**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： 郑贵峰

|  |  |  |  |
| --- | --- | --- | --- |
| 年级 | 2015 | 专业（方向） | 移动信息工程（M2） |
| 学号 | 15352193 | 姓名 | 梁公勋 |
| 电话 | 15066299571 | Email | 1104695389@qq.com |
| 开始日期 | 2017.10.19 | 完成日期 | 2017.10.24 |

# 实验题目

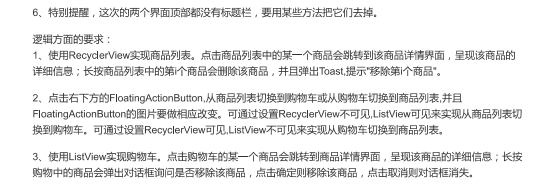
Intent、Bundle的使用和RecyclerView、ListView的应用。

# 实现内容



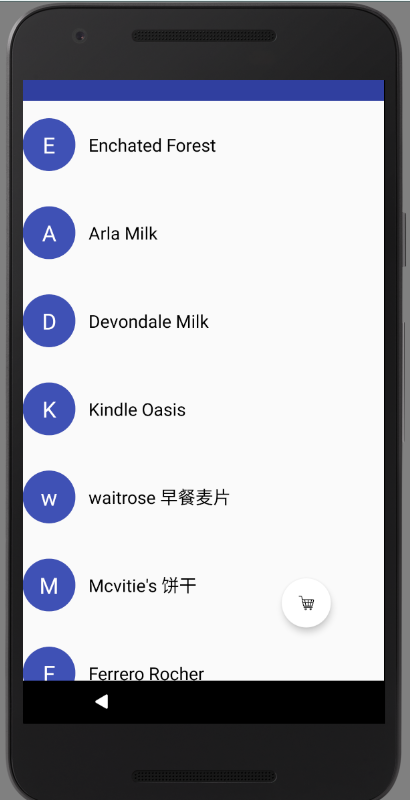
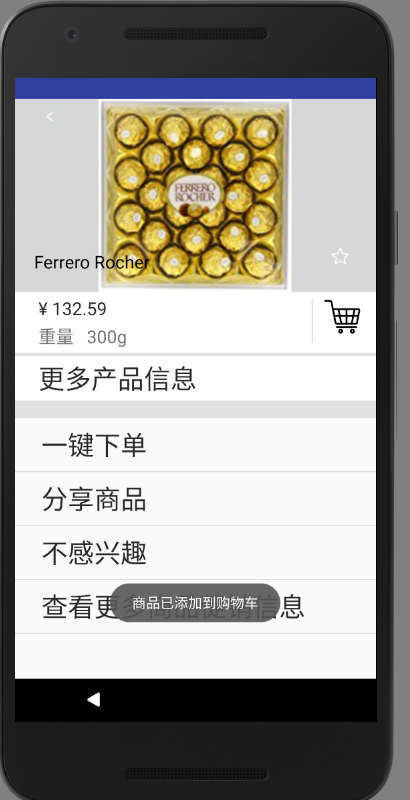


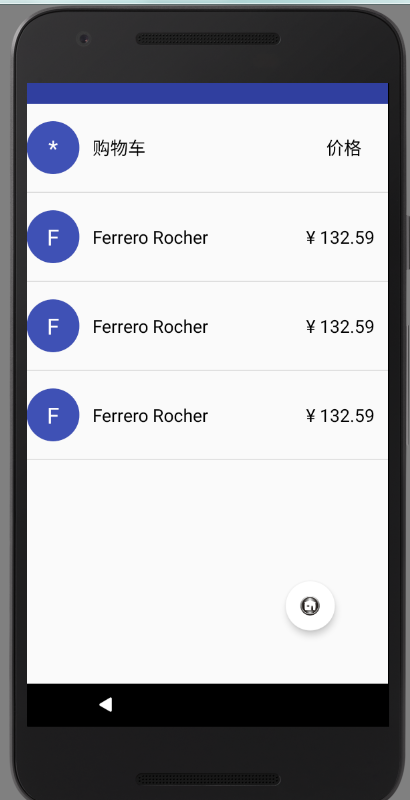


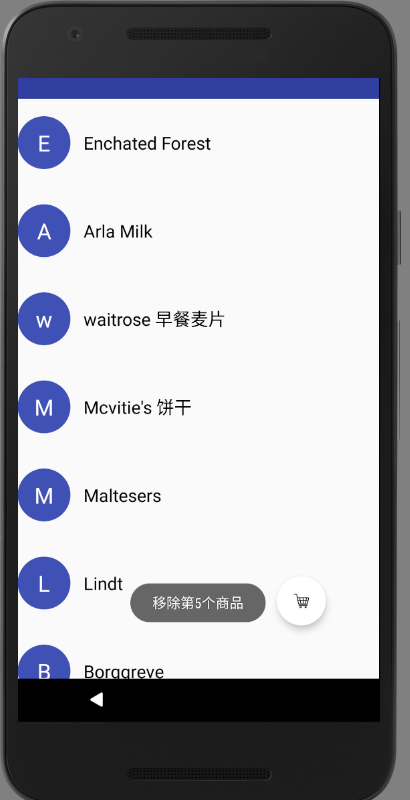


# 课堂实验结果

1. 实验截图

1. 实验步骤以及关键代码

1、初始界面XML布局的设计

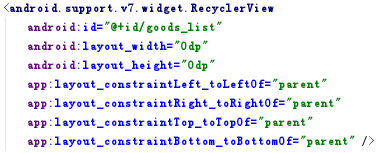
初始界面一共有三个控件。用于实现商品列表的RecyclerView，用于实现购物车列表的ListView以及用于两者切换的FloatingActionButton**。**

