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	2017.10.27	完成日期	2017.10.28

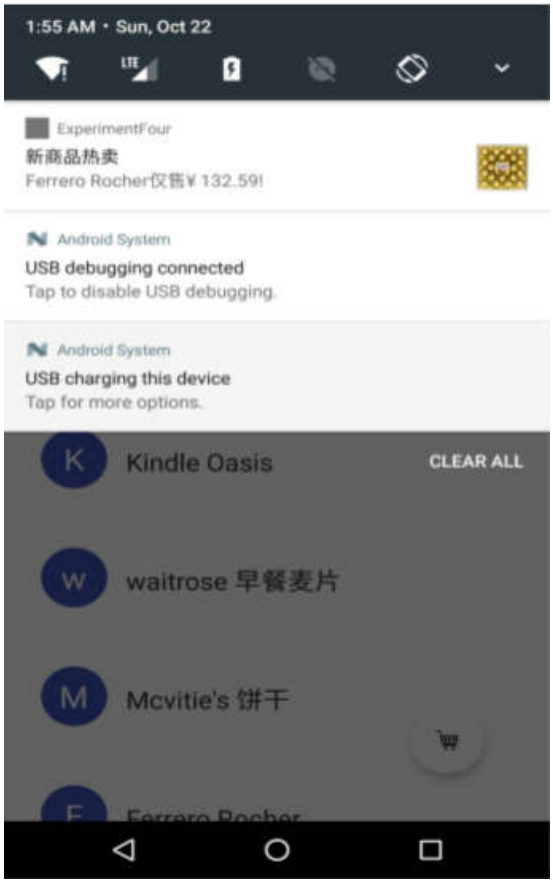
一、 实验题目

Broadcast 使用

二、 实现内容

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：

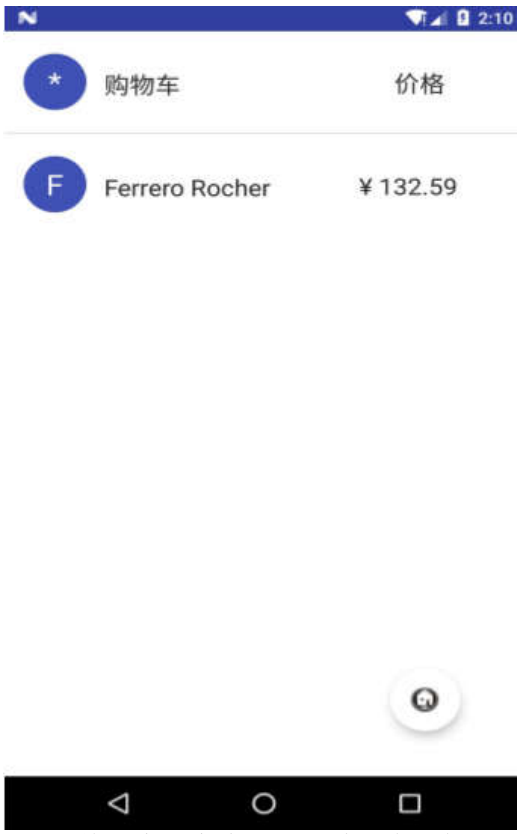
(1)在启动应用时，会有通知产生，随机推荐一个商品:



(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:

