**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 |  | 完成日期 |  |

# 实验题目

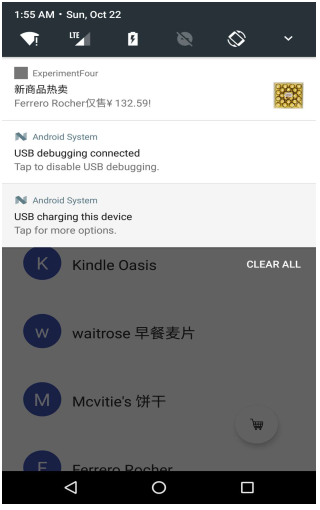
Broadcast 使用

# 实现内容

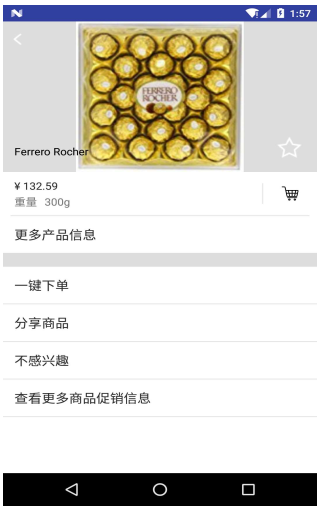
【实验内容】 在实验三的基础上，实现静态广播、动态广播两种改变Notification 内容的方法。

具体要求：

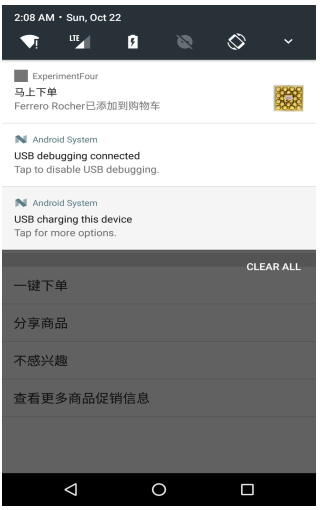
(1)在启动应用时，会有通知产生，随机推荐一个商品:



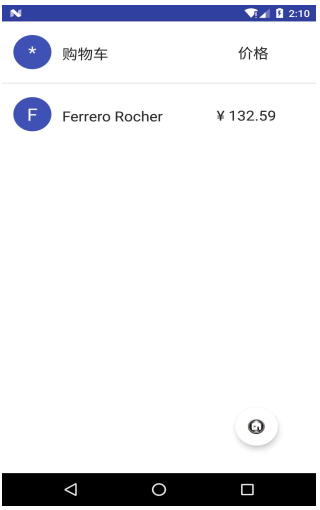
(2)点击通知跳转到该商品详情界面:



(3)点击购物车图标，会有对应通知产生，并通过Eventbus在购物车列表更新数据:



(4)点击通知返回购物车列表:



(5)实现方式要求:启动页面的通知由静态广播产生，点击购物车图标的通知由动态广播产生。

# 课堂实验结果

1. 实验截图

1. 实验步骤以及关键代码

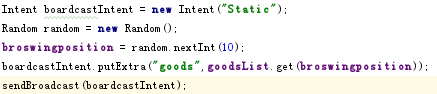
**1、静态广播的实现**

在APP开始运行时，要发送一条静态广播，随机挑选一个商品，然后向广播发送这条商品的信息。点击通知后，进入该商品的详情界面。

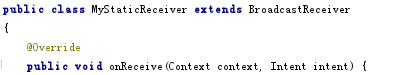
首先，注册静态广播，代码如下，位于AndroidManifest文件中。这里，MyReceiver是自定义的接收器，Static是定义的广播的action，也可以理解为广播名称，应当是一个惟一的标识。



然后是广播的发送。我们先声明一个Intent，并将广播的名称传入，接着使用随机函数产生一个0-9的随机数，然后根据随机数从商品列表中挑选一个商品放入Intent，接着调用sendBroadcast函数发送Intent至广播。

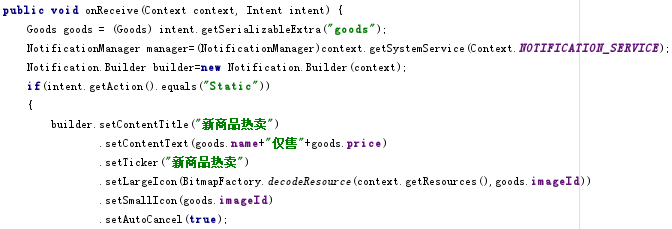


接下来是自定义接收器的部分。自定义接收器MyReceiver继承自父类BroadcastReceiver。然后重载onReceive函数。

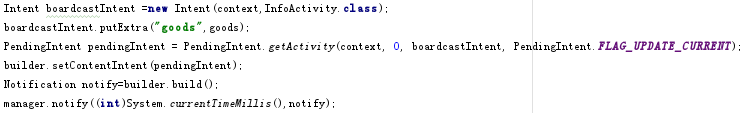


onReceive函数如下。我们首先接受intent中的数据。然后是创建了并从系统获取了一个NotificationManager类。NotificationManager是通知管理类，负责管理Notification。接下来使用Notification.Builer以建造者模式构建 Notification 对象。