为了能够使用RecyclerView，我们首先在gradle文件中添加如下依赖。





添加依赖后即可使用RecyclerView控件，如下将其添加至XML文件中。



设置了宽度和高度，并添加了到父容器的依赖。

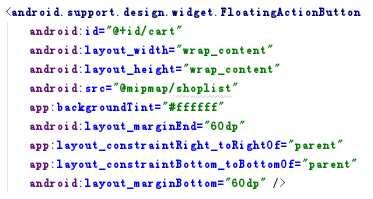
添加ListView控件。也是设置高度宽度后，添加到父容器的依赖。



为了使用FloatingActionButton控件，我们需要在gradle文件中添加如下依赖。



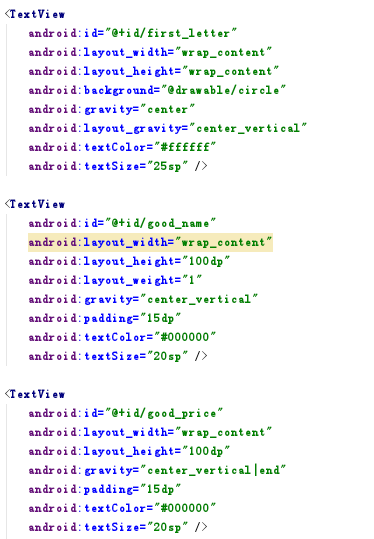
然后向布局文件中添加FloatingActionButton控件。高度和宽度为适用内容大小。然后设置其初始背景图片为那个购物车的图片，背景颜色设为全白。将其约束至父容器的右下方。



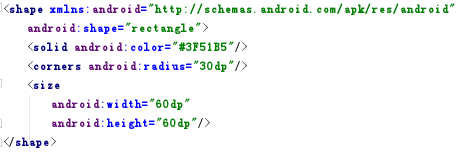
2、RecyclerView和ListView中每一项item的布局设计。

接下来要设计RecyclerView和ListView中每一项item的布局。这个布局中是三个TextView控件。其中，第一个用来放置首字母，第二个用来放置 商品名称，第三个用来商品价格。RecyclerView和ListView都是使用这个item布局，不同的是，RecycleView中没有将商品价格与布局中的第三个TextView进行绑定，所以在显示的时候就不会将价格显示出来。

三个TextView的代码如下。首字母的那个使用了自定义背景circle，字体设为白色。然后内容居中，控件竖直居中。名字的那个也是竖直居中，然后使用了layout\_gravity属性，这样将价格的文本框挤到最右边。字体颜色设为了黑色。价格文本框和名字文本框属性基本相同。两者都是固定高度，宽度适用内容大小。设置gravity属性使字体居中显示



首字母文本框的自定义背景如下。使用shape标签，将圆角半径设为宽和高的一半即可呈现圆形效果。



3、存放商品的类Goods

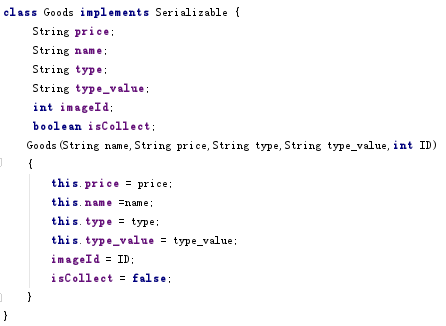
到这里尚有商品详情的XML布局没有解释。这里先放一放，讲完商品列表和购物车列表的实现后，再回去讲商品详情的布局及其Activity。这里所要讲的是最基本的一个商品的类Goods。

这个类的代码如下，为了方便调用，所有的变量都是public的。（可能不太符合代码规则）。除此之外就只有一个构造函数。

变量price代表商品价格，name代表商品名称，type代表类型，type\_value代表产品信息，（类型、产品信息可能不同，这里全部使用字符串存放，就无须关心其为何种类型，产品信息为何种变量。），imageId为产品图片的布局ID，布尔变量isCollect代表这个商品是否被收藏。

关键词Serializable将这个类序列化，以便其能在Activity间传递。

所有的变量在构造函数中初始化。

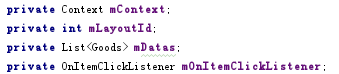


4、RecyclerView的自定义Adapter的实现

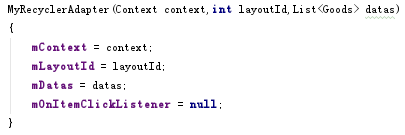
声明类的代码如下。我们声明了一个MyRecyclerAdapter的类，这个类继承自RecyclerView.Adapter。



