# 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

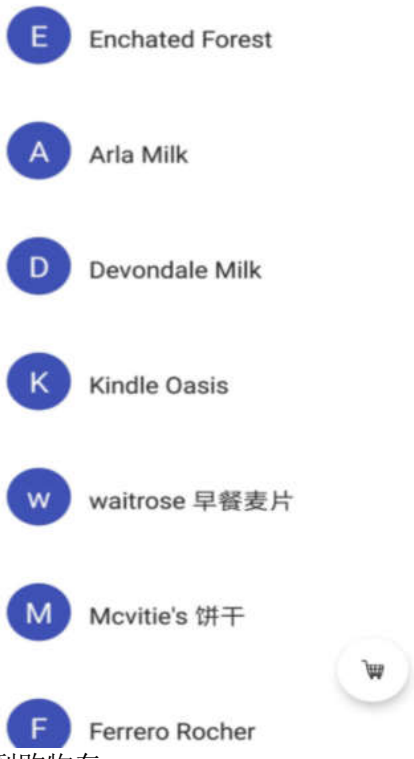
| 年级 | 2015 | 专业（方向） | 移动互联网 |
|---|---|---|---|
| 学号 | 15352344 | 姓名 | 吴文标 |
| 电话 | 13112312320 | Email | wwb.bill@qq.com |
| 开始日期 | 2017.10.20 | 完成日期 | 2017.10.26 |

## 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。

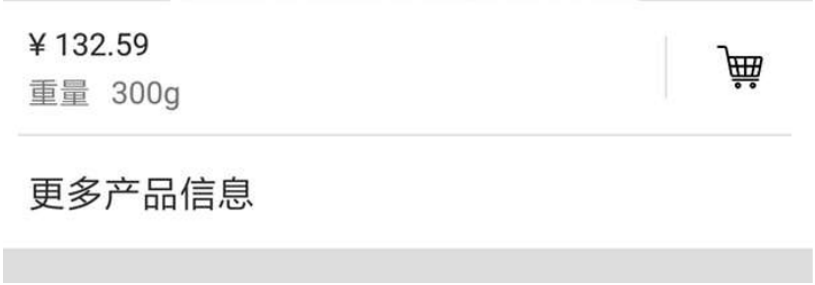
2、 购物车列表界面

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#05000000 ， 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#1 E000000 ， 右边购物车符号的左边也有一条分割线 ，argb 编 码值也是#1 E000000 ，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


这个没什么说的 ，内容和样式都很清楚。

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

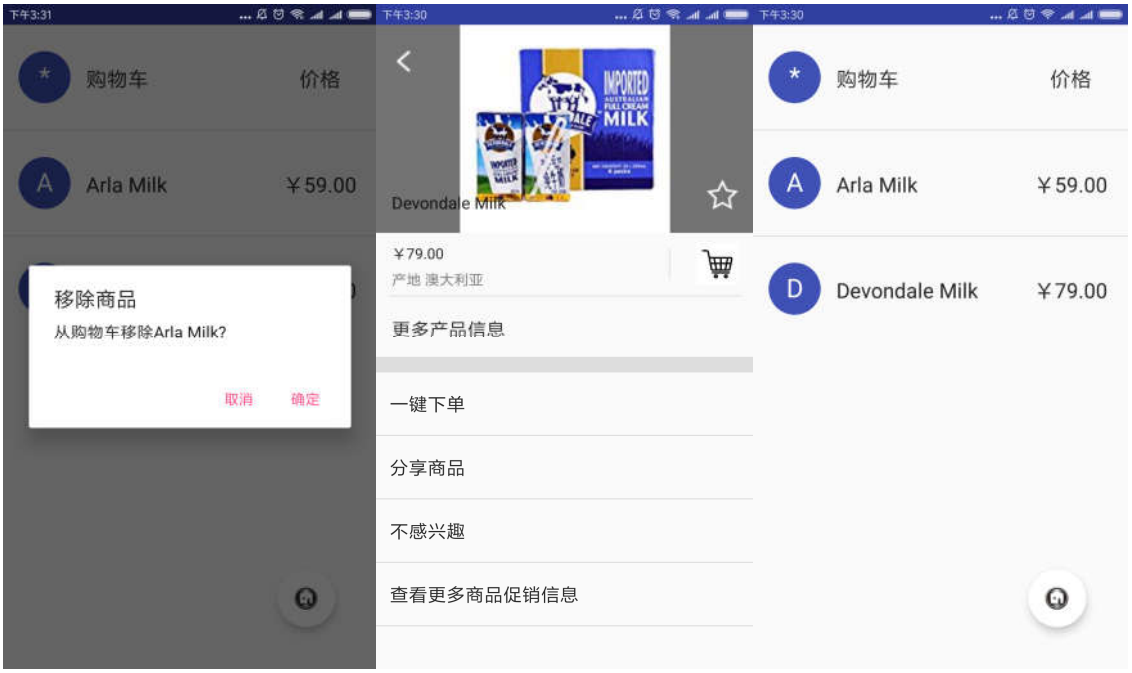
个商品’
2、点击右下方的 FloatingActionButton，从商品列表切换至 U 购物车或从购物车切换到商品列表，并且 FloatingActionButton 的图片要做相应改变。可通过设置 RecyclerView 不可见，ListView 可见来实现从商品列表切换至 U 购物车。可通过设置 RecyclerView 可见，ListView 不可见来实现从购物车切换到商品列表。
3、使用 ListView 实现购物车。点击购物车的某一个商品会跳转到商品详情界面 ，呈现该商品的详细信息 ；长按购物中的商品会弹出对话框询问是否移除该商品 ，点击确定则移除该商品 ，点击取消则对话框消失。


