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

姓名	学号	班级	电话	邮箱
张子豪	15352427	15M1	15989046143	ahzzh1998@163.com

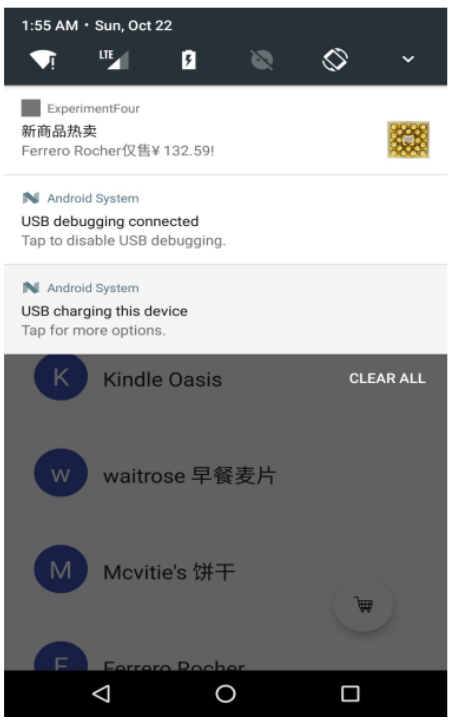
1.实验题目

Broadcast使用

2.实现内容

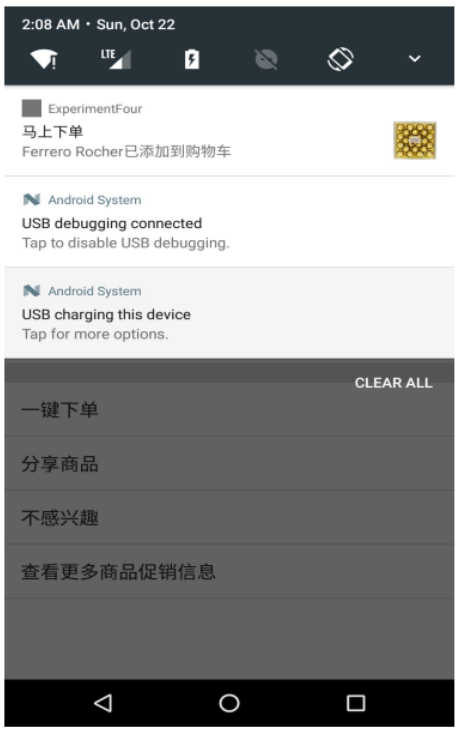
在实验三的基础上，实现实现静态广播、动态广播两种改变Notification内容的方法 。

1. 在启动应用时，会有通知产生，随机推荐一个商品:

