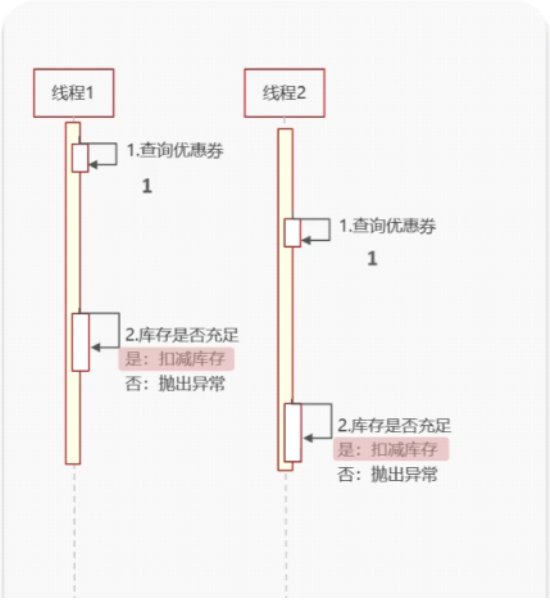
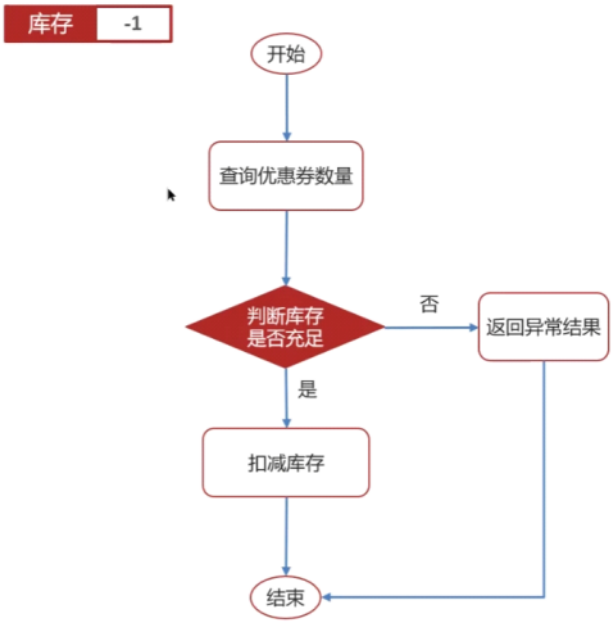
分布式锁