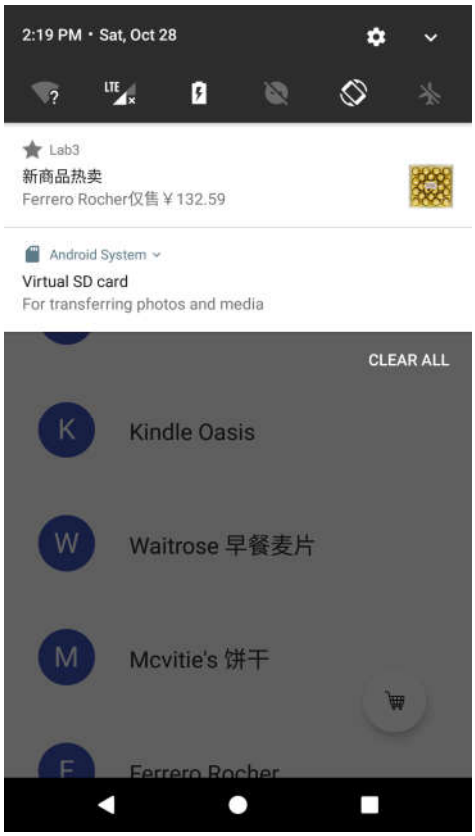
(4)点击通知返回购物车列表:

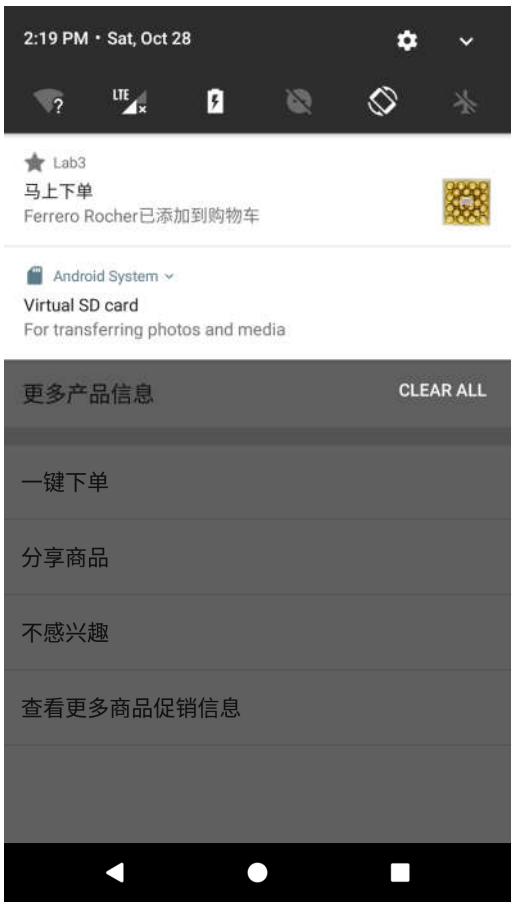


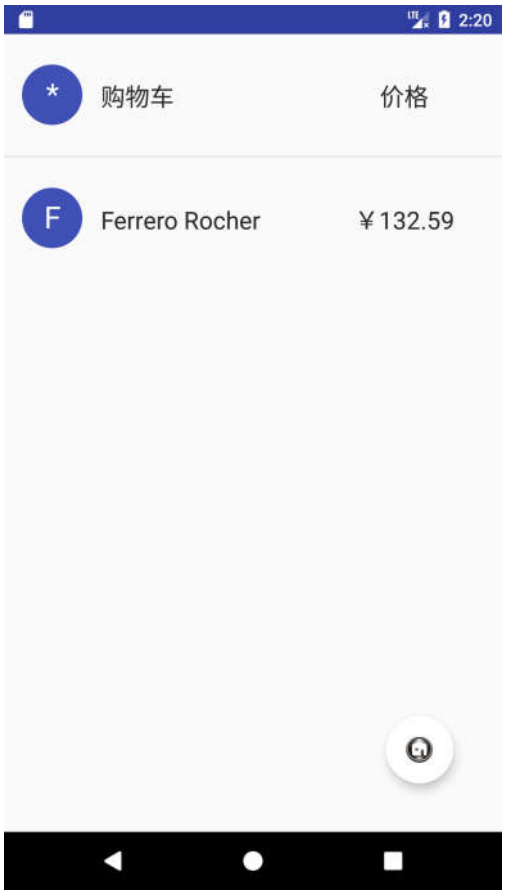
(5)实现方式要求:启动页面的通知由静态广播产生，点击购物车图标的通知由动态广播产生。