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


购物车
价格
Arla Milk
￥59.00
移除商品
从购物车移除Arla Milk?
取消
确定
Devondale Milk
￥79.00
产地 澳大利亚
更多产品信息
一键下单
分享商品
不感兴趣
查看更多商品促销信息
购物车
价格
Arla Milk
￥59.00
Devondale Milk
￥79.00


Enchated Forest
Arla Milk
Devondale Milk
Kindle Oasis
Waitrose 早餐麦片
Mcvitie's 饼干
购物车
价格
Enchated Forest
Arla Milk
Devondale Milk
Kindle Oasis
Waitrose 早餐麦片
Mcvitie's 饼干

Figure 1 全览

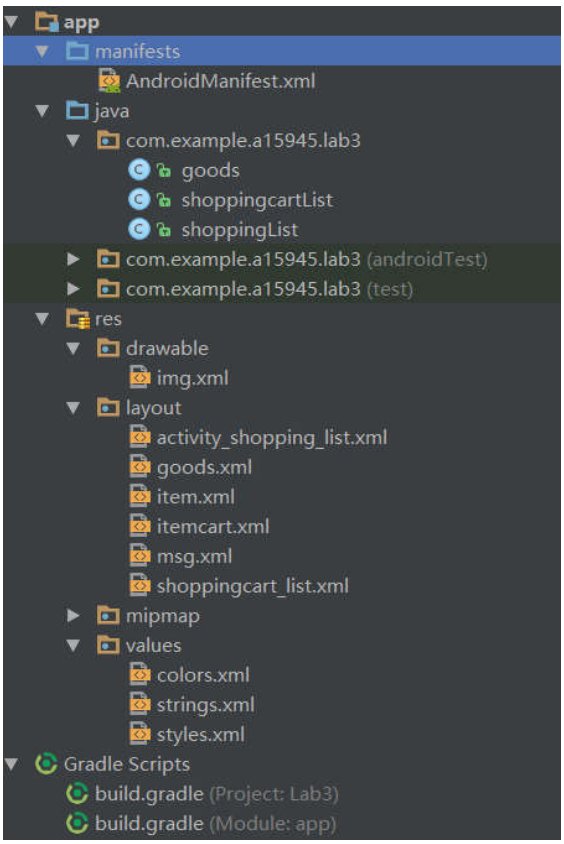



Figure 2 主体代码

（2） 实验步骤以及关键代码

步骤 1：新建 app 文件，构造第一个主界面，按要求使用了 RelativeLayout 相对布局，以及 recyclerView 控件。

关键代码：

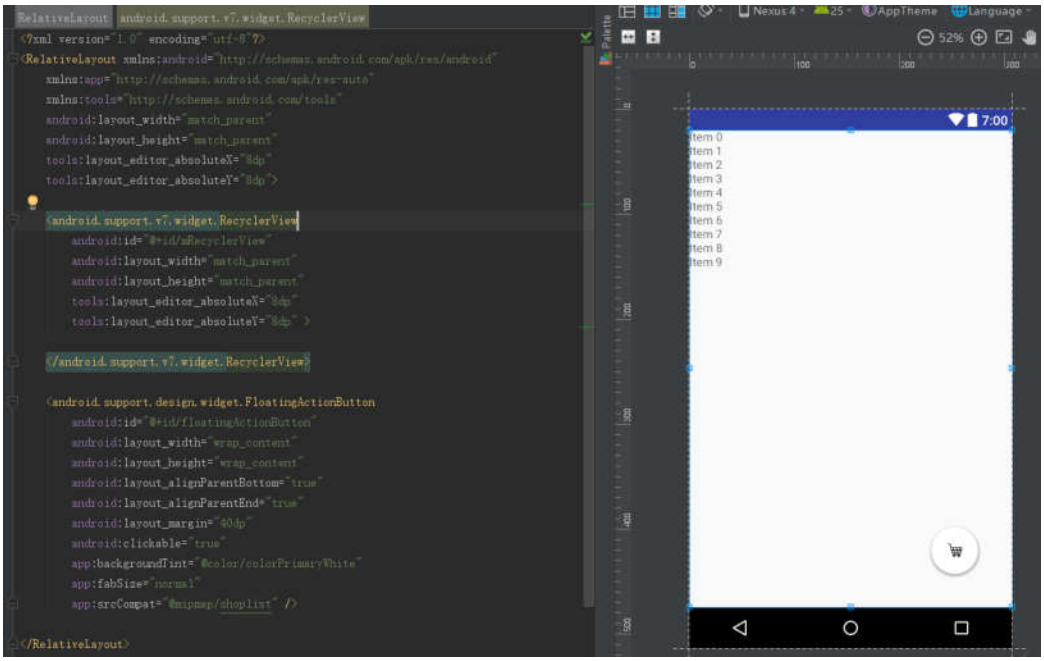


Figure 3 activity_shoping_list.xml

步骤 2：构造 item 布局。ImageView 导入 drawable 样式再控制长度宽度来构造一个圆，通过 Android：gravity 等来进行文字居中处理。

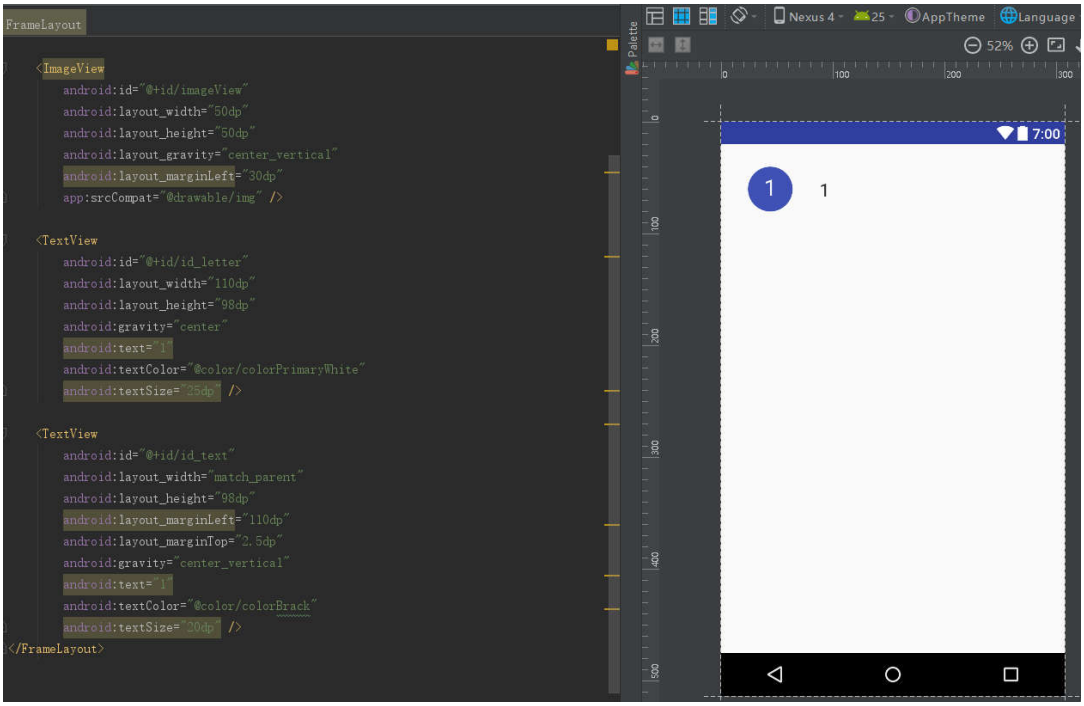


Figure 4 activity_shoping_list.xml

步骤 3：给 recyclerView 自定义适配器

1） 自定义 ViewHolder：

ViewHolder 在适配器中主要是在 listView、RecyclerView 滚动时快速设置值，而不必每次都重新设置，提升性能。

创建 MyViewHolder 类，关联 item 布局的控件：

```
class MyViewHolder extends ViewHolder
{
    TextView tv;
    TextView tv1;
    ImageView imgV;

    public MyViewHolder(View view)
    {
        super(view);
        tv = (TextView) view.findViewById(R.id.id_letter);
        tv1 = (TextView) view.findViewById(R.id.id_text);
        imgV = (ImageView) view.findViewById(R.id.imageView);
    }
}
```

获取 ViewHolder 实例，连接 item 布局：

```
@Override
public MyViewHolder onCreateViewHolder(ViewGroup parent, int viewType){
    MyViewHolder holder = new MyViewHolder(LayoutInflater.from(
            shoppingList.this).inflate(R.layout.item, parent,
            false));
    return holder;
}
```

对 item 布局的控件进行操作，绑定数据：

```
    @Override
    public void onBindViewHolder(final MyViewHolder holder, int position)
    {
        holder.tv.setText(myDatas[position]);
        holder.tv1.setText(myDatas1[position]);
        if(myDatas[position] == null){
            holder.imgV.setVisibility(View.INVISIBLE);
        }
```

