**进程知识**

**参考：<https://developer.android.com/guide/components/processes-and-threads.html?hl=zh-cn#Threads>**

**https://dev.qq.com/topic/57ac4a0ea374c75371c08ce8**

**进程生命周期**

**Android系统将尽**量长时**间的保持应用进程，但为了新建进程或运行更重要的进程，最终要移除旧的进程来回收内存。**为了确定保留或终止哪些进程，系统会根据进程中正在运行的组件以及这些组件的状态，将每个进程放入“重要性层次结构”中。 必要时，系统会首先消除重要性最低的进程，然后是清除重要性稍低一级的进程，依此类推，以回收系统资源。

**进程优先级**

“重要性层次结构”一共分为5级，也叫作进程的优先级，从上至下依次变低：

* **前台进程**

用户当前操作所必需的进程。如果一个进程满足以下任一条件，即视为前台进程：

1. 托管用户正在交互的 [Activity](https://developer.android.com/reference/android/app/Activity.html?hl=zh-cn)（已调用 [Activity](https://developer.android.com/reference/android/app/Activity.html?hl=zh-cn) 的 [onResume()](https://developer.android.com/reference/android/app/Activity.html?hl=zh-cn" \l "onResume()) 方法）
2. 托管某个 [Service](https://developer.android.com/reference/android/app/Service.html?hl=zh-cn)，后者绑定到用户正在交互的 Activity
3. 托管正在“前台”运行的 [Service](https://developer.android.com/reference/android/app/Service.html?hl=zh-cn)（服务已调用 [startForeground()](https://developer.android.com/reference/android/app/Service.html?hl=zh-cn" \l "startForeground(int, android.app.Notification))）
4. 托管正执行一个生命周期回调的 [Service](https://developer.android.com/reference/android/app/Service.html?hl=zh-cn)（[onCreate()](https://developer.android.com/reference/android/app/Service.html?hl=zh-cn" \l "onCreate())、[onStart()](https://developer.android.com/reference/android/app/Service.html?hl=zh-cn" \l "onStart(android.content.Intent, int)) 或 [onDestroy()](https://developer.android.com/reference/android/app/Service.html?hl=zh-cn" \l "onDestroy())）
5. 托管正执行其 [onReceive()](https://developer.android.com/reference/android/content/BroadcastReceiver.html?hl=zh-cn" \l "onReceive(android.content.Context, android.content.Intent)) 方法的 [BroadcastReceiver](https://developer.android.com/reference/android/content/BroadcastReceiver.html?hl=zh-cn)

通常，在任意给定时间下前台进程数量不会很多。只有在内存不足以支持它们同时继续运行这一万不得已的情况下，系统才会终止它们。 此时，设备往往已达到内存分页状态，因此需要终止一些前台进程来确保用户界面正常响应。

* **可见进程**

没有任何前台组件、但仍会影响用户在屏幕上所见内容的进程。 如果一个进程满足以下任一条件，即视为可见进程：

1. 托管不在前台、但仍对用户可见的 [Activity](https://developer.android.com/reference/android/app/Activity.html?hl=zh-cn)（已调用其 [onPause()](https://developer.android.com/reference/android/app/Activity.html?hl=zh-cn" \l "onPause()) 方法）的进程。例如，如果前

台 Activity 启动了一个对话框，允许在其后显示上一 Activity，则有可能会发生这种情况。

1. 托管绑定到可见（或前台）Activity 的 [Service](https://developer.android.com/reference/android/app/Service.html?hl=zh-cn)的进程。

可见进程被视为是极其重要的进程，除非为了维持所有前台进程同时运行而必须终止，否则系统不会终止这些进程。

* **服务进程**

1. 正在运行已使用 [startService()](https://developer.android.com/reference/android/content/Context.html?hl=zh-cn" \l "startService(android.content.Intent)) 方法启动的服务且不属于上述两个更高类别进程的进程。

尽管服务进程与用户所见内容没有直接关联，但是它们通常在执行一些用户关心的操作（例如，在后台播放音乐或从网络下载数据）。因此，除非内存不足以维持所有前台进程和可见进程同时运行，否则系统会让服务进程保持运行状态。

* **后台进程**

1. 包含目前对用户不可见的 Activity 的进程（已调用 Activity 的 [onStop()](https://developer.android.com/reference/android/app/Activity.html?hl=zh-cn" \l "onStop()) 方法）。

这些进程对用户体验没有直接影响，系统可能随时终止它们，以回收内存供前台进程、可见进程或服务进程使用。 通常会有很多后台进程在运行，因此它们会保存在 LRU （最近最少使用）列表中，以确保包含用户最近查看的 Activity 的进程最后一个被终止。如果某个 Activity 正确实现了生命周期方法，并保存了其当前状态，则终止其进程不会对用户体验产生明显影响，因为当用户导航回该 Activity 时，Activity 会恢复其所有可见状态。 有关保存和恢复状态的信息，请参阅 [Activity](https://developer.android.com/guide/components/activities.html?hl=zh-cn" \l "SavingActivityState)文档。

* **空进程**

1. 不含任何活动应用组件的进程。

保留这种进程的的唯一目的是用作缓存，以缩短下次在其中运行组件所需的启动时间。 为使总体系统资源在进程缓存和底层内核缓存之间保持平衡，系统往往会终止这些进程。

**进程优先级的确定**

1. 进程的优先级评定是根据这个进程可以到达最高的级别来最终确定的。为使总体系统资源在进程缓存和底层内核缓存之间保持平衡，系统往往会终止这些进程。
2. 此外，一个进程的级别可能会因其他进程对它的依赖而有所提高，即服务于另一进程的进程其级别永远不会低于其所服务的进程。 例如，如果进程 A 中的内容提供程序为进程 B 中的客户端提供服务，或者如果进程 A 中的服务绑定到进程 B 中的组件，则进程 A 始终被视为至少与进程 B 同样重要。

**进程回收策略**

Android 中对于内存的回收，主要依靠 Lowmemorykiller 来完成，是一种根据 OOM\_ADJ 阈值级别触发相应力度的内存回收的机制。在 Lowmemorykiller 回收内存时会根据进程的级别优先杀死 OOM\_ADJ 比较大的进程，对于优先级相同的进程则进一步受到进程所占内存和进程存活时间的影响。如果一个进程中不包含任何组件，这个进程就认为是空进程，例如一个应用只有一个Activity，当这个Activity销毁时，该进程就变成了一个空进程；当Android结束一个进程时，并不会将它立即从系统中删除，而是将它标记为cache进程，当再次启动新进程时，会优先使用cache进程，这样可以加快应用启动速度。下面是OOM\_ADJ 阈值级别：

OOM\_ADJ < 0：表示非Android进程（纯Linux进程）。

0 <= OOM\_ADJ < 4：表示不容易被杀死的Android进程。

OOM\_ADJ >= 4：容易被杀死的Android进程。



Android 手机中进程被杀死可能有如下情况：

