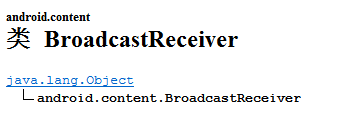
Broadcast Receiver(广播接受者)



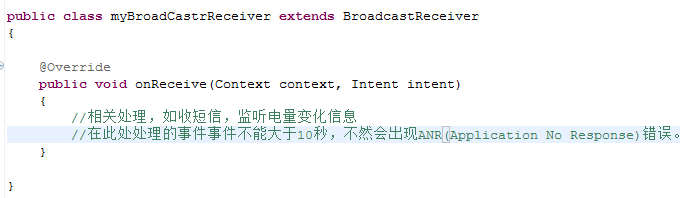
BroadcastReceiver本质上是一种全局的监听器，用于监听系统全局的广播消息。它可以接收来自系统和应用的的广播。由于BroadcastReceiver是一种全局的监听器，因此它可以非常方便地实现系统不同组件之间的通信。比如Activity与通过startService()方法启动的Service之间通信，就可以借助于BroadcastReceiver来实现。

1. BroadcastReceiver启动步骤（发出广播）
2. 创建需要启动的BroadCastReceiver的Intent，将需要发送的信息装入一个Intent对象中；
3. 调用Context的sendBroadcast()方法或者sendOrderBroadcast ()方法启动指定的BroadcastReceiver。其中sendBroadcast ()发出的是普通广播，sendOrderBroadcast ()发出的是有序广播；
4. 当应用发出一个Broadcast Intent之后与该Intent匹配的组件都可能被启动。

**1.1、创建BroadcastReceiver步骤**

**第一步：创建BroadcastReceiver的子类**

由于BroadcastReceiver本质上是一种监听器，所以创建BroadcastReceiver的方法也非常简单，只需要创建一个BroadcastReceiver的子类然后重写onReceive (Context context, Intentintent)方法即可。



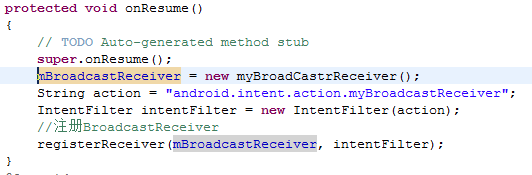
**第二步：注册BroadcastReceiver**

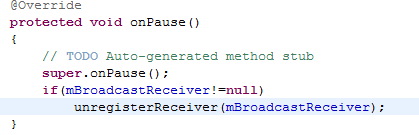
一旦实现了BroadcastReceiver，接下就应该指定该BroadcastReceiver能匹配的Intent即注册BroadcastReceiver。BroadcastReceiver注册分为2种：动态注册和静态注册。简单的说动态注册就是在代码中注册，静态注册就是在xml文件中注册。

（1）动态注册BroadcastReceiver

这种方法是通过代码在.[Java](http://www.2cto.com/kf/ware/Java/)文件中进行注册。通过这种方式注册的广播为**非常驻型**广播，即它会跟随Activity的生命周期，所以在Activity结束前我们需要调用unregisterReceiver(receiver)方法移除它，否则会发生异常。

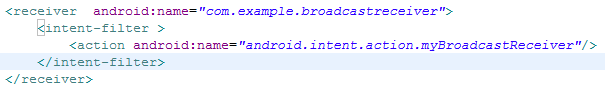
在官方文档中描述，在 Activity 中代码注册广播建议在： **onResume() 中注册**；**在onPause()中注销**。You won't receive intents when paused, and this will cut down on unnecessary system overhead. Do not unregister in [**Activity.onSaveInstanceState()**](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/android/app/Activity.html#onSaveInstanceState(android.os.Bundle)), because this won't be called if the user moves back in the history stack.

**//注销BroadcastRecevier**

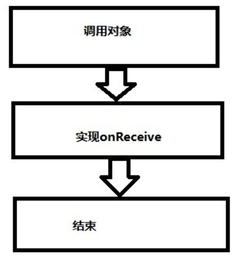


（2）静态注册BroadcastReceiver

这种方法是在配置AndroidManifest.xml配置文件中注册，通过这种方式注册的广播为**常驻型**广播，也就是说如果应用程序关闭了，有相应事件触发，程序还是会被系统自动调用运行。



**1.2、广播的生命周期**



**生命周期只有十秒左右，如果在 onReceive() 内做超过十秒内的事情，就会报错ANR**。

**怎么用好BroadcastReceiver？**如果需要处理一些耗时操作，应该通过发送Intent给Service，由Service来完成，这里不能使用子线程来解决，因为BoradcastReceiver的生命周期太短，子线程可能还没有结束，BroadcastReceiver就先结束了。BroadcastReceiver一旦结束 , 此时 BroadcastReceiver 的所在进程很容易在系统需要内存时被优先杀死，因为它属于空进程 ( 没有任何活动组件的进程 )，如果它的宿主进程被杀死，那么正在工作的子线程也会被杀死，所以采用子线程来解决是不可靠的。