然后我们对这个对象设置相关属性。setContentTitle设置通知的标题，setContentText是这时通知的内容。setTicker是通知刚弹出来的时候显示的内容。setLargeIcon是显示大图标，就是下拉通知栏后的那个图标。这里的参数是一个位图资源。我们使用BitmapFactory.decodeResource函数将图片转换为位图。setSmallIcon是设置小图标，由于国内第三方系统的原因，小图标默认显示为应用图标，这个属性失效。setAutoCancel是点击后是否取消。



然后声明一个Intent，写入我们点击后要跳转的Acitivity。向这个Intent中传入数据。然后使用PendingIntent和getActivity从系统取得一个用于启动一个Activity的PendingIntent对象。PendingIntent不是立刻执行的，而是满足某些条件后执行，例如我们要用的Notification。然后将这个PendingIntent装入build。接下来使用build生成一个Notification对象。并为其绑定id。这里的id使用了系统时间，保证了每个的不同，这样才能发出多个广播。



这样就发送了静态广播并在点击通知后，跳转到商品详情界面。

**2、动态广播的实现**

动态广播和静态广播的不同是，静态广播是在AndroidManifest文件中事先注册好，而动态广播是用时注册，用完销毁。

我们在商品详情界面中发送了动态广播，所以在这个Activity中使用类MyDynamicReceiver声明了动态接收器的对象，然后声明一个动态广播。接着添加Action。最后使用registerReceiver函数进行注册。

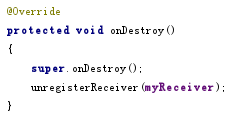




发送时出来Action的不同其它都一样。



最后的时候要将这个动态广播注销掉。



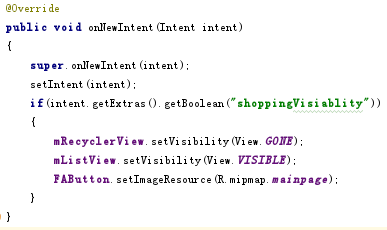
动态广播的自定义接收器MyDynamicReceiver和静态广播的基本一样，仅仅展示的内容有所不同，这里不做详细解释。点击通知后，要返回MainAcitivity且显示购物车列表，这里传回了一个布尔变量。



为了不新建另一个MainAcitivity，而是在原来MainAcitivity的基础上继续运行，我们在注册文件中修改launchMode为singleInstance。这样系统会为MainAcitivity单独维护 一个运行栈，当跳转回来的时候就不会新建而是继续运行



为了能接受数据，我们要重载onNewIntent函数。onNewIntent函数是跳转回一个Activity时运行的。我们在这里接受了刚刚通知中的数据，然后更新列表。setIntent函数重新设置了intent。

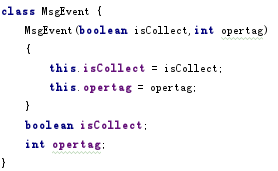


**3、使用EventBus传递数据。**

使用EventBus时要先添加如下依赖。



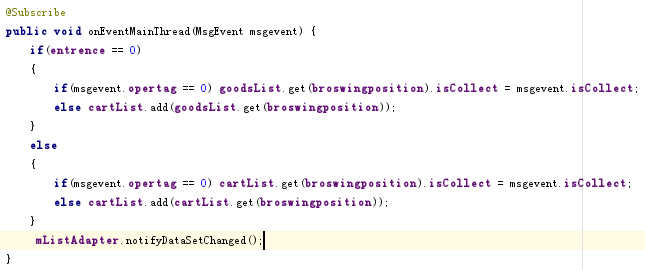
然后要声明一个事件类，在EventBus中使用这个事件类传递数据。这个事件类中，isCollect是商品收藏状态，opertag是一个标识码，用来标识是添加购物车操作还是收藏操作。



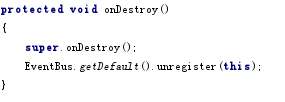
然后使用如下语句，在接受信息的Activity中注册一个EventBus。



然后是实现订阅方法，这里重载了onEventMainThread函数，这一部分也是在接受信息的Activity里。这个函数会在每次使用EventBus发送数据中调用。这个如下函数中的内容是修改购物车列表及商品收藏状态。上一次使用intent实现的已经被删掉了。



