# 多线程基础

## 相关概念

进程是一个应用程序在处理机上的一次执行过程，线程是进程的一部分。

多线程的优势：

进程之间不能共享内存，线程之间共享内存非常容易。

线程比进程代价小。

并发和并行

并行：多个CPU同时执行一段处理逻辑。是真正的同时。

并发：cpu调度。

线程安全：指并发时，代码经多线程使用，结果不受影响。

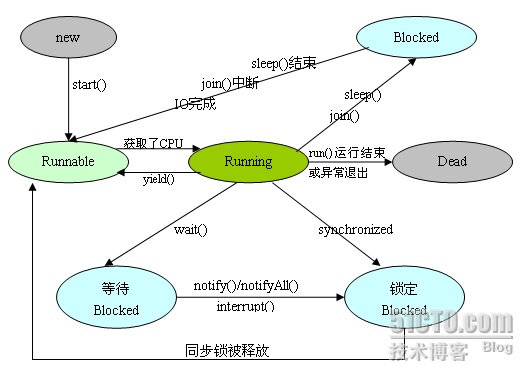
同步：人为控制和调度，保证共享资源时运行结果准确。

线程安全的优先级高于性能。

## 线程的状态

java.lang.Thread.State线程状态

1. NEW:  
   至今尚未启动的线程的状态。
2. RUNNABLE:  
   可运行线程的线程状态。处于可运行状态的某一线程正在 Java 虚拟机中运行，但它可能正在等待操作系统中的其他资源，比如处理器。
3. BLOCKED ：  
   受阻塞并且正在等待监视器锁的某一线程的线程状态。处于受阻塞状态的某一线程正在等待监视器锁，以便进入一个同步的块/方法，或者在调用 Object.wait 之后再次进入同步的块/方法。
4. WAITING ：  
   某一等待线程的线程状态。某一线程因为调用下列方法之一而处于等待状态：  
   -》不带超时值的 Object.wait  
   -》不带超时值的 Thread.join  
   -》LockSupport.park  
   处于等待状态的线程正等待另一个线程，以执行特定操作。 例如，已经在某一对象上调用了 Object.wait() 的线程正等待另一个线程，以便在该对象上调用 Object.notify() 或 Object.notifyAll()。已经调用了 Thread.join() 的线程正在等待指定线程终止。
5. TIMED\_WAITING  
   具有指定等待时间的某一等待线程的线程状态。某一线程因为调用以下带有指定正等待时间的方法之一而处于定时等待状态：  
   -》Thread.sleep  
   -》带有超时值的 Object.wait  
   -》带有超时值的 Thread.join  
   -》LockSupport.parkNanos  
   -》LockSupport.parkUntil
6. TERMINATED  
   已终止线程的线程状态。线程已经结束执行。



# 线程同步

synchronized

synchronized, wait, notify 是任何对象都具有的同步工具, wait/notify必须存在于synchronized块中