数据的来源：

```
protected void initData() {
    myDatas = getResources().getStringArray(R.array.item_letter_array);
    myDatas1 = getResources().getStringArray(R.array.item_name_array);
}
```

```
<string-array name="item_letter_array">
    <item>E</item>
    <item>A</item>
    <item>D</item>
    <item>K</item>
    <item>W</item>
    <item>M</item>
    <item>F</item>
    <item>M</item>
    <item>L</item>
    <item>B</item>
</string-array>
<string-array name="item_name_array">
    <item>Enchated Forest</item>
    <item>Arla Milk</item>
    <item>Devondale Milk</item>
    <item>Kindle Oasis</item>
    <item>Waitrose 早餐麦片</item>
    <item>Mcvitie\'s 饼干</item>
    <item>Ferrero Rocher</item>
    <item>Maltesers</item>
    <item>Lindt</item>
    <item>Borggreve</item>
</string-array>
```

获取 item 总数：

```
    @Override
    public int getItemCount() { return myDatas.length; }
```

2） 添加点击事件

接口添加（在 adapter 类外部添加）：

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

设置点击事件方法：

```
//设置点击事件的方法
public void setItemClickListener(OnItemClickListener itemClickListener){
    this.itemClickListener = itemClickListener;
}
```

在 onBindViewHolder（）添加点击事件：

```
public void onBindViewHolder(final MyViewHolder holder, int position)
{
    holder.tv.setText(myDatas[position]);
    holder.tv1.setText(myDatas1[position]);
    if(myDatas[position] == null){
        holder.imgV.setVisibility(View.INVISIBLE);
    }
    if(itemClickListener!=null){
        holder.itemView.setOnClickListener((v) -> {
                itemClickListener.onClick(holder.getAdapterPosition());
        });
        holder.itemView.setOnLongClickListener((v) -> {
                itemClickListener.onLongClick(holder.getAdapterPosition());
                return false;
        });
    }
}
```

3） 调用适配器并添加动画

这里只需要添加源：

```
compile 'jp.wasabeef:recyclerview-animators:2.2.5'
```

```
mRecyclerView = (RecyclerView) findViewById(R.id.mRecyclerView);
mRecyclerView.setLayoutManager(new LinearLayoutManager(this));
mAdapter = new commonAdapter();

ScaleInAnimationAdapter animationAdapter = new ScaleInAnimationAdapter(mAdapter);
animationAdapter.setDuration(1000);
mRecyclerView.setAdapter(animationAdapter);
mRecyclerView.setItemAnimator(new OvershootInLeftAnimator());
```

