Java多线程系列---"基础篇"14之 wait,sleep,join,yield,park,unpark,notify等通信机制对比

1. 线程让步： yield()

yield()的作用是让步。它能让当前线程**由"运行状态"进入到"就绪状态"**，从而让其它具有相同优先级的等待线程获取执行权；但是，并不能保证在当前线程调用yield()之后，其它具有相同优先级的线程就一定能获得执行权；也有可能是当前线程又进入到"运行状态"继续运行！

2. yield() 与 wait()的比较

我们知道，wait()的作用是让当前线程由"运行状态"进入"等待(阻塞)状态"的同时，也会释放同步锁。而yield()的作用是让步，它也会让当前线程离开"运行状态"。它们的区别是：
(1) wait()是让线程由"运行状态"进入到"等待(阻塞)状态"，而不yield()是让线程由"运行状态"进入到"就绪状态"。
(2) wait()是会线程释放它所持有对象的同步锁，而yield()方法不会释放锁。

3. 线程休眠：sleep()
sleep() 定义在Thread.java中。
sleep() 的作用是让当前线程休眠，即当前线程会从"运行状态"进入到"休眠(阻塞)状态"。sleep()会指定休眠时间，线程休眠的时间会大于/等于该休眠时间；在线程重新被唤醒时，它会由"阻塞状态"变成"就绪状态"，从而等待cpu的调度执行。

4. sleep() 与 wait()的比较：
我们知道，wait()的作用是让当前线程**由"运行状态"进入"等待(阻塞)状态"的同时**，也会释放同步锁。而sleep()的作用是也是让当前线程**由"运行状态"进入到"休眠(阻塞)状态"**。
但是，wait()会释放对象的同步锁，而sleep()则不会释放锁。

5. sleep()与yield()的比较：

sleep和yield的区别在于，sleep可以使优先级低的线程得到执行的机会，而yield只能使同优先级的线程有执行的机会.

6. Object中的wait()和notify()
使用注意事项：

- 因为wait需释放锁，所以必须在synchronized中使用（没有锁时使用会抛出IllegalMonitorStateException）
- notify也要在synchronized使用，并且应该指定对象
- synchronized(),wait(),notify() 对象必须一致，一个synchronized()代码块中只能有1个线程wait()或notify()

7.LockSupport中的park() 和 unpark()

总结一下，LockSupport比Object的wait/notify有两大优势：

①LockSupport不需要在同步代码块里 。所以线程间也不需要维护一个共享的同步对象了，实现了线程间的解耦。说明：park和wait的区别。wait让线程阻塞前，必须通过synchronized获

取 同步锁。

②unpark函数可以先于park调用，所以不需要担心线程间的执行的先后顺序。

8.join

等待该线程终止。
等待调用join方法的线程结束，再继续执行。如：t.join();//主要用于等待t线程运行结束，若无此句，main则会执行完毕，导致结果不可预测。

9.interrupt

interrupt方法不会中断一个正在运行的线程.就是指线程如果正在运行的过程中, 去调用此方法是没有任何反应的. 为什么呢, 因为这个方法只是提供给 被阻塞的线程, 即当线程调用了.Object.wait, Thread.join, Thread.sleep三种方法之一的时候, 再调用interrupt方法, 才可以中断刚才的阻塞而继续去执行线程.