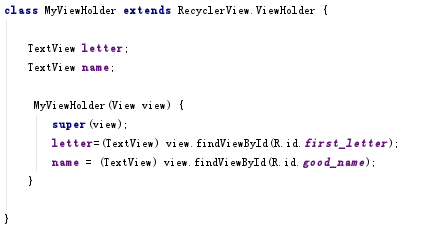
这个类有如下的变量。mContext是一应用接口，mLayoutId时item的布局文件ID，mDatas是在Adapter中要和RecyclerView中的item绑定的数据，mOnItemClickListener是我们自定义的一个监听器。



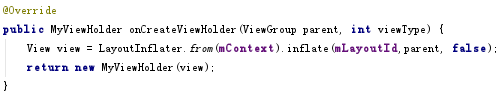
构造函数设计如下：



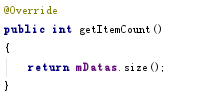
另外这个类中还有一个自定义的子类MyViewHolder，继承自RecyclerView的ViewHolder。在这个ViewHolder中，有两个控件letter和name，分别代表首字母和名字并在构造函数中获取布局中的控件id。然后我们将使用这个MyViewHolder替代控件中的View去进行数据绑定，以提高效率。



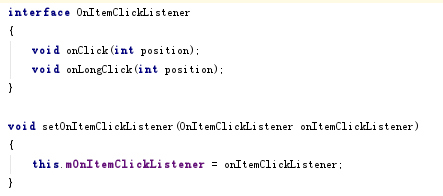
onCreateViewHolder函数创建item的view，返回相应的ViewHolder。这里的返回类型是我们自定义的一个继承自ViewHolder的类MyViewHolder。这里创建item视图时需要刚刚提到的布局ID。第二行代码的意思就是将通过id引用布局并将其加载到context的主布局中。



函数getItemCount，返回item的数目，也就是相应绑定数据的mDatas的数目。



接下来是与点击事件有关的接口及绑定监听器的函数。由于RecyclerView中没有单击和长按事件，所以需要自定义监听器。实现接口，接口中提供onClick和onLongClick函数。绑定监听器的函数中，将传入的监听器赋值给类的监听器。



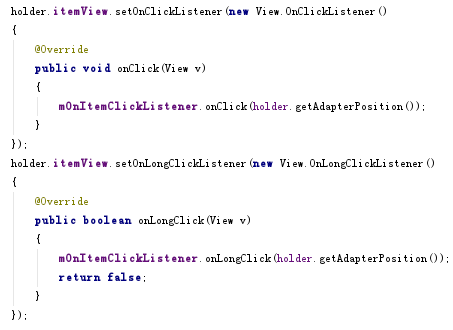
接下来是数据绑定的函数onBindViewHolder，声明如下。holder是用于绑定数据的变量，position是数据在List中的下标。这个函数中除了数据绑定部分外，还有设置监听器的函数，我们分开解释。



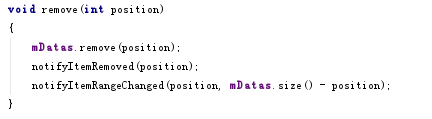
数据绑定的部分如下。letter绑定的时根据position在mDatas中找到相应的商品，再获取其名字，然后取其第一个字母，和holder中的letter绑定。name绑定的数据就是刚刚获取的名字。



然后是与点击事件相关的部分。这里为其中的一项item绑定监听器，分别绑定单击和长按事件的监听器。然后再使用类的监听器调用单击和长按函数。并将点击的item的下标传入。



最后是移除item的函数remove。形参position是移除项的下标。首先将数据项中的这个数据移除，再用notifyItemRemoved将RecyclerView中的这一项item删除。最后用notifyItemRangeChanged调整RecyclerView中删除的这一项之后的item下标，以防止下标中没有Item而发生错误。



5、ListView的自定义Adapter的实现

我们用ListView实现购物车列表，这里我们自定义一个适配器供ListView的使用。这个适配器的声明如下，继承于BaseAdapter类。

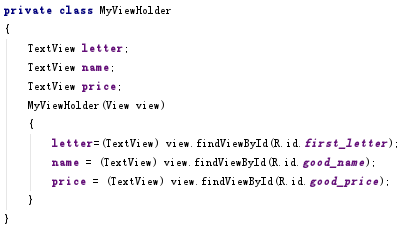


MyListAdapter类具有以下变量。mdatas就是要与View绑定的数据。mContext则是上下文接口。

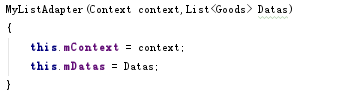


这个适配器也有一个私有的类变量MyViewHolder。与RecyclerView中的MyViewHolder不同，这里的MyViewHolder是我们自定义的类，不需要继承。而在RecyclerView中，重新封装了ViewHolder，那里的MyViewHolder是继承于RecyclerView中的ViewHolder的。

