# 第10章

1. 线程和进程的区别是什么？

所谓“进程”，是一个自包含的执行程序，每个进程含有自己独立的地址空间和系统资源。多任务的操作系统通过周期性地将CPU的时间分配给不同的任务而达到“同时”运行多个进程（程序）的目的。

将上述多任务的原理应用到程序的更低一层中进行发展就是多线程。“线程（Thread）”是线程控制流的简称，它是进程内部的一个控制序列流，一个进程中可以具有多个线程，每个线程执行不同的任务，而这些线程也可以像多个进程一样并发执行。能够同时运行两个或两个以上线程的程序叫做多线程程序。

多线程和多进程之间是有区别的，最基本的区别是每个进程都有属于自己的代码和数据空间，而线程则共享这些这些数据。

1. 创建线程的方式有哪两种？如何启动一个线程？

可以通过两种方式创建一个线程：一种是扩展Thread类；另一种是实现Runnable接口。通过start()方法启动线程。

1. 线程有哪几种状态？状态之间是如何转换的？

* 新建状态（New Thread）
* 可运行状态（Runnable）
* 运行状态（Running）
* 阻塞状态（Blocked）
* 死亡状态（Dead）

当使用New关键字新创建一个线程时，这时线程处于新建状态（New Thread）。一旦一个线程调用了start()方法，线程就进入可运行状态（Runnable）。处于可运行状态（Runnable）的线程，一旦得到了时间片，就进入了运行状态（Running）。导致一个线程进入阻塞状态可能有如下几个原因：

1) 线程调用了sleep()方法；

2) I/O流中发生了线程阻塞；

3) 线程调用了wait()方法；

4) 线程调用了suspend()方法（已过时）；

5) 线程要锁住一个对象，但该对象已被另一个线程锁住。

一个处于可运行状态的线程如果发上上述几种情况，就会进入到阻塞状态。处于阻塞状态的线程要想重新回到可运行状态，需要分别满足不同的特定条件：

1) 如果由于调用sleep()而进入阻塞状态，则当睡眠时间到达所规定的毫秒数，就可以离开该状态；

2) 如果由于等待I/O操作，那么当这个I/O操作完成后，就可以离开该状态；

3) 如果由于调用了wait()方法，那么只有等另一个线程调用了notify()或notifyAll()方法后，才能离开该状态；

4) 如果由于调用suspend()方法被挂起，则需调用resume()后才能离开该状态；

5) 如果正在等待其他线程拥有的对象锁，只有该线程放弃锁后，才能离开该状态。

线程进入死亡状态一般有两种原因：

1) 线程执行完，自然死亡。

2) 异常终止run()方法（如调用stop()方法，stop()方法已过时）。

1. 什么叫“临界资源”？什么叫“临界区”？什么叫“同步方法”？

我们把这种一次只允许一个线程使用的资源称为“临界资源”，访问这种临界资源的代码称为“临界区”。Java中采用对象锁的机制来处理临界区的互斥问题。Java引入了synchronized关键字，可以用来修饰实例方法，一旦一个方法被该关键字修饰，这个方法被称为“同步方法”。同步方法在其执行期间是不会被中断的。

1. Java中的同步方法是如何处理临界区的互斥问题的？

每个对象具有一把锁，我们将其称为“对象锁”，当线程没有访问同步方法时，对象锁的状态是打开的。如果线程要进入对象中的同步方法，即被synchronized修饰的实例方法，需要先检查其对象锁的状态，如果锁是打开的，则线程可以进入该同步方法，同时关闭对象锁，此时线程持有该对象锁。如果这时其他线程也要访问该同步方法，就会进入阻塞状态，到对象的锁等待池中等待，直到前面的线程退出同步方法，释放对象锁后才能继续执行。