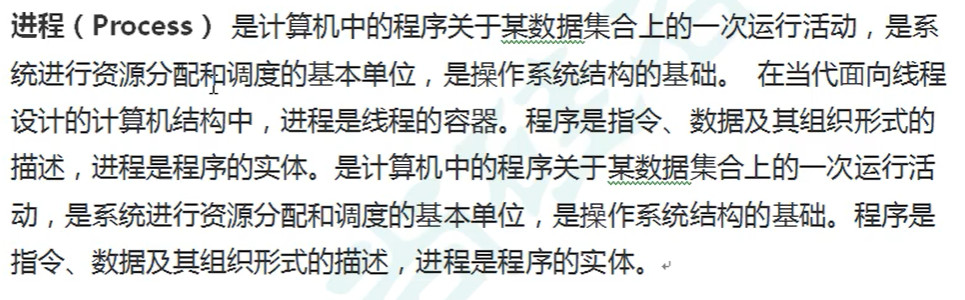
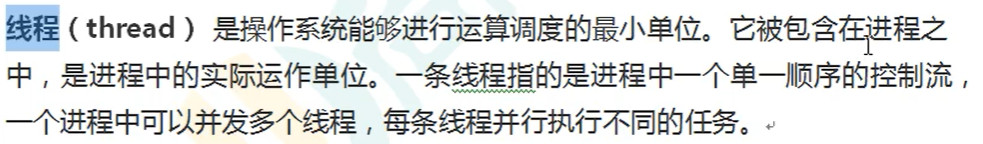
[信号量与管程 - kk\_kk - 博客园 (cnblogs.com)](https://www.cnblogs.com/kkkkkk/p/5543799.html)

[(2条消息) 信号量与管程\_fl\_sw的博客-CSDN博客](https://blog.csdn.net/qq_36391130/article/details/82809344)

