

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	15M1	专业（方向）	移动互联网
学号	15352008	姓名	蔡荣裕
电话	13727021990	Email	897389207@qq.com
开始日期	2017/10/24	完成日期	2017/10/30

一、 实验题目

Broadcast 使用

二、 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

三、 实验内容

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，如图 1
2. 点击通知跳转到该商品详情界面，如图 2
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，如图 3

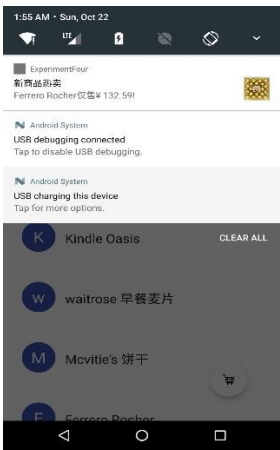


图 1



图 2



图 3

- 4. 点击通知返回购物车列表如图 4
- 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。

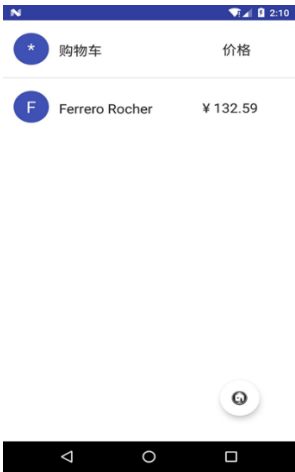
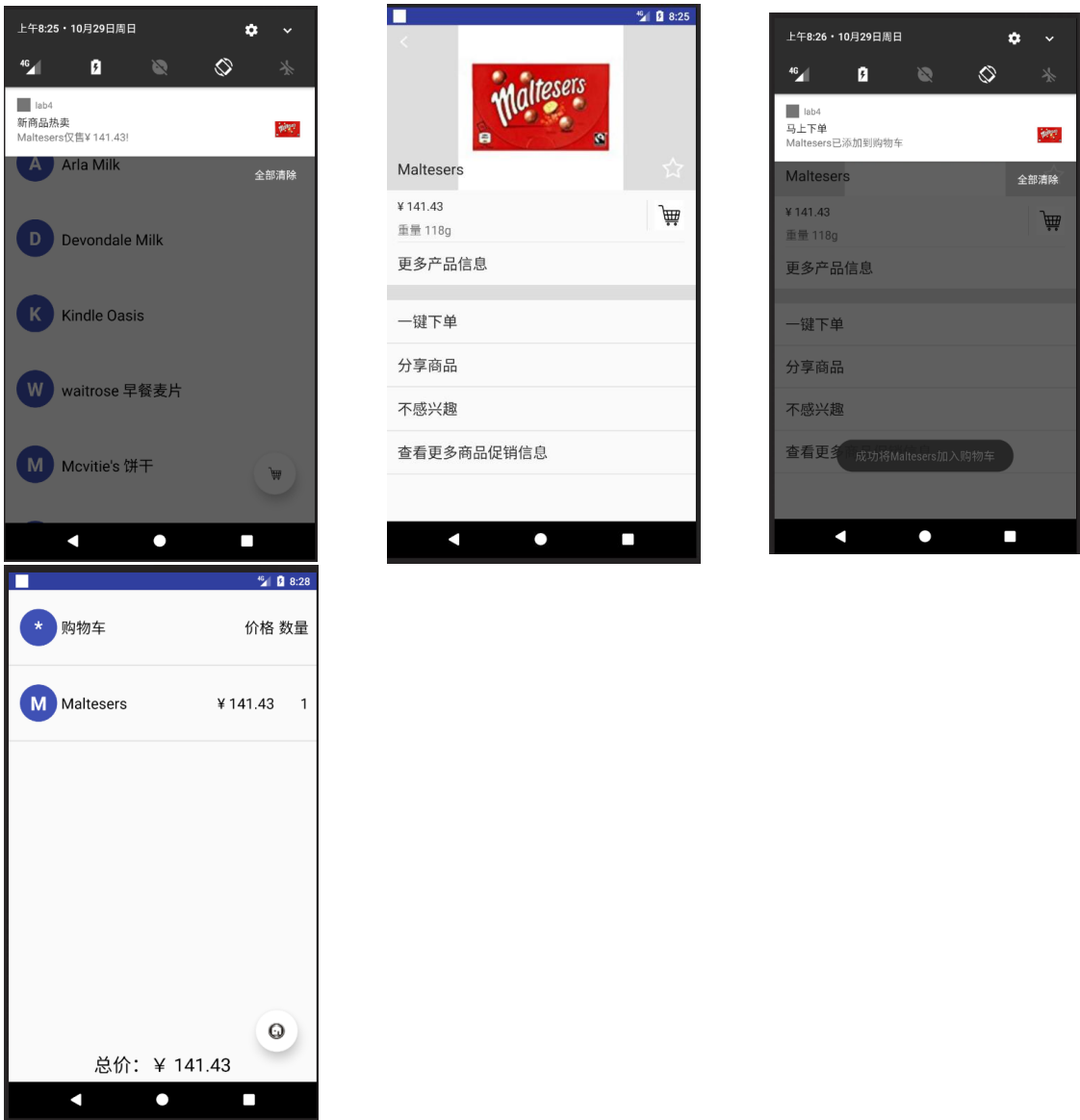