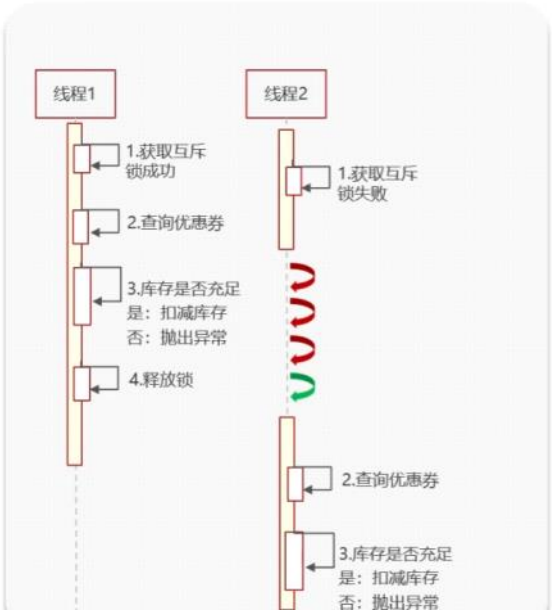
2024年4月11日 18:44

抢券执行流程





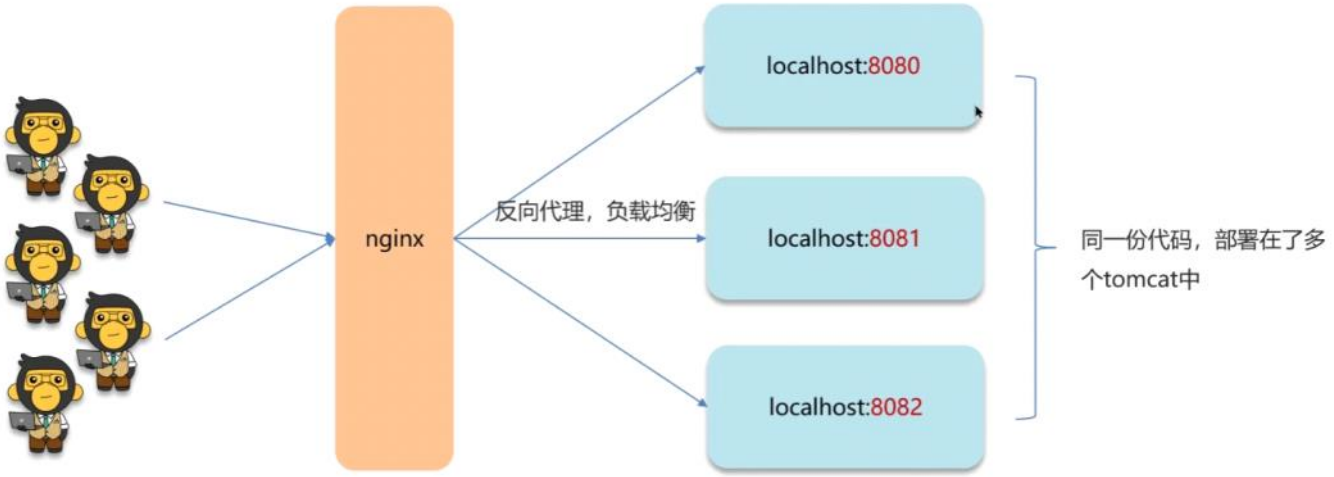
抢券执行流程



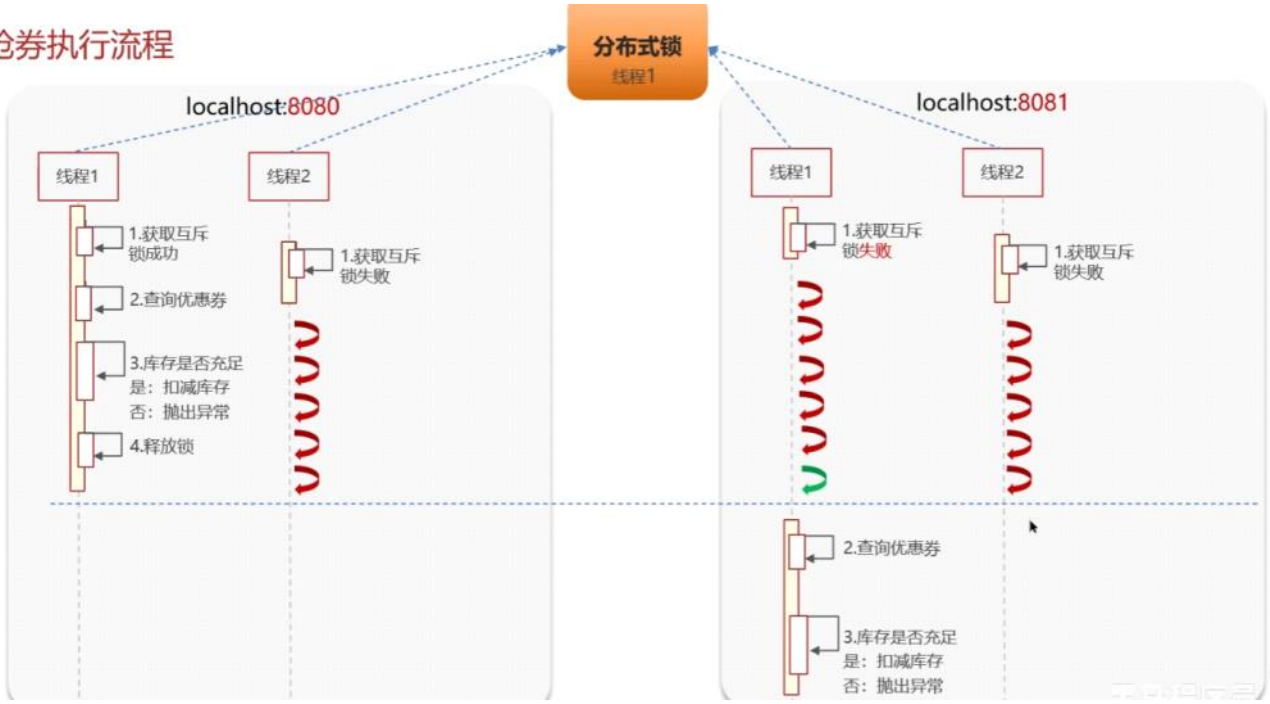
库存 1

```
public void rushToPurchase() throws InterruptedException {
    synchronized (this){
        //查询优惠券数量
        Integer num = (Integer) redisTemplate.opsForValue().get("num");
        //判断是否抢完
        if (null == num || num <= 0) {
            throw new RuntimeException("商品已抢完");
        }
        //优惠券数量减一（减库存）
        num = num - 1;
        //重新设置优惠券的数量
        redisTemplate.opsForValue().set("num", num);
    }
}
```

