中山大学移动信息工程学院本科生实验报告

（2016 年秋季学期）

课程名称：移动应用开发　　任课教师：郑贵锋

年级	14	专业（方向）	物联网
学号	14353259	姓名	商家煜
电话	13211143626	Email	shangjy3@mail2.sysu.edu.cn
开始日期	2016.9.22	完成日期	2016.9.22

一、 实验题目

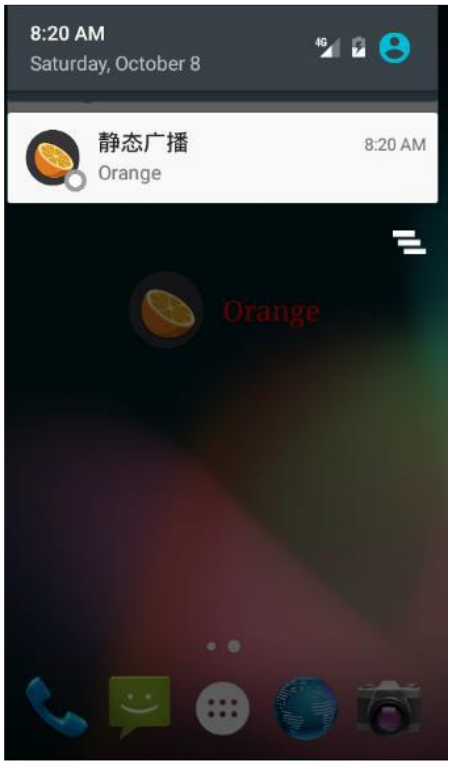
1、掌握 Broadcast 编程基础
2、掌握动态注册 Broadcast 和静态注册 Broadcast
3、掌握Notification 编程基础

二、 实现内容

实现一个Android 应用，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） 该界面为应用启动后看到的界面。

