# 线程中断的几种方式

[https://mp.weixin.qq.com/s?\_\_biz=MzA5MTkxMDQ4MQ==&mid=2648933111&idx=1&sn=0a3592e41e59d0ded4a60f8c1b59e82e&chksm=88621ac9bf1593df5f8342514d6750cc8a833ba438aa208cf128493981ba666a06c4037d84fb&token=2041017112&lang=zh\_CN&scene=21#wechat\_redirect](https://mp.weixin.qq.com/s?__biz=MzA5MTkxMDQ4MQ==&mid=2648933111&idx=1&sn=0a3592e41e59d0ded4a60f8c1b59e82e&chksm=88621ac9bf1593df5f8342514d6750cc8a833ba438aa208cf128493981ba666a06c4037d84fb&token=2041017112&lang=zh_CN&scene=21" \l "wechat_redirect)

本文主要探讨一下中断线程的几种方式。

## 通过一个变量控制线程中断

代码：



代码中启动了一个线程，线程的run方法中有个死循环，内部通过exit变量的值来控制是否退出。 TimeUnit.SECONDS.sleep(3);让主线程休眠3秒，此处为什么使用TimeUnit？TimeUnit使用更方便一些，能够很清晰的控制休眠时间，底层还是转换为Thread.sleep实现的。程序有个重点：volatile关键字，exit变量必须通过这个修饰，如果把这个去掉，程序无法正常退出。volatile控制了变量在多线程中的可见性，关于volatile前面的文章中有介绍，此处就不再说了。

## 通过线程自带的中断标志控制

示例代码：



运行上面的程序，程序可以正常结束。线程内部有个中断标志，当调用线程的interrupt()实例方法之后，线程的中断标志会被置为true，可以通过线程的实例方法isInterrupted()获取线程的中断标志。

## 线程阻塞状态中如何中断？

示例代码：



运行上面代码，发现程序无法结束。

在此先补充几点知识：

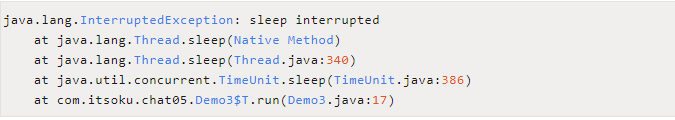
1、调用线程的interrupt()实例方法，线程的中断标志会被置为true

2、当线程处于阻塞状态时，调用线程的interrupt()实例方法，线程内部会触发InterruptedException异常，并且会清除线程内部的中断标志（即将中断标志置为false）

那么上面代码可以调用线程的interrupt()方法来引发InterruptedException异常，来中断sleep方法导致的阻塞，调整一下代码，如下：



运行结果：



程序可以正常结束了，分析一下上面代码，注意几点：

1、main方法中调用了t.interrupt()方法，此时线程t内部的中断标志会置为true

2、然后会触发run()方法内部的InterruptedException异常，所以运行结果中有异常输出，上面说了，当触发InterruptedException异常时候，线程内部的中断标志又会被清除（变为false），所以在catch中又调用了this.interrupt();一次，将中断标志置为true

1. run()方法中通过this.isInterrupted()来获取线程的中断标志，退出循环（break）

## 总结

1、当一个线程处于被阻塞状态或者试图执行一个阻塞操作时，可以使用 Thread.interrupt()方式中断该线程，注意此时将会抛出一个InterruptedException的异常，同时中断状态将会被复位(由中断状态改为非中断状态)

2、内部有循环体，可以通过一个变量来作为一个信号控制线程是否中断，注意变量需要volatile修饰

3、文中的几种方式可以结合起来灵活使用控制线程的中断。