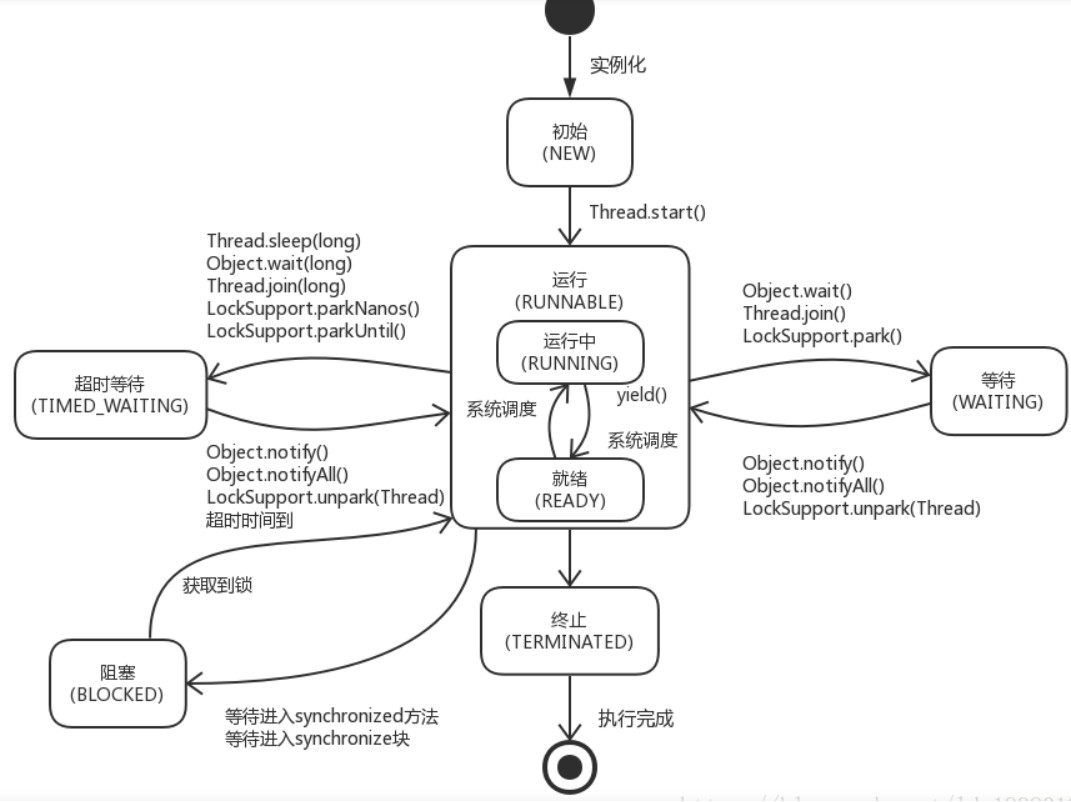


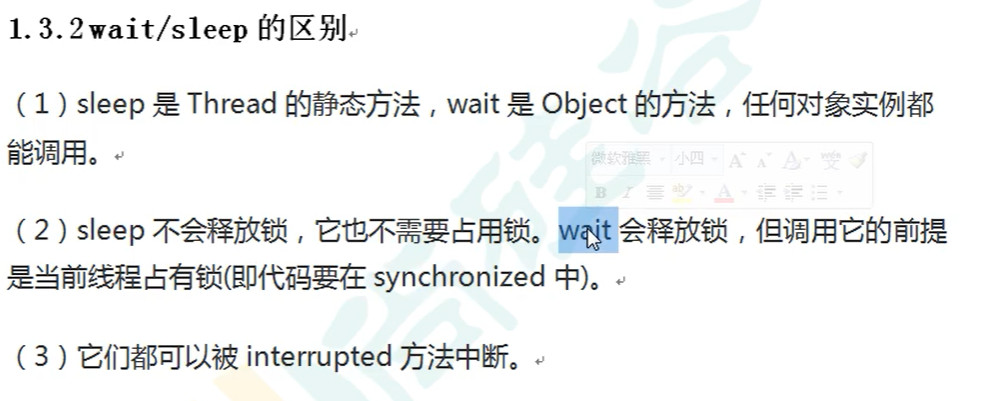


[(2条消息) Java 线程的6种状态（简单介绍）\_爱你の大表哥的博客-CSDN博客\_线程的6种状态](https://blog.csdn.net/ldx19980108/article/details/81675241)

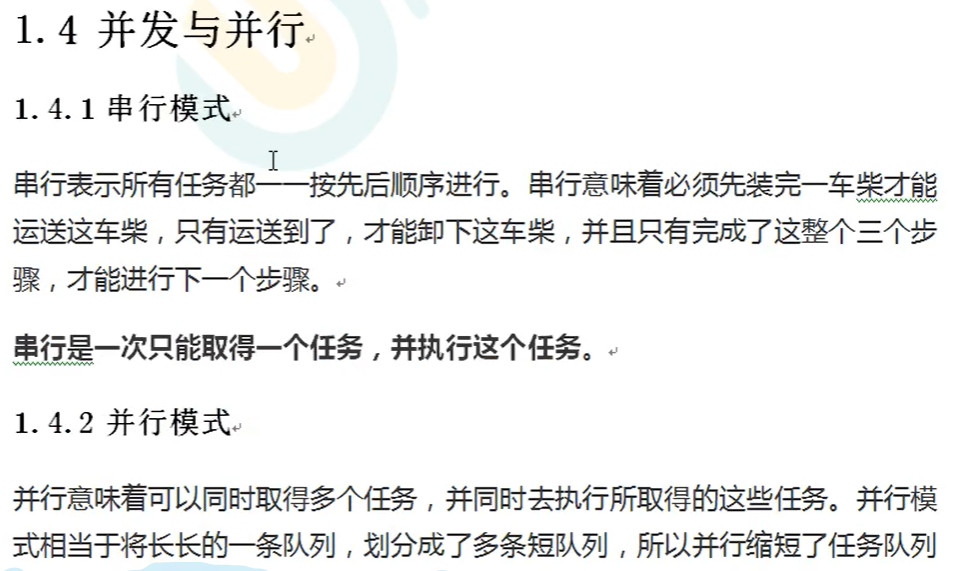
[Java线程状态 与 操作系统线程状态 - 它山之玉 - 博客园 (cnblogs.com)](https://www.cnblogs.com/codeclock/p/13803898.html)