（2） 点击静态注册按钮，跳转至如下界面

点击表单项目。如 orange。会有对应通知产生，点击通知返回主界面：



（3） 点击动态注册按钮，跳转至如下界面。

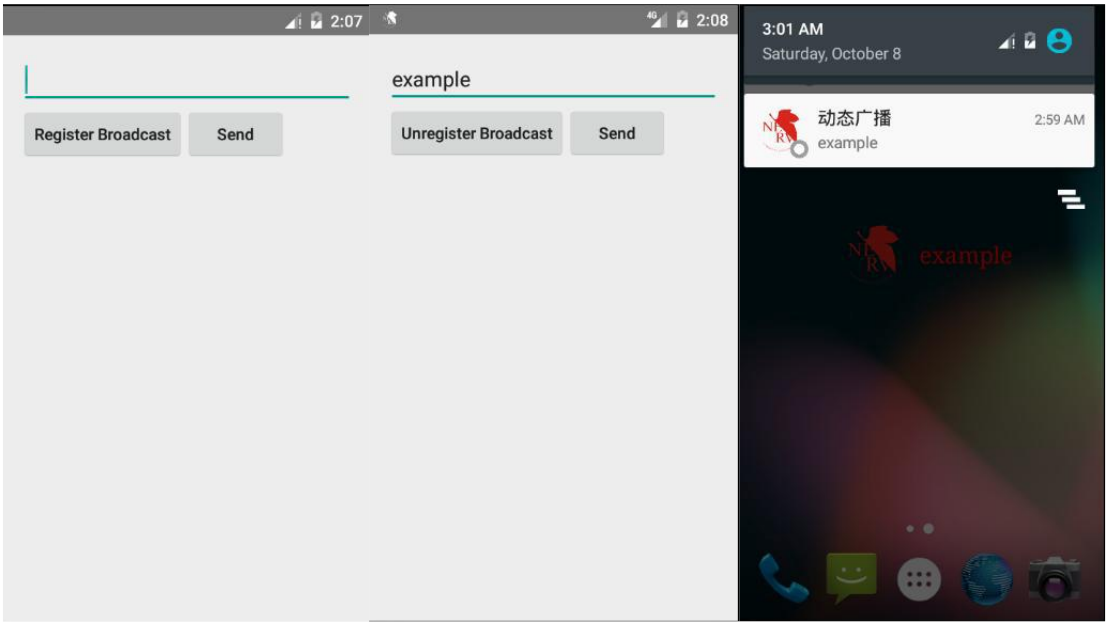
实现以下功能：

a)可以编辑广播的信息，点击Send 按钮发送广播。

b)设置一个按钮进行广播接收器的注册与注销。

c)广播接收器若已被注册，发送出的广播信息会产生一个对应通知。

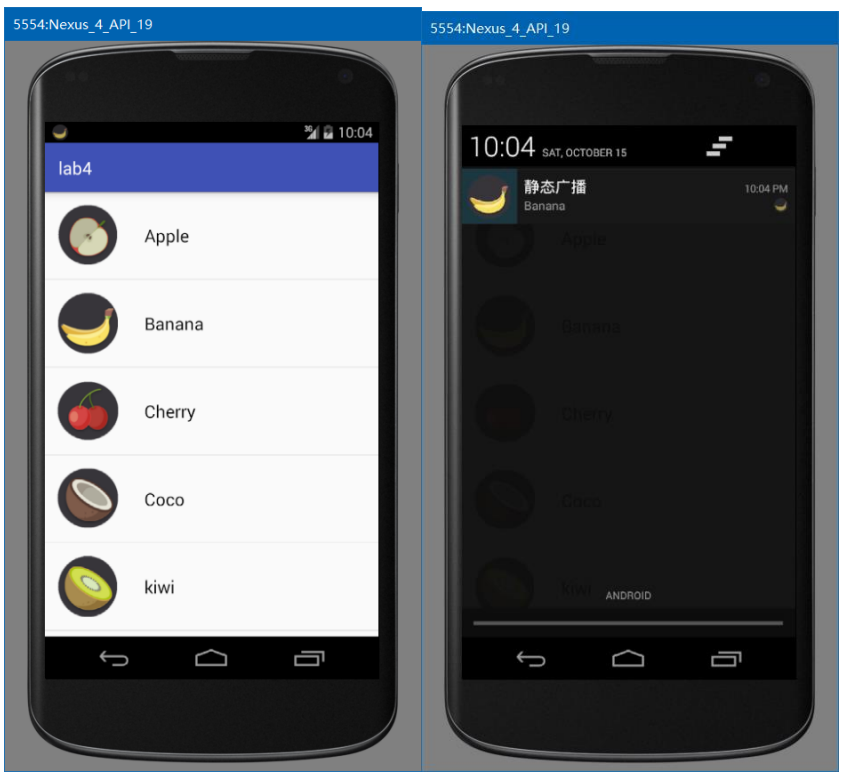
d)点击 Notification 可以跳转回主界面。

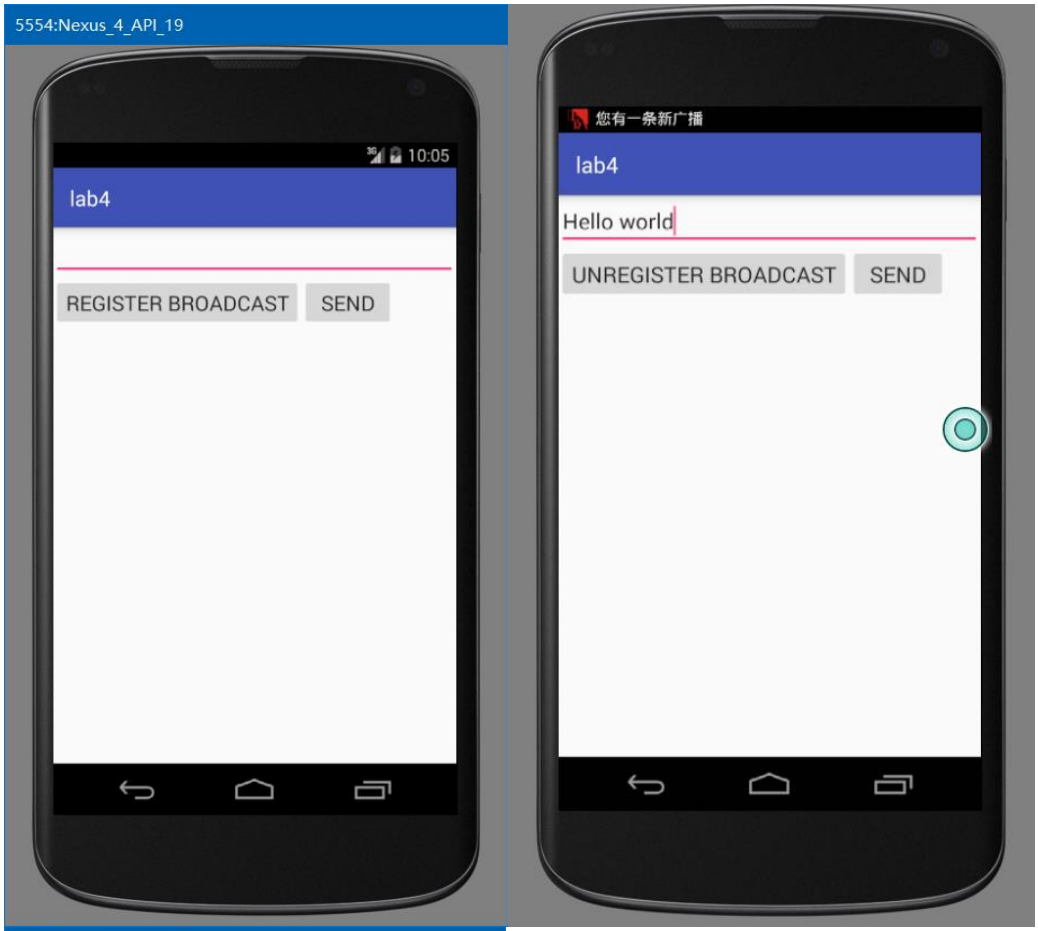


（未注册广播）　　（已注册广播）　　（已注册后点击发送）

三、 课堂实验结果

（1） 实验截图




5554:Nexus_4_API_19
10:05
lab4
REGISTER BROADCAST
SEND
您有一条新广播
lab4
Hello world
UNREGISTER BROADCAST
SEND


5554:Nexus_4_API_19
10:05 SAT, OCTOBER 15
动态广播
Hello world
10:05 PM
ANDROID

（2） 实验步骤以及关键代码

1) 静态注册设置

- AndroidMainfest.xml

```
<receiver
    android:name=".static_receiver"
    android:enabled="true"
    android:exported="true">
    <intent-filter>
        <action android:name="com.example.user.lab4.static_receiver" />
    </intent-filter>
</receiver>
```

- XML 界面

```
<ListView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/list_view"></ListView>
```

图：XML 界面

XML 界面根据要求只需要设置一个 ListView 即可，然后再 item 界面中，设置一个 imageView 和 TextView 即可，基本过程与上一次的相似。

- java 代码

```
final int[] num = {R.mipmap.apple, R.mipmap.banana, R.mipmap.cherry,
                   R.mipmap.coco, R.mipmap.kiwi, R.mipmap.orange, R.mipmap.pear,