三、 课堂实验结果

（1） 实验截图







（2） 实验步骤以及关键代码

步骤 1：静态广播设置。

1）AndroidMainfest.xml 中注册 receiver 类，以及定义广播名称

```
<receiver android:name=".StaticReceiver">
    <intent-filter>
        <action android:name="com.example.a15945.lab3.staticReceiver"></action>
    </intent-filter>
</receiver>
```

2）新建静态广播类 StaticReceiver，重写 onReceive 方法

```java
public class StaticReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        if(intent.getAction().equals(STATICACTION)){
            Bundle bundle = intent.getExtras();
            Bitmap bm = BitmapFactory.decodeResource(context.getResources(), bundle.getInt("imgID"));

            Intent intent1 = new Intent(context,goods.class);
            intent1.putExtra("name",bundle.getString("name"));
            PendingIntent pendingIntent = PendingIntent.getActivity(context,0,intent1,PendingIntent.FLAG_UPDATE_CURRENT);

            //Notification部分
            Notification.Builder builder = new Notification.Builder(context);
            builder.setContentTitle("新商品热卖")
                    .setContentText(bundle.getString("name")+"仅售"+bundle.getString("price"))
                    .setTicker("您有一条新消息")
                    .setWhen(System.currentTimeMillis())
                    .setLargeIcon(bm)
                    .setSmallIcon(R.mipmap.full_star)
                    .setAutoCancel(true)
                    .setContentIntent(pendingIntent);
            //获取状态栏管理
            NotificationManager manager = (NotificationManager)context.getSystemService(Context.NOTIFICATION_SERVICE);
            Notification notification = builder.build();
            //绑定Notification，发送通知请求
            manager.notify(0,notification);
        }
    }
}
```

点击通知栏广播把 name 信息发送到详情页面 activity，而后生成对应的页面。

步骤 2：动态广播设置。

1）在购物车点击事件中注册动态广播，以及广播发送

```java
//注册动态广播
IntentFilter dynamic_Filter = new IntentFilter();
dynamic_Filter.addAction(DYNAMICACTION);
DynamicReceiver dynamicReceiver = new DynamicReceiver();
registerReceiver(dynamicReceiver,dynamic_Filter);
//unregisterReceiver(dynamicReceiver);

//发送动播
Intent intent = new Intent();
intent.setAction(DYNAMICACTION);
Bundle bundle = new Bundle();
bundle.putString("name",str);
bundle.putInt("imgID",data_img_id[pos]);
intent.putExtras(bundle);
sendBroadcast(intent);
```

2）新建静态广播类 StaticReceiver，重写 onReceive 方法

```
public class DynamicReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        if(intent.getAction().equals(DYNAMICACTION)){
            Bundle bundle = intent.getExtras();
            Bitmap bm  = BitmapFactory.decodeResource(context.getResources(),
                    bundle.getInt("imgID"));
            //Notification部分
            Notification.Builder builder = new Notification.Builder(context);
            builder.setContentTitle("马上下单")
                    .setContentText(bundle.getString("name")+"已添加到购物车")
                    .setTicker("您有一条新消息")
                    .setLargeIcon(bm)
                    .setSmallIcon(R.mipmap.full_star)
                    .setAutoCancel(true)
                    //通知栏点击跳转
                    .setContentIntent(PendingIntent.getActivity(context,0,new Intent(context,shoppingcartList.class),0));
            //获取状态栏管理
            NotificationManager manager = (NotificationManager)context.getSystemService(Context.NOTIFICATION_SERVICE);
            Notification notification = builder.build();
            //绑定Notification，发送通知请求
            manager.notify(0,notification);

        }
    }
}
```

