服务在其托管进程的主线程中运行，它既**不**创建自己的线程，也**不**在单独的进程中运行（除非另行指定）。

1. 启动服务的两种方式

* startService
* bindService

当应用组件通过调用 bindService() 绑定到服务时，服务即处于“绑定”状态。绑定服务提供了一个客户端-服务器接口，允许组件与服务进行交互、发送请求、获取结果，甚至是利用进程间通信 (IPC) 跨进程执行这些操作。 仅当与另一个应用组件绑定时，绑定服务才会运行。 多个组件可以同时绑定到该服务，但全部取消绑定后，该服务即会被销毁。

为了避免浪费系统资源和消耗电池电量，应用必须在工作完成之后停止其服务。 如有必要，其他组件可以通过调用 [stopService()](https://developer.android.com/reference/android/content/Context.html" \l "stopService(android.content.Intent)) 来停止服务。*即使为服务启用了绑定，一旦服务收到对 [onStartCommand()](https://developer.android.com/reference/android/app/Service.html" \l "onStartCommand(android.content.Intent, int, int)) 的调用，您始终仍须亲自停止服务。*

1. 常用回调方法

* onStartCommand()

当另一个组件（如 Activity）通过调用 startService() 请求启动服务时，系统将调用此方法。一旦执行此方法，服务即会启动并可在后台无限期运行。 如果您实现此方法，则在服务工作完成后，需要由您通过调用 stopSelf() 或 stopService() 来停止服务。（如果您只想提供绑定，则无需实现此方法。）

* onBind()

当另一个组件想通过调用 bindService() 与服务绑定（例如执行 RPC）时，系统将调用此方法。在此方法的实现中，您必须通过返回 IBinder 提供一个接口，供客户端用来与服务进行通信。请务必实现此方法，但如果您并不希望允许绑定，则应返回 null。

* onCreate()

首次创建服务时，系统将调用此方法来执行一次性设置程序（在调用 onStartCommand() 或 onBind() 之前）。如果服务已在运行，则不会调用此方法。

* onDestroy()

当服务不再使用且将被销毁时，系统将调用此方法。服务应该实现此方法来清理所有资源，如线程、注册的侦听器、接收器等。 这是服务接收的最后一个调用。

*如果组件通过调用 startService() 启动服务（这会导致对 onStartCommand() 的调用），则服务将一直运行，直到服务使用 stopSelf() 自行停止运行，或由其他组件通过调用 stopService() 停止它为止。*

*如果组件是通过调用 bindService() 来创建服务（且未调用 onStartCommand()，则服务只会在该组件与其绑定时运行。一旦该服务与所有客户端之间的绑定全部取消，系统便会销毁它。*

*如果通过startService启动服务并且有组件绑定了服务或者绑定后调用了startService，只有所有组件取消绑定后stopService或者调用stopService后所有组件取消绑定才会停止服务。*

1. IntentService

这是 [Service](https://developer.android.com/reference/android/app/Service.html) 的子类，它使用工作线程逐一处理所有启动请求。如果您不要求服务同时处理多个请求，这是最好的选择。 您只需实现 [onHandleIntent()](https://developer.android.com/reference/android/app/IntentService.html" \l "onHandleIntent(android.content.Intent)) 方法即可，该方法会接收每个启动请求的 Intent，使您能够执行后台工作。

IntentService 执行以下操作：

创建默认的工作线程，用于在应用的主线程外执行传递给 onStartCommand() 的所有 Intent。

创建工作队列，用于将 Intent 逐一传递给 onHandleIntent() 实现，这样您就永远不必担心多线程问题。

在处理完所有启动请求后停止服务，因此您永远不必调用 stopSelf()。

提供 onBind() 的默认实现（返回 null）。

提供 onStartCommand() 的默认实现，可将 Intent 依次发送到工作队列和 onHandleIntent() 实现。

1. onStartCommand的三种返回值

START\_NOT\_STICKY

如果系统在 onStartCommand() 返回后终止服务，则除非有挂起 Intent 要传递，否则系统不会重建服务。这是最安全的选项，可以避免在不必要时以及应用能够轻松重启所有未完成的作业时运行服务。

START\_STICKY

如果系统在 onStartCommand() 返回后终止服务，则会重建服务并调用 onStartCommand()，但不会重新传递最后一个 Intent。相反，除非有挂起 Intent 要启动服务（在这种情况下，将传递这些 Intent ），否则系统会通过空 Intent 调用 onStartCommand()。这适用于不执行命令、但无限期运行并等待作业的媒体播放器（或类似服务）。

START\_REDELIVER\_INTENT

如果系统在 onStartCommand() 返回后终止服务，则会重建服务，并通过传递给服务的最后一个 Intent 调用 onStartCommand()。任何挂起 Intent 均依次传递。这适用于主动执行应该立即恢复的作业（例如下载文件）的服务。

1. 绑定服务

onServiceDisconnected()

Android 系统会在与服务的连接意外中断时（例如当服务崩溃或被终止时）调用该方法。当客户端取消绑定时，系统“不会”调用该方法。

只有 Activity、服务和内容提供程序可以绑定到服务 — 您无法从广播接收器绑定到服务。