步骤 4：构造购物车页面，这里需要用到的是 ListView，布局使用相对布局

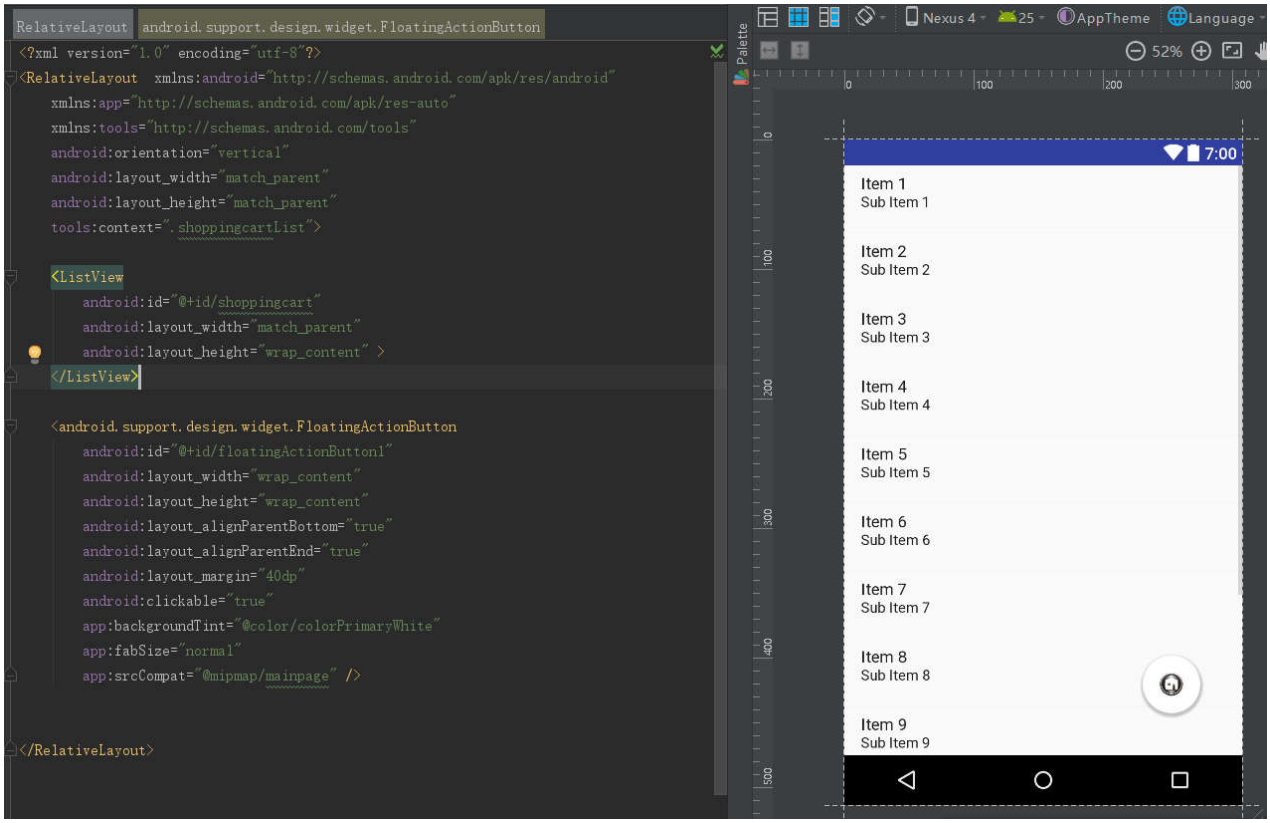



Figure 5 shoppingcart_list.xml

这里由于是新建了一个 activity 所以需要在 AndroidManifest 文件中添加：

```
<activity android:name=".shoppingcartList">

</activity>
```

使用 SimpleAdapter 适配器给 ListView 添加数据

```
simpleAdapter = new SimpleAdapter(this,list,R.layout.itemcart,
        new String[]{"img","letter","name","price"},
        new int[]{R.id.imageView1,R.id.id_letter1,R.id.id_text1,R.id.id_price1});
simpleAdapter.notifyDataSetChanged();
listView.setAdapter(simpleAdapter);
```

初始数据：

```
public List<Map<String,Object>> getData(){
    List<Map<String, Object>> list1 = new ArrayList<>();

    Map<String, Object> map = new LinkedHashMap<>();
    map.put("img", R.drawable.img);
    map.put("letter", "*");
    map.put("name", "购物车");
    map.put("price","价格");
    list1.add(map);
```

定义 push 函数方便给 list 添加数据：

```
public Map<String,Object> push(int position){
    Map<String, Object> map = new LinkedHashMap<>();
    switch (position){
        case 0:
            map.put("img", R.drawable.img);
            map.put("letter", "E");
            map.put("name", "Enchated Forest");
            map.put("price","￥5.00");
            break;
        case 1:
            map.put("img", R.drawable.img);
            map.put("letter", "A");
            map.put("name", "Arla Milk");
            map.put("price","￥59.00");
            break;
        case 2:
            map.put("img", R.drawable.img);
            map.put("letter", "D");
            map.put("name", "Devondale Milk");
            map.put("price","￥79.00");
            break;
        case 3:
            map.put("img", R.drawable.img);
            map.put("letter", "K");
            map.put("name", "Kindle Oasis");
            map.put("price","￥2399.00");
            break;
```

步骤 5：页面跳转。这里通过 FloatingActionButton 的点击事件实现。

```
FAB = (FloatingActionButton)findViewById(R.id.floatingActionButton);
FABListener fabListener = new FABListener();
FAB.setOnClickListener(fabListener);
```

```
class FABListener implements View.OnClickListener {
    @Override
    public void onClick(View v) {
            Intent intent = new Intent();
            intent.setClass(shoppingList.this,shoppingcartList.class);
            startActivity(intent);
           // overridePendingTransition(R.anim.open_in,  R.anim.open_out);
    }
}
```

```
floatingActionButton = (FloatingActionButton)findViewById(R.id.floatingActionButton1);
floatingActionButton.setOnClickListener((v) -> {
        Intent intent = new Intent();
        intent.setClass(shoppingcartList.this,shoppingList.class);
        startActivity(intent);
});
```