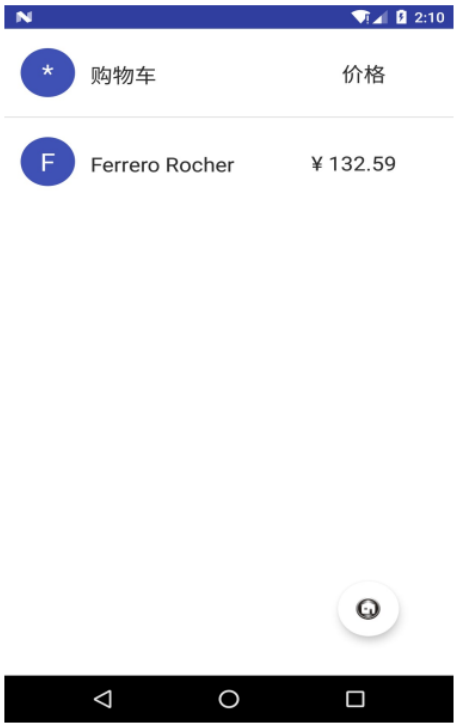


2. 点击通知跳转到该商品详情界面：

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：



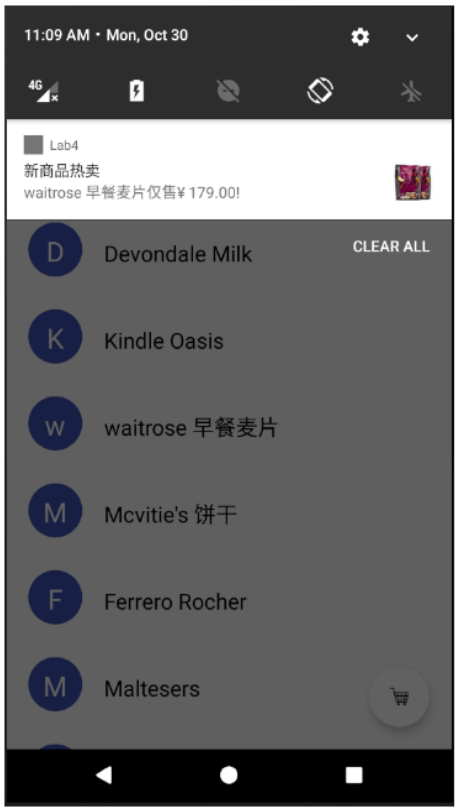
4. 点击通知返回购物车列表：



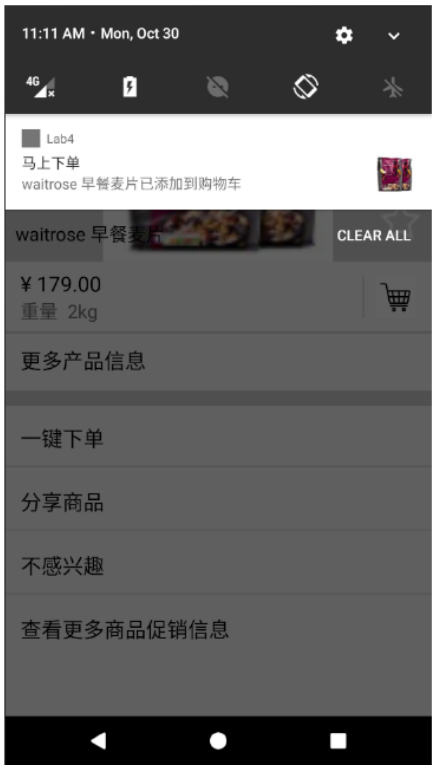
5. 实现方式要求：启动页面的通知由静态广播产生，点击购物车图标的通知由动态广播产生。

3.课堂实验结果

- 实验截图



进入app后由静态广播产生的通知如上



点击通知后进入商品详情，点击购物车添加商品，产生由动态广播产生的通知如上

点击通知后进入购物车，如上

- 实验步骤以及关键代码

  smallIcon与LargeIcon的区别：当setSmallIcon()与setLargeIcon()同时存在时,smallIcon显示在通知的右下角,largeIcon显示在左侧；当只设置setSmallIcon()时,smallIcon显示在左侧。当然这点也取决于手机的系统版本。

  动态广播：

```
Intent intent=new Intent();//在商品详情中发送广播
intent.setAction("action.refreshShopcar");
Bundle bundle=new Bundle();
bundle.putInt("key",pos);
intent.putExtras(bundle);
sendBroadcast(intent);
```

```
IntentFilter intentFilter = new IntentFilter();//注册接收
intentFilter.addAction("action.refreshShopcar");
registerReceiver(mBroadcastReceiver, intentFilter);

unregisterReceiver(mBroadcastReceiver);//onDestroy中取消注册
```

```
private BroadcastReceiver mBroadcastReceiver = new BroadcastReceiver() {//动态广播的接收
@RequiresApi(api = Build.VERSION_CODES.JELLY_BEAN)
@Override
public void onReceive(Context context, Intent intent) {
    String action = intent.getAction();
    if (action.equals("action.refreshShopcar")) {
        Bundle bundle = intent.getExtras();
        int pos = bundle.getInt("key");//获得被点击商品的下标
        Notification.Builder builder=new Notification.Builder(context);
        builder.setContentTitle("马上下单");
        builder.setContentText(name[pos]+"已添加到购物车");
        builder.setTicker("您有一条新消息");
        builder.setSmallIcon(img[pos]);
 builder.setLargeIcon(BitmapFactory.decodeResource(MainActivity.this.getResources(),img[pos
]));
        builder.setAutoCancel(true);
        flag=false;//表示通过点击通知返回MainActivity
        Intent mainIntent=new Intent(context,MainActivity.class);
        PendingIntent
pendingIntent=PendingIntent.getActivity(context,0,mainIntent,PendingIntent.FLAG_UPDATE_CURR
ENT);
        builder.setContentIntent(pendingIntent);
        NotificationManager manager=(NotificationManager)
context.getSystemService(Context.NOTIFICATION_SERVICE);
        Notification notify=builder.build();
        manager.notify(0,notify);
    }
}
};
```