在最后还要注销掉这个EventBus



在发送数据的Activity中，我们只需要使用EventBus.getDefault().post函数将之前实现的事件类压入传送数据就可以。

1. 实验遇到困难以及解决思路

1、点击通知回到MainAcitivity时，无法接收信息并调出购物车列表。

这个主要是没有正确理解MainAcitivity的运行过程。我们将MainAcitivity的launchMode修改为singleInstance后，系统会为其单独维护一个运行栈。每次要跳转到这个Activity时，系统会从这个栈中拿出这个Activity重新运行而不会新建。

如果不是新建的话，跳转时就不会运行onCreat函数。所以我们需要重载onNewIntent函数来接收数据并进行处理。

2、多次点击购物车时，只发送一条广播。

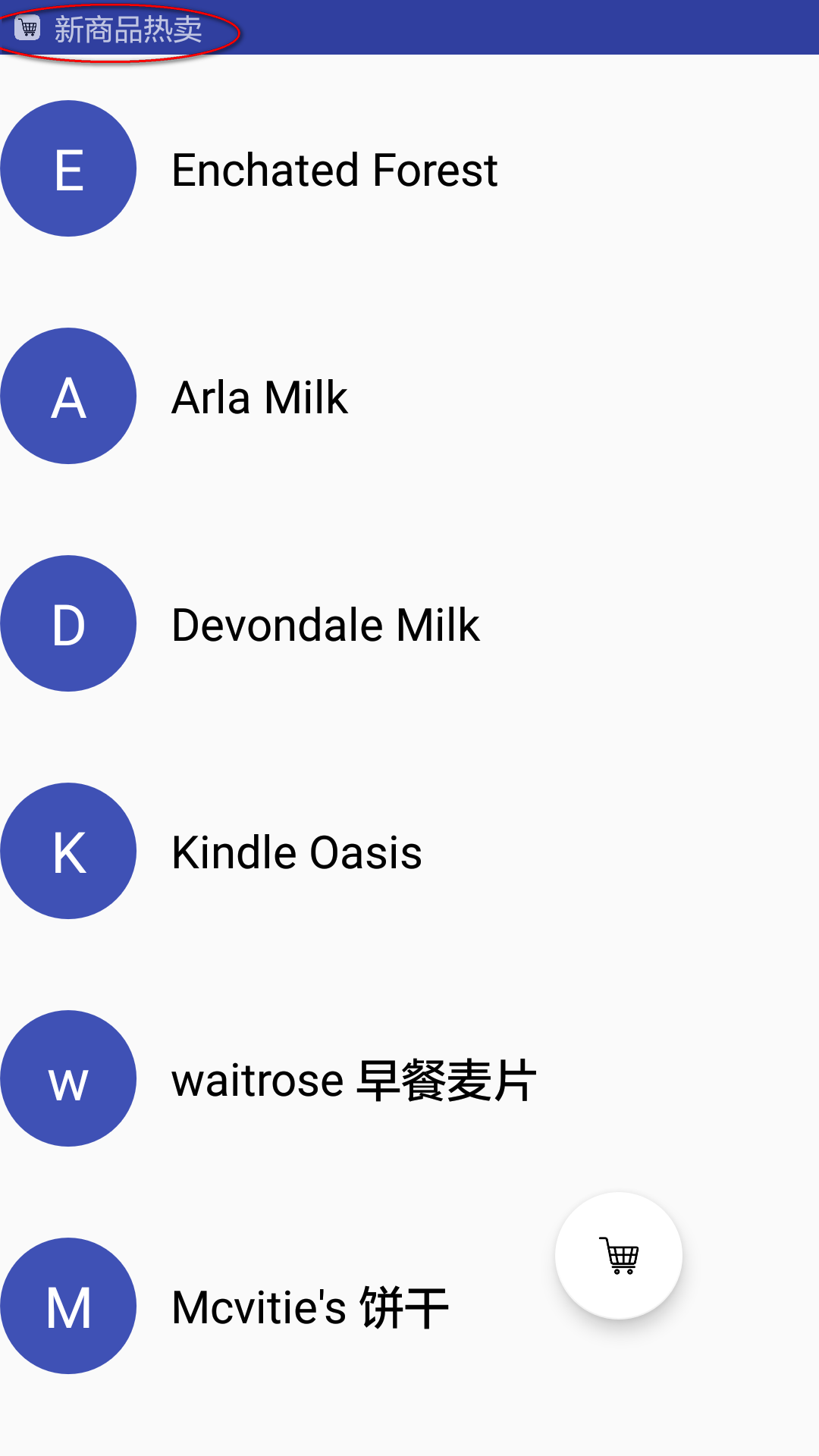