服务集群部署



抢券执行流程



redis分布式锁

Redis实现分布式锁主要利用Redis的setnx命令。setnx是SET if not exists(如果不存在，则 SET)的简写。

- 获取锁：

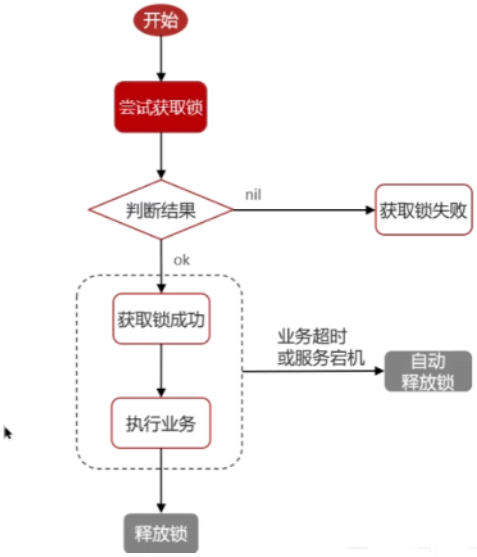
```
# 添加锁，NX是互斥、EX是设置超时时间
SET lock value NX EX 10
```

- 释放锁：

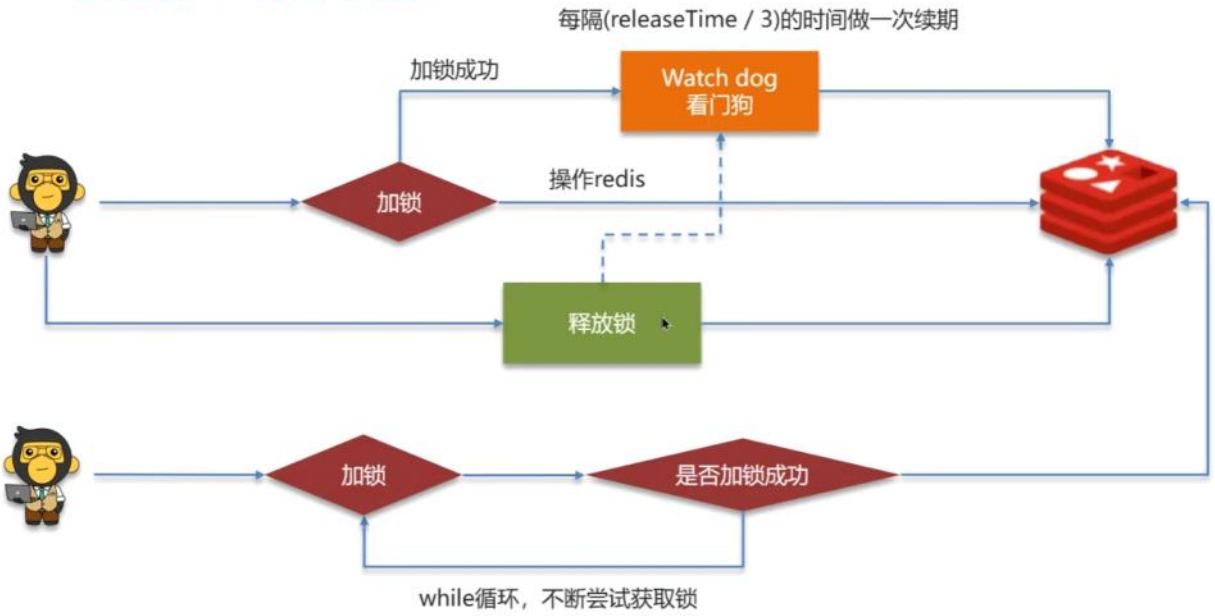
```
# 释放锁，删除即可
DEL key
```

Redis实现分布式锁如何合理的控制锁的有效时长？

根据业务执行时间预估　　给锁续期



redisson实现的分布式锁-执行流程




```
public void redisLock() throws InterruptedException {
    //获取锁（重入锁），执行锁的名称
    RLock lock = redissonClient.getLock( s: "heimalock");
    //尝试获取锁，参数分别是：获取锁的最大等待时间（期间会重试），锁自动释放时间，时间单位
    //boolean isLock = lock.tryLock(10,30, TimeUnit.SECONDS);
    boolean isLock = lock.tryLock( time: 10, TimeUnit.SECONDS);
    //判断是否获取成功 加锁、设置过期时间等操作都是基于lua脚本完成
    if (isLock) {
        try {
            System.out.println("执行业务");
        } finally {
            //释放锁
            lock.unlock();
        }
    }
}
```

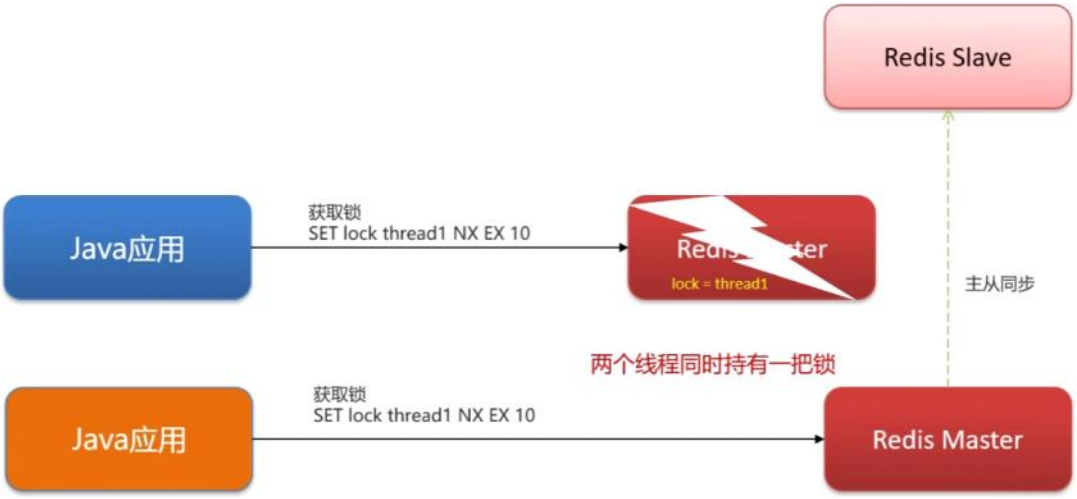
redisson实现的分布式锁-可重入

```
public void add1(){
    RLock lock = redissonClient.getLock("heimalock");
    boolean isLock = lock.tryLock();
    //执行业务
    add2();
    //释放锁
    lock.unlock();
}
public void add2(){
    RLock lock = redissonClient.getLock("heimalock");
    boolean isLock = lock.tryLock();
    //执行业务
    //释放锁
    lock.unlock();
}
```