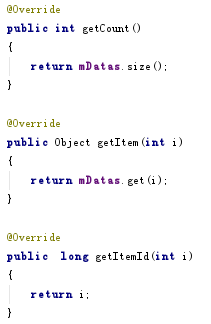
这个类有三个变量，letter和name已经提到过，price就是价格，因为购物车列表中需要显示价格。然后同样要在构造函数中获取控件ID。



MyListAdapter的构造函数如下，相应赋值即可。



实现MyListAdapter要重载四个函数。如下三个相对简单。分别是返回item的个数，返回item，返回item的位置。相应处理即可。



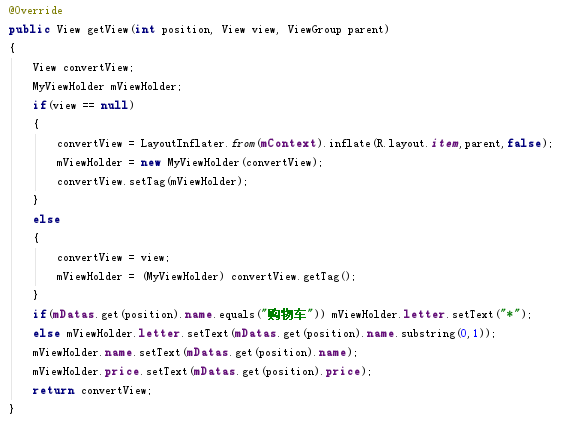
接下来是MyListAdapter中最重要的getView函数，这里返回填充了数据项的View。

三个形参里，position时列表项所要绑定的数据的下标，view是要绑定的item布局，parent是view的父容器。

然后声明了一个View变量convertView作为返回的View，声明了一个MyViewHolder的变量用来绑定数据，这里绑定数据的方法和RecyclerView那里相差无几。

如果View为空，先获取item的View并赋值给convertView，然后利用convertView创建一个ViewHolder，再使用setTag将这个ViewHolder放到convertView中。如果不为空的话，则将View直接赋值给convertView，然后使用getTag从中取出ViewHolder。

最后就是使用ViewHolder将数据与item视图绑定，由于ListView实现的是购物车列表，所有List里的第一项时购物车等表头，所以我们使用if语句，将第一项的首字母单独设置。其余的就是相应绑定数据即可。最后返回convertView。

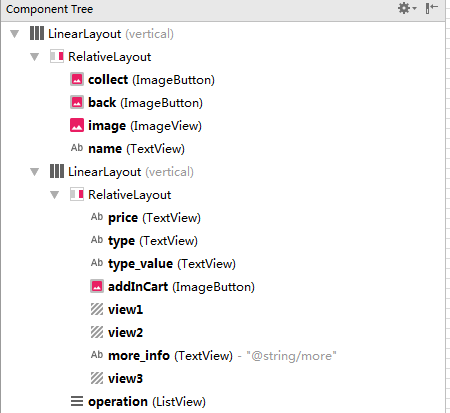


6、商品详情界面的布局文件

接下来讲商品详情界面的布局文件activity\_info。这个文件的控件比较多，先贴个可视化的图。

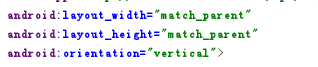


这个布局的树结构如下。

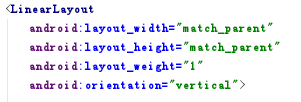
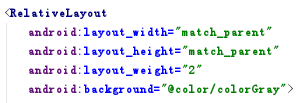


最外围是一个LinnerLayout布局，上部分占1/3的是一个RelativeLayout布局，放置图片，星标，返回，名字等信息。下部是一个占2/3的LinnerLayout布局。这个布局里又有一个RelativeLayout布局和一个ListView。RelativeLayout布局中放置了价格，类型，信息文本框，购物车图标、“更多产品信息”文本框，和一些分割线的View。ListView实现详情界面底部的四个信息。

首先是最外围的LinnerLayout的属性，要求竖直放置，大小匹配父容器



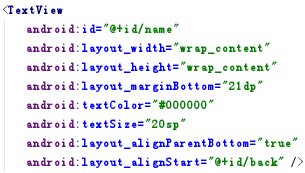
然后是上下两个RelativeLayout和LinnerLayout的属性。这里我们使用了layout\_weight设置了两个布局的相对大小。（按理说，RelativeLayout的权重应该为1，LinnerLayout的权重应该是2，这样才是RelativeLayout占1/3，但是不知为何我只有反着写它才占1/3）。



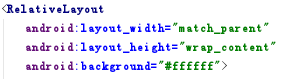
第一个RelativeLayout中的所有控件都是适应字体大小的。RelativeLayout中有一些属性可以与某元素或者是父元素的边缘对齐。以第一个ImageButton为例，这个控件是那个星星图标，宽高适应内容大小。layout\_alignBottom属性代表与哪个元素的底部边缘对齐，这里是与名字那个文本框对齐了。layout\_alignParentEnd代表与父容器右侧边缘对齐，用layout\_marginEnd设置了间距18dp。其他控件的对齐也基本如此。