步骤六：详情页面制作

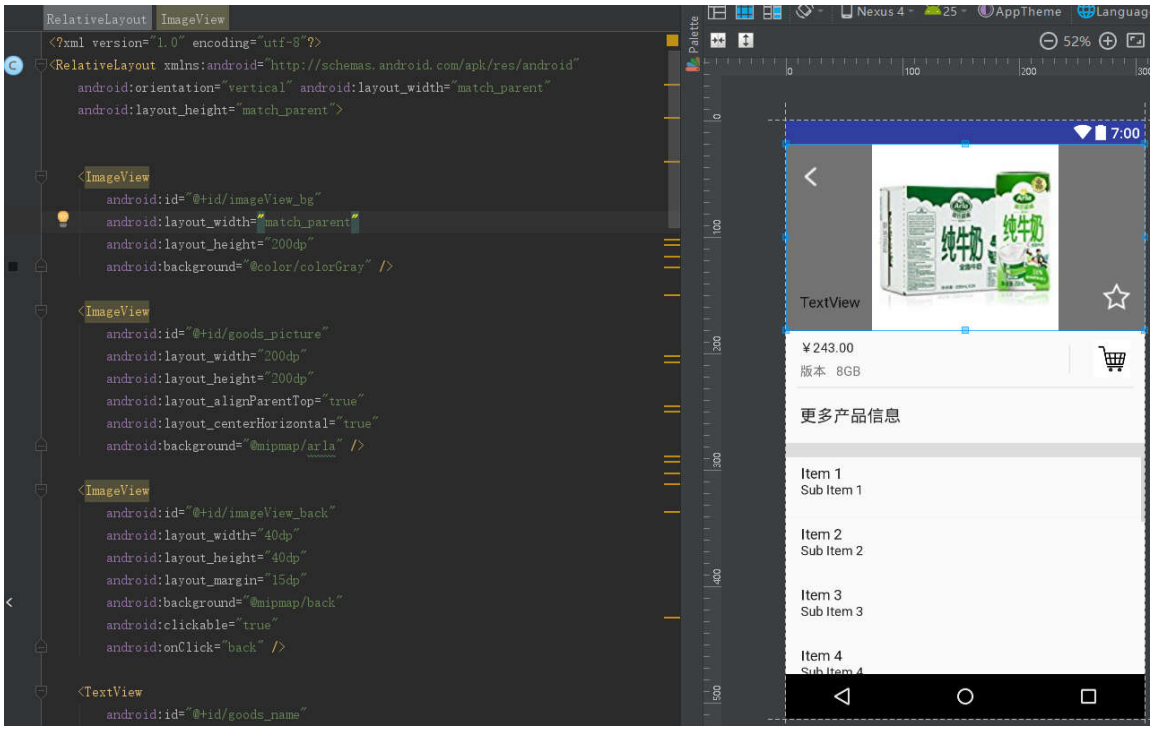


Figure 6 goods.xml

这里需要用到比例来设置高度占比，提供两种方法：

一是，可以把 android：layout_height 设置为 3/1.5/1.5/4，物品图片设为 3，在 xml 里面就会解析为 3/（3+1.5+1.5+4）也就是占高度的 30%。

二是，在 java 文件里动态获取屏幕高度然后设置图片高度为 30%。

这里由于已经做好了整体布局，由于时间问题，我并没有写进去再修改，不过以上两种方法都是可行的设置方法。

分割线实现：

```
<View
    android:layout_width="match_parent"
    android:layout_height="1dp"
    android:layout_marginTop="260dp"
    android:layout_gravity="center_vertical"
    android:layout_marginLeft="12dp"
    android:layout_marginRight="12dp"
    android:background="@color/colorLine"/>
```

页面点击事件实现部分代码：

返回：

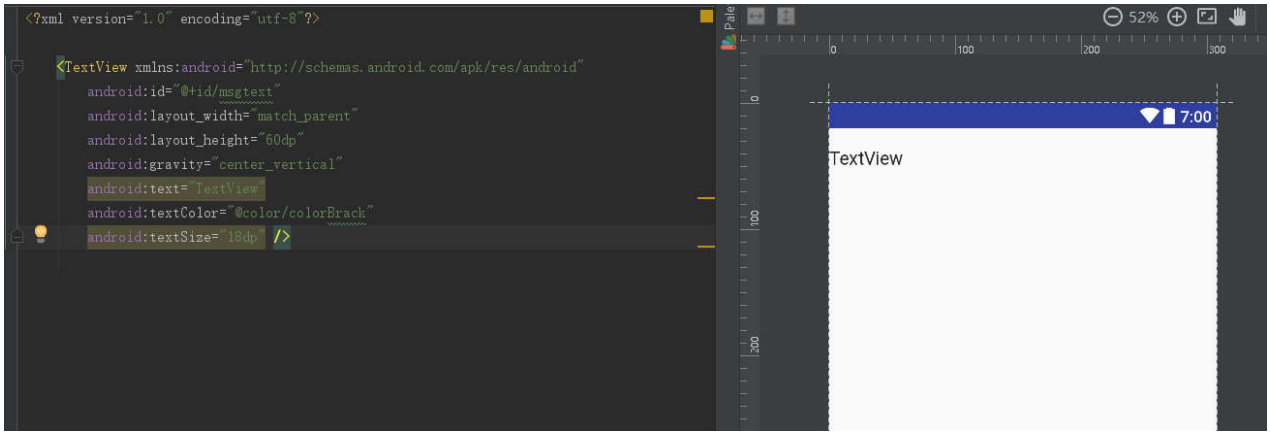
```
public void back(View view) { goods.this.finish(); }
```

*PS：此方式放回的页面无法刷新，后面有修复*

Star：

```
public void star(View view){
    ImageView imageView = (ImageView)findViewById(R.id.imageStar);
    if(flag == true) {
        //Toast.makeText(getApplication(),"s",Toast.LENGTH_SHORT).show();
        imageView.setImageResource(R.mipmap.full_star);
        flag = false;
    }
    else {
        imageView.setImageResource(R.mipmap.empty_star);
        flag = true;
    }
}
```

