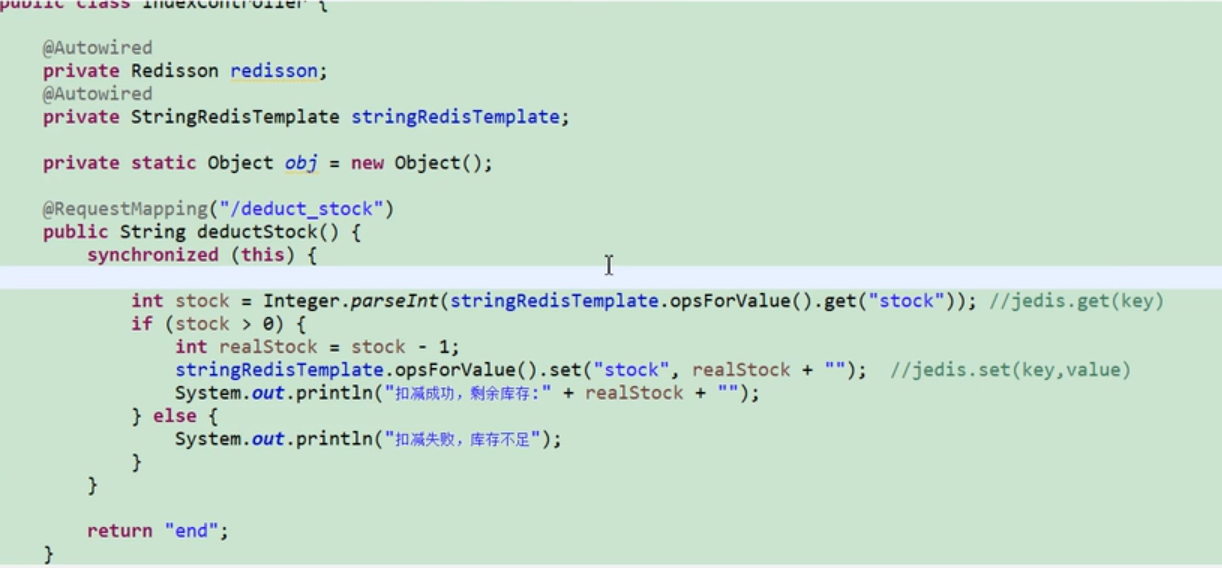
# 分布式锁实战



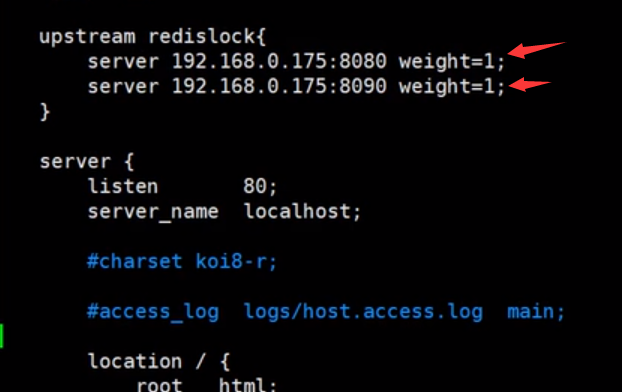
单应用直接加sychronized(this)关键词。

多服务（集群）端：Nginx(作为反向代理，负载均衡到几个实例)管理几个tomcat端口。Sychronized就没有用了

步骤：

Step1:启动2个springboot实例，端口分别为8080、8090，通过JMeter压力测试。

Step2:配置Nginx



Step3:客户端通过访问linux的ip来访问程序。

Step4:压力测试，发现sychronized锁失效。

原因：sychronized只在单个JVM有效，在两个JVM上面是锁不住的。

分布式锁：

方法1？库存减少的少，因为大多线程都被返回了，但不会重复扣库存：使用redis锁。SETNX：多个线程访问一个API时，仅有一个成功，其他返回。







问题：一个线程抛异常不释放锁。

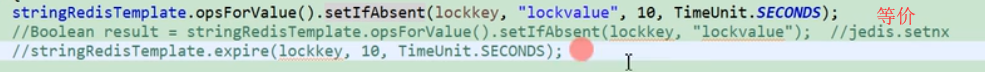
解决：在外面包围try catch finally。

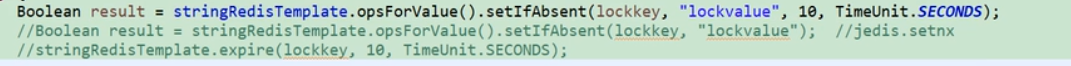
问题：执行到一半服务器挂掉，不释放锁

解决1：设置超时时间-不是原子性，刚拿到锁就挂，还没执行到设置超时时间的代码。

解决2：set()api，但是10秒够用吗？







还有bug：10秒不够，永久失效-线程1要执行15s，线程2执行10s，线程1的锁在第10s时释放，但继续执行，在第15s手动释放，但此时是把线程2的锁释放掉了，然后线程2释放了线程3的锁，如此循环达成永久失效。

