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：

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

在 MainActivity 中设置 Adapter，为了实现动画效果，调用 ScaleInAnimationAdapter 类。

```
final HomeAdapter myAdapter = new HomeAdapter();
 ScaleInAnimationAdapter animationAdapter = new ScaleInAnimationAdapter(myAdapter);
 animationAdapter.setDuration(1000);
 final RecyclerView myRecView = (RecyclerView) findViewById(R.id.id_recyclerview);
 myRecView.setLayoutManager(new LinearLayoutManager(this));
 myRecView.setAdapter(animationAdapter);
 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 setOnItemClickLitener(OnItemClickLitener mOnItemClickLitener)；

RecyclerView 没有点击和长按事件，需要自己定义。OnItemClickLitener 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 OnItemClickLitener mOnItemClickLitener;
14.
15.     public void setOnItemClickLitener(OnItemClickLitener mOnItemClickLitener)
16.     {
17.         this.mOnItemClickLitener = mOnItemClickLitener;
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 (mOnItemClickLitener != null)
26.         {
27.             holder.itemView.setOnClickListener(new View.OnClickListener()
28.             {
```

```
                @Override
                public void onClick(View v)
                {
                    int pos = holder.getLayoutPosition();
                    mOnItemClickLitener.onItemClick(holder.itemView, pos);
                }
            });

            holder.itemView.setOnLongClickListener(new View.OnLongClickListener()
            {
                @Override
                public boolean onLongClick(View v)
                {
                    int pos = holder.getLayoutPosition();
                    mOnItemClickLitener.onItemLongClick(holder.itemView, pos);
                    return false;
                }
            });
        }
    }

    public void removeData(int position) {
        mDatas.remove(position);
        notifyItemRemoved(position);
    }

    @Override
    public int getItemCount()
    {
        return mDatas.size();
    }

    class MyViewHolder extends RecyclerView.ViewHolder
    {

        TextView tv;
        Button btn;

        public MyViewHolder(View view)
        {
            super(view);
            tv = (TextView) view.findViewById(R.id.id_num);
            btn = (Button) view.findViewById(R.id.button);
        }
    }
}
```

自定义的 OnItemClickLitener 接口：

```
interface OnItemClickLitener {
    void onItemClick(View view, int position);
    void onItemLongClick(View view , int position);
}
```

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。

```
myAdapter.setOnItemClickLitener(new OnItemClickLitener()
    {

```

```
        @Override
        public void onItemClick(View view, int position)
        {
            Intent intent = new Intent(MainActivity.this, NewActivity.class);
            String extra = mDatas.get(position);
            intent.putExtra("passname", extra);
            MainActivity.this.startActivity(intent);
        }

        @Override
        public void onItemLongClick(View view, int position)
        {
            Toast.makeText(MainActivity.this, "移除第" + position + "个商品",
                    Toast.LENGTH_SHORT).show();
            myAdapter.removeData(position);
        }
    });
```

RecyclerView 的布局，使用 FrameLayout 布局实现：

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    xmlns:android="http://schemas.android.com/apk/res/android">

        />
    <Button
        android:id="@+id/button"
        android:layout_width="60dp"
        android:layout_height="60dp"
        android:background="@drawable/shape"
        android:text="E"
        android:textSize="35dp"
        android:textColor="@color/buttontext"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="53dp"
        />

    <TextView
        android:id="@+id/id_num"
        android:layout_width="match_parent"
        android:layout_height="50dp"
        android:gravity="left"
        android:text="1"
        android:textColor="@color/black"
        android:textSize="27dp"
        android:layout_marginLeft="90dp"
        android:layout_marginTop="66dp"
      />

</FrameLayout>
```

RecyclerView 的数据，使用 initData 进行初始化：

```
private void initData(){
       mDatas = new ArrayList<String>();
       mDatas.add("Enchated Forest");
       mDatas.add("Arla Milk");
       mDatas.add("Devondale Milk");
       mDatas.add("Kindle Oasis");
       mDatas.add("waitrose 早餐麦片");
```

```
        mDatas.add("Mcvitie's 饼干");
        mDatas.add("Ferrero Rocher");
        mDatas.add("Maltesers");
        mDatas.add("Lindt");
        mDatas.add("Borggreve");
    }
```

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：

