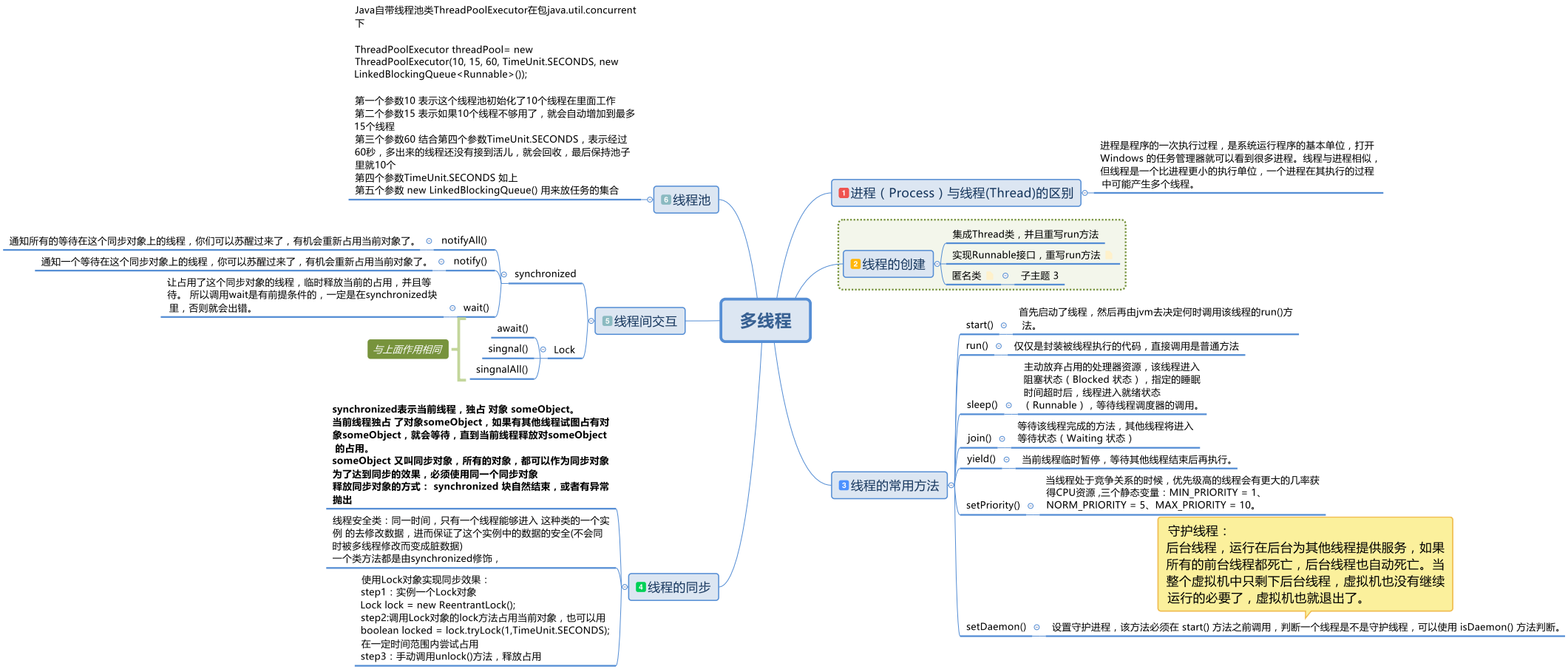
多线程
1 进程（ Process ）与线程(Thread)的区别
进程是程序的一次执行过程，是系统运行程序的基本单位，打开Windows 的任务管理器就可以看到很多进程。线程与进程相似，但线程是一个比进程更小的执行单位，一个进程在其执行的过程中可能产生多个线程。
2 线程的创建
集成Thread类，并且重写run方法
实现Runnable接口，重写run方法
匿名类
子主题 3
3 线程的常用方法
start()
首先启动了线程，然后再由jvm去决定何时调用该线程的run()方法。
run()
仅仅是封装被线程执行的代码，直接调用是普通方法
sleep()
主动放弃占用的处理器资源，该线程进入阻塞状态（ Blocked 状态 ），指定的睡眠时间超时后，线程进入就绪状态（ Runnable ），等待线程调度器的调用。
join()
等待该线程完成的方法，其他线程将进入等待状态（ Waiting 状态 ）
yield()
当前线程临时暂停，等待其他线程结束后再执行。
setPriority()
当线程处于竞争关系的时候，优先级高的线程会有更大的几率获得CPU资源 ,三个静态变量：MIN_PRIORITY = 1、NORM_PRIORITY = 5、MAX_PRIORITY = 10。
setDaemon()
设置守护进程，该方法必须在 start() 方法之前调用，判断一个线程是不是守护线程，可以使用 isDaemon() 方法判断。
守护线程：
后台线程，运行在后台为其他线程提供服务，如果所有的前台线程都死亡，后台线程也自动死亡。当整个虚拟机中只剩下后台线程，虚拟机也没有继续运行的必要了，虚拟机也就退出了。
4 线程的同步
线程安全类：同一时间，只有一个线程能够进入 这种类的一个实例 的去修改数据，进而保证了这个实例中的数据的安全(不会同时被多线程修改而变成脏数据)
一个类方法都是由synchronized修饰，
synchronized表示当前线程，独占 对象 someObject。
当前线程独占 了对象someObject，如果有其他线程试图占有对象someObject，就会等待，直到当前线程释放对someObject 的占用。
someObject 又叫同步对象，所有的对象，都可以作为同步对象
为了达到同步的效果，必须使用同一个同步对象
释放同步对象的方式： synchronized 块自然结束，或者有异常抛出
使用Lock对象实现同步效果：
step1：实例一个Lock对象
Lock lock = new ReentrantLock();
step2:调用Lock对象的lock方法占用当前对象，也可以用
boolean locked = lock.tryLock(1,TimeUnit.SECONDS);
在一定时间范围内尝试占用
step3：手动调用unlock()方法，释放占用
5 线程间交互
synchronized
notifyAll()
通知所有的等待在这个同步对象上的线程，你们可以苏醒过来了，有机会重新占用当前对象了。
notify()
通知一个等待在这个同步对象上的线程，你可以苏醒过来了，有机会重新占用当前对象了。
wait()
让占用了这个同步对象的线程，临时释放当前的占用，并且等待。 所以调用wait是有前提条件的，一定是在synchronized块里，否则就会出错。
Lock
await()
singnal()
singnalAll()
与上面作用相同
6 线程池
Java自带线程池类ThreadPoolExecutor在包java.util.concurrent下
ThreadPoolExecutor threadPool= new ThreadPoolExecutor(10, 15, 60, TimeUnit.SECONDS, new LinkedBlockingQueue<Runnable>());
第一个参数10 表示这个线程池初始化了10个线程在里面工作
第二个参数15 表示如果10个线程不够用了，就会自动增加到最多15个线程
第三个参数60 结合第四个参数TimeUnit.SECONDS，表示经过60秒，多出来的线程还没有接到活儿，就会回收，最后保持池子里就10个
第四个参数TimeUnit.SECONDS 如上
第五个参数 new LinkedBlockingQueue() 用来放任务的集合