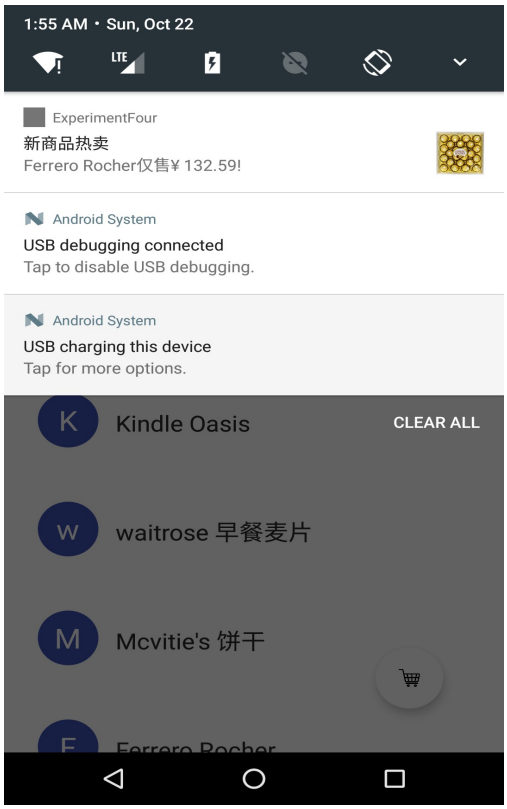
实验四

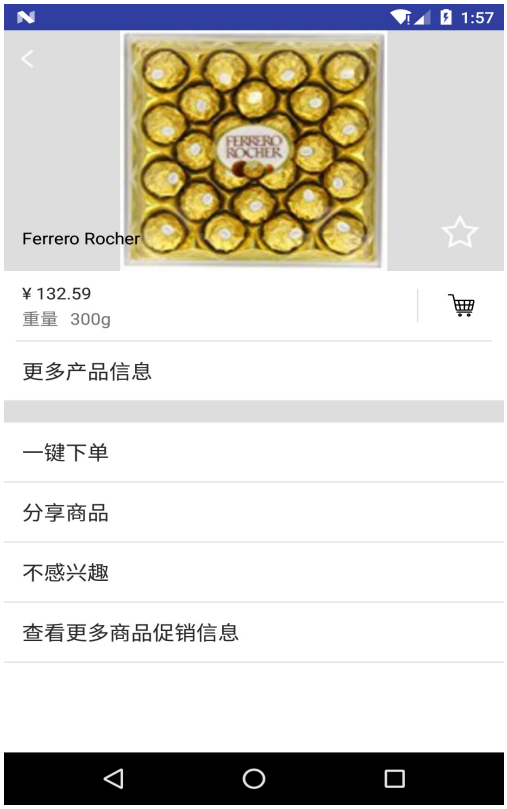
Broadcast 使用

【实验目的】
1、掌握 Broadcast 编程基础
2、掌握动态注册 Broadcast 和静态注册 Broadcast
3、掌握Notification 编程基础
4、掌握 EventBus 编程基础

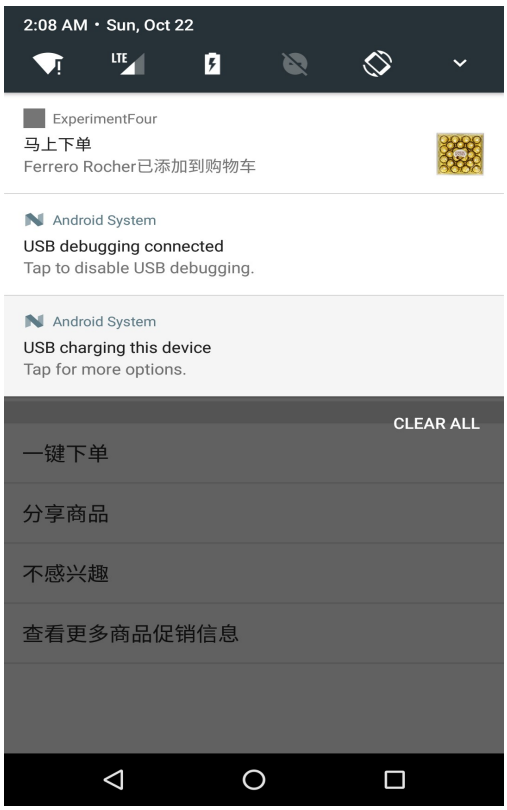
【实验内容】 在实验三的基础上，实现静态广播、动态广播两种改变Notification 内容的方法。具体要求：
(1)在启动应用时，会有通知产生，随机推荐一个商品:



