**[Java多线程的6种状态](http://www.cnblogs.com/mindandaction/p/4921678.html)**

线程可以有以下6种状态：

1. New(新建)

    当用new操作符创建一个新线程时，如 new Thread(r), 该线程还没有开始运行。这意味着它的状态是new。

    当一个线程处在new状态，程序还没有开始运行线程中的代码。在线程运行之前还有一些基础工作要做。

2. Runnable(可运行)

   一旦调用start()方法，线程就处于runnable状态。可以可运行的线程可能正在运行也可能没有运行，这取决于操作系统给线程提供运行的时间(这就是为什么这个状态成为可运行而不是运行)

   事实上，运行中的线程被中断，目的是为了让他们线程获得运行机会。线程调度的细节依赖于操作系统提供的服务。抢占式调度系统给每一个可运行线程一个时间片来执行任务。当时间片用完，操作系统剥夺该线程的运行权，并给另一个线程可运行机会。当选择下一个线程时，操作系统考虑线程的优先级。

3. Blocked(被阻塞)

|

4. Waiting(等待)

|

5.Timed Waiting(计时等待)

   当线程处于被阻塞或等待状态时，它暂时不活动。它不运行任何代码且消耗最少的资源。直到线程调度器重新激活它。细节取决于它是怎样达到非活动状态的。

   \* 当一个线程试图获取一个内部的对象锁(而不是java.util.concurrent库里的锁), 而该锁被其他线程持有，则该线程进入阻塞状态。当其他线程释放该锁，并且线程调度器允许本线程持有它的时候，该线程将变成非阻塞状态。

   \* 当线程通知另一个线程通知调度器一个条件时，它自己进入等待状态。在调用Object.wait方法或Thread.join方法，或者是等待java.util.concurrent库中的Lock或Condition时，就会出现这种情况。实际上，被阻塞状态与被等待状态是有很大不同的。

   \* 有几个方法有一个超时参数。调用它们导致线程进入计时等待(timed waiting)状态。这一状态将一直保持到超时期满或者接收到适当的通知。带有超时参数的方法有Thread.sleep和Object.wait, Thrad.join, Lock.tryLock以及Condition.await的计时版。

6. Terminated(被终止)

   线程因如下两个原因之一而被终止:

   1) 因为run方法正常退出而自然死亡

   2) 因为一个没有捕获的异常终止了run方法二意外死亡