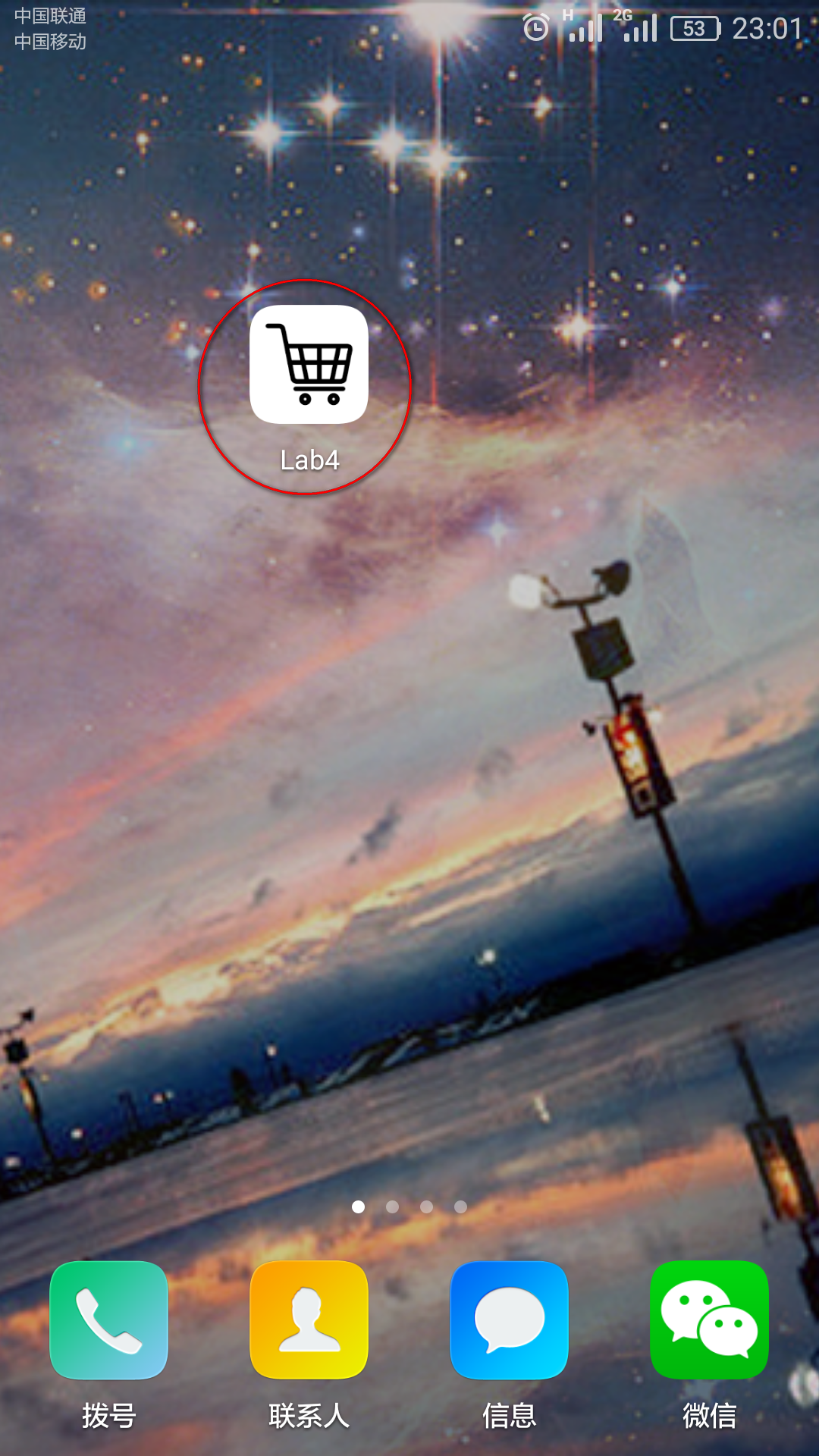
这个主要是在为Notification设置ID时使用了固定id这样每次发送广播就会将前一个Notification覆盖，所以就始终只有一条通知，最后使用系统时间作为id。

# 课后实验结果

由于国内部分安卓系统小图标默认显示应用图标，所以，为了好看，换了一个应用图标，将其换为购物车那个图片。在注册文件中修改如下语句即可更换应用图标。



更换后效果如下。

## 实验思考及感想

这次的实验比上次的真的要简单的多，遇到的问题也少。通过这次的实验我学习了动态广播和静态广播的使用。然后学习了EventBus这个更为有效的传送数据的容器。另外，基于launchMode的原因，了解了一下不同launchMode的Activity的运行栈的切换情况。

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。