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：

年级	15	专业（方向）	互联网
学号	15352006	姓名	蔡丽芝
电话	13538489980	Email	314749816@qq.com
开始日期	2017/10/31	完成日期	2017/10/31

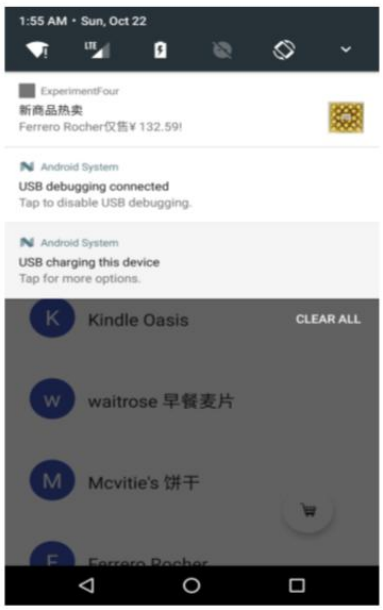
一、 实验题目

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

启动 app

点击通知

点击购物车

点击通知

（2） 实验步骤以及关键代码

A. 启动 app 时，由静态广播产生一个随机物品的通知

```
<receiver android:name=".StaticReceiver"
    android:permission="true"
    android:exported="true">
    <intent-filter>
        <action android:name="com.example.lizhicai.MyStaticFilter"></action>
    </intent-filter>
</receiver>
```

exported：属性表示当前 activity 能否被另一个 application 组件启动：true 允许被启动，false 不运行被启动

静态广播需要在 AndroidManifest.xml 中进行注册。

```
final String STATICACTION = "com.example.lizhicai.MyStaticFilter";
int seed = good.size();
Random random = new Random();
int r = random.nextInt(seed);
Intent intentBroadcast = new Intent(STATICACTION);
intentBroadcast.putExtra("name", good.get(r).getName());
intentBroadcast.putExtra("price", good.get(r).getPrice());
intentBroadcast.putExtra("info", good.get(r).getInfo());
sendBroadcast(intentBroadcast);
```

在 mainActivity.java 文件中进行广播的发送：

通过 STATICACTION 为广播命名一个特定的广播名称，通过 random 函数随机生成 0 到 10 的数字，从而随机生成商品。利用 intent 和 bundle 进行图片，名字和价格的存储，并通过 sendBroadcast 将 intentBroadcast 发送出去。

```java
public class StaticReceiver extends BroadcastReceiver {
    private static final String STATICACTION = "com.example.lizhicai.MyStaticFilter";
    private String name;
    private String price;
    private String info;
    @Override
    public void onReceive(Context context, Intent intent) {
        if(intent.getAction().equals(STATICACTION)) {
            name = intent.getStringExtra("name");
            price = intent.getStringExtra("price");
            info = intent.getStringExtra("info");
            Bitmap bitmap = BitmapFactory.decodeResource(context.getResources(), getImage.MapImage(name));
            NotificationManager mNotificationManager = (NotificationManager) context.getSystemService(Context.NOTIFICATION_SERVICE);
            NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(context);
            mBuilder.setContentTitle("新商品热卖")
                    .setContentText(name+"仅售"+price)
                    .setTicker("您有一条新信息")
                    .setLargeIcon(bitmap).setSmallIcon(getImage.MapImage(name))
                    .setAutoCancel(true);
            Intent myIntent = new Intent(context, ShowDetails.class);
            myIntent.putExtra("Name", name);
            myIntent.putExtra("Price", price);
            myIntent.putExtra("Info", info);
            PendingIntent pendingIntent = PendingIntent.getActivity(context, 0, myIntent, PendingIntent.FLAG_CANCEL_CURRENT);
            mBuilder.setContentIntent(pendingIntent);
            mBuilder.setDefaults(Notification.DEFAULT_ALL);
            Notification notification = mBuilder.build();
            mNotificationManager.notify(1, mBuilder.build());
        }
    }
}
```

在 staticReceiver.java 中重写 onReceive 方法，自定义收到广播时需要实现的数据处理和动作。

```java
if(intent.getAction().equals(STATICACTION)) {
```

首先必须判断接受到的广播是否为静态广播，通过判断接受到的 intent.getAction()是否和 STATICACTION 相等，只有是需要处理的广播，才会进行执行下面的代码。

```java
name = intent.getStringExtra("name");
price = intent.getStringExtra("price");
info = intent.getStringExtra("info");
Bitmap bitmap = BitmapFactory.decodeResource(context.getResources(), getImage.MapImage(name));
```

获取 intnet 中的数据 name，price，info，通过 BitmapFactory 的方式将图片资源转化为位图文件，用于设定大图标

```java
// 获取状态通知栏管理
NotificationManager mNotificationManager = (NotificationManager) context.getSystemService(Context.NOTIFICATION_SERVICE);
NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(context);
mBuilder.setContentTitle("新商品热卖")                    // 设置通知标题栏
        .setContentText(name+"仅售"+price)               // 设置通知栏显示内容
        .setTicker("您有一条新信息")                      // 通知首次出现在通知栏，会带上升动画效果
        .setLargeIcon(bitmap).setSmallIcon(getImage.MapImage(name))   //设置通知大ICON和小ICON
        .setAutoCancel(true);                            //当用户单击面板的时候，通知将自动取消
```