                   R.mipmap.strawberry, R.mipmap.watermelon};
final String[] name = {"Apple", "Banana", "Cherry", "Coco", "kiwi", "Orange",
              "Pear", "Strawberry", "Watermelon"};
for(int i = 0; i < 9; i++){
    Map<String, Object> temp = new LinkedHashMap<>();
    temp.put("num", num[i]);
    temp.put("name", name[i]);
    data.add(temp);
}
```

图：ListView 部分

java 编程方面，ListView 部分与上次不同的地方在于，通过一个数组来存储对应的图片的 id，然后其他方面也与上一次的代码一致。

```
listview.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    private static final String STATICATION = "com.example.user.lab4.static_receiver"
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        Intent intent = new Intent(STATICATION);
        Bundle bundle = new Bundle();
        bundle.putString("name", name[position]);
        bundle.putInt("image", num[position]);
        intent.putExtras(bundle);
        sendBroadcast(intent);
    }
});
```

图：发送广播

发送广播的地方，首先我们先设置我们广播的名称，然后利用 bundle 的功能，存入后面 receiver 中生成消息栏中所需要用到的参数，包括图像 id 和名称字符串，然后通过 sendBroadcast 函数发送广播。

- Broadcast receiver

```
if(intent.getAction().equals(STATICATION)){
    Bundle bundle = intent.getExtras();
    String name = bundle.getString("name");
    int image = bundle.getInt("image");
```

