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	1516	专业（方向）	软件工程（移动信息工程）
学号	15352354	姓名	肖想
电话	13247651998	Email	1368994233@qq.com
开始日期	10.22	完成日期	10.28

一、 实验题目

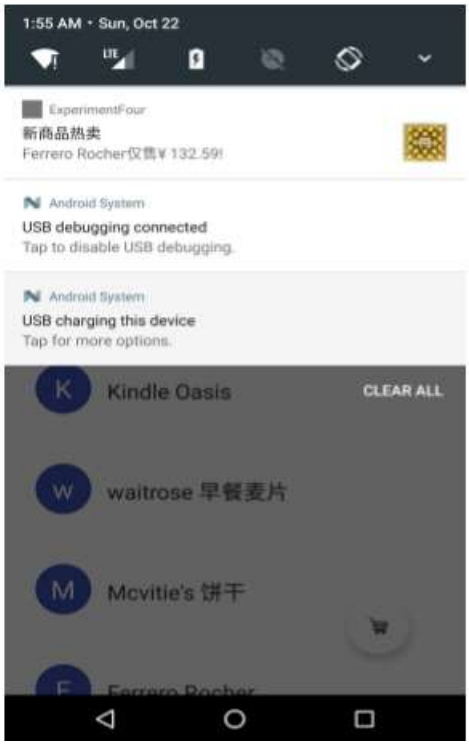
Broadcast 使用

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：

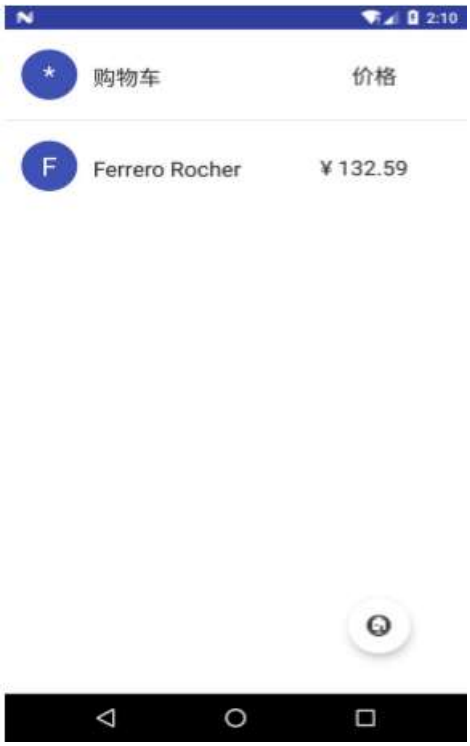
(1)在启动应用时，会有通知产生，随机推荐一个商品:



(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

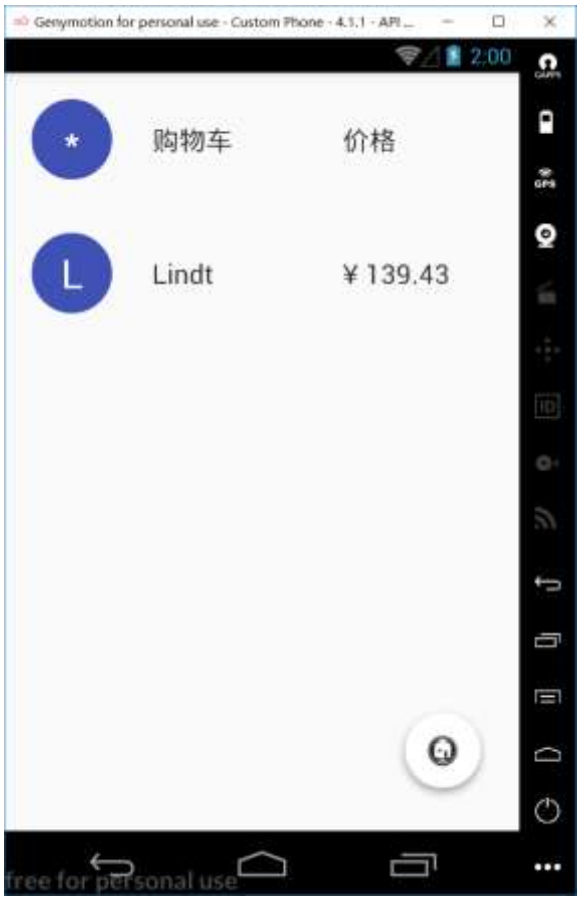
（1） 实验截图

① 启动应用，显示通知：

② 点击通知，跳转页面：

③点击加入购物车，弹出通知：

④点击通知，进入购物车界面：



（2） 实验步骤以及关键代码

静态广播：

①在 AndroidManifest 中注册，达到静态效果。

```
<receiver android:name=".Receiver">
    <intent-filter>
        <action android:name="StartByStatic"/>
    </intent-filter>
</receiver>
```

②具体的 onReceive 实现。

```
public class Receiver extends BroadcastReceiver {
    public NotificationManager nm;
    @Override
    public void onReceive(Context context, Intent intent) {
        MainActivity m = new MainActivity();

        nm = (NotificationManager) context.getSystemService(NOTIFICATION_SERVICE);

        NotificationCompat.Builder nBuilder = new NotificationCompat.Builder(context);

        int rd = new Random().nextInt(10);
        PendingIntent pendingintent = PendingIntent.getActivity(context,0
                ,new Intent(context,ItemDetail.class).putExtra("id",rd)
                ,PendingIntent.FLAG_UPDATE_CURRENT);
        nBuilder.setContentIntent(pendingintent).setWhen(System.currentTimeMillis())
                .setPriority(Notification.PRIORITY_DEFAULT).setAutoCancel(true);
        nBuilder.setTicker("lab4").setContentTitle("
                .setContentText(String.format("%s%s!",m.getbyid(rd).getItem_name(),m.getbyid(rd).getPrice()))
                .setSmallIcon(R.drawable.shoplist);

        Notification notification = nBuilder.build();
        nm.notify(R.layout.activity_main,notification);
    }
}
```