图 4

四、 课堂实验结果

(一) 实验截图：

1. 虚拟机上效果如下：



(二) 关键步骤:

1. 在启动应用时, 会有通知产生, 随机推荐一个商品
 - i. 首先是需要产生一个随机数, 然后将这个随机数作为参数发出去, 因为已经前一个实验已经实现数据类。

```
private static final String STATICACTION = "cn.chonor.lab4.staticreceiver";

private void broad_init() {
    Random random=new Random();
    int i=random.nextInt(10);
    Intent iBroad=new Intent(STATICACTION);
    Bundle bundle = new Bundle();
    bundle.putInt("position", i);
    iBroad.putExtra("mainActivity",bundle);
    sendBroadcast(iBroad);
}
```

STATICACTION 为自己设定的广播名称。由于是静态注册所以需要在 Manifest.xml 中进行注册, 同时为了防止重复生成 Main 这个页面还需要修改 launchMode (其他两个界面同时修改)。

```
<activity android:name=".MainActivity" android:launchMode="singleTask">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

<receiver android:name=".MainActivity$StaticReceiver" android:enabled="true">
    <intent-filter>
        <action android:name="cn.chonor.lab4.staticreceiver" />
    </intent-filter>
</receiver>
```

此处静态广播就注册了, 因为我们把广播写到 MainActivity 所以此处需要用.MainActivity\$StaticReceiver

- ii. 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。

此处要用 Notification.Builder, 动态设置 Notification。

```
public static class StaticReceiver extends BroadcastReceiver {
    public StaticReceiver() {}
    @Override //静态广播接收器执行的方法
    public void onReceive(Context context, Intent intent) {
        if(intent.getAction().equals(STATICACTION)) {
            Bundle extras = intent.getBundleExtra("mainActivity");
            Data data=new Data();
            int id=extras.getInt("position"); //获得数据位置
            Bitmap bm=BitmapFactory.decodeResource(context.getResources(),data.ID[id]);
            String name=data.getGood_list_index(id).getGoodName();
            String price=data.getGood_list_index(id).getGoodPrice();
        }
    }
}
```

```

        NotificationManager notifyManager = (NotificationManager)
context.getSystemService(Context. NOTIFICATION_SERVICE);

        Intent intent1 =new Intent (context,Good_Info.class); //点击事件和传输
        Bundle bundle = new Bundle();
        bundle.putInt("position", id);
        bundle.putParcelableArrayList("data",data.getGood_list());
        bundle.putParcelableArrayList("cart", data.getCart_list());
        intent1.putExtra("mainActivity", bundle);
        PendingIntent pi = PendingIntent.getActivities(context, 0, new
Intent[] {intent1}, PendingIntent.FLAG_CANCEL_CURRENT);

        //实例化 NotificationCompat.Buide 并设置相关属性
        Notification.Builder builder = new Notification.Builder(context)
            //设置小图标
            .setSmallIcon(data.ID[id])
            .setLargeIcon(bm)
            //设置通知标题
            .setContentTitle("新商品热卖")
            //设置通知内容
            .setContentText(name+"仅售"+price+"!")
            .setContentIntent(pi)
            .setAutoCancel(true);
        notifyManager.notify(0, builder.build());
    }
}
}