**1.3、广播的类型**

（1）普通广播（Normal broadcast）：发送一个广播，所有监听广播的广播接收器都可以监听到该广播，但是不能进行处理也不能截断。

（2）有序广播（Ordered broadcast）：按照接受者的优先级顺序接受广播，优先级别在<intent-filter>中得priority属性中声明，优先级在-1000到1000之间，值越大，优先级越高。可以对广播进行处理，也可以截断。截断广播用abortBroadcast()。

（3）异步广播：当处理完之后的Intent，依然存在，这时候registerReceiver(BroadcastReceiver, IntentFilter) 还能收到它的值，直到你把它去掉，不能将处理结果传给下一个接收者，无法终止广播。

**1.4、广播的收发**

（1）普通广播

public abstract void [sendBroadcast](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/src-html/android/content/Context.html#line.607)([Intent](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/android/content/Intent.html) intent)

public abstract void [sendBroadcast](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/src-html/android/content/Context.html#line.633)([Intent](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/android/content/Intent.html) intent, [String](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/java/lang/String.html) receiverPermission)

receiverPermission表示接受者的优先级，级别值在-1000到1000之间的整数，值越大，优先级越高。

sendBroadcast(intent);

Intent intent = **new** Intent(String action );

sendBroadcast(intent);

当某个广播接受者的action和发送广播的action相同时，就可以接受到广播。若未设定广播接受者的action，则所有的监听者都可以收到广播，其接受顺序按照其设定的优先级；若未设定优先级，则接受广播的先后顺序是随机的，并且代码注册的广播接收者会最先收到。

（2）有序广播

public abstract void [sendOrderedBroadcast](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/src-html/android/content/Context.html#line.656)([Intent](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/android/content/Intent.html) intent, [String](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/java/lang/String.html) receiverPermission)

public abstract void [sendOrderedBroadcast](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/src-html/android/content/Context.html#line.701)([Intent](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/android/content/Intent.html) intent, [String](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/java/lang/String.html) receiverPermission,

[BroadcastReceiver](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/android/content/BroadcastReceiver.html) resultReceiver,

[Handler](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/android/os/Handler.html) scheduler, int initialCode,

[String](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/java/lang/String.html) initialData, [Bundle](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/android/os/Bundle.html) initialExtras)

所有与该广播意图intent相同的，即匹配这action的接受者会收到该广播。

receiverPermission表示接受者的优先级，级别值在-1000到1000之间的整数，值越大，优先级越高。优先级也可通过intentfilter.setPriority(10)方式设置。

resultReceiver您自己 BroadcastReceiver 来当作最后的广播接收器。 调度自定义处理程序，用以安排 resultReceiver 回调，如果为 null 将在上下文的主线程中进行。   
initialCode 一种结果代码的初始值。通常为 Activity.RESULT\_OK 。这个值是 -1 ；为其他 int型也可以，如 0,1,2 ；   
initialData 一种结果数据的初始值。通常情况下为空，是 String 类型。  
initialExtras 一种结果额外的初始值。通常情况下为空，是 Bundle。

另外有序广播的接收者可以通过abortBroadcast()的方式取消广播的传播，也可以通过setResultData和setResultExtras方法将处理的结果存入到Broadcast中，传递给下一个接收者。然后，下一个接收者通过getResultData()和getResultExtras(true)接收高优先级的接收者存入的数据。

（3）异步广播（sendStickyBroadcast）

public abstract void [sendStickyBroadcast](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/src-html/android/content/Context.html#line.724)([Intent](mk:@MSITStore:E:\android资料\Android官方API文档完整版.chm::/android/content/Intent.html) intent)

当处理完之后的Intent，依然存在，直到你把它去掉。

**发这个广播需要权限：**

<uses-permission android:name=*"android.permission.BROADCAST\_STICKY"* />

去掉该广播是用方法removeStickyBroadcast(intent)，但别忘了在执行这个方法的应用的**AndroidManifest.xml** 同样要加上面的权限。

用此这个方法发来广播，代码注册方式收到广播先后次序为：注明优先级的、代码注册的、没有优先级的；如果都没有优先级，代码注册收到为最先。

注：如果在代码中注册广播接收者，也在xml文件中注册了，广播这会接收到两次信息。在代码中多次注册同一个广播接受者，最后只会接收到一次广播信息。