViewBounds="true"
40.         android:src="@mipmap/background"
41.         app:layout_constraintTop_toTopOf="parent"
42.         app:layout_constraintBottom_toBottomOf="@+id/guideline1"
43.         app:layout_constraintLeft_toLeftOf="parent"
44.         app:layout_constraintRight_toLeftOf="@+id/photo"
45.     />
46.     <ImageView
47.         android:id="@+id/rightBlank"
48.         android:layout_width="wrap_content"
49.         android:layout_height="0dp"
50.         android:adjustViewBounds="true"
51.         android:src="@mipmap/background"
52.         app:layout_constraintTop_toTopOf="parent"
53.         app:layout_constraintBottom_toBottomOf="@+id/guideline1"
54.         app:layout_constraintRight_toRightOf="parent"
55.         app:layout_constraintLeft_toRightOf="@+id/photo"
56.     />
57.     <ImageView
58.         android:id="@+id/photo"
59.         android:layout_width="wrap_content"
60.         android:layout_height="0dp"
61.         android:adjustViewBounds="true"
62.         android:src="@mipmap/ferrero"
63.         app:layout_constraintTop_toTopOf="parent"
64.         app:layout_constraintBottom_toBottomOf="@+id/guideline1"
65.         app:layout_constraintLeft_toLeftOf="@+id/guideline2"
66.         app:layout_constraintRight_toRightOf="@+id/guideline3"
67.     />
68.     <TextView
69.         android:id="@+id/shopName"
70.         android:layout_width="wrap_content"
71.         android:layout_height="wrap_content"
72.         android:text="Enchated Forest"
73.         android:textColor="@color/black"
74.         android:textSize="20sp"
75.         android:layout_marginLeft="20dp"
76.         app:layout_constraintLeft_toLeftOf="parent"
77.         app:layout_constraintBottom_toBottomOf="@+id/guideline1"
78.         android:layout_marginBottom="20dp"
79.     />
80.     <ImageView
81.         android:id="@+id/star"
82.         android:layout_width="35dp"
83.         android:layout_height="35dp"
84.         android:src="@mipmap/empty_star"
85.         app:layout_constraintBottom_toBottomOf="@+id/shopName"
86.         app:layout_constraintRight_toRightOf="parent"
87.         android:layout_marginRight="20dp"
88.     />
89.     <TextView
90.         android:id="@+id/shopPrice"
91.         android:layout_width="wrap_content"
92.         android:layout_height="wrap_content"
93.         android:text="¥ 5.00"
94.         android:textColor="#D5000000"
95.         android:layout_marginLeft="20dp"
96.         app:layout_constraintLeft_toLeftOf="parent"
97.         app:layout_constraintTop_toBottomOf="@id/leftBlank"
98.         android:layout_marginTop="10dp"
99.     />
100.    <TextView
101.        android:id="@+id/shopInfo"

```

```

102.     android:layout_width="wrap_content"
103.     android:layout_height="wrap_content"
104.     android:text="作者 Johanna Basford"
105.     android:textColor="#8A000000"
106.     android:layout_marginLeft="20dp"
107.     app:layout_constraintLeft_toLeftOf="parent"
108.     app:layout_constraintTop_toBottomOf="@+id/shopPrice"
109.     android:layout_marginTop="10dp"
110.     />
111. <ImageView
112.     android:id="@+id/shopBus"
113.     android:layout_width="30dp"
114.     android:layout_height="30dp"
115.     android:foreground="@mipmap/shoplist"
116.     app:layout_constraintTop_toBottomOf="@+id/guideline1"
117.     app:layout_constraintRight_toRightOf="parent"
118.     android:layout_marginRight="20dp"
119.     android:layout_marginTop="20dp"
120.     />
121. <View
122.     android:layout_height="0dp"
123.     app:layout_constraintTop_toBottomOf="@+id/guideline1"
124.     android:layout_width="1dp"
125.     android:background="#1E000000"
126.     android:orientation="vertical"
127.     android:layout_marginTop="15dp"
128.     app:layout_constraintRight_toLeftOf="@+id/shopBus"
129.     app:layout_constraintBottom_toBottomOf="@+id/shopBus"
130.     android:layout_marginRight="20dp"
131.     />
132. <View
133.     android:layout_width="fill_parent"
134.     app:layout_constraintTop_toBottomOf="@+id/guideline4"
135.     android:layout_height="1dp"
136.     android:background="#1E000000"
137.     android:orientation="horizontal"
138.     android:layout_marginLeft="10dp"
139.     android:layout_marginRight="10dp"
140.     />
141. <TextView
142.     android:id="@+id/more"
143.     android:layout_width="match_parent"
144.     android:layout_height="wrap_content"
145.     android:text="更多产品信息"
146.     android:textColor="#D5000000"
147.     android:textSize="20sp"
148.     app:layout_constraintTop_toTopOf="@id/guideline4"
149.     android:layout_marginTop="10dp"
150.     android:layout_marginLeft="20dp"
151.     />
152. <View
153.     android:id="@+id/moreLine"
154.     android:layout_width="fill_parent"
155.     app:layout_constraintTop_toBottomOf="@+id/more"
156.     android:layout_height="20dp"
157.     android:background="#1E000000"
158.     android:orientation="horizontal"
159.     android:layout_marginTop="10dp"
160.     />
161. <ImageView
162.     android:id="@+id/back"
163.     android:layout_width="30dp"
164.     android:layout_height="30dp"
165.     android:foreground="@mipmap/back"
166.     app:layout_constraintTop_toTopOf="parent"
167.     app:layout_constraintLeft_toLeftOf="parent"
168.     android:layout_marginLeft="20dp"

```