```
<FrameLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <android.support.design.widget.FloatingActionButton
            android:id="@+id/floatButton"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="bottom|end"
            android:layout_margin="45dp"
            android:elevation="20dp"
            android:src="@mipmap/shoplist"
            android:background="#FFF"
            app:backgroundTint="#FFFFFF" />
    </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：

```
final FloatingActionButton floatBtn = (FloatingActionButton) findViewById(R.id.floatButton);
floatBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (tag == 0) {
            myRecView.setVisibility(View.INVISIBLE);
            shopList.setVisibility(View.VISIBLE);
            floatBtn.setImageResource(R.mipmap.mainpage);
            tag = 1;
        }
        else if (tag == 1){
            shopList.setVisibility(View.INVISIBLE);
            myRecView.setVisibility(View.VISIBLE);
            floatBtn.setImageResource(R.mipmap.shoplist);
            tag = 0;
        }
    }
});
```

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：

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:descendantFocusability="blocksDescendants"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <Button
        android:id="@+id/shopButton"
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:background="@drawable/shape"
        android:text="*"
        android:textSize="20dp"
        android:textColor="@color/buttontext"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="22dp"
        />
    <TextView
        android:id="@+id/goods"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="15dp"
        android:textSize="15sp"
        android:textColor="#000000"
        android:layout_marginTop="15dp"
        android:layout_marginBottom="15dp"
        android:text="购物车购物车购物车"
        />
    <TextView
        android:id="@+id/price"
        android:layout_weight="1"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:gravity="end"
        android:padding="15dp"
        android:textSize="15sp"
        android:textColor="#000000"
        android:layout_marginTop="15dp"
        android:layout_marginBottom="15dp"
        android:text="价格"
        />
</LinearLayout>
```

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

```
<ListView
    android:id="@+id/shoppinglist"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:clickable="true"
    xmlns:android="http://schemas.android.com/apk/res/android">

```

```
</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：

```
final ListView shopList = (ListView) findViewById(R.id.shoppinglist);
  String[] goods = new String[]{"购物车"};
  String[] price = new String[]{"价格"};
  for (int i = 0; i < goods.length; i++) {
      Map<String, String> temp = new LinkedHashMap<String, String>();
      temp.put("goods", goods[i]);
      temp.put("price", price[i]);
      if (goods[i] == "购物车"){
          temp.put("shopButton", "*");
      }
      else{
          temp.put("shopButton", goods[i].substring(0, 1));
      }
      data.add(temp);
  }
```

使用 SimpleAdapter：

```
simpleAdapter = new SimpleAdapter(MainActivity.this, data, R.layout.item_shop,
             new String[]{"goods", "price","shopButton"}, new int[]{R.id.goods, R.id.price, R.id.shopButton});
     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：

```
shopList.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            if (position == 0){}
            else {
                Intent intent = new Intent(MainActivity.this, NewActivity.class);
                Map<String, String> extra = data.get(position);
                intent.putExtra("passname", extra.get("goods"));
                MainActivity.this.startActivity(intent);
            }
        }
    });
    shopList.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener(){
        @Override
        public boolean onItemLongClick(AdapterView<?> arg0, View arg1,
                                       final int arg2, long arg3) {
            AlertDialog.Builder dialogbuilder = new AlertDialog.Builder(MainActivity.this);
            dialogbuilder.setTitle("上传头像");
            String temp = data.get(arg2).get("goods");
```

```
            dialogbuilder.setMessage("从购物车移除" + temp +"?");
            dialogbuilder.setNegativeButton("取消", click1);
            dialogbuilder.setPositiveButton("确定",
                new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialogInterface, int i) {
                    data.remove(arg2);
                    simpleAdapter.notifyDataSetChanged();
                }
            });
            AlertDialog dialog = dialogbuilder.create();
            dialog.show();
            return true;
        }
        private DialogInterface.OnClickListener click1 = new DialogInterface.OnClickList
ener(){
            @Override
            public void onClick(DialogInterface arg0, int arg1){
                arg0.cancel();
            }
        };
    });
```

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，底部更多商品信息同样使用 ListView 实现。布局文件如下：

```
<android.support.constraint.ConstraintLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FFF"
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
  <android.support.constraint.Guideline
      android:id="@+id/guideline1"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:orientation="horizontal"
      app:layout_constraintGuide_percent="0.333333"
      />
   <android.support.constraint.Guideline
       android:id="@+id/guideline2"
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       android:orientation="vertical"
       app:layout_constraintGuide_percent="0.25"
       />
   <android.support.constraint.Guideline
       android:id="@+id/guideline3"
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       android:orientation="vertical"
       app:layout_constraintGuide_percent="0.75"
       />
 <android.support.constraint.Guideline
     android:id="@+id/guideline4"
     android:layout_width="match_parent"
     android:layout_height="match_parent"
     android:orientation="horizontal"
     app:layout_constraintGuide_percent="0.46"
     />
```