静态广播：

```
<receiver android:name=".BroadcastRecTest">//在AndroidMainifest中注册receiver
    <intent-filter>
        <action android:name="static_broadcast"></action>
    </intent-filter>
</receiver>
```

```
Intent static_intent=new Intent("static_broadcast");//发送静态广播
sendBroadcast(static_intent);
```

```
//自定义的类中重载BroadcastReceiver来接收静态广播
public void onReceive(Context context, Intent intent) {
    String action=intent.getAction();
    if(action.equals("static_broadcast")){
        Random random=new Random();
        int pos=random.nextInt(name.length);//产生随机数
        Notification.Builder builder=new Notification.Builder(context);
        builder.setContentTitle("新商品热卖");
        builder.setContentText(name[pos]+"仅售"+price[pos]+"!");
        builder.setTicker("您有一条新消息");
        builder.setSmallIcon(img[pos]);

 builder.setLargeIcon(BitmapFactory.decodeResource(context.getResources(),img[pos]));
        builder.setAutoCancel(true);
        Intent mainIntent=new Intent(context,ProductActivity.class);//指向商品详情的Activity
        Bundle bundle=new Bundle();
        bundle.putInt("key",pos);
        mainIntent.putExtras(bundle);
        PendingIntent
pendingIntent=PendingIntent.getActivity(context,0,mainIntent,PendingIntent.FLAG_UPDATE_CURR
ENT);
        builder.setContentIntent(pendingIntent);
        NotificationManager manager=(NotificationManager)
context.getSystemService(Context.NOTIFICATION_SERVICE);
        Notification notify=builder.build();
        manager.notify(0,notify);
    }
}
```

Eventbus：

```
EventBus.getDefault().post(new MessageEvent(pos));//发送消息

EventBus.getDefault().register(this);//在MainActivity的onCreate中注册接收

EventBus.getDefault().unregister(this);//onDestroy中取消注册
```

```
public class MessageEvent {//自定义一个类用来传输消息，这里的消息是商品在数组中的下标
    public int pos;
    public MessageEvent(int p){
        this.pos=p;
    }
}
```

```
@Subscribe(threadMode = ThreadMode.MAIN)//接收Eventbus并进行事件处理
public void onMessageEvent(MessageEvent messageEvent){
    int pos=messageEvent.pos;
    Map<String, String> tmp = new HashMap<>();
    tmp.put("icon", String.valueOf(name[pos].charAt(0)));
    tmp.put("name", name[pos]);
    tmp.put("price", price[pos]);
    list.add(tmp);
    simpleAdapter.notifyDataSetChanged();//更新数据配器使得添加的商品能显示在购物车中
}
```

onRestart中判断是通过哪种方式返回MainActivity(点击通知和点击back箭头)

```
@Override
protected void onRestart(){
    super.onRestart();
    if(flag==false)//点击通知返回MainActivity，需要显示购物车
    {
        findViewById(R.id.shopcar).setVisibility(View.VISIBLE);
        findViewById(R.id.mainpage).setVisibility(View.VISIBLE);
        findViewById(R.id.RecyclerView).setVisibility(View.GONE);
        findViewById(R.id.shoplist).setVisibility(View.GONE);
        ProductActivity.instance.finish();//finish留在栈中的ProductActivity
    }
    else{//点击back箭头返回MainActivity，需要显示商品列表
        findViewById(R.id.shopcar).setVisibility(View.GONE);//设置购物车不可见
        findViewById(R.id.mainpage).setVisibility(View.GONE);//设置mainpage不可见
        findViewById(R.id.RecyclerView).setVisibility(View.VISIBLE);//设置RecyclerView可见
        findViewById(R.id.shoplist).setVisibility(View.VISIBLE);//设置购物车图标可见
    }
}
```

- 实验遇到困难以及解决思路

    1. 在我点击动态广播产生的通知返回购物车的时候，切换回商品列表，再点击任意一个商品进入该商品详情，进入的页面都会是动态广播的通知对应的商品详情。(MainActivity为购物车/商品列表对应的Activity，ProductActivity为商品详情对应的Activity)

        这是由于我在切换回MainActivity的时候没有将ProductActivity结束掉，导致这个Activity依然留在栈中，所以我在商品列表中点击商品试图start ProductActivity时会直接从栈中Restart，也就是动态广播那条通知对应的商品。解决方案是在我点击通知返回购物车时finish掉ProductActivity，在商品详情这个类里设置一个静态变量等于this——`public static ProductActivity instance=this`然后在MainActivity中使用`ProductActivity.instance.finish()`就可以finish掉留在栈中的这个Activity。

    2. 点击动态广播的通知应该回到购物车界面，点击商品详情的back箭头应该回到商品列表页面，这二者没有同时做到。

        考虑到Activity的生命周期，从栈中重新调用后是onRestart>onStart->onResume的过程，因此我设置了一个flag变量来判断是点击通知返回MainActivity还是点击back返回MainActivity，然后在onRestart中根据对应情况设置View的可见性。

4.实验思考及感想

由于在上一次实验中我已经使用过了广播机制，因此这次实验做起来也相对容易，了解到了Android中传输数据除了使用广播还可以使用Eventbus。静态广播的运行要比动态广播慢上一些，这点在真机上可能体现不出来，但是在模拟机这种硬件极差的机器上表现的很明显。此外的话，上面所说的实验中碰到的一些困难，也让我对Android中的Activity理解更加深刻。

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。