```

这样我们启动后的推荐就完成了。

2. 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。
 - i. 实现 BroadcastReceiver 子类，并且重写 onReceive 方法，与静态广播类中类似，此处因为需要先打开购物车界面注册，但是一开始没有打开，所以此处广播还是使用 Intent 传参（课后实验中使用粘性 EventBus 解决，不需要 intent 传参）。

```

public class DynamicReceiver extends BroadcastReceiver {
    public static final String DYNAMICATION= "cn.chonor.lab4.dynamicreceiver";
    public DynamicReceiver() {}
    @Override
    public void onReceive(Context context, Intent intent) {
        if(intent.getAction().equals(DYNAMICATION)) {
            NotificationManager notifyManager = (NotificationManager)
context.getSystemService(Context. NOTIFICATION_SERVICE);

            Intent intent1 =new Intent (context,Shoppingcart.class); //点击事件和传输
            Bundle bundle = new Bundle();
            bundle.putParcelableArrayList("data", data.getGood_list());
            bundle.putParcelableArrayList("cart", data.getCart_list());
            intent1.putExtra("mainActivity", bundle);
            PendingIntent pi = PendingIntent.getActivities(context, 0, new
Intent[] {intent1}, PendingIntent.FLAG_CANCEL_CURRENT);

```

```

        //实例化 NotificationCompat.Builder 并设置相关属性
        Notification.Builder builder = new Notification.Builder(context)
            //设置小图标
            .setSmallIcon(data.ID[id])
            .setLargeIcon(BitmapFactory.decodeResource(context.getResources(), data.ID[data.map.get(names)]))
            //设置通知标题
            .setContentTitle("马上下单")
            //设置通知内容
            .setContentText(names+"已添加到购物车")
            .setContentIntent(pi)
            .setAutoCancel(true);
        notifyManager.notify(0, builder.build());
    }
}
}

```

进行动态广播注册，我们只需要在 onCreate 的时候调用一下

```

public static final String DYNAMICATION= "cn.chonor.lab4.dynamicreceiver";
private DynamicReceiver dynamicReceiver=new DynamicReceiver();
private IntentFilter dynamic_filter =new IntentFilter();

private void broad_init(){
    dynamic_filter.addAction(DYNAMICATION);
    registerReceiver(dynamicReceiver,dynamic_filter);
}

```

之后我们在 onDestroy()进行销毁

```

@Override
protected void onDestroy() {
    super.onDestroy();
    unregisterReceiver(dynamicReceiver);
}

```

ii. 然后我们只要在，购物车点击事件中增加发送广播，此处还有 EventBus 的传输。

```

cart.setOnClickListener(new View.OnClickListener() { //购物车按钮
    @Override
    public void onClick(View view) {
        if(flag){
            int incart= -1; //确定购物车里有没有
            for(int i=0;i<data.getCart_list().size();i++){
                if(data.getGood_list_index(id).getGoodName().equals(data.getCart_list_index(i).getGoodName()))
                    incart=i;
            }
            if(incart== -1) //没有加入购物车
                data.addCart_list(data.getGood_list_index(id));
            else //有数量+1
                data.setCart_list_Cnt(incart,data.getCart_list_index(incart).getCnt()+1);
            Toast.makeText(Good_Info.this,"成功将"+data.getGood_list_index(id).getGoodName()+"加入购物车",Toast.LENGTH_SHORT).show();
        }
        else {
            data.setCart_list_Cnt(id,data.getCart_list_index(id).getCnt()+1);
        }
    }
}

