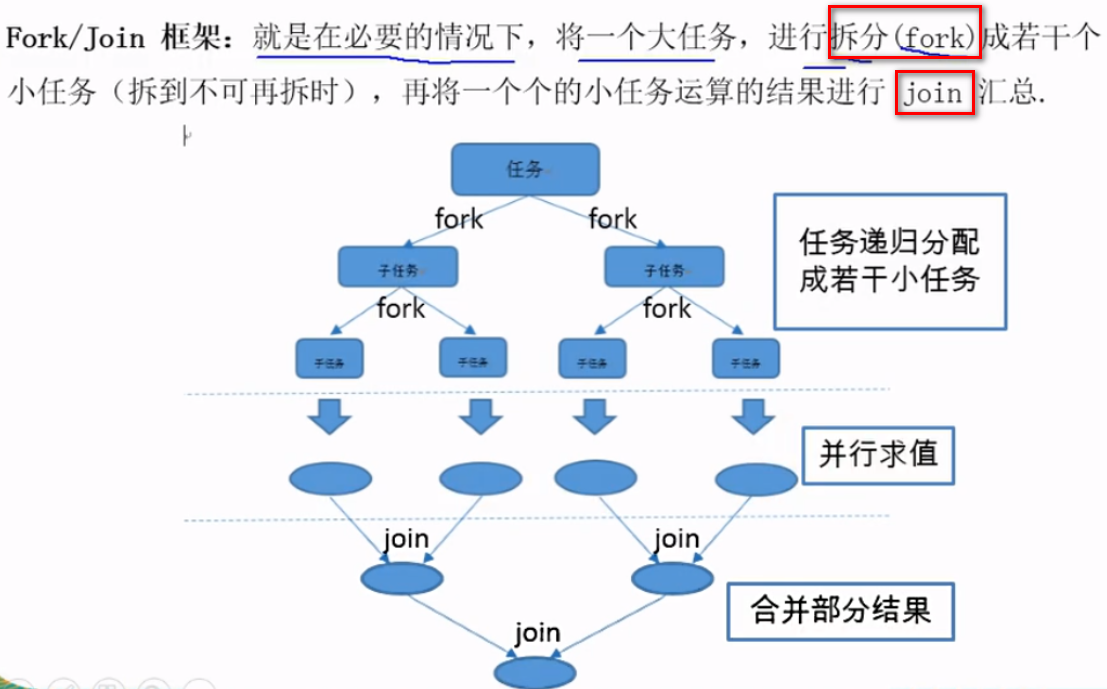
了解Fork/Join框架



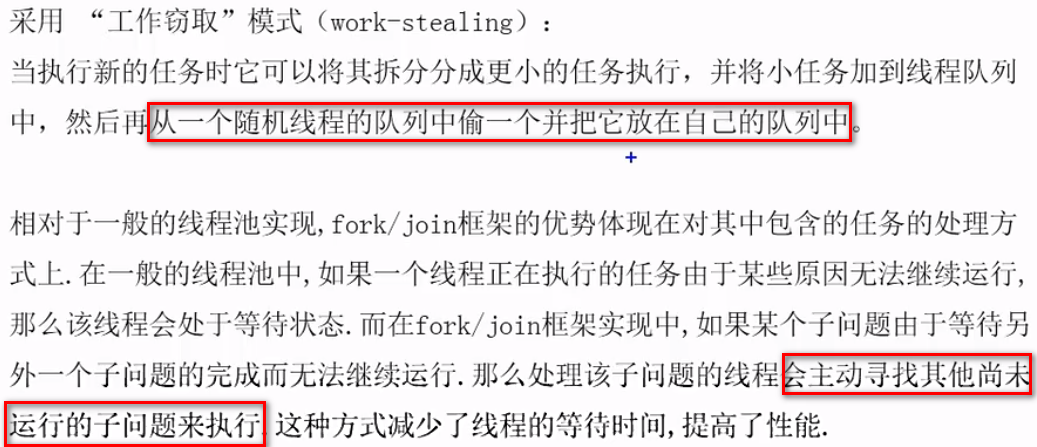
总结：

简单的说就是把大任务拆分成小任务执行，再合并

好处是可以提高CPU的使用权（使用工作窃取模式），效率高

使用：一般使用在大数据运算中，数据量越大，效率的提升越明显，而在数据量少的时候，反而效率低，因为拆分任务，合并任务也要消耗性能，所以先了解

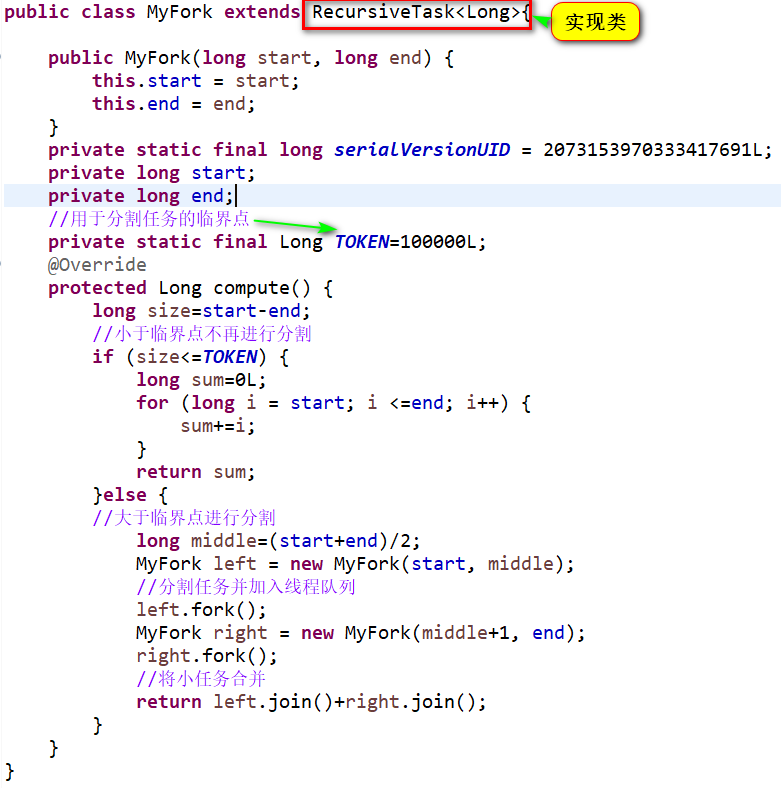
Fork/join与传统线程池的区别



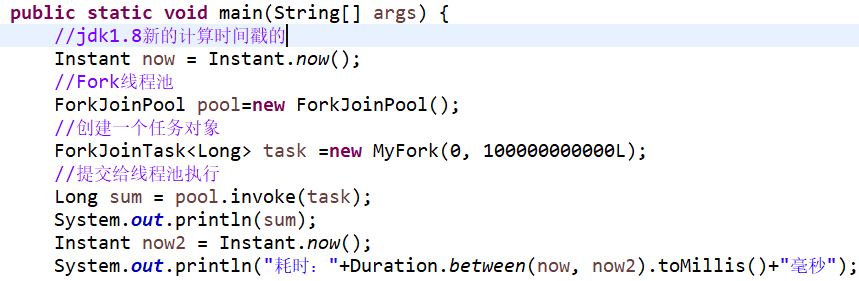
简单的说：就是当一个cpu（线程）任务执行完了，而其他线程的任务队列还有任务。它会随机在其他线程的任务队列的队尾偷一个任务放到自己的任务队列来执行，这样是为了保证CPU的利用率

