线程通讯

什么是线程通讯?

- 线程通信就是当多个线程共同操作共享的资源时，互相告知自己的状态，以避免资源争夺导致出现混乱。
  消息传递时常用 wait（休眠）/ notify(唤醒) / notifyAll 方法

wait（休眠）/ notify(唤醒) / notifyAll 方法的使用

- 1.wait(long timeout)让当前线程处于“等待(阻塞)状态”，“直到其他线程调用此对象的notify()方法或notifyAll() 方法，或者超过指定的时间量”，当前线程被唤醒(进入“就绪状态”)。
- 2.notify()和notifyAll()的作用是唤醒当前对象上的等待线程；notify()是唤醒单个线程，而 notifyAll() 是唤醒所有的线程。
- 3.wait 之前需要请求锁，而wait执行时会先释放锁，等被唤醒时再重新请求锁。
- 注意
  - 1. wait()方法只能在同步方法中或同步块中调用。如果调用wait()时，没有持有适当的锁，会抛出异常
  - 2. wait()方法执行后，当前线程释放锁，线程与其它线程竞争重新获取锁。
  - 3.一组 wait 和 notfiy / notfiyAll 必须是同一对象。
  - 4.notifyAll 只能唤醒当前对象的所有等待线程。

wait 为什么要加锁?

- 因为 wait 在使用时会暂时释放锁

wait 为什么要释放锁?

- 因为这样其他线程就可以获取到 wait 释放掉的对象锁，让给其他需要的人。不像 sleep 休眠的同时还一直占有着资源

Thread.leep() 和 Object.wait() 区别?

- 相同点:
  - 1. 都可以让当前线程休眠
  - 2. 都必须要处理一个 Interrupt 异常
- 不同点:
  - 1.wait 来自于 Object 中的一个方法： 而Sleep 来自于 Thread
  - 2.传参不同，wait可以没有参数，而 Sleep 必须又一个大于0的参数
  - 3.wait 之前需要请求锁，而wait执行时会先释放锁，等被唤醒时再重新请求锁。， sleep 使用时不需要加锁

- 4.wait 使用时会释放锁，而Sleep 不会释放锁

- 5.wait 默认不传参的情况下会进入 WAITING 状态， 而Sleep 会进入 TIME_WAITING
- 6. sleep(0) 立即触发一次 CPU 资源的抢占 wait(0) 永久的等待下去

为什么 wait 会放在 Object 中而不是 Thread?

- 它可以让同步方法或者同步块暂时放弃对象锁,而将它暂时让给其它需要对象锁的人(这里应该是程序块,或线程)用，是对对象级别的操作而不是线程级别（线程和锁的关系是一对多的关系，也就是一个线程可以拥有多把锁），为了灵活起见（一个线程当中会有多把锁），就把 wait 放到 Object 中了

LockSupport

因为 notify 唤醒唤醒的时候是随机唤醒的，所以可以使用 LockSupport

堵塞：LockSupport.park();

- 通过 native 方法来调用操作系统的原语，来将当前线程挂起

- 指定线程的唤醒：LockSupport.unpark(Thread t);

wait 和 LockSupport 区别

相同点：

- 1.两个都可以进行休眠
- 2.二者都可以传参或者不传参，并且二者线程状态也是一致的

不同点：

- 1. wait 必须配合 sy 一起使用（必须加锁），而 LockSupport 不许加锁
- 2. wait 只能唤醒性全部和随机的一个线程，而LockSupport 可以唤醒指定线程