```

```

        Toast.makeText(Good_Info.this, "成功将"+data.getCart_list_index(id).getGoodName()+"加入购物车", Toast.LENGTH_SHORT).show();
    }
    EventBus.getDefault().post(new MessageEvent(data)); //发送数据
    Intent iBroad=new Intent(DYNAMICATION); //发送广播
    sendBroadcast(iBroad);
}
});

```

- iii. 因为此处我们要用 EventBus 进行数据传输所以我们需要在购物车界面中注册 EventBus，首先在 onCreate 中注册

```

public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.shopping);
    init(); //初始化
    get_set_data(); //接收数据
    init_listener(); //事件监听
    EventBus.getDefault().register(this);
}

```

之后 onDestroy()注册取消。

```

@Override
protected void onDestroy() {
    super.onDestroy();
    //取消注册事件
    EventBus.getDefault().unregister(this);
}

```

- iv. 之后在购物车中写好订阅

```

@Subscribe(threadMode = ThreadMode.MAIN)
public void onMessageEvent(MessageEvent event) {
    data=event.getData();
}

```

- v. 此处我们只需要简单的实现一下事件类，用来传递我们的 Data 类

```

public class MessageEvent {
    private Data data;
    public MessageEvent(Data data) {
        this.data = data;
    }
    public Data getData() {
        return data;
    }
    public void setData(Data data) {
        this.data = data;
    }
}

```

至此我们的动态广播和数据传输就设置好了。

(三) 实验中遇到的困难和解决思路：

1. 一开始无法接受到静态和动态广播，之后发现这个是 8.0 的权限要求问题，换回 7.1.1 即可使用。
2. 在使用 launchMode 为 singleTask 时候，发现不会更新数据，然后百度了一下因为此时不会再次执行 onCreate 函数，但是 onNewIntent 可以解决问题。
3. 使用 EventBus 时候发现这个页面需要先打开注册一次，因为之前实验中使用了 3 页面，所以放弃了全部使用 EventBus 传参，还是留了一部分 Parcelable 接口。但是在课后实验中发现可以直接使用 Sticky EventBus 进行实现，这部分在课后拓展。
4. 对于实验中给出的思考题大 ICON 如何设置？bm 是什么？bm 实际是 bitmap，只需要转化一下就好了。

五、课后实验结果

1. 使用 Sticky EventBus 进行传参，解决了 3 页面时购物车页面没有打开造成 EventBus 没有注册进而收不到消息。相对于之前的代码可以实现购物车按钮和界面的完全同步，只需要修改两个地方，然后我们就可删除之前的 Intent 传递数据部分了
首先是 EventBus 的发送