```

169.         android:layout_marginTop="12dp"
170.     />
171.     <ListView
172.         android:id="@+id/moreInfo"
173.         android:layout_width="match_parent"
174.         android:layout_height="wrap_content"
175.         app:layout_constraintTop_toBottomOf="@+id/moreLine"
176.     >
177.
178.     </ListView>
179.
180.
181. </android.support.constraint.ConstraintLayout>

```

ListView 实现类似，直接上代码：

Xml：

```

1. <LinearLayout
2.     android:layout_width="match_parent"
3.     android:layout_height="match_parent"
4.     xmlns:android="http://schemas.android.com/apk/res/android">
5.
6.     <TextView
7.         android:id="@+id/infoItem"
8.         android:layout_width="wrap_content"
9.         android:layout_height="wrap_content"
10.        android:paddingLeft="0dp"
11.        android:textSize="20sp"
12.        android:textColor="#D5000000"
13.        android:layout_marginLeft="20dp"
14.        android:layout_marginTop="13dp"
15.        android:layout_marginBottom="13dp"
16.        android:text="一键下单"
17.    />
18. </LinearLayout>

```

Java：

```

1. ListView moreInfo = (ListView) findViewById(R.id.moreInfo);
2. List<Map<String, String>> data = new ArrayList<Map<String, String>>();
3. String[] infoItem = new String[]{"一键下单", "分享商品", "不感兴趣", "查看更多商品促销信息"};
4. for(int i = 0; i < infoItem.length; i++){
5.     Map<String, String> temp = new LinkedHashMap<String, String>();
6.     temp.put("infoItem", infoItem[i]);
7.     data.add(temp);
8. }
9. SimpleAdapter simpleAdapter = new SimpleAdapter(NewActivity.this, data, R.layout.more_info,
10.     new String[]{"infoItem"}, new int[]{R.id.infoItem});
11. moreInfo.setAdapter(simpleAdapter);

```

点击星标空心和实心切换，使用 flag 来“记住”状态：

```

1. final ImageView star = (ImageView) findViewById(R.id.star);
2. star.setOnClickListener(new View.OnClickListener() {
3.     @Override
4.     public void onClick(View v) {

```

```

5.         if (flag == 0){
6.             star.setImageResource(R.mipmap.full_star);
7.             flag = 1;
8.         }
9.         else if (flag == 1){
10.            star.setImageResource(R.mipmap.empty_star);
11.            flag = 0;
12.        }
13.    }
14. });

```

设置返回按钮，点击后调用 finish()即可：

```

1. ImageView back = (ImageView) findViewById(R.id.back);
2. back.setOnClickListener(new View.OnClickListener() {
3.     @Override
4.     public void onClick(View v) {
5.         finish();
6.     }
7. });

```

至此，新页面除了数据以外，基本的布局已经完成。最后讲讲页面之间的信息传递。

5.前面已经讲到，主界面的 RecyclerView 或者 ListView 被点击之后，被点击的商品名称会作为 Intent 的参数传递给新页面并打开新界面。之所以只传递商品名称，是因为我在新页面的 java 文件中保存了所有的商品信息，通过商品名称就可以 map 到其他的信息再将它们渲染在新页面中。比较棘手的是图片的信息，可以将图片的 id 转化为 String 与商品名称存在 map 中，使用时再将它转化为 int 类型的 id。如下：

商品信息：