解决：自己释放自己的锁-每一个客户端定义唯一的id，

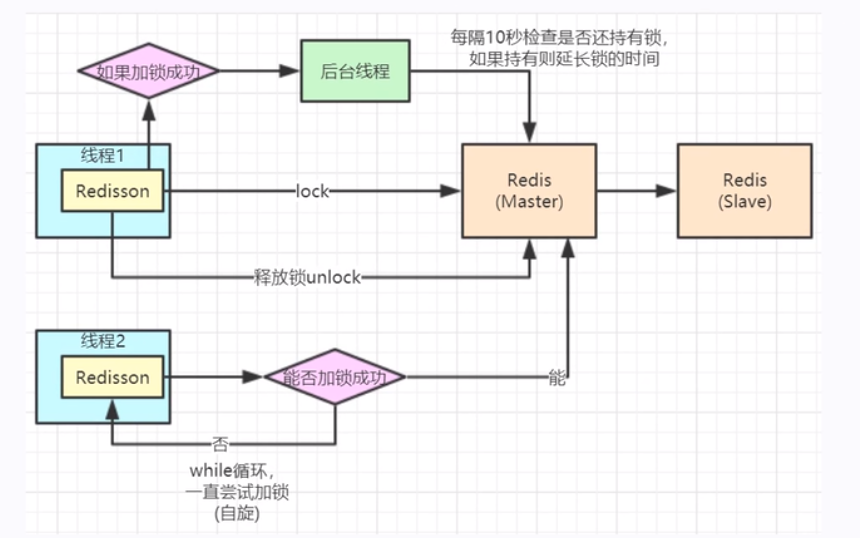






问题：可能存在线程1释放锁时线程2，3同时拿到锁

解决：后台开个分线程，搞个定时任务，定期看看是否执行完，如果没有就可以延期。网上有框架redisson

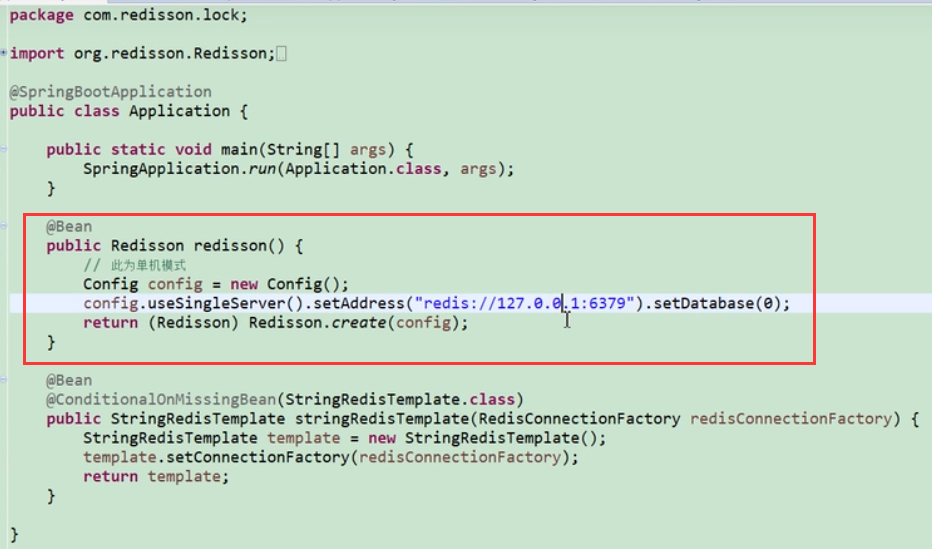


步骤:

Step1:引用redisson依赖



Step2:配置redisson



Step3:加锁



其实做的工作主要就是开个线程然后执行上面那两行代码。

DONE

问题：Redis是主从架构，Redis刚把锁设置成功，线程1拿到锁，Redis还没来得及把锁对应的Key同步到从节点上去，Redis主节点挂了，Redis从节点要选新的主节点，其他线程再去加锁就有可能对新的Redis Master加锁，新的master没有这把还没同步过来的锁，这时出现多个线程执行锁的代码逻辑。



面试可能会问：Spring怎么配置事务，Mybatis怎么配置sqlsession工厂。深入到底层源码。

===========================DONE========================