ListView：




ArrayAdapter 适配器便可轻松实现：

```
listView = (ListView)findViewById(R.id.listView_goods);
String[] Msg = new String[]{"  一键下单","  分享商品","  不感兴趣","  查看更多商品促销信息"};
ArrayAdapter<String> arrayAdapter = new ArrayAdapter<>(this,R.layout.msg,Msg);
listView.setAdapter(arrayAdapter);
listView.addFooterView(new ViewStub(this));
```

点击购物车给后台添加数据：

```
public void shopping(View view){
    Toast.makeText(getApplication(),"商品已加到购物车",Toast.LENGTH_SHORT).show();
    textView = (TextView)findViewById(R.id.goods_name);
    String str = (String)textView.getText();
    int pos = 0;

    for(int i = 0;i<data_name.size();i++){
        if(str.equals(data_name.get(i))){
            pos = i;
            SharedPreferences.Editor editor = getSharedPreferences("data",MODE_PRIVATE).edit();
            editor.putInt("number"+j,pos);
            editor.commit();
       //     Toast.makeText(getApplicationContext(), pos+"设置成功", Toast.LENGTH_SHORT).show()
            j++;
            break;
        }
    }
    SharedPreferences.Editor editor = getSharedPreferences("data",MODE_PRIVATE).edit();
    editor.putInt("j",j);
    editor.commit();
```

这里为了实现可以一直点击添加商品，一般的传输数据无法实现（或者只是自己不知道怎么实现）然后动用了数据库这一骚操作。这里使用的是 SharedPreferences 能以键值对的方式存储数据在 xml 文件里面，几番折腾后才写出来的。

购物车里面的数据获取代码实现：

```
SharedPreferences read = getSharedPreferences("data", MODE_PRIVATE);
int value = read.getInt("j",0);
int p = 0;
list = getData();
for (int j = 0;j<value;j++){
    p = read.getInt("number"+j,100);
    if(p != 100)
    list.add(push(p));
    else{
        //p == 100 说明已经删除数据，下面用替换的方式删除数据源
        SharedPreferences.Editor editor = getSharedPreferences("data",MODE_PRIVATE).edit();
        int k = 0;
        for(k = j;k<value-1;k++){
            int k1 = read.getInt("number"+(k+1),100);
            if(k1 != 100) {
                editor.putInt("number" + k, k1);
                editor.commit();
            }
                else break;
        }
        editor.remove("number"+ k);
        editor.commit();

    }
```

这里也因为删除事件的存在费了好大笔墨。

步骤 7：onClick 点击事件。这里需要实现对应跳转。

1）主界面

```
mAdapter.setItemClickListener(new OnItemClickListener() {
    @Override
    public void onClick(int position) {
        for(int i = 0;i<rmBug;i++){
            if(rmBugArr[i] <= position){
                position++;
            }
        }

        Intent intent = new Intent();
        intent.putExtra("position",position);
        intent.setClass(shoppingList.this,goods.class);
        startActivity(intent);
```

由于数据存储在数据库（strings.xml 或其他）中，传递过去的信息只需要是一个标记数字即可。

该部分实现对应的方法主要是通过静态数组来存储每一次的删除（如果有的话）的位置，然后比较当前点击的 position 和数组里面的删除位置信息再来适当调整，最终发送的 position 便依旧能对应着相应的标记号，也就不会被后面的长按删除 item 所影响。

2）购物车界面

```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        Map<String,Object> map = list.get(position);
        String name = (String)map.get("name");
        if(!name.equals("购物车")) {
            Intent intent = new Intent();
            intent.putExtra("name", name);
            intent.setClass(shoppingcartList.this, goods.class);
            startActivity(intent);
        }else
        Toast.makeText(getApplication() ,"我要剁手", Toast.LENGTH_LONG).show();

    }
});
```

购物车的数据存储方式不同于前面的，这里是

```
private List<Map<String, Object>> list;
```

所以可以通过 get（）方法获取当前位置的数据，也就可以通过“name”这一独一无二的标签来判断，然后就是详情页面判断收到的信息，进行转化：

```
int i = 10000;
position = intent.getIntExtra("position", i);

if(position == 10000) {
    String name = intent.getStringExtra("name");
    for (int k = 0; k < data_name.size(); k++) {
        if (name.equals(data_name.get(k))) {
            position = k;
            break;
        }
    }
}
```

（由于会收到来自两个 acticity 的 intent，所以需要做一些判断）

详情页面收到 position 信息后就可以分配数据了，如下：

```
textView = (TextView)findViewById(R.id.goods_name);
textView.setText(data_name.get(position));
textView = (TextView)findViewById(R.id.goods_price);
textView.setText(data_price.get(position));
textView = (TextView)findViewById(R.id.goods_msg);
textView.setText(data_msg.get(position));

data_img_id = new int[]{R.mipmap.enchatedforest,R.mipmap.arla,R.mipmap.devondale,R.
imageView = (ImageView)findViewById(R.id.goods_picture);
imageView.setImageResource(data_img_id[position]);
```

