



链滴

从分布式锁角度理解 Java 的 synchronized 关键字

作者: [pattyq](#)

原文链接: <https://ld246.com/article/1532699313015>

来源网站: 链滴

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

分布式锁

分布式锁就以zookeeper为例,zookeeper是一个分布式系统的协调器,我们将其理解为一个文件系统,以在zookeeper服务器中创建或删除文件夹或文件.设D为一个数据系统,不具备事务能力,在并发状态可能出现对单个数据同时读写.客户端A,B是数据系统D提供的客户端,能够对其读写.

几个关键角色已经登场,D是一个不提供事务行为的数据系统,其存放的数据可被读写,在单客户端条件可以保证数据的可靠,但是在两个客户端可能并发请求时就变得不可靠,A写的数据可能被B覆盖,B读的数据可能是A没有写完的数据.在不修改D源码为其提供原子性操作的前提下,我们可以考虑分布式锁.

客户端的原始API

```
void write(){
    // 写数据
}
void read(){
    // 读数据
}
```

如何做呢,核心就是以zookeeper为媒介.

修改客户端API,如果读或者写,我们首先要在zookeeper上创建一个文件夹,如果创建成功,我们就执行写操作,如果不成功(代表已经有人创建了)我们就一直尝试创建,直到创建成功.当创建成功时,对D系统数据进行读写操作,完成后删除zookeeper上创建的文件夹并退出读写方法.

```
void write(){
    while(在zookeeper上创建文件夹) {
        写操作;
        删除zookeeper的刚创建的文件夹;
    }
    return;
}
```

这样就完成了一个分布式并发行为的同步.假设A已经创建了文件夹,B就没办法进行后续操作,会一直尝试创建文件夹,A执行完操作之后删除文件夹,B才有机会进行在D系统上的操作.这个例子中产生了一个临界资源:D系统的数据,和一个竞态条件:A和B并发读写.(实际上zookeeper中是可以为监听者发送文件情的,比如我创建文件夹没有成功,可以监听该文件夹,当文件夹变化时会通知你,这时候你就可以再尝试创建文件夹了(很像wait和notify),但是为了不把重心放在zookeeper上就没有改成监听模式)

Java中Synchronized关键字

那我们再回到Java中的synchronized关键字上来,如果你还没理解上面的行为和synchronized的关系可以继续往后看.

synchronized究竟锁住的是什么呢,锁住的是一块内存,我们在内存中的某个位置设置一个值为0,当该为0时,一个线程为了修改临界资源将其设置为1,然后读写操作,读写结束将其设置为0,其他线程可以再将将其改为1,进行读写,结束后再将其置为0.....这块内存存储在每个对象的对象头中,我们称其为锁标志位实际上所标志位不是简单的0和1,锁标志位分为四种状态:无锁,偏向锁,轻量级锁,和重量级锁,并发中还及锁升级等,但本文不做细究,有兴趣的读者可以查阅相关资料).

当进入synchronized代码块或者方法的时候,你会像上面说的分布式锁那样"创建一个文件夹",其他线程无法"创建文件夹"就会一直去尝试,当代码块或方法结束的时候会"删除文件夹",其他线程可以去抢夺创

文件夹的机会。`synchronized`可以看做一道门,锁住的对象可以看做是门票.千万不要认为`synchronized`住的是临界资源,`synchronized`是以某个对象的锁标志作为入场券去约束竞态线程之间行为:我只有一门票,我只给一个人.

看一下`synchronized`的用法,多个线程之间的竞态行为一定要是用相同的门票去约束.

```
// 这锁住的是
class Foo{
    // 这锁住的是类对象
    static synchronized void bar();
    // 这锁住的是this,即实例对象
    synchronized void bar();
    // 等价于static synchronized
    void bar(){
        synchronized(Foo.class){};
    }
    // 等价于synchronized实例方法
    void bar(){
        synchronized(this){}
    }
}
```

对于静态方法需要注意一点(假如你想观察偏向锁的变化),实例方法可以直接观察实例对象的对象头上锁标志位,但是静态方法需要观察`Foo.class`对象的锁标志位,如果观察实例对象的锁标志位会竹篮打水哟.