链滴

高并发程序设计（3）—— 线程操作进阶篇

作者：wanglei0622

原文链接：https://ld246.com/article/1488524969478

来源网站：链滴

<h2>volatile（易变的）与Java内存模型(JMM)</h2>

Java内存模型都是围绕原子性、有序性、可见性展开的
volatile:可以保证数据操作的原子性，保证操作结束，才能被读取，告诉Java虚拟机，被修饰的量可能被别的线程修改。

可以解决long类型数据在32位操作系统上并发循环赋值读取时，读取数据错误的问题。
但是不能解决并发下不同的线程读取到共享资源同样的结果问题，比如多个线程对int i =0 ; i++ 加到10000为止，最终结果会小于10000，加上volatile也如此。
不能替代锁，不能保证并发下，共享资源的复合操作的原子性。

<h2>线程组</h2>

ThreadGroup:线程组

activeCount():获得活动线程的总数，但是线程是动态的，因此是个预估值，不准确
stop():和线程的stop()一样，不建议使用。
list():打印线程组内所有线程信息，对调试有帮助。

<h2>守护线程（Daemon）</h2>

垃圾回收线程，JIT线程就是守护线程，与之对应的就是用户线程，用户线程可以理解为系统工作程，它会完成这个程序应该要完成的业务操作。如果用户线程都结束了，以为只程序也无事可做了，护线程要守护的对象已经不在，那么整个应用就自然要结束了，在一个Java应用内，当只有守护线程Java虚拟机就会自然退出。
setDaemon(true):注意要在start()，之前设置，不然会抛线程状态异常，然后继续以用户线程执。设置不生效。

<h2>线程优先级</h2>

setPriority(): 1-10数字越大优先级越高，但建议在应用层解决线程调度问题，优先级是个概率问，和底层操作系统相关，并不严格保证执行机会。

<h2>线程安全和synchronized</h2>

线程安全是并行程序的根本和根基，volatile并不能抱枕线程安全，只能抱枕一个线程修改了数后，其他线程能看到改动，但当两个线程同时去修改时，有可能读到同样的值。

必须保证多个线程对数据操作时完全同步，A在写入时，B不能写，也不能读。

synchronized: 保证多个线程对数据操作时完全同步，A在写入时，B不能写，也不能读被限制的多个线程是串行的。

指定对象：进入同步代码前，要获得对象的锁。
指定方法：等同于指定对象，要先获得对象实例的锁。
指定静态方法：相当于给类加锁，当操作的是static数据时，static synchronized 法，如果不加static锁就加对象上了，获取的是对象锁，还是会并发。
不要给常量加同步，如 int integer 等，他们的赋值是一个新的引用，同步还作用在原来引用上。

<h2>无提示的错误</h2>

如int数字的加减乘除运算，如果值超出了4个字节的范围，结果就会错误，这就是溢出问题，但并没有错误日志。在并行程序中非常容易产生错误结果。

<h2>线程调试工具</h2>

jstack：jps命令显示当前系统所有的java进程， jstack打印指定进程的内部线程及堆栈。

