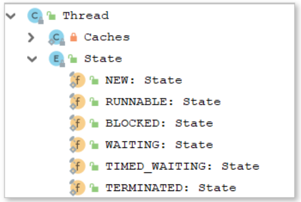
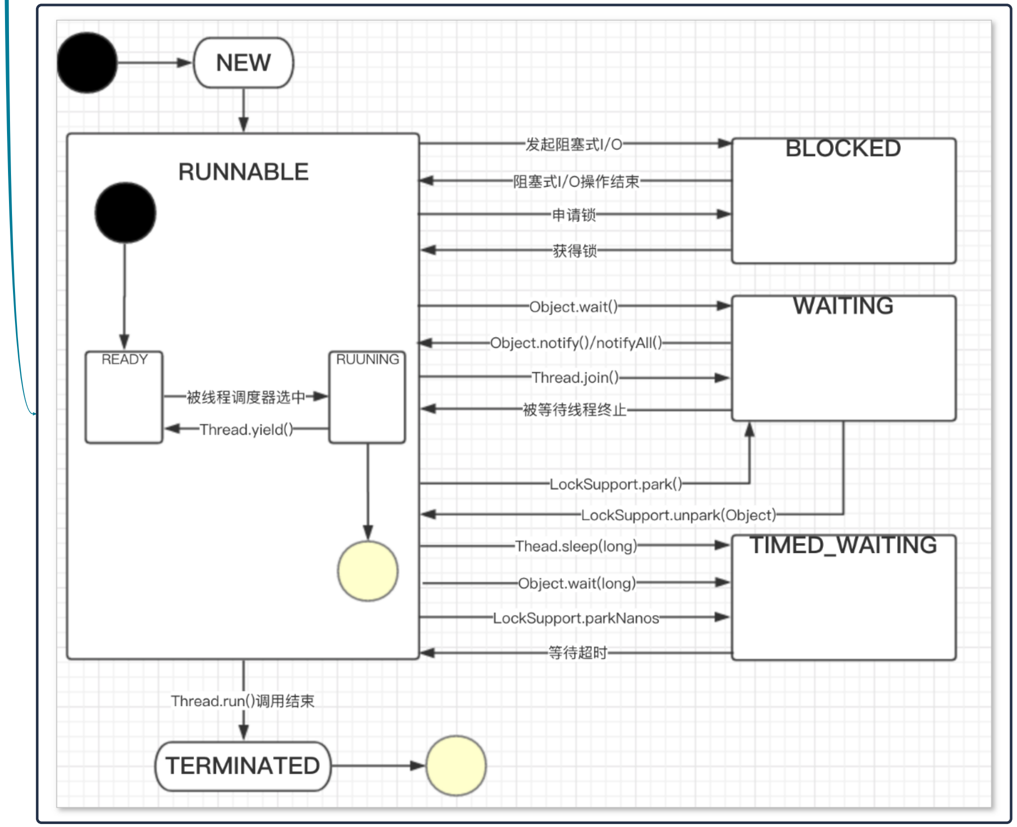
JAVA线程的状态或者生命周期

- **JAVA.LANG.THREAD.STATE一种有六种状态**
  - a. 新建（NEW）
    - 表示线程新建出来还没有被启动的状态，比如：Thread t = new MyThread();
  - b. 就绪/运行（RUNNABLE）
    - ：该状态包含了经典线程模型的两种状态：就绪(Ready)、运行(Running)：
  - c. 阻塞（BLOCKED）
    - 通常与锁有关系，表示线程正在获取有锁控制的资源，比如进入synchronized代码块，获取ReentryLock等；发起阻塞式IO也会阻塞，比如字符流字节流操作。
  - d. 等待（WAITING）
    - 线程在等待某种资源就绪。
  - e. 超时等待（TIMED_WAIT）
    - 线程进入条件和等待类似，但是它调用的是带有超时时间的方法。
  - f. 终止（TERMINATED）
    - 线程正常退出或异常退出后，就处于终结状态。也可以叫线程的死亡。
- **经典线程模型包含5种状态**
  - 新建、就绪、运行、等待、退出
- **经典线程的就绪、运行**
  - RUNNABLE
- **传统模型的等待状态**
  - BLOCKED、WAITING、TIMED_WAITING
- **进入等待状态**
  - 1. 当一个线程执行了Object.wait()的时候，它一定在等待另一个线程执行Object.notify()或者Object.notifyAll()。
  - 2. 一个线程thread，其在主线程中被执行了thread.join()的时候，主线程即会等待该线程执行完成。
  - 3. 当一个线程执行了LockSupport.park()的时候，其在等待执行LockSupport.unpark(thread)。
- **进入超时等待状态**
  - 该状态不同于WAITING，它可以在指定的时间后自行返回
    - 1. Object.wait(long)
    - 2. Thread.join(long)
    - 3. LockSupport.parkNanos()
    - 4. LockSupport.parkUntil()
    - 5. Thread.sleep(long)



```
Thread
  Caches
  State
    NEW: State
    RUNNABLE: State
    BLOCKED: State
    WAITING: State
    TIMED_WAITING: State
    TERMINATED: State
```


NEW
RUNNABLE
READY
RUUNING
被线程调度器选中
Thread.yield()
BLOCKED
发起阻塞式I/O
阻塞式I/O操作结束
申请锁
获得锁
WAITING
Object.wait()
Object.notify()/notifyAll()
Thread.join()
被等待线程终止
LockSupport.park()
LockSupport.unpark(Object)
TIMED_WAITING
Thead.sleep(long)
Object.wait(long)
LockSupport.parkNanos
等待超时
Thread.run()调用结束
TERMINATED