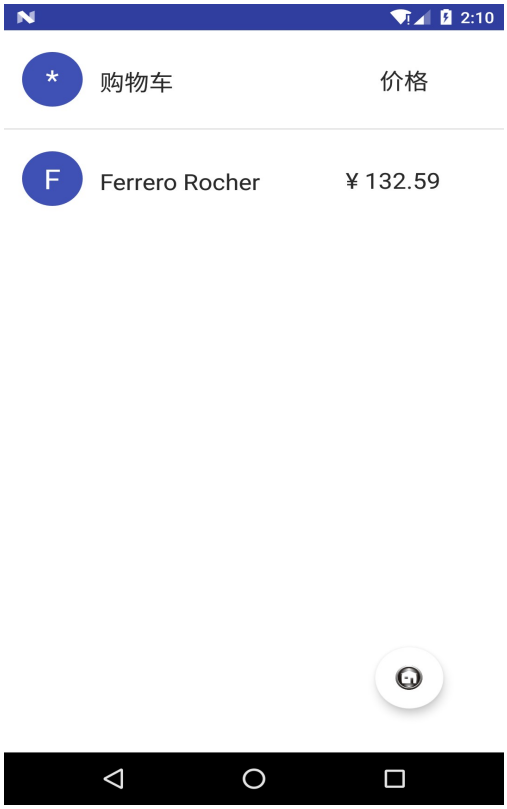
(2)点击通知跳转到该商品详情界面:



(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。

【参考内容】
(1)静态广播部分: 使用随机数:

```
Random random=new Random();
random.nextInt(n);//返回一个0到n-1的整数
```

利用bundle 和intent 将图片与文字内容发送出去:

```
Intent intentBroadcast = new Intent(STATICACTION); // 定义Intent
intentBroadcast.putExtras(bundle);
sendBroadcast(intentBroadcast);
```

参考代码中的STATICATION 为自己设定的广播名称。由于是静态注册所以需要在AndroidMainfest.xml 中进行注册。

```
<receiver android:name=".Receiver.Receiver">
    <intent-filter>
        <action android:name="com.example.ex4.MyStaticFliter" />
    </intent-filter>
</receiver>>
```

在静态广播类StaticReceiver 中重写onReceive 方法，当接收到对应广播时进行数据处理，产生通知。

```
public void onReceive(Context context, Intent intent) {
    if (intent.getAction().equals(STATICACTION)) {    //动作检测
        Bundle bundle=intent.getExtras();
        //TODO:添加Notification部分
    }
}
```

(2)动态广播部分 1.实现 BroadcastReceiver 子类（这里命名为DynamicReceiver），并且重写 onReceive 方法，修改方法与静态广播类中类似。

```
public class DynamicReceiver extends BroadcastReceiver {
    private static final String DYNAMICACTION = "com.example.ex4.MyDynamicFliter";    //动态广播的Action字符串

    public void onReceive(Context context, Intent intent) {
        if (intent.getAction().equals(DYNAMICACTION)) {    //动作检测
            Bundle bundle = intent.getExtras();
            //TODO:添加Notification部分
        }
    }
}
```

2.注册广播关键代码:

```
IntentFilter dynamic_filter = new IntentFilter();
dynamic_filter.addAction(DYNAMICACTION);          //添加动态广播的Action
registerReceiver(dynamicReceiver, dynamic_filter);    // 注册自定义动态广播消息
```

注销广播关键代码:

```
unregisterReceiver(dynamicReceiver);
```

其中dynamicReceiver 为我们之前创建的DynamicReceiver 类。用registerReceiver与 unregisterReceiver 分别对其进行注册与注销。