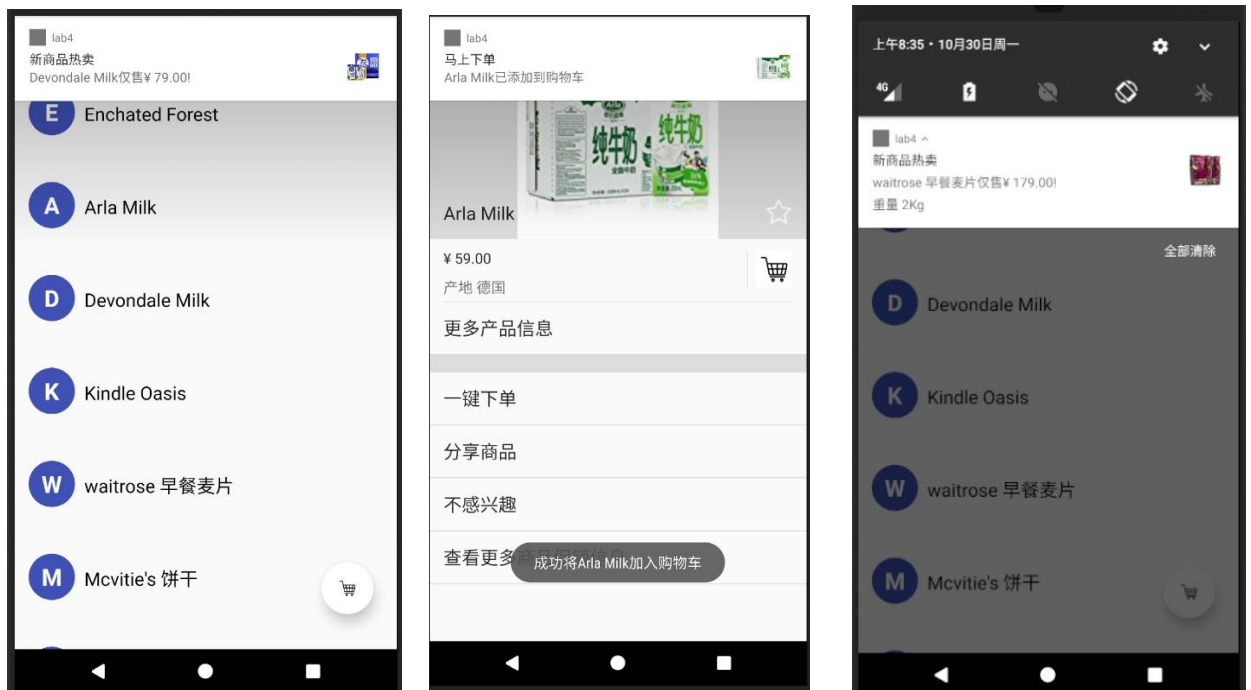
```
EventBus.getDefault().postSticky(new MessageEvent(data)); //发送数据
```

之后是 EventBus 的接收

```
@Subscribe(threadMode = ThreadMode.POSTING, sticky = true)
public void onMessageStickyEvent(MessageEvent event) {
    data=event.getData();
    get_set_data();//重设数据
}
```

2. 优化了一下弹窗模式

如下图，点击这个弹出的通知栏可以直接转跳，并增加了详细信息。



代码修改如下：

```
Notification.InboxStyle inboxStyle = new Notification.InboxStyle()
    .setBigContentTitle("新商品热卖")
    .addLine(name+"仅售"+price+"!")
    .addLine(data.getGood_list_index(id).getGoodTypes() + " " +
data.getGood_list_index(id).getGoodInfo());

Notification.Builder builder = new Notification.Builder(context)
    //设置小图标
    .setSmallIcon(data.ID[id])
    .setLargeIcon(bm)
    //设置通知标题
    .setTicker("TickerText:" + "您有新短消息，请注意查收！")
    .setContentTitle("新商品热卖")
    //设置通知内容
    .setContentText(name+"仅售"+price+"!")
    .setStyle(inboxStyle)
    .setContentIntent(pi)
    .setAutoCancel(true)
    .setDefaults(Notification.DEFAULT_SOUND | Notification.DEFAULT_VIBRATE)
    .setWhen(System.currentTimeMillis())
    .setPriority(Notification.PRIORITY_HIGH)
    .setVisibility(Notification.VISIBILITY_SECRET);
```

六、实验思考及感想

这次实验比较简单只需要在上一个实验的基础上进行修改，不需要太多的处理，广播部分比较简单，就是注册和发送，但是一直搞不懂为什么 8.0 的系统怎么样都接收不到，但是换到 7.1.1 之后正常，在 8.0 的系统中额外增加了获取权限也是没有成功实现。

之后就是 Eventbus 的使用，因为上一个实验中使用了 3 页面的模式，所以造成了普通的 Eventbus，无法再购物车界面未启动的时候传输数据，但是最后使用了粘性 Eventbus 处理了这个问题，使得数据能够正常传输。

最后的部分给整个广播系统增加了一下弹出功能和声音震动，优化一下使用体验。