NotificationManager 负责将 notifacation 在状态栏的通知和取消
通过 NotificationCompat.Builder 动态设置 notification 的属性。

```
Intent myIntent = new Intent(context, ShowDetails.class);   //定义一个intent
myIntent.putExtra("Name", name);              // 将name, price, info传入intent中存储数据
myIntent.putExtra("Price", price);
myIntent.putExtra("Info", info);
PendingIntent pendingIntent = PendingIntent.getActivity(context, 0, myIntent, PendingIntent.FLAG_CANCEL_CURRENT);
mBuilder.setContentIntent(pendingIntent);
```

点击通知 noticificaiton，跳转到指定的 activity，主要使用 PendingIntent 和 setContentIntent

B. 点击购物车图标，动态广播产生广播

```
private static String DYNAMICACTION = "com.example.lizhicai.MyDynamicFilter";
private DynamicReceiver dynamicReceiver = new DynamicReceiver();
IntentFilter intentFilter = new IntentFilter();
intentFilter.addAction(DYNAMICACTION);    // 添加动态广播的Action
registerReceiver(dynamicReceiver, intentFilter);//注册自定义动态广播信息
```

注册动态广播

```
public class DynamicReceiver extends BroadcastReceiver {
    private String name;
    private static final String DYNAMICACTION = "com.example.lizhicai.MyDynamicFilter";
    @Override
    public void onReceive(Context context, Intent intent) {
        if(intent.getAction().equals(DYNAMICACTION)) {
            Toast.makeText(context,"动态广播",Toast.LENGTH_SHORT).show();
            name = intent.getStringExtra("name");
            Bitmap bitmap = BitmapFactory.decodeResource(context.getResources(), getImage.MapImage(name));
            NotificationManager mNotificationManager = (NotificationManager) context.getSystemService(Context.NOTIFICATION_SERVICE);
            Intent myIntent = new Intent(context, MainActivity.class);
            PendingIntent pendingIntent=PendingIntent.getActivity(context,1,myIntent,PendingIntent.FLAG_UPDATE_CURRENT);

            NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(context);
            mBuilder.setContentTitle("马上下单")
                    .setContentText(name+"已添加到购物车")
                    .setLargeIcon(bitmap)
                    .setSmallIcon(getImage.MapImage(name))
                    .setTicker("您有一条新信息")
                    .setAutoCancel(true);
            mBuilder.setContentIntent(pendingIntent);
//            mBuilder.setContent(contentView);
            mBuilder.setDefaults(Notification.DEFAULT_ALL);
            Notification notification = mBuilder.build();
            mNotificationManager.notify(2, mBuilder.build());
        }
    }
}
```

和静态广播一样，需要在 DynamicReceiver 重写 onReceive 函数，思路和静态广播的基本一致，这里不加赘述

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

注销动态广播

```
<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>
```

当点击通知时，为了不会新建一个新的购物车列表，需要将 launchMode 设为 singleTask

C. Eventbus 的使用

```
public class eventBus {
ackage-private more... (Ctrl+F1)
    private String price;
    private String info;

    public eventBus(int cnt, String name, String price, String info) {
        this.cnt = cnt;
        this.name = name;
        this.price = price;
        this.info = info;
    }
    public int getCnt() { return cnt; }
    public String getName() { return name; }
    public String getPrice() { return price; }
    public String getInfo() { return info; }
}
```

声明一个事件类 eventBus，用于传递商品信息

```
@Subscribe(threadMode = ThreadMode.MAIN)
public void first(eventBus messageEvent) {
    int cnt = messageEvent.getCnt();
    shopCarGood.add(new Goods(messageEvent.getName(), messageEvent.getInfo(), messageEvent.getPrice()));
    mshopCarAdapter.notifyDataSetChanged();
}
```

准备订阅者，并选定线程模式为 ThreadMode.MAIN

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

注册订阅者

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

当退出的时候，注销订阅者

```
holder.addgoods.setOnClickListener((view) -> {
        Intent intentBroadcast = new Intent();
        intentBroadcast.setAction(DYNAMICACTION);
        intentBroadcast.putExtra("name", Name);
        context.sendBroadcast(intentBroadcast);
        Toast.makeText(context, "商品已添加到购物车", Toast.LENGTH_SHORT).show();
        EventBus.getDefault().post(new eventBus(0, Name, Price, Info));
});
```

当点击购物车图标的时候，通过 EventBus 传递商品信息

(3) 实验遇到困难以及解决思路

实验主要遇到的问题是设置通知的大图标和小图标，由于手机版本和 api 版本不同的原因，通过发送到真机上，我的手机无法自定义图标，所以必须通过使用虚拟机的方式进行模拟，然而 Android studio 自带的虚拟机启动非常慢，所以一开始折腾了很久，后来放弃使用 Android studio 自带的虚拟机，使用 Genymotion 模拟器并在 Android studio 上安装 Genymotion 插件，终于可以比较快速的启动虚拟机，但是还是遇到一个问题，就是有些 API 版本无法正常现实小图标，所以尝试了很多个 api，后来终于使用 4.4 的可以正常显示大图标和小图标

四、 **课后实验结果**

解决了 TA 给的示范 apk 存在的 bug，bug 如下

当点击购买商品 a -》出现动态广播 -》点击动态广播的通知 -》进入购物车列表 -》点击悬浮按钮回到商品列表 -》 任意点击非 a 的商品 -》却都会进入 a 的商品详情页面
出现 bug 的原因：
当点击了动态广播的通知后，a 商品详情并没有 finish
所以解决方式：
在 mainActivity 里发送广播，商品详情的类里接受广播把自己 finish

五、 **实验思考及感想**

通过本次实验，学会了如何使用广播，在遇到闪退问题时候，第一次通过 debug 的方式解决了问题，终于不用像以前一样不明白自己哪个地方不懂，一点思路都没有，通过 debug 的方式，对安卓的学习有了更深的了解。