再看一下名字文本框的属性。宽高也是适应内容大小，还设置了字体颜色与字号。这里使用layout\_alignStart属性与返回图标的控件右对齐。使用layout\_alignParentBottom属性使这个控件与父容器底部边缘对齐，使用layout\_marginBottom属性设置了其间距。

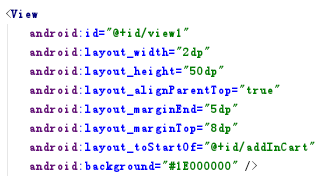


第一个RelativeLayout就将这些，接下来是底部的LinnerLayout。这个布局里也有一个RelativeLayout它的属性如下，我们将背景色设为了纯白，宽度充满父容器，高度适用内容。

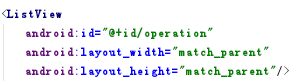


这个布局里是有TextView、 mageButton和一些分割线，TextView、 mageButton的属性和布局等代码我们之前都已熟悉，属性的值实验文档也有给出，这里不做过多解释。我们主要看一下分割线。这个RelativeLayout里一共有三个分割线，这些分割线全部是使用View控件实现的。这里以购物车图标左侧的分割线为例解释一下属性。

我们只需要设置好宽度和高度，以及其背景颜色，就可以将一个View以分割线的形式表现出来。不同的分割线只要相应修改这些属性值即可。属性layout\_toStartOf将其放置在购物车图标的左边，然后设置间距等即可。



底部LinnerLayout剩余部分是一个ListView，这个代码很简单。



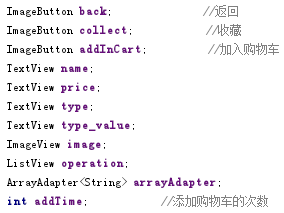
这个ListView的每一项item的布局文件是oper，这个布局文件的代码也很简单。这个布局只有一个文本框，设置好宽度高度，颜色字体。设置Gravity属性使字体居中显示。注意由于我们使用的这个ListView的适配器是ArrayAdapter，所有这个布局文件的根控件必须是TextView。



7、商品详情界面的Activity的实现

到这里还有MainAcitivity以及商品详情界面的InfoAcivity尚未解释。这里我们先讲解InfoAcivity。它的主布局就是我们刚刚讲的activity\_info。这个Activity的主要功能是将商品添加到购物车，收藏，返回这几个按钮。

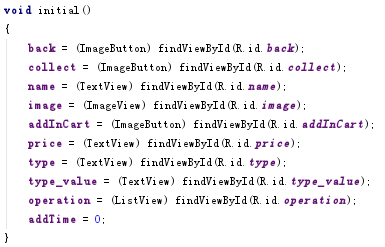
首先看一下它的变量，前三个变量是ImageButton，也是我们要在这个Activity中要实现的主要部分。几个TextView就是显示一些名字价格信息等。ListView是底部的列表，然后为它声明一个ArrayAdapter的适配器。addTime是点击购物车的次数，我们需要记录并传回MainAcitivity



然后使用如下代码设置了主布局以及变量初始化。



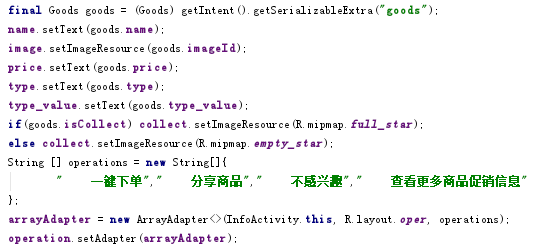
初始化的函数如下，addTime初始化为0，其他变量就是根据ID获取布局文件的控件。



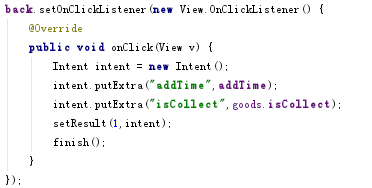
下边这段代码主要是向我们的布局中填入数据。首先，我们声明一个Goods类的对象，获取intent所携带的由MainAcitivity中传来的数据，这个goods就是我们展示详情的商品。

然后我们根据这对象的数据，为我们的布局设置数据，包括价格、名字、类型、信息、图片，并根据其收藏状态，显示星星图片。

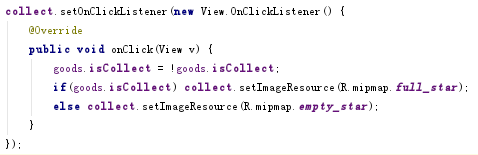
然后我们新建一个字符串operations，并想起填入我们要展示的数据，再使用这个字符串，和之前讲的item布局文件oper创建一个ArrayAdapter，再将这个适配器设置为ListView的适配器。



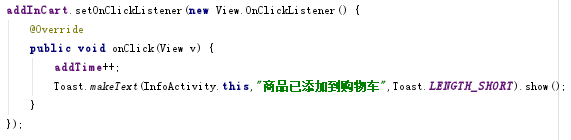
接下来讲各个按钮的点击事件。首先是back按钮的点击事件。刚开始也是绑定监听器，检测到单击后，我们新建一个Intent，然后向这个Intent中放入数据。一个是我们将这个商品添加到购物车的次数addTime，还有一个时这个商品的当前的收藏状态。放置后使用setResult函数，设置应答码并将Intent返回。最后调用finish函数结束这个Activity。