利用hash结构记录线程id和重入次数

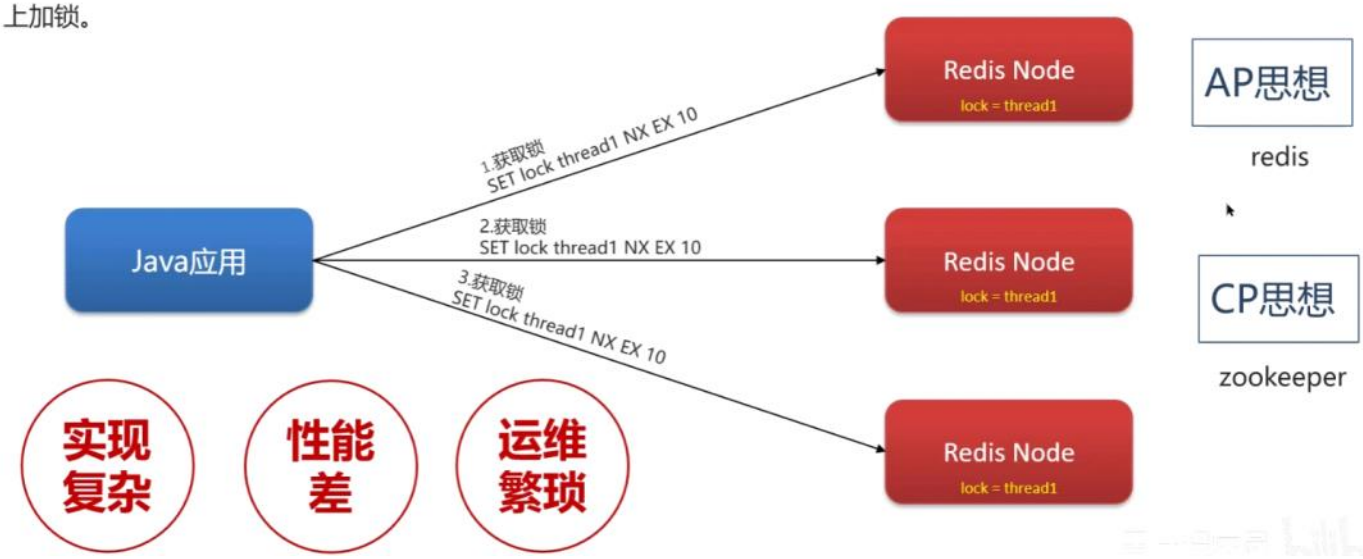
KEY	VALUE	
	field	value
heimalock	thread1	0

redisson实现的分布式锁-主从一致性



redisson实现的分布式锁-主从一致性

RedLock(红锁)：不能只在一个redis实例上创建锁，应该是在多个redis实例上创建锁(n / 2 + 1)，避免在一个redis实例上加锁。



redis分布式锁，是如何实现的?

- 先按照自己简历上的业务进行描述分布式锁使用的场景
- 我们当使用的redisson实现的分布式锁，底层是setnx和lua脚本（保证原子性）

Redisson实现分布式锁如何合理的控制锁的有效时长?

在redisson的分布式锁中，提供了一个WatchDog(看门狗），一个线程获取锁成功以后，WatchDog会给持有锁的线程续期（默认是每隔10秒续期一次）

Redisson的这个锁，可以重入吗?

可以重入，多个锁重入需要判断是否是当前线程，在redis中进行存储的时候使用的hash结构，来存储线程信息和重入的次数

Redisson锁能解决主从数据一致的问题吗

不能解决，但是可以使用redisson提供的红锁来解决，但是这样的话，性能就太低了，如果业务中非要保证数据的强一致性，建议采用zookeeper实现的分布式锁