3.发送方法与静态注册时一直，仅需修改广播名称即可。（使用sendBroadcast(intent)）

4.注意在 Android 主界面中将 launchMode 设置为 singleInstance，使得点击Notification 后不会另外新建一个购物车列表：

```
<activity android:name=".Activity.GoodsList"
    android:launchMode="singleInstance" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

(3)Notification 的使用 Notification 可以提供持久的通知，位于手机最上层的状态通知栏中。用手指按下状态栏，并从手机上方向下滑动，就可以打开状态栏查看提示消息。开发Notification 主要涉及以下3个类：

1.Notification.Builder：用于动态的设置Notification 的一些属性。

```
Notification.Builder builder = new Notification.Builder(context);
//对Builder进行配置，此处仅选取了几个
builder.setContentTitle("动态广播")    //设置通知栏标题:发件人
        .setContentText(bundle.getString("name"))        //设置通知栏显示内容：短信内容
        .setTicker("您有一条新消息")        //通知首次出现在通知栏，带上升动画效果的
        .setLargeIcon(bm)                  //设置通知大ICON
        .setSmallIcon(R.mipmap.dynamic)          //设置通知小ICON(通知栏)
        .setAutoCancel(true);                //设置这个标志当用户单击面板就可以让通知将自动取消
```

思考：大ICON 如何设置？bm 是什么？

2.NotificationManager：负责将Notification 在状态显示出来和取消；

```
//获取状态通知栏管理
NotificationManager manager = (NotificationManager) context.getSystemService(Context.NOTIFICATION_SERVICE);
```

3.Notification：设置Notification 的相关属性。

```
//绑定Notification，发送通知请求
Notification notify = builder.build();
manager.notify(0, notify);
```

4.点击notification，就可以跳转到我们intent 中指定的activity。主要使用到setContentIntent 与PendingIntent。

关于Notification，不同版本的API 显示可能会有所不同。本次实验中必须实现的部分是**标题、大图标、内容、小图标**。其中标题为新商品热卖或马上下单；大图标与广播发送的内容相关，为对应商品图片；内容为"商品名"+仅售+"商品价格"!或"商品名"+已添加到购物车；小图标与大图标内容一样。

图片的使用方面请尽量使用mipmap 目录下的image asset。否则在某些API 中可能会出现Icon 过大的情况。

(4)Eventbus的使用
Eventbus可以简化组件之间的沟通。
1.添加依赖:

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

2.声明一个事件类(传递商品信息):

```
public static class MessageEvent { /* Additional fields if needed */ }
```

3.准备订阅者：声明并注释您的订阅方法，可选地指定线程模式(在购物车所在Activity声明这个方法)：

```
@Subscribe(threadMode = ThreadMode.MAIN)
public void onMessageEvent(MessageEvent event) {/* Do something */};
```

注册订阅者(注册购物车所在Activity为订阅者):

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

注销订阅者(退出时要注销订阅者):
EventBus.getDefault().unregister(this);

4.传递事件(点击购物车图标时候，传递商品信息):

```
EventBus.getDefault().post(new MessageEvent());
```

【检查内容】

1、静态广播：启动应用是否有随机推荐商品的通知产生。点击通知是否正确跳转到商品详情界面。

2、动态广播：点击购物车后是否有提示商品已加入购物车的通知产生。同时注意设置launchMode。点击通知是否跳转到购物车列表。

3、Eventbus:点击购物车图标是否正确添加商品到购物车。每点击一次添加一件该商品到购物车。

【提交说明】

1、 deadline：下一次实验课前一天晚上12点

2、 提交作业地址：ftp://edin.sysu.edu.cn

3、 文件命名及格式要求：学号*姓名*labX.zip（姓名中文拼音均可）

4、 目录结构：

```
15331111_huashen_lab1 --
                       |
                       --  lab1实验报告.pdf
                       |
                       --  lab1_code（包含项目代码文件）
```

其中项目代码文件为项目文件夹，*提交之前先 clean