然后是星星图标那个收藏按钮。每次检测到点击事件后，我们就将这个商品的收藏状态取反，然后根据新的收藏状态设置星星图标的背景。



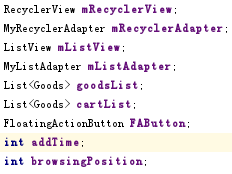
这是最后一个点击事件，也就是单击购物车图标后将商品添加到购物车的单击事件。，每次单击后我们只需要将之后返回的添加购物车的次数的变量addTime加一即可。然后就弹出一条Toast信息提示商品已添加至购物车。



最后，在创建这个Acitivity的时候，一定要将其在AndroidManifest文件中注册，这样这个Acitivity才能使用。

8、MainAcitivity的实现

终于来到最后一份代码，MainAcitivity。这个是APP初始的Acitivity。首先看一下这个Acitivity的变量。



mRecyclerAdapter是自定义适配器MyRecyclerAdapter对象，用来作为mRecyclerView的适配器。mListAdapter是一个自定义适配器MyListAdapter的对象，用来作为ListView的适配器。

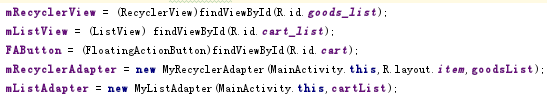
goodsList是存储商品列表的里的商品数据。cartList是购物车列表里的商品数据。FAButton是悬浮按钮

addTime是用来接受跳转到商品详情Activity时传回的添加到购物车的数量。browsingPosition是进入商品详情的那个商品在goodList中的下标。这个下标可能是goodsList的也可能是cartList的，稍后讲我们怎么用一个下标分开处理两个List的。

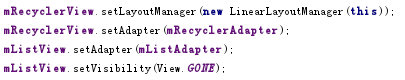
接下来看一下变量的初始化。首先是两个List的初始化，分别新建一个ArrayList。然后向goodsList中加入所有商品的信息。cartList最初是没有商品信息的数据的，但是它有一行购物车，所有我们把这个当作商品先填进去，没有的信息随便填，也不会显示。



然后是其他变量的初始化，一个是要从布局中根据ID获取控件。还有就是新建适配器，并将其item所要绑定的布局以及数据传入。



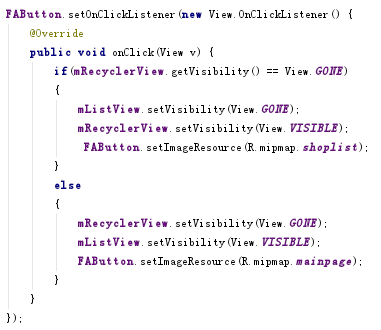
接下来这一部分，一个是为RecyclerView设置布局方式为线性布局。然后为RecyclerView和ListView绑定适配器。然后先将ListView设为不可见。



接下来是悬浮按钮的点击事件。检测到单击后，我们先判断是哪个View是可见状态。

如果是RecyclerView可见，我们就将其设为不可见，然后将ListView设为可见，然后重新设置悬浮按钮的背景图片，这样就完成了从商品列表到购物车列表的切换。

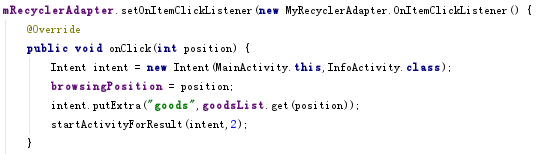
如果RecyclerView不可见，就将ListView设为不可见，再将RecyclerView设为可见，然后重新设置悬浮按钮的背景图片，这样就完成了从购物车到商品列表的切换。



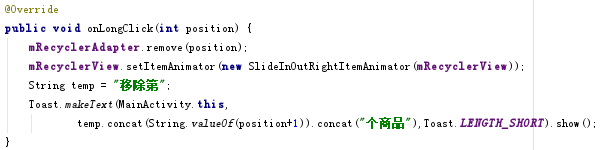
接下来是RecyclerView中的单击事件和长按事件。这两个函数是我们在MyRecyclerAdapter自定义实现的，先前已经解释过。

下边这个是单击事件。监听到单击事件后，我们要跳转到商品的详情界面。参数position是单击item项的下标。然后我们声明一个Intent变量进行跳转并传递数据。然后对browsingPosition赋值，记录我们进入的是哪个商品，等到之后得到回传的数据后对其进行处理。然后将这个商品放到intent中，将其传到InfoAvtivity。如果先前我们没用将Goods类使用关键词Serializable标记为序列化，这里是无法传递的。

最后我们使用startActivityForResult启动另一个Activity。如果不需要回调的数据，则使用startActivity即可，startActivityForResult是需要回调数据时调用的，intent是用来携带参数，数字2是一个请求码，设置请求码的主要目的是后边根据请求码分开处理从RecyclerView和ListView跳转进商品详情的情况。因为两个View中的List并不相同。