```
    <ImageView
        android:id="@+id/leftBlank"
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:adjustViewBounds="true"
        android:src="@mipmap/background"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="@+id/guideline1"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toLeftOf="@+id/photo"
        />
    <ImageView
        android:id="@+id/rightBlank"
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:adjustViewBounds="true"
        android:src="@mipmap/background"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="@+id/guideline1"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintLeft_toRightOf="@+id/photo"
        />
   <ImageView
       android:id="@+id/photo"
       android:layout_width="wrap_content"
       android:layout_height="0dp"
       android:adjustViewBounds="true"
       android:src="@mipmap/ferrero"
       app:layout_constraintTop_toTopOf="parent"
       app:layout_constraintBottom_toBottomOf="@+id/guideline1"
       app:layout_constraintLeft_toLeftOf="@+id/guideline2"
       app:layout_constraintRight_toRightOf="@+id/guideline3"
       />
   <TextView
       android:id="@+id/shopName"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:text="Enchated Forest"
       android:textColor="@color/black"
       android:textSize="20sp"
       android:layout_marginLeft="20dp"
       app:layout_constraintLeft_toLeftOf="parent"
       app:layout_constraintBottom_toBottomOf="@+id/guideline1"
       android:layout_marginBottom="20dp"
       />
    <ImageView
        android:id="@+id/star"
        android:layout_width="35dp"
        android:layout_height="35dp"
        android:src="@mipmap/empty_star"
        app:layout_constraintBottom_toBottomOf="@+id/shopName"
        app:layout_constraintRight_toRightOf="parent"
        android:layout_marginRight="20dp"
        />
     <TextView
         android:id="@+id/shopPrice"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="￥5.00"
         android:textColor="#D5000000"
         android:layout_marginLeft="20dp"
         app:layout_constraintLeft_toLeftOf="parent"
         app:layout_constraintTop_toBottomOf="@id/leftBlank"
         android:layout_marginTop="10dp"
         />
    <TextView
     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"
```

```
        android:layout_marginTop="12dp"
        />
    <ListView
        android:id="@+id/moreInfo"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toBottomOf="@+id/moreLine"
        >

    </ListView>

</android.support.constraint.ConstraintLayout>
```

ListView 实现类似，直接上代码：

Xml：

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <TextView
        android:id="@+id/infoItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:paddingLeft="0dp"
        android:textSize="20sp"
        android:textColor="#D5000000"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="13dp"
        android:layout_marginBottom="13dp"
        android:text="一键下单"
        />
</LinearLayout>
```

Java：

```
ListView moreInfo = (ListView) findViewById(R.id.moreInfo);
    List<Map<String, String>> data = new ArrayList<Map<String, String>>();
    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
"};
    for(int i = 0; i < infoItem.length; i++){
        Map<String, String> temp = new LinkedHashMap<String, String>();
        temp.put("infoItem", infoItem[i]);
        data.add(temp);
    }
    SimpleAdapter simpleAdapter = new SimpleAdapter(NewActivity.this, data, R.layout.mor
e_info,
            new String[]{"infoItem"}, new int[]{R.id.infoItem});
    moreInfo.setAdapter(simpleAdapter);
```

点击星标空心和实心切换，使用 flag 来“记住”状态：

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

```
            if (flag == 0){
                star.setImageResource(R.mipmap.full_star);
                flag = 1;
            }
            else if (flag == 1){
                star.setImageResource(R.mipmap.empty_star);
                flag = 0;
            }
        }
    });
```

设置返回按钮，点击后调用 finish()即可：

```
    ImageView back = (ImageView) findViewById(R.id.back);
        back.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
```