在 Broadcast receiver 中，重写 onReceive 函数。通过判断广播的消息是否与该接收器需要接受的消息相同来决定 receiver 是否工作，然后读取 intent 中传递的参数到该类中以备后序程序执行使用。

```
NotificationManager manager = (NotificationManager) context.getSystemService(Context.NOTIFICATION_SERVICE);
Notification.Builder builder = new Notification.Builder(context);
Bitmap bm = BitmapFactory.decodeResource(context.getResources(), image);

builder.setContentTitle("静态广播").setContentText(name).setSmallIcon(image)
        .setTicker("您有一条新广播").setLargeIcon(bm)
        .setWhen(System.currentTimeMillis())
        .setVibrate(new long[] {0, 500, 0, 0})
        .setAutoCancel(true);

Intent mIntent = new Intent(context, MainActivity.class);
PendingIntent mPendingIntent = PendingIntent.getActivity(context, 0, mIntent, 0);
builder.setContentIntent(mPendingIntent);

Notification notify = builder.build();
manager.notify(0, notify);
```

图：Notification 创建

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。这里我们通过实例化一个 NtificationManager 对象负责将 Notification 在状态显示出来和取消。然后再实例化一个 Notification.Builder 对象来动态的设置 Notification 中的一些属性。这里我们设定的属性包括，提示消息的 Title，包含信息，小图标，提示信息，大图标，产生时间，设置震动提醒以及是否可以被点击取消。其中大图标需要实例化一个 Bitmap 对象，来进行获取对应的图片信息（通过 id 和 getResource）

最后，实例化 Notification 对象来设置 Notification 的相关属性，通过绑定刚才创建好的 builder 对象，并通过 manager.notify 函数来执行这个跳转，第一个参数表示 notify 的 id

2) 动态注册设置

- XML 界面

```
<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textSize="20dp"
    android:id="@+id/edit_text"/>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/RegisterBroadcast"
        android:id="@+id/button1"
        android:textSize="20dp"/>
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Send"
        android:id="@+id/button2"
        android:textSize="20dp"/>
```

XML 界面中，我们设置一个 EditText 和两个 button 来完成实验要求所展示的界面布局，EditText 用以接收用户需要再动态注册后的信息提示中的提示信息。

- java 代码

```
button1.setOnClickListener(new View.OnClickListener(){
    int i = 1;
    private static final String DYNAMICATION = "com.example.user.lab4.dynamicreceiver";
    final BroadcastReceiver br = new dynamic_receiver();
    @Override
    public void onClick(View v) {

        if(i == 1){
            i = 0;
            button1.setText("Unregister Broadcast");

            IntentFilter dynamic_filter = new IntentFilter();
            dynamic_filter.addAction(DYNAMICATION);
            registerReceiver(br, dynamic_filter);
        }
        else{
            i = 1;
            button1.setText("Register Broadcast");
            unregisterReceiver(br);
        }
    }
});
```

图：注册和注销按钮

对于注册/注销按钮，我们先创建一个自己已经编写好的实例化对象 dynamic_receiver 以备后序使用通过标志位来判断当前行为是进行注册还是进行注销。在注册过程中，我们将按下后的按钮显示的文字修改为注销。然后创建一个 IntentFilter 实例。通过调用 addAction 函数来添加广播信息，最后调用 registerReceiver 函数，进行动态注册。第一个参数为刚才实例化的 BroadcastReceiver 类，第二个参数为刚实例化后并添加了广播信息的 dynamic_filter。

```
button2.setOnClickListener(new View.OnClickListener(){
    private static final String DYNAMICATION = "com.example.user.lab4.dynamicreceiver";
    @Override
    public void onClick(View v){
        if(button1.getText().toString().equals("Unregister Broadcast")) {
            Intent intent = new Intent(DYNAMICATION);
            Bundle bundle = new Bundle();
            bundle.putString("name", text.getText().toString());
            intent.putExtras(bundle);
            sendBroadcast(intent);
        }
    }
});
```

图：发送广播

首先判断当前是否已经注册了一个 BroadcastReceiver，如果是，则像静态注册一样操作即可，在 DynamicReceiver 中的代码编写也与之前 StaticReceiver 基本一致，这里就不再赘叙。

（3） 实验遇到困难以及解决思路

本次的实验过程比较简单，基本按照老师和TA上课所讲的内容加以实现即可，没有碰到什么麻烦。

四、 实验思考及感想

本次实验课后对 Broadcast 的使用有了比较详细的了解和时间操作，并尝试在实验要求的基础上尝试着自己添加一些小的 Notification.builder 的功能，试验中需要注意的是在 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 MainActivity。实验过后将继续复习上课讲授的实质和学习新的有关安卓的基本知识，以备下一次的实验使用