一个小例子连接Fork/Join框架：计算1-1百亿的和

首先：要使用Fork/Join要实现一个类RecursiveTask<T>（有返回值）或者RecursiveAction（无返回值）

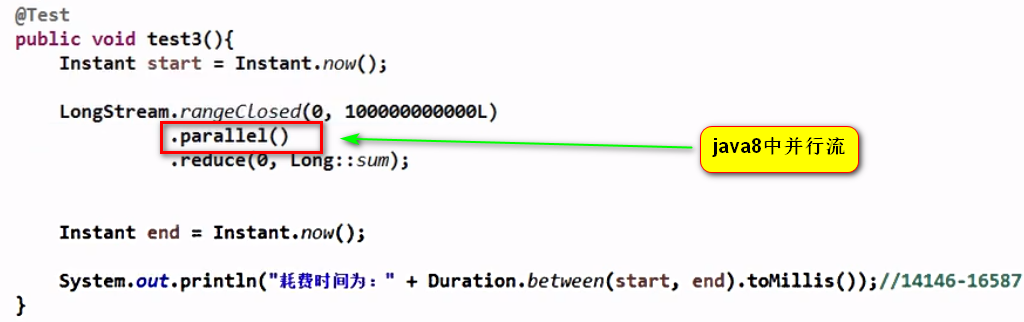


其次：运行它需要Fork线程池的支持



总结：它就是使用递归的思想进行拆分合并，然后本身作为一个任务提交给Fork线程池去执行即可，具体底层的不用关心

JDK8中的并行流解决这个问题：



说明：使用Stream API 解决这个问题，使用parall()方法见普通的串行流转化为并行流（其实并行流的底层是使用Fork/Join框架实现的）

注意：先了解，找到它是jdk对于性能提高的优化即可