然后是处在同一个函数下的长按事件。长按事件就是在商品列表中删除对应的商品。这里直接调用了在MyRecyclerAdapter中自定义的remove函数，这个函数先前已经讲过，这里不做解释。接下来setItemAnimator函数是为删除item设置删除动画效果，然后参数是一个展示动画效果的对象SlideInOutRightItemAnimator。这个效果是从左向右滑动删除，这个我们稍微再讲。然后将点击项的下标position转换成字符串后与其他的内容相连接后，使用Toast弹出消息提示。



然后说一说这个动画效果。RecyclerView是只有一个默认的动画效果的，但保留了重载动画效果的类。起初确实是想自己写一些，但是发现原来写这个动画效果相当麻烦，刚好在百度的时候，发现了一个别人已经写好动画效果的库，就拿来用了，其中有 几种动画效果，我们用的是向右滑动消失。

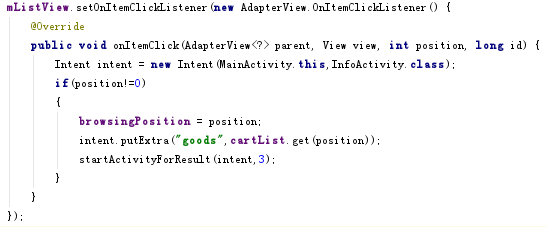
在gradle文件中加入如下部分代码，即可引用这个库，然后直接调用动画效果。



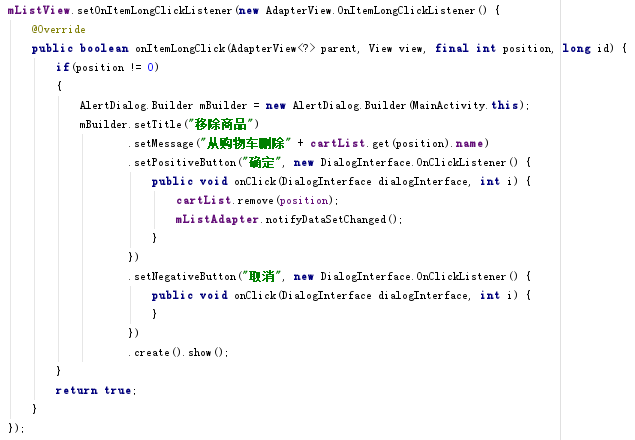


接下来是ListView中的单击事件和长按事件。ListView中本身封装了这个事件监听器，所以我们不需要重写，直接使用就好。

下边的是单击事件。购物车中的单击事件同样要跳转到商品详情列表，所以这里的代码和RecyclerView中单击跳转的代码基本相同。两处不同，一个是我们向InfoActivity中传入的商品是取自cartList，而不是goodsList。还有这里的请求码改为了3，以区别在RecyclerView中跳转所回调的数据，然后可以分开处理这两种从cartList和cartList跳转的不同情况。



然后是ListView中的长按事件。购物车中长按是弹出对话框，提示是否删除商品，如果确认就将其从购物车中删除。对话框的使用上个实验已经练习过，这里不做过多解释。这里主要是在上个的基础上又添加了对话框确定按钮的点击事件。单击了确定后，我们就将这个数据从cartList中移除，然后调用notifyDataSetChanged函数重新调整ListView的列表项。如果点击了取消，就不做操作。



然后是最后一段代码。这段代码的主要作用是处理Activity跳转后回调的数据。

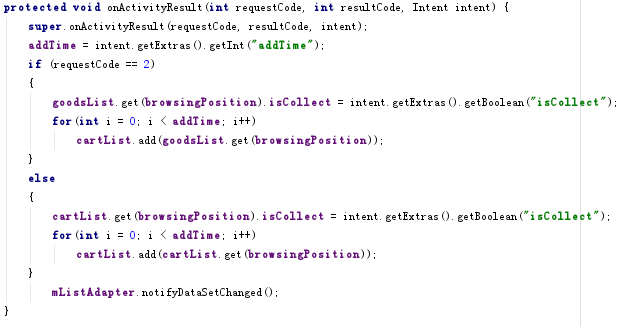
这个onActivityResult函数有以下几个参数 ，requestCode是请求码，是对跳转到另一个Activity的一个标记， resultCode是应答码是对Activity跳转回来时，在那个Activity里的标记。我们就是要根据这两个标记分别处理一些情况。, intent则是携带数据的容器。

我们先将addTime从intent中读取出来，这个变量之前也提到过，就是在商品详情界面，点击添加到购物车按钮的次数，然后我们将这个商品添加这么多次到购物车也就是cartList中。

然后我们根据请求码分别处理从商品列表进入商品详情，和从购物车列表进入商品详情两种情况。它们的请求码分别是2和3.（商品详情的Activity里，返回数据的地方只有一个，所以应答码也只有一个，这里不需要区别。）

然后进行if判断，如果请求码为2，说明是从商品列表进入的商品详情。接着，我们将从intent中取出商品的收藏状态并赋值给先前传入的这个商品。由于是从goodsList中调取这个商品。然后根据addTime，使用for循环，将这个商品再添加到cartList中。