③在应用生成的 oncreate 方法里面发送广播。

```
Intent intent = new Intent("StartByStatic");
sendBroadcast(intent);
```

动态广播：

①通过在 oncreate 中注册，在 ondestroy 中销毁来实现动态的注册。

```
private Receiver2 rc = new Receiver2();
```

oncreate 中：

```
IntentFilter filter = new IntentFilter();
filter.addAction("StartByDynamic");
filter.setPriority(IntentFilter.SYSTEM_HIGH_PRIORITY);
registerReceiver(rc,filter);
```

ondestroy 中：

```
unregisterReceiver(rc);
```

②具体的 onReceive 实现。

```
public class Receiver2 extends BroadcastReceiver {
    public NotificationManager nm;
    private int times = 0;
    @Override
    public void onReceive(Context context, Intent intent) {
        MainActivity m = new MainActivity();

        nm = (NotificationManager) context.getSystemService(NOTIFICATION_SERVICE);
        NotificationCompat.Builder nBuilder = new NotificationCompat.Builder(context);
        PendingIntent pendingintent = PendingIntent.getBroadcast(context,0,new Intent(context,Receiver3.class)
                ,PendingIntent.FLAG_UPDATE_CURRENT);

        nBuilder.setContentIntent(pendingintent).setWhen(System.currentTimeMillis())
                .setPriority(Notification.PRIORITY_DEFAULT).setAutoCancel(true);
        nBuilder.setTicker("lab4").setContentTitle("[illegible]")
                .setContentText(String.format("%s[illegible]",m.getbyid(intent.getIntExtra("id",0)).getItem_name()))
                .setSmallIcon(R.drawable.shoplist);

        Notification notification = nBuilder.build();
        nm.notify(times,notification);
        times ++;
    }
}
```

因为我在这里需要判断该通知是否被点击了来设置页面，由于通知是系统服务，无法直接得到 click 操作，因此在这里使用了间接的方法来判断通知是否被点击。

Receive3 采用静态注册，具体的 onReceive 实现为：

```
public class Receiver3 extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        EventBus.getDefault().post(new MessageEvent(0,-1,false));

        Intent newIntent = new Intent(context, MainActivity.class).addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        context.startActivity(newIntent);
    }
}
```

③在需要的地方发送广播。

```
EventBus.getDefault().post(new MessageEvent(getId,1,true));
Intent intent = new Intent("StartByDynamic");
intent.putExtra("id",getId);
sendBroadcast(intent);
```

EventBus 对事件的处理：

①定义数据类型（时间类）。

```java
public class MessageEvent {
    private int id;

    private Boolean flag;

    private int times;

    public MessageEvent(int id,int times,Boolean flag) {
        this.id = id;
        this.times = times;
        this.flag = flag;
    }

    public int getId() {
        return id;
    }

    public int getTimes() {
        return times;
    }

    public Boolean getFlag() {
        return flag;
    }

}
```

②定义实现方法。

```java
@Subscribe(threadMode = ThreadMode.MAIN)
public void onShowMessageEvent(MessageEvent messageEvent) {
    if(!messageEvent.getFlag() && flag)
    {
        flag = messageEvent.getFlag();
        initial();
    }
    if(messageEvent.getTimes() == -1);
    else if(messageEvent.getTimes() == -2) Data[messageEvent.getId()].setFlag(!Data[messageEvent.getId()].getFlag());
    else
        for(int i = 0;i < messageEvent.getTimes();i ++)
            AddMyData(messageEvent.getId());
}
```

（若直接返回进入主界面，则返回原来的界面，不修改页面，若通过通知进入主界面，则不论原来界面是什么，将页面修改为显示购物车的页面；若 times 为-1，则不处理，若 times 为-2，则将星星图标的状态改变：空变实，实变空，其它则根据物品的 id 在购物车的数据里面增加该物品）

③在需要的地方调用。

a.通知被点击，修改页面为购物车页面：

```java
EventBus.getDefault().post(new MessageEvent(0,-1,false));
```

b.点击星星图标时，在主界面的数组中修改星星的属性值。

```java
EventBus.getDefault().post(new MessageEvent(getId,-2,true));
```

c.点击购物车，更新购物车列表数据。

```java
EventBus.getDefault().post(new MessageEvent(getId,1,true));
```

（3） 实验遇到困难以及解决思路

最主要的问题在于，最初实现的时候并未考虑到从商品详情界面直接返回到主界面去的时候，原来是什么界面就应该显示为什么界面，而不是一味地显示成购物车界面，因此就需要通过考虑到通知是否被点击来更改主界面的显示。但是因为通知是系统服务，不能为其的点击设计自定义事件，因此需要通过间接的方法来判断通知是否被点击。

四、 课后实验结果

五、 实验思考及感想

最初完成的时候并未注意到很多细节问题，只是单纯的在原实验的基础上增加了广播，通过通知进行的页面跳转以及用 EventBus 来实现为购物车列表增加数据，仔细查看要求后，发现原来通过 StartActivityForResult 传输的数据都应该修改为用 EventBus 传输，同时返回主界面后的显示也应该根据实际情况和准确的逻辑来判断。后来为了这些细节也改了挺久的，特别是思考怎样根据实际情况和准确的逻辑来显示主界面。