多线程

多线程深入(参考http://www.cnblogs.com/wxd0108/p/5479442.html)

一. 线程与进程

二. 多线程

1. 定义:指的是这个程序（一个进程）运行时产生了不止一个线程.

并行与并发：

- 并行：多个cpu实例或者多台机器同时执行一段处理逻辑，是真正的同时。
- 并发：通过cpu调度算法，让用户看上去同时执行，实际上从cpu操作层面不是真正的同时。并发往往在场景中有公用的资源，那么针对这个公用的资源往往产生瓶颈，我们会用TPS或者QPS来反应这个系统的处理能力。
- 线程安全：经常用来描绘一段代码。指在并发的情况之下，该代码经过多线程使用，线程的调度顺序不影响任何结果。这个时候使用多线程，我们只需要关注系统的内存，cpu是不是够用即可。反过来，线程不安全就意味着线程的调度顺序会影响最终结果
- 同步：Java中的同步指的是通过人为的控制和调度，保证共享资源的多线程访问成为线程安全，来保证结果的准确。如上面的代码简单加入`@synchronized`关键字。在保证结果准确的同时，提高性能，才是优秀的程序。线程安全的优先级高于性能。

2. 存在的意义:

用多线程只有一个目的，那就是更好的利用cpu的资源，因为所有的多线程代码都可以用单线程来实现。说这个话其实只有一半对，因为反应“多角色”的程序代码，最起码每个角色要给他一个线程吧，否则连实际场景都无法模拟，当然也没法说能用单线程来实现：比如最常见的“生产者，消费者模型”。

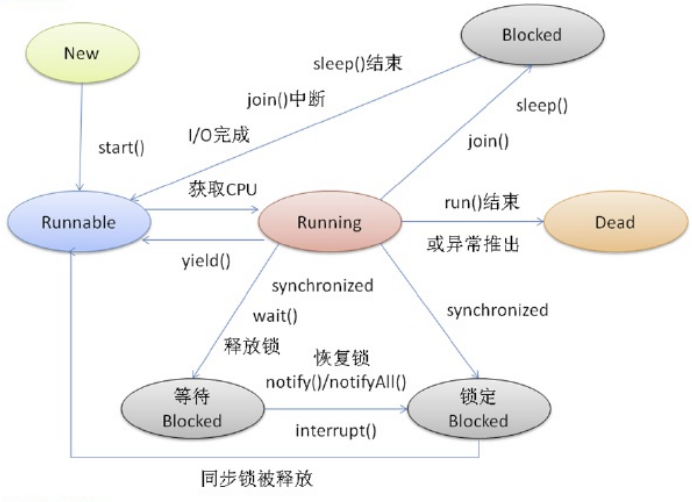
public static enum Thread.State
extends Enum<Thread.State>

A thread state. A thread can be in one of the following states:

- NEW
  A thread that has not yet started is in this state.
- RUNNABLE
  A thread executing in the Java virtual machine is in this state.
- BLOCKED
  A thread that is blocked waiting for a monitor lock is in this state.
- WAITING
  A thread that is waiting indefinitely for another thread to perform a particular action is in this state.
- TIMED_WAITING
  A thread that is waiting for another thread to perform an action for up to a specified waiting time is in this state.
- TERMINATED
  A thread that has exited is in this state.

A thread can be in only one state at a given point in time. These states are virtual machine states which do not reflect any operating system thread states.

线程状态



线程状态转换