zookeeper可靠性比redis强太多，只是效率低了点，如果并发量不是特别大，追求可靠性，首选zookeeper。为了效率，则首选redis实现。

Redis分布式锁

Redis官方支持原生的分布式锁，采用了RedLock算法，原理如下:

1. 使用一个key作标记（代表资源），如果客户端使用setnx设置这个key成功，则表示加锁成功。
2. 释放锁：把这个key删除

开源框架：Redission提供了分布式锁的支持，RLock是可重入锁的分布式实现，原理：

Redission原理和Redis原生分布式锁相似，只是使用Lua脚本来保证操作的原子性

Redis分布式锁缺点：

如果master宕机，数据还没同步到slave，会导致重复加锁，也就是多个客户端同时获取了同一把锁。

Zookeeper分布式锁

zk分布式锁可用性更好，但是性能上不如redis分布式锁。ZK中创建和删除节点只能通过Leader服务器来执行，然后将数据同步到所有的Follower机器上。原理如下：

1. 客户端A发起加锁请求，则会在要加锁的node下创建一个临时顺序节点
2. 判断自己创建的那个顺序节点是不是排在第一个，如果是就表示加锁成功
3. 如果不是，表示加锁失败。对他的上一个顺序节点加一个监听器，监听这个节点是否被删除等变化
4. 释放锁其实就是把自己在ZK里创建的那个顺序节点删除掉
5. 客户端B的监听器感知到了上一个顺序节点被删除，就会通知客户端B重新尝试去获取锁。

缺点：zk分布式锁不能承载高并发请求（每次在创建锁和释放锁的过程中，都要动态创建、销毁节点）。而redis可以