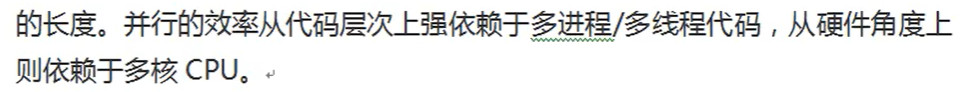


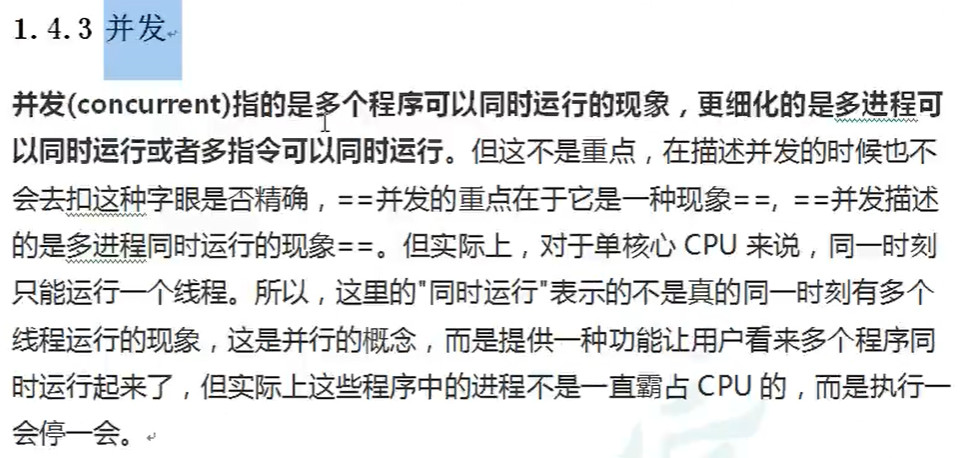


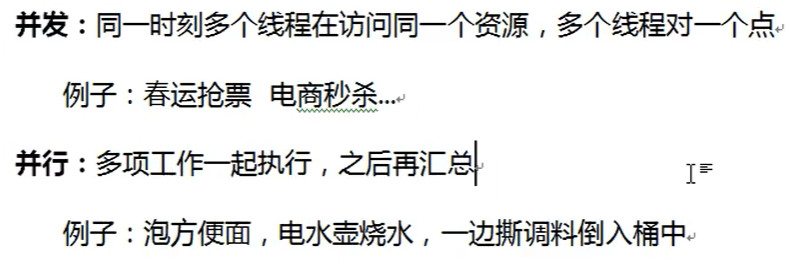


（每次等待都会保存线程的上下文，所以下次接着执行）









**（并发的场景就是多用户访问临界资源。要做到的目的就是：让表面上所有用户都使用到该资源。而具体使用怎样的实现方法达到这个目的。**

**现实场景：多用户访问同一服务器、访问同一程序、多线程请求cpu，多程序请求同一文件。。。。。。并发的概念提出是为了解决临界资源不够用，但又能让一定的用户都能正常使用。）**

**（先有资源不够用的问题->想出交替使用解决方法->取名为并发->使用学术性语言进行描述定义->我们一来就看定义不一定明白为什么）**

**（所谓的提高并发就是要提高在资源有限的情况下的保证用户正常使用资源的用户数。）**

**（并发的方式虽然能一定程度解决资源使用问题，但是也会有新的问题需要解决：比如每个用户的使用状态，对数据产生的影响。。。。。所以还需要一系列的解决方案，比如多线程的访问同一资源，因为随时可能切换，所以容易导致数据不一致性问题。这时候就有引出同步的概念）**

**（上面仅仅只是对并发抽象的描述，具体到场景、问题、其他需求，并发可能引出不一样的信问题，这就还需要考虑一些细节处理机制。比如多线程编程使用同步、比如分布式服务之间的通信、比如多台数据库服务器与应用服务器之间数据的一致性。。。。。。。。）**

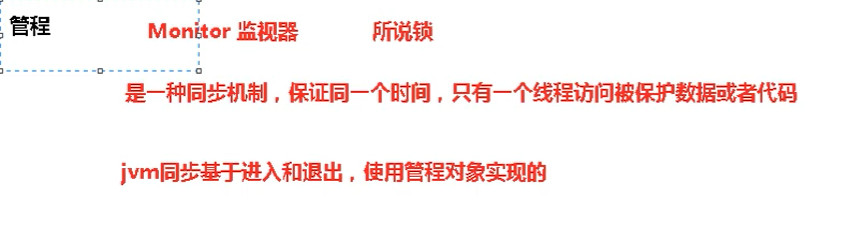
**（**[(2条消息) 面试官常说的多线程并发是要解决什么问题？\_xk4848123的博客-CSDN博客\_多线程是为了解决什么问题](https://blog.csdn.net/xk4848123/article/details/113077595)**）**

**（**[Java并发编程的本质是解决这三大问题 - 程序员薛师兄 - 博客园 (cnblogs.com)](https://www.cnblogs.com/onblog/p/13043365.html)**）**

管程(monitor-监控器，通俗说的锁)

管程(monitor)只是保证了同一时刻只有一个进程在管程内活动,即管程内定义的操作在同一时刻只被一个进程调用(由编译器实现).但是这样并不能保证进程以设计的顺序执行,因此需要设置condition变量,让进入管程而无法继续执行的进程阻塞自己.（这在Java中不就是synchronized的显式作用吗，对应到的不就是操作系统中的上开锁吗，所以它就是一种同步机制，那信号灯就对应着信号量。称呼区别而已。）

（再进一步回顾关于Java中的锁：每个对象都有唯一个monitor，而将这个对象作为锁，上锁的过程就是获取monitor的过程，释放锁的过程就是释放monitor。那么synchronized就是基于monitor进行实现。）



**用户线程与守护线程**：

