Android移动应用开发基础教程

讲授：葛新

第4章 广播机制

本章主要内容：

- 广播机制简介
- 使用广播接收器
- 广播接收器优先级与有序广播

第4章 广播机制

- 什么场景下需要广播?
- 谁发送?
- 发送什么?
- 谁接收?

4.1 广播机制简介

- Android的广播机制非常灵活。广播可来自于系统，也来自其他应用，甚至于应用内部的其他模块。应用程序可以只对感兴趣的广播进行注册，也只有注册了的广播才可能接收到。
- Android中的广播可分为两种类型：标准广播和有序广播。
  - **标准广播**：标准广播在发出后，所有接收器均可接收到广播消息。各个接收器之间没有先后顺序之分。标准广播发出后，不可能被中断。
  - **有序广播**：有序广播在发出后，同一时间只有优先级较高的一个接收器接收到广播消息。只有在优先级较高的接收器处理完广播消息后，广播才能继续向优先级较低的接收器继续传递。在当前接收器中，可中断广播，使后继接收器无法收到广播消息。
- Android提供了一套完整的API用于发送和接收广播。发送广播时，可类似于Activity使用Intent对象来传递数据。接收广播使用广播接收器（BroadcastReceiver）。

第4章 广播机制

本章主要内容：

- 广播机制简介
- 使用广播接收器
- 广播接收器优先级与有序广播

4.2 使用广播接收器

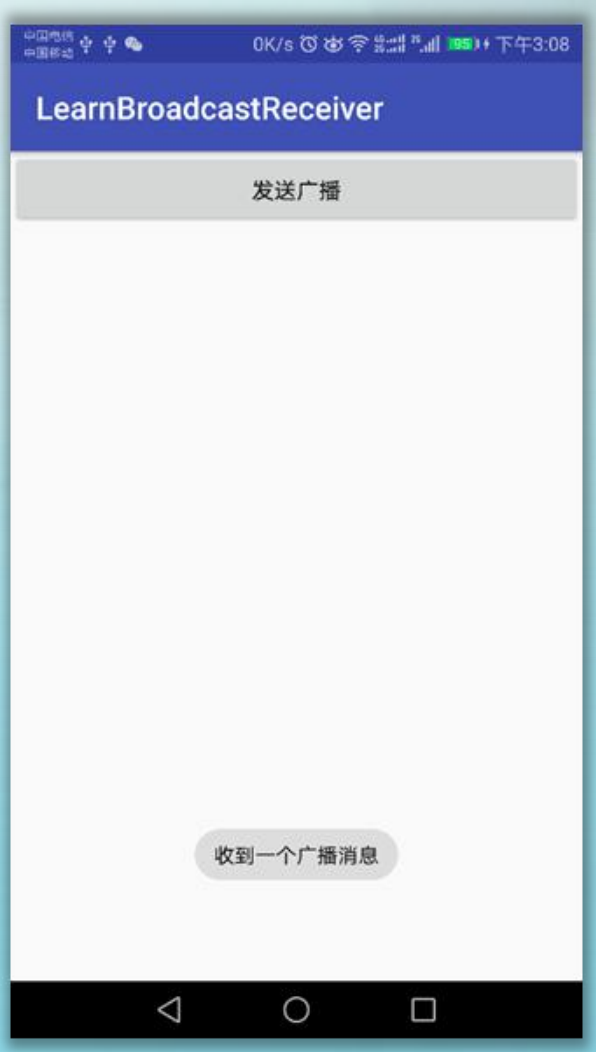
Android提供了一个BroadcastReceiver类，通过开展该类，并重写onReceive()方法，即可创建一个广播接收器。接收到广播消息时，onReceive()方法被执行。

本节主要内容：

1. 静态注册广播接收器
2. 动态注册和注销广播接收器
3. 接收系统广播
4. 发送本地广播

4.2.1 静态注册广播接收器

- 通过创建BroadcastReceiver的子类实现一个广播接收器后，首先需要对广播接收器进行注册。只有经过了注册的广播接收器才能接收到广播消息。静态注册是指在应用程序的清单文件AndroidManifest.xml中添加广播接收器的注册信息。
- 下面的实例通过静态注册的方式来使用广播接收器，具体操作步骤如下：（实例项目：源代码\04\LearnBroadcastReceiver）



编写MyReceiver.java

```
package com.example.xbg.learnbroadcastreceiver;
import android.content.BroadcastReceiver;
……
public class MyReceiver extends BroadcastReceiver {
    public MyReceiver() {
    }
    @Override
    public void onReceive(Context context, Intent intent) {
        Toast.makeText(context,"收到一个广播消息",Toast.LENGTH_LONG).show();
    }
}
```

实现用于发送广播消息的sendMsg()方法

```
package com.example.xbg.learnbroadcastreceiver;
……
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void sendMsg(View view){
        sendBroadcast(new Intent(this,MyReceiver.class));
    }
}
```

4.2.2 动态注册和注销广播接收器

- 动态注册和注销广播接收器是指通过执行持续代码来注册和注销广播接收器，从而可由用户来控制是否启用接收器来接收广播。
- 下面的实例说明如何动态注册和注销广播接收器，具体操作步骤如下：（实例项目：源代码\04\LearnBroadcastReceiver2）