```

1. final Map<String, String> name_price = new LinkedHashMap<String, String>();
2. name_price.put("Enchated Forest", "¥5.00");
3. name_price.put("Arla Milk", "¥59.00");
4. name_price.put("Devondale Milk", "¥79.00 ");
5. name_price.put("Kindle Oasis", "¥2399.00");
6. name_price.put("waitrose 早餐麦片", "¥179.00");
7. name_price.put("Mcvitie's 饼干", "¥14.90");
8. name_price.put("Ferrero Rocher", "¥132.59 ");
9. name_price.put("Maltesers", "¥141.43");
10. name_price.put("Lindt", "¥139.43");
11. name_price.put("Borggreve", "¥28.90 ");
12. Map<String, String> name_info = new LinkedHashMap<String, String>();
13. name_info.put("Enchated Forest", "作者 Johanna Basford");
14. name_info.put("Arla Milk", "产地 德国");
15. name_info.put("Devondale Milk", "产地 澳大利亚");
16. name_info.put("Kindle Oasis", "版本 8GB");
17. name_info.put("waitrose 早餐麦片", "重量 2Kg");
18. name_info.put("Mcvitie's 饼干", "产地 英国");
19. name_info.put("Ferrero Rocher", "重量 300g");
20. name_info.put("Maltesers", "重量 118g");
21. name_info.put("Lindt", "重量 249g");

```

```

22. name_info.put("Borggreve", "重量 640g");
23. Map<String, String> name_pic = new LinkedHashMap<String, String>();
24. name_pic.put("Enchated Forest", String.valueOf(R.mipmap.enchatedforest));
25. name_pic.put("Arla Milk", String.valueOf(R.mipmap.arla));
26. name_pic.put("Devondale Milk", String.valueOf(R.mipmap.devondale));
27. name_pic.put("Kindle Oasis", String.valueOf(R.mipmap.kindle));
28. name_pic.put("waitrose 早餐麦片", String.valueOf(R.mipmap.waitrose));
29. name_pic.put("Mcvitie's 饼干", String.valueOf(R.mipmap.mcvitie));
30. name_pic.put("Ferrero Rocher", String.valueOf(R.mipmap.ferrero));
31. name_pic.put("Maltesers", String.valueOf(R.mipmap.maltesers));
32. name_pic.put("Lindt", String.valueOf(R.mipmap.lindt));
33. name_pic.put("Borggreve", String.valueOf(R.mipmap.borggreve));

```

使用 Bundle 提取其他页面传递过来的信息，然后设置本页面的信息：

```

1. Bundle bundle = getIntent().getExtras();
2. if(bundle != null){
3.     passname = bundle.getString("passname");
4. }
5. TextView shopname = (TextView) findViewById(R.id.shopName);
6. shopname.setText(passname);
7. String price = name_price.get(passname);
8. TextView shopprice = (TextView) findViewById(R.id.shopPrice);
9. shopprice.setText(price);
10. String info = name_info.get(passname);
11. TextView shopinfo = (TextView) findViewById(R.id.shopInfo);
12. shopinfo.setText(info);
13. ImageView photo = (ImageView) findViewById(R.id.photo);
14. photo.setImageResource(Integer.parseInt(name_pic.get(passname)));

```

另外一个消息传递是，点击新页面的购物车按钮后，该商品被添加到主页面的购物车中去，可以使用 EventBus 实现。

新建类 FirstEvent，设定需要发布的信息：

```

1. public class FirstEvent {
2.     private String namemsg;
3.     private String pricemsg;
4.     public FirstEvent(String str1, String str2) {
5.         namemsg = str1;
6.         pricemsg = str2;
7.     }
8.     public String getNamemsg(){
9.         return namemsg;
10.    }
11.    public String getPricemsg(){
12.        return pricemsg;
13.    }
14. }

```

在新页面中设置发布者：

```

1. ImageView shopbus = (ImageView) findViewById(R.id.shopBus);
2. shopbus.setOnClickListener(new View.OnClickListener() {
3.     @Override
4.     public void onClick(View v) {

```

```

5.         Toast.makeText(NewActivity.this, "商品已加入购物车
", Toast.LENGTH_SHORT).show();
6.         EventBus.getDefault().post(
7.             new FirstEvent(passname, name_price.get(passname)));
8.     }
9. });

```

在主界面中设置注册，解除注册和处理时间：