点击通知栏回到购物车 activity，点击购物车图标时，购物车列表便已经更新，所以无需传入数据便可实现本实验的全部功能。

步骤 3：EventBus 使用

1） 添加依赖

```
compile 'org.greenrobot:eventbus:3.0.0'
```

2） 声明一个事件类

```
public class Event0 {
    private int message;

    public Event0(int message) { this.message = message; }

    public int getMessage() { return message; }
```

3） 订阅者

```
        EventBus.getDefault().register(this);
    }

    @Subscribe(threadMode = ThreadMode.MAIN)
    public void onEventMain(Event0 event){
        int[] number = new int[]{0};
        number[0] = event.getMessage();
        msg.add(number);
      //  Toast.makeText(getApplication(),""+number[0],Toast.LENGTH_SHORT).show();
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        //反注册
        EventBus.getDefault().unregister(this);
    }
```

4) 传递事件

```
EventBus.getDefault().post(new Event0(pos));
```

（3） 实验遇到困难以及解决思路

1）点击 notification 跳转指定的 activity，一开始不知道这个是 notification 的一个方法，以及在数据传输上不太了解，最后通过查阅资料 get 到相关的知识点解决。

2） EventBus 的使用上不知道遇到什么问题不能在购物车列表和详情页面这两个 activity 中传输数据，使用就会导致购物车列表这个 activity 闪退，而且经过实验 EventBus 在主列表 activity 和购物车列表或者详情页面之间传递数据是没有问题的。未很好地解决。

3）问题（2）的替代方案：在主列表 activity 中注册订阅者，设置全局静态变量，在购物车 activity 中获取该变量，代码：

```
public static List<int[]> msg = new ArrayList<>();
```

```
number[0] = event.getMessage();
msg.add(number);
```

```
import static com.example.a15945.lab3.shoppingList.msg;
```

```
list = getData();
for(int i = 0;i<msg.size();i++){
    list.add(push(msg.get(i)[0]));
}
```

用这种方式实现购物车列表的添加比在 lab3 中使用数据库存取方便了很多，于是本次实验在这里做出了一定的修改，简化了代码。

4） EventBus 的 onEvent 重复多次执行。经过测试，购物车点击事件只执行了一次，点击事情不存在问题，那么问题就是 onEvent 多次执行了，想到这个问题其他人没有遇到，大概是个人使用过多 activity 的又一后果，于是在网上查询 EventBus 的相关特性，以及看了一下 GitHub 源码，发现大概的原因便是在购物车 activity 中发布事件之后，用了 startActivity()方法重新回到主列表 Activity，这样主列表 Activity 就会再次从

onCreate()执行，导致 EventBus 重复注册。EventBus 的注册，意味着将注册者（我这里就是主列表 Activity）中所有的订阅事件保存起来，这种保存是放在一个 HashMap 中的，允许重复元素，多次订阅就导致订阅事件多次保存。而在发布事件之后，在这个 HashMap 中就会找到多个相同的订阅事件，这些订阅事件都会得到执行。解决方法：

```
intent.setClass(shoppingcartList.this,shoppingList.class);
intent.addFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP);//设置不要刷新将要跳到的界面
intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);//它可以关掉所要到的界面中间的activity
startActivity(intent);
```

在每一个跳转中加入上面两句，即可防止多次的 onCreate 重复执行。

5）【未解决】未知问题，点击动态广播通知栏后跳转回到主列表或者购物车列表，点击任何的 item 均是打开上一次点击的 item，返回再打开一次后便回复正常。猜测：详情页面 activity 一旦打开后便一直在后台中，打开后原数据还存在着，不过经过一些测试发现这个猜测似乎并不正确。留个疑问。

四、 实验思考及感想

比较简单的一次实验，主要遇到的问题是 EventBus 的使用，对 startActivity 之后的 Activity 的生命周期过程模糊，对 EventBus 注册的实现方式和原理不清楚，好在在实验过程中不停地查找解决办法从而了解到了许多，不过之后还是应该系统性地学习一番，不然总觉得很虚。

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。