编写MyReceiver2.java代码，实现广播接收器

```
package com.example.xbg.learnbroadcastreceiver2;
……
public class MyReceiver2 extends BroadcastReceiver {
    public static String ACTION="learnbroadcastreceiver2.MyReceiver2";//定义操作
    public MyReceiver2() {
    }
    @Override
    public void onReceive(Context context, Intent intent) {
        Toast.makeText(context,"收到一个广播消息",Toast.LENGTH_LONG).show();
    }
}
```

实现注册广播接收器、注销广播接收器和发送广播消息的方法

```
package com.example.xbg.learnbroadcastreceiver2;

……

public void registerMyReceiver(View view){
        //注册广播接收器
        if(receiver==null){
            receiver=new MyReceiver2();
            registerReceiver(receiver,new IntentFilter(MyReceiver2.ACTION));
        }
```

```
public  void  unRegisterMyReceiver(View view){
        //注销广播接收器
        if(receiver!=null){
            unregisterReceiver(receiver);
            receiver=null;
        }
    }
    public void sendMsg(View view){
        sendBroadcast(new Intent(MyReceiver2.ACTION));//发送广播
    }
}
```

4.2.3 接收系统广播

- Android提供了一系列系统广播，在系统中发生某种事件时，系统就会自动发送对应的广播消息。例如，在系统WIFI断开或连接时，系统会发送包含了android.net.wifi.STATE_CHANGE操作字符串的Intent的广播消息，接收器接收到该消息时，可判定当前WIFI连接是否可用。
- 在Android SDK安装目录下的platforms\android-25\data文件夹中的broadcast_actions.txt文件中，可查看对应Android版本支持的系统广播操作字符串。
- 要使接收器响应系统广播，需要在注册接收器时，在IntentFilter指明可响应的操作。例如，要让接收器监听WIFI连接状态变化，可在AndroidManifest.xml文件中用如下代码来注册接收器：（实例项目：源代码\04\ReceiveSystemBroadcast）

在AndroidManifest.xml文件中注册接收器

```
<receiver android:name=".SysReceiver">
    <intent-filter>
        <action android:name="android.net.wifi.WIFI_STATE_CHANGED"/>
    </intent-filter>
</receiver>
```

自定义的广播接收器类SysReceiver

```
package com.example.xbg.receivesystembroadcast;
import android.content.BroadcastReceiver;
……
public void onReceive(Context context, Intent intent) {
        int state= intent.getIntExtra(WifiManager.EXTRA_WIFI_STATE,0);
        if(state==WifiManager.WIFI_STATE_DISABLED){
            Toast.makeText(context,"WIFI连接已关闭！",Toast.LENGTH_SHORT).show();
        }else if(state==WifiManager.WIFI_STATE_ENABLED){
            Toast.makeText(context,"WIFI已连接！",Toast.LENGTH_SHORT).show();
        }
……
```

4.2.4 发送本地广播

- 当在活动中直接调用sendBroadcast()方法发送广播时，广播默认为系统全局广播，即可被其他应用中的接收器接收。如果不希望关键的广播消息被其他应用接收，则可使用本地广播。本地广播只能被当前应用中的接收器接收。
- Android提供了一个LocalBroadcastManager（本地广播管理器）来管理本地广播的注册、注销和发送等操作。
- 下面的实例说明了如何使用本地广播。（实例项目：源代码\04\LocalBroadcast）

```
public class MainActivity extends AppCompatActivity {
    private MyReceiver localReceiver;
    private  LocalBroadcastManager localBroadcastManager;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //获得当前本地广播管理器
        localBroadcastManager= LocalBroadcastManager.getInstance(this);
        IntentFilter intentFilter=new IntentFilter("MyLocalBroadcastReceiver");
        localReceiver=new MyReceiver();//创建广播接收器对象
        localBroadcastManager.registerReceiver(localReceiver,intentFilter);//注册本地广播接收器
    }
```

```
protected void onDestroy() {
        super.onDestroy();
        localBroadcastManager.unregisterReceiver(localReceiver);//注销本地广播接收器
    }
    public void sendMyBroadcst(View view){
        Intent  intent=new Intent("MyLocalBroadcastReceiver");//用注册的操作创建Intent
        localBroadcastManager.sendBroadcast(intent);
    }
```

```
public static class MyReceiver extends BroadcastReceiver {
        public MyReceiver() { }
        @Override
        public void onReceive(Context context, Intent intent) {
            Toast.makeText(context,"收到一个本地广播消息",
                                        Toast.LENGTH_LONG).show();
        }
    }
```

4.3 广播接收器优先级与有序广播

- 在前面的内容中，对同一个广播消息，接收器之间没有先后顺序之分，所有接收器同时接收到广播。
- 在注册广播接收器时，可以为接收器的IntentFilter设置优先级，优先级越高的接收器先接收到广播。
- 只有等优先级高的接收器处理完广播后，优先级较低的才能接收到广播。
- 在AndroidManifest.xml中静态注册接收器时，可使用<intent-filter>标签的android:priority属性来设置广播接收器优先级。
- 例如：（实例项目：源代码\04\PriorityOrderBroadcast）

```
<receiver android:name=".MyReceiver1" >
    <intent-filter android:priority="3">
    <action android:name=
                "com.example.xbg.priorityorderbroadcast.ACTION"/>
    </intent-filter>
</receiver>
<receiver android:name=".MyReceiver2">
    <intent-filter android:priority="4">
    <action android:name=
                    "com.example.xbg.priorityorderbroadcast.ACTION"/>
    </intent-filter>
</receiver>
```