数据的定义：

```
protected void initData() {
    String[] myData = getResources().getStringArray(R.array.item_name_array);
    String[] myData1 = getResources().getStringArray(R.array.goods_price_array);
    String[] myData2 = getResources().getStringArray(R.array.goods_msg_item);

    data_name = Arrays.asList(myData);
    data_price = Arrays.asList(myData1);
    data_msg = Arrays.asList(myData2);
}
```

步骤 8：长按删除事件实现。

1）主界面

```
@Override
public void onLongClick(int position) {
    mAdapter.removeItem(position);
    String toast = "删除第"+position+"个商品";
    rmBugArr[rmBug] = position;
    rmBug++;
    Toast.makeText(getApplication(), toast, Toast.LENGTH_SHORT).show();
}
```

静态数组 rmBugArr 存储每一次删除的位置信息，和前面的点击适配。

removeItem（）：

```
public void removeItem(int position){
    rmData(myDatas,position);
    rmData(myDatas1,position);
    notifyItemRemoved(position);//刷新被删除的地方
    notifyItemRangeChanged(position, getItemCount()); //刷新被删除数据，以及其后面的数据
```

删除数据源然后刷新。

2）购物车界面

```
listView.setOnItemLongClickListener((parent, view, position, id) -> {
        Map<String,Object> map = list.get(position);
        String name = (String)map.get("name");
        if(!name.equals("购物车")) {
            new AlertDialog.Builder(shoppingcartList.this)
                    .setTitle("移除商品")
                    .setMessage("从购物车移除" + name + "?")
                    .setPositiveButton("确定", new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                          //  Toast.makeText(getApplication(), "您选择了[确定]", Toast.LENGTH_SHORT).show();
                            list.remove(position);
                            simpleAdapter.notifyDataSetChanged();
                            SharedPreferences.Editor editor = getSharedPreferences("data",MODE_PRIVATE).edit();
                            editor.remove("number"+(position-1));
                            editor.commit();
                        }
                    })
                    .setNegativeButton("取消", (dialog, which) -> {
                                Toast.makeText(getApplication(), "您选择了取消", Toast.LENGTH_SHORT).show();
                            }
                    ).show();
        }
        return false;
});
```

这里的删除实现起来比较简单，主要是删除数据源麻烦了些，麻烦也是因为个人用了 SharedPreferences 数据库造成的。

（3） 实验遇到困难以及解决思路

1）一开始布局文件使用的是线性布局 LinearLayout，会出现一些问题，比如首界面使用 recyclerView 后无法对齐窗口的上顶部和左顶部，后来去查阅了一下关于安卓布局的文档，了解到相对布局和线性布局的一些区别，最后改用 RecyclerLayout 解决。

2） Activity 的初始化需要在 AndroidManifest 中添加相关代码，而的跳转需要用到 Intent，一开始不知道导致出来不少问题，后来在相关的博客中看到相关描述解决。

3）点击返回图标时购物车界面不能自动刷新，后来在查询了相关的 API 和文档后发现了通过定义函数来实现打开页面或转到该页面便刷新的功能。代码如下：

```
//自动刷新
@Override
protected void onResume() {
    super.onResume();
    onCreate(null);
}
```

4）在做分割线的时候遇到的一个问题，大体就是元素没有大写，因为是 View 这个自己实现的控件，不同于其他控件自动生成不会出错，这里我一不小心就写成了 view 所以错了，还一

直找不到问题，后来把 log 信息复制到网上去查询才找到的原因。

5） 删除事件的实现遇到了非常非常多的 bug，能写出来是经过了无数次百度 google，以及和小伙伴讨论的结果。

## 四、 课后实验结果

实现了动画的添加。

## 五、 实验思考及感想

难度系数直线飙升的一个实验，遇到了无尽的 bug，为此 chrome 第一次因为标签页打开太多而崩溃。做完后回去看看实验文档以及和同学交流发现其实很多 bug 都可以避免，例如 commonAdapter 如果是配置得足够好是完全可以无视删除之一操作的，因为只需要一行代码！这时才惊觉多和队友同学交流是多么重要，感觉可以节省一半时间！然后有点想吐槽的是实验文档还是太模糊了，如果是一开始做的话是根本看不懂实验文档，到了后期看得懂了却又没办法修改已经做了的部分，所以还是很希望实验文档能够给更多的指导，起码逻辑哟啊足够清晰，对新手友好。

作业要求：

1. 命名要求： 学号_姓名_实验编号，例如 15330000_林 XX_lab1。
2. 实验报告提交格式为 pdf。
3. 实验内容不允许抄袭，我们要进行代码相似度对比。如发现抄袭，按 0 分处理。