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：

```
    final Map<String, String> name_price = new LinkedHashMap<String, String>();
     name_price.put("Enchated Forest", "¥5.00");
     name_price.put("Arla Milk", "¥59.00");
     name_price.put("Devondale Milk", "¥79.00 ");
     name_price.put("Kindle Oasis", "¥2399.00");
     name_price.put("waitrose 早餐麦片", "¥179.00");
     name_price.put("Mcvitie's 饼干", "¥14.90");
     name_price.put("Ferrero Rocher", "¥132.59 ");
     name_price.put("Maltesers", "¥141.43");
     name_price.put("Lindt", "¥139.43");
     name_price.put("Borggreve", "¥28.90 ");
     Map<String, String> name_info = new LinkedHashMap<String, String>();
     name_info.put("Enchated Forest", "作者 Johanna Basford");
     name_info.put("Arla Milk", "产地 德国");
     name_info.put("Devondale Milk", "产地 澳大利亚");
     name_info.put("Kindle Oasis", "版本 8GB");
     name_info.put("waitrose 早餐麦片", "重量 2Kg");
     name_info.put("Mcvitie's 饼干", "产地 英国");
     name_info.put("Ferrero Rocher", "重量 300g");
     name_info.put("Maltesers", "重量 118g");
     name_info.put("Lindt", "重量 249g");
```

```
name_info.put("Borggreve", "重量 640g");
Map<String, String> name_pic = new LinkedHashMap<String, String>();
name_pic.put("Enchated Forest", String.valueOf(R.mipmap.enchatedforest));
name_pic.put("Arla Milk", String.valueOf(R.mipmap.arla));
name_pic.put("Devondale Milk", String.valueOf(R.mipmap.devondale));
name_pic.put("Kindle Oasis", String.valueOf(R.mipmap.kindle));
name_pic.put("waitrose 早餐麦片", String.valueOf(R.mipmap.waitrose));
name_pic.put("Mcvitie's 饼干", String.valueOf(R.mipmap.mcvitie));
name_pic.put("Ferrero Rocher", String.valueOf(R.mipmap.ferrero));
name_pic.put("Maltesers", String.valueOf(R.mipmap.maltesers));
name_pic.put("Lindt", String.valueOf(R.mipmap.lindt));
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：

```
Bundle bundle = getIntent().getExtras();
if(bundle != null){
    passname = bundle.getString("passname");
}
TextView shopname = (TextView) findViewById(R.id.shopName);
shopname.setText(passname);
String price = name_price.get(passname);
TextView shopprice = (TextView) findViewById(R.id.shopPrice);
shopprice.setText(price);
String info = name_info.get(passname);
TextView shopinfo = (TextView) findViewById(R.id.shopInfo);
shopinfo.setText(info);
ImageView photo = (ImageView) findViewById(R.id.photo);
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：

```
public class FirstEvent {
    private String namemsg;
    private String pricemsg;
    public FirstEvent(String str1, String str2) {
        namemsg = str1;
        pricemsg = str2;
    }
    public String getNamemsg(){
        return namemsg;
    }
    public String getPricemsg(){
        return pricemsg;
    }
}
```

在新页面中设置发布者：

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

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

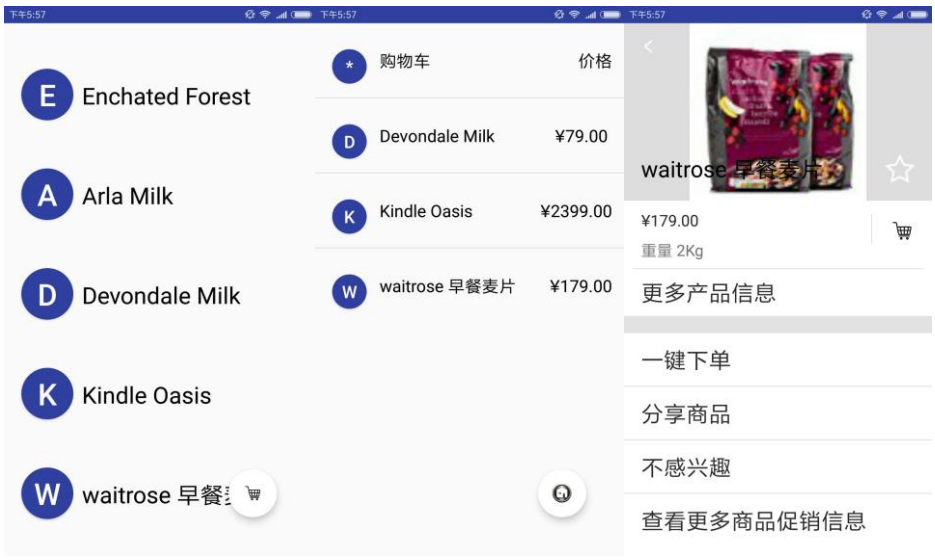
在主界面中设置注册，解除注册和处理时间：

```
EventBus.getDefault().register(this);
    protected void onDestroy(){
        super.onDestroy();
        EventBus.getDefault().unregister(this);//反注册 EventBus
    }
    @Subscribe
    public void onEventMainThread(FirstEvent event) {
        Map<String, String> temp = new LinkedHashMap<String, String>();
        temp.put("goods", event.getNamemsg());
        temp.put("price", event.getPricemsg());
        temp.put("shopButton", event.getNamemsg().substring(0,1));
        data.add(temp);
        simpleAdapter.notifyDataSetChanged();
    }
```

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。