```

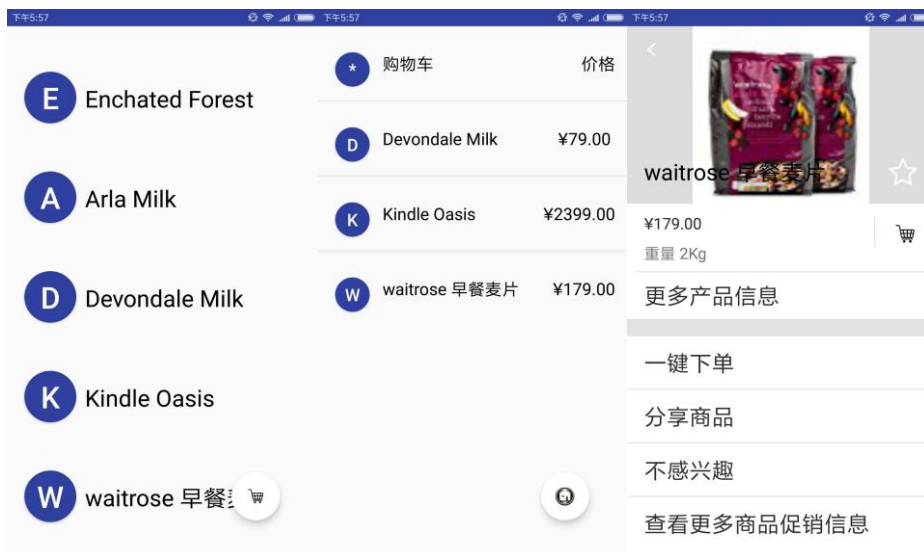
1. EventBus.getDefault().register(this);
2.     protected void onDestroy(){
3.         super.onDestroy();
4.         EventBus.getDefault().unregister(this); //反注册 EventBus
5.     }
6.     @Subscribe
7.     public void onEventMainThread(FirstEvent event) {
8.         Map<String, String> temp = new LinkedHashMap<String, String>();
9.         temp.put("goods", event.getNamemsg());
10.        temp.put("price", event.getPricemsg());
11.        temp.put("shopButton", event.getNamemsg().substring(0,1));
12.        data.add(temp);
13.        simpleAdapter.notifyDataSetChanged();
14.    }

```

至此，实验完成。

### 三、 实验结果

主界面：                      购物车界面：                      商品详情界面：



### 四、 实验遇到的困难和解决方法

首先遇到的困难是 RecyclerView 的使用，网上关于 RecyclerView 的博客有很多，但内容出入很大，改了很多次都不成功。后来耐心理解了 RecyclerView 各个方法的含义，才真正学会了使用 RecyclerView。

遇到的另一个困难是商品详情页面的布局。最顶部占屏幕的 1/3，而且图片还要居中，刚开始尝试了各种各样的布局，都搞不定，可能是用不熟悉的缘故。后来用约束布局的引导线，轻松解决，现在真心觉得约束布局是最好用的布局。

还遇到一个问题，就是点击购物车图标后商品添加到购物车。最开始想到的还是 Intent，但是 Intent 貌似一定要跳转到其他页面去……后来经过 TA 提示，才知道可以用 EventBus。于是就百度了 EventBus 的用法。按照教程一步一步添加代码后，还是不行。最后偶然看到一篇博客，才知道大多数博客中漏掉了 @Subscribe 这一句。

## 五、实验思考及感想

这次实验，难度相较前几次实验，有了质的飞跃……以前碰到这种老大难的作业，大概会很急躁，自暴自弃。但是由于对 android 比较感兴趣，所以就一直鼓励自己，只要多花一点时间，就一定能够完成。结果，差不多花了两三天的时间，终于完整地做出来了。对比自己的 app 和给的 demo，基本没有什么差异，真的蛮有成就感。这次感触比较深的关于博客的使用。刚开始接触一个知识点，什么都不会，首先想到的当然是去网上看看人家怎么用，然后再依样画葫芦，慢慢熟悉这个知识点。网上的博客不乏精品，的确给了我很大的帮助。但是也有一些博客，断章取义，代码这摘一段那摘一段，让人十分糊涂，白白浪费很多时间，所以总结一点就是，遇到新的知识点，可以先查看官方文档，努力去理解，而不是照搬博客的内容，遇到它们

不对的地方也能够有辨识力，不至于往坑里跳。