如果应答码不是为2，说明是从购物车列表进入的商品详情，进行的操作都是一样的，不过调取商品时要把goodsList替换为cartList。



对于如何取消掉标题栏，我们在res文件中找到style.xml文件，然后在其中将主题修改为NoActionBar，这样就取消了标题栏。



1. 实验遇到困难以及解决思路

这次的实验真的时太多困难，怕也无法完全记着能列出来。有关百度安卓的历史记录都好几页了，这里就列一些我记着的吧。

1、一调用新建的Activity程序就崩溃。

这一个起初百度了很多，大多数都说是这个Activity的主布局文件出了问题。直到看见一篇博客说已经注册了Activity但还是崩溃，这才将注意力转移到注册上。最后就直接在文件列表新建一个Activity，而不是新建一个类了。

2、为ListView绑定ArrayAdapter的时候报错。

也看不懂报的时什么错，直接copy下来去百度，才知道ArrayAdapter为每一项item绑定的布局文件的根布局必须是TextView。

3、自定义RecyclerView的适配器时遇到的种种问题。

这里遇到的问题简直太多了，很多报错，很多不懂的地方，估计也无法一一列举出来，这里就写一下我的一些理解吧。

RecyclerView中的Adapter是已经封装好的抽象类。我们自定义的适配器就是继承这个抽象类，但是我们只有重载这个抽象类的所有虚函数，才能新建这个类的对象。重点，虚函数的名字不能打错。

另外，这个Adapter类还是一个模板类，所以我们在继承的时候，要声明它的ViewHolder的类型，然后才能用。

4、移除RecyclerView的item项时，程序下标混乱。

这一个是一个很典型的问题，百度一下也很容易找到解决方法。主要是移除了数据之后，下标会发生变化，所以要调用notifyItemRangeChanged函数重新设置下标。ListView中则使用notifyDataSetChanged函数。

5、从购物车列表进入商品详情，再点击收藏的那个星星之后，没有效果。

这其实是一个bug，主要原因是在Activity回调数据时没有利用好请求码。应该是根据请求码，对从购物车进入商品详情和从商品列表进入商品详情分开进行处理。

6、对于页面布局的一些问题

不知道为什么，感觉页面布局总是很玄学。像先前提到的那个layout\_weight属性的问题。当然也可能是我还没有发现问题所在。最后很多布局就是直接用边缘对齐和间距那两个属性强行解决，而不能很好的运用gravity和layout\_gravity属性

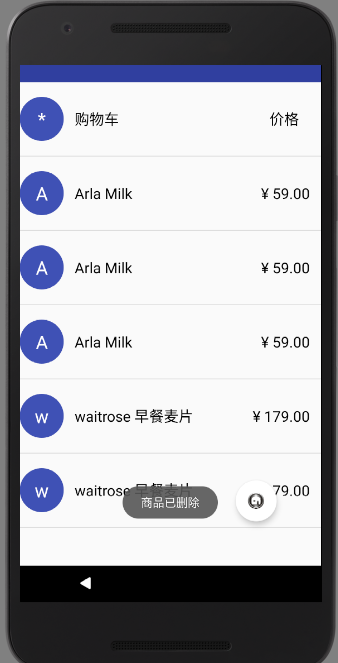
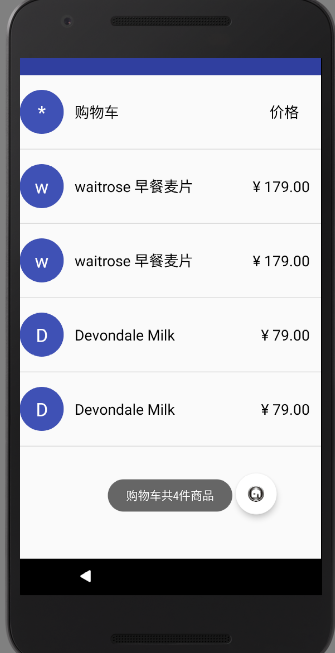
# 课后实验结果

这里没有运用新知识，也没有添加什么控件，只是增加了几条Toast信息，提高一下用户交互的效果。

第一个是购物车删除商品的时候弹出一条Toast信息，提示商品已删除。

第二个是长按购物车列表的购物车那一项时，弹出Toast信息提示购物车里商品的总数量。

第三个是点击星星图标的时候，弹出Toast信息，提示商品已收藏（取消收藏）。

## 实验思考及感想

这次实验的内容真的很多很多，而且大多要自己去百度，找代码样例，外加自己比较渣，做了差不多三天才做完。做的时候周围的人基本没做，所有有什么问题也不能讨论，只能自己解决。

不过通过这次实验也学到了很多有用的属性，对于安卓底层封装的一些类也有了一些了解。有时候别人问问题我也能比较清楚的知道是哪个地方的代码出了问题。不过除此之外，还是有很多地方并没有完全搞懂，只是知道用法。（百度的负面效应啊）但是能基本自己独立做完这个实验，还是很欣慰的。

最后感慨一句，感觉自己大学学到的最有用的东西就是学会了去百度。

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。