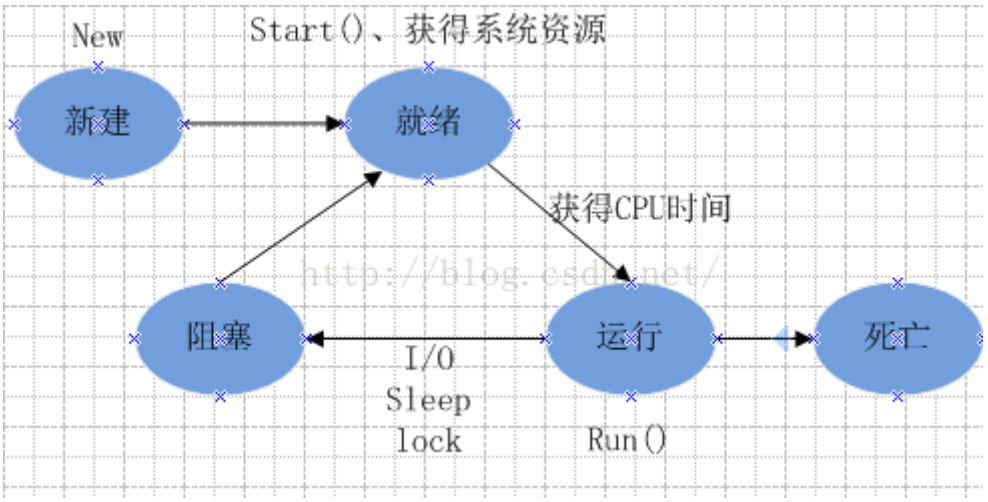
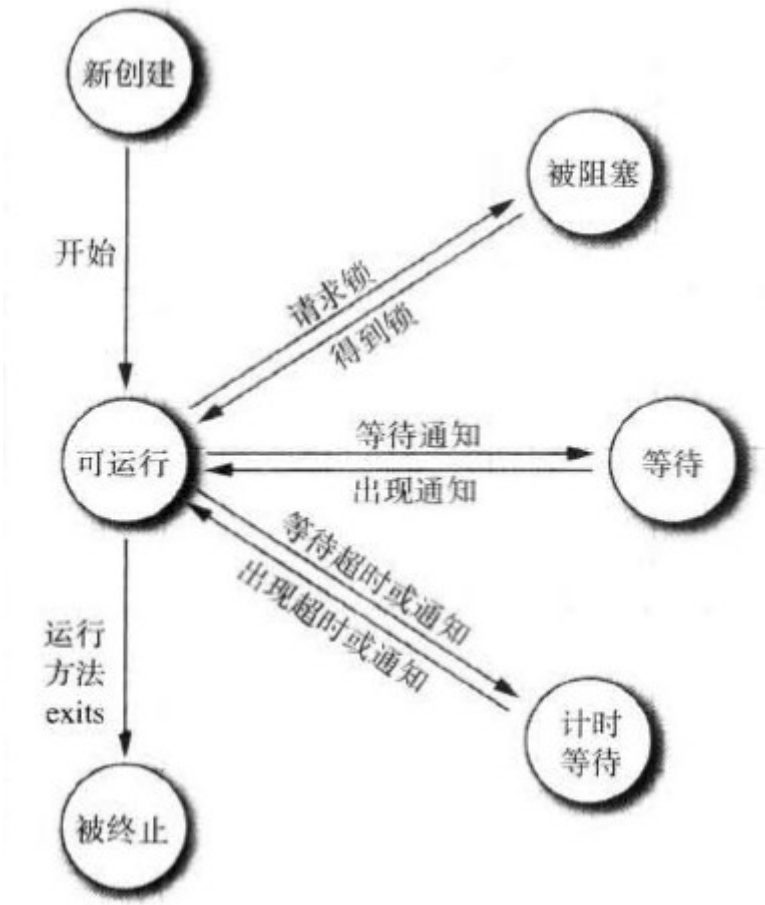
1.sleep 是thread类的方法,不会释放锁.

wait是Object的方法,会释放锁,进入等待队列,直到调用notify方法,才会进入锁池,进入运行状态.

线程的状态:





2.线程的五种状态及其转化:

1.New 新生状态 :创建一个新线程时,如New Thread(r),此时的状态就是新生状态.此时有了相应的内存空间和其他资源,但是还没有开始执行.

2.Runnable 就绪状态:新建对象后,调用start方法就可以启动线程.当线程启动之后,线程就进入就绪状态.由于没有分配cpu,线程进入线程队列,等待cpu服务.

当系统挑选一个等待执行的线程对象后,他就会从等待状态变为运行状态.

3.运行状态

当就绪状态的线程被调用并获得处理器资源时，线程就进入了运行状态。此时，自动调用该线程对象的 run()方法。

run()方法定义了该线程的操作和功能。运行状态中的线程执行自己的run方法中代码。直到调用其他方法或者发生阻塞

而终止。

4.阻塞状态

一个正在执行的线程在某些特殊情况下，如被人为挂起或需要执行耗时的输入输出操作时，将让出 CPU 并暂时中止

自己的执行，进入堵塞状态。在可执行状态下，如果调用 sleep()、 suspend()、 wait()等方法，线程都将进入堵塞状态。

堵塞时，线程不能进入排队队列，只有当引起堵塞的原因被消除后，线程转入就绪状态。重新到就绪队列中排队等待，

这时被CPU调度选中后会从原来停止的位置开始继续执行。

记住：阻塞被消除后是回到就绪状态，不是运行状态。

5、死亡状态

线程调用 stop()方法、destory()方法或 run()方法执行结束后，线程即处于死亡状态。处于死亡状态的线程不具有继续运行的能力。

常见问题：

1、造成线程阻塞的方法？

阻塞线程的方法：join、yield、sleep 和Object的wait()方法

2、Java的守护进程（后台进程）？

设置线程为后台进程运行：setDaemon(true) 如果一个进程中只有后台线程在运行，这个进程就会结束。

3、造成线程阻塞后，线程回到哪个状态了？

通过join、yield、sleep造成线程阻塞后是回到了就绪状态

3、哪些状态之后是回到就绪状态？

a)通过join、yield、sleep造成线程阻塞后是回到了就绪状态

b)遇到synchronized后

c)遇到Object的等待wait方法后

4、sleep会释放锁吗？

sleep不会释放锁【它会抱着锁睡觉】

5、线程都有哪些状态？具体是怎么运行的？

线程有：创建、就绪、运行、阻塞、终止。5种状态

1.通过new关键字创建后，进入到新生状态

2.调用start后进入就绪状态

3.CPU调度到本线程后，本线程开始执行。进入到运行状态

4.运行中遇到join，yield，sleep造成阻塞，进入阻塞状态。阻塞完成后，又回到就绪状态

5.线程正常执行完，或者遇到异常终止后，进入死亡状态

6、终止线程有哪几种方法？

线程调用 stop()方法、destory()方法或 run()方法执行结束后，线程即处于死亡状态。处于死亡状态的线程不具有继续运行的能力。

推荐使用boolen标识来停止线程.