# Quanta基础知识- 广播+ Service详解、通知和权限

**Part 1：广播详解**

1. 什么是广播

Android中的每个应用程序都可以对自己感兴趣的广播进行注册，这样该程序就只会接收到自己所关心的广播内容，这些广播可能是来自于系统的，也可能是来自其他应用的。发送广播的方法是借助Intent，而接收广播的方式则是需要Broadcast Receiver。

1. 广播类型

标准广播：对所有的广播接收器同时发出，无法被截断

有序广播：同一时刻只有一个广播接收器能够收到这条广播信息，可以被截断

1. 动态注册广播

创建广播接收器，让它继承自BroadcastReceiver，并重写父类的onReceive（）方法，当有广播时，具体的逻辑就可以在这个方法中得到处理。

我们的广播接受器想要监听什么广播就在Intent里添加相应的action，然后调用registerReceiver（广播对象，intentFilter对象）方法，最后，动态注册的广播一定都要取消注册，可以（根据具体需要）在onDestory（）中调用unregisterReceiver（）方法来实现。

1. 静态注册广播

在AndroidManifest.xml文件中注册使用，在<intent-filter>标签里添加相应的action。

注意：不要在onReceive（）方法中添加过多的逻辑或者进行任何耗时操作，因为在广播接收器中是不允许开启线程的，广播接收器跟多的是扮演一种打开程序其他组件的角色。

1. 发送广播

方法：Intent intent = new Intent(“action名”)；

sendBroadcast(intent); //标准广播

sendOrderedBroadcast(intent); //有序广播，可以在注册广播接收器的时候设置优先级android：priority = “100”

可以自Intent中携带一些数据传递给广播接收器，有序广播可以在onReceive（）方法中调用abortBroadcast（）方法，截断广播。

考虑到安全性的问题，可以发生本地广播，防止其他应用程序截获敏感数据，方法是通过LocalBroadcastManager的getInstance（）方法得到它的一个实例，接受广播时调用它的registerReceiver（）方法，发送广播时调用它的sendBroadcast（）方法。

**Part 2：服务详解**

1. 什么是服务

（1）Service是Android中一个类，它是Android四大组件之一

（2）使用Service可以在后台执行长时间的操作（ perform long-running operations in the background ），Service并不与用户产生UI交互。

（3）其他的应用组件可以启动Service，即便用户切换了其他应用，启动的Service仍可在后台运行。

（4）一个组件可以与Service绑定并与之交互，甚至是跨进程通信（IPC）。例如，一个Service可以在后台执行网络请求、播放音乐、执行文件读写操作或者与 content provider交互等。

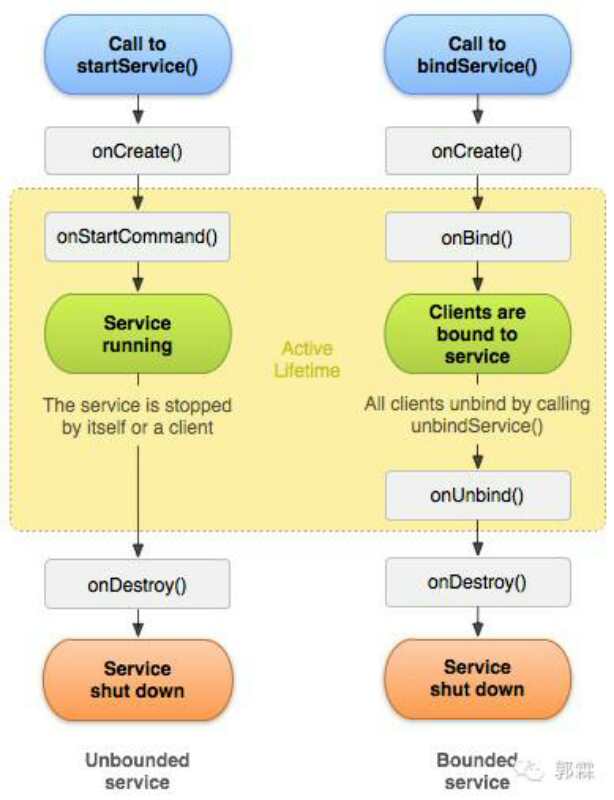
2.服务的启动方式

Started：其他组件调用startService()方法启动一个Service。一旦启动，Service将一直运行在后台（run in the background indefinitely）即便启动Service的组件已被destroy。通常，一个被start的Service会在后台执行单独的操作，也并不给启动它的组件返回结果。比如说，一个start的Service执行在后台下载或上传一个文件的操作，完成之后，Service应自己停止。

Bound：其他组件调用bindService()方法绑定一个Service。通过绑定方式启动的Service是一个client-server结构，该Service可以与绑定它的组件进行交互。一个bound service仅在有组件与其绑定时才会运行（A bound service runs only as long as another application component is bound to it），多个组件可与一个service绑定，service不再与任何组件绑定时，该service会被destroy。

当然，service也可以同时在上述两种方式下运行。这涉及到Service中两个回调方法的执行：onStartCommand()（通过start方式启动一个service时回调的方法）、onBind()（通过bind方式启动一个service回调的方法）。

1. 服务的生命周期



1. Sever各方法详解

为了创建Service，需要继承Service类。并重写它的回调方法，这些回调方法反应了Service的生命周期，并提供了绑定Service的机制。最重要的Service的生命周期回调方法如下所示：

onStartCommand()：当其他组件调用startService()方法请求启动Service时，该方法被回调。一旦Service启动，它会在后台独立运行。当Service执行完以后，需调用stopSelf() 或 stopService()方法停止Service。

onBind()：当其他组件调用bindService()方法请求绑定Service时，该方法被回调。该方法返回一个IBinder接口，该接口是Service与绑定的组件进行交互的桥梁。若Service未绑定其他组件，该方法应返回null。

onCreate()：当Service第一次创建时，回调该方法。该方法只被回调一次，并在onStartCommand() 或 onBind()方法被回调之前执行。若Service处于运行状态，该方法不会回调。

onDestroy()：当Service被销毁时回调，在该方法中应清除一些占用的资源，如停止线程、接触绑定注册的监听器或broadcast receiver 等。该方法是Service中的最后一个回调。

**Part 3：通知和权限**

1. 什么是通知

当某个应用程序希望向用户发出一些提示信息，而该应用程序又不在前台运行时，就可以借助通知来实现。

1. 通知的使用

通知可以设置非常多属性来满足不同场景的需求。示例代码如下：



1. 运行时权限

每个安卓版本对于运行时权限的处理有所差别，有时候权限问题没处理好会出现可怕的闪退现象。

步骤：①在AndroidManifest.xml文件中声明所需权限

②判断用户是否已经授权该权限给应用，借助的是ContextCompat.checkSelfPermission()方法

③如果没有申请的话，则需要调用ActivityCompat.requestPermissions()方法来想用户申请授权

④回调onRequestPermissionsResult()方法，授权结果会封装在grantResults参数中，判断授权结果，执行相应逻辑