两程序员玩“锁”，一人抢救无效身亡

哎呀 treenpool 2018-03-14 12:08

房间里灯光昏暗，两个男人相对而坐，

良久，眼睛男率先打破僵局，

眼睛男，知道锁么

帅气男，知道些，

眼睛男：什么是锁？（先发制人）

一种保护机制，在多线程的情况下，保证操作数据的正确性/一致性，

眼镜男：有哪几种分类？

悲观锁，乐观锁，独占锁，共享锁，公平锁，非公平锁，分布式锁，自旋锁

眼睛男：讲讲乐观锁悲观锁吧（顺藤摸瓜）

带秀

一般喜欢放在数据库来讲(其实这两个概念是属于计算机的，不要被误导)，就说mysql吧，悲观锁，主要是表锁，行锁还有间隙锁，叶锁，读锁，因为这些锁在被触发的时候势必引起线程阻塞，所以叫悲观

另外乐观锁其实在mysql本身中不存在的，但是mysql提供了种mvcc的机制，支持乐观锁机制，

眼睛男：**mvcc**是咋回事？（诱敌深入）

*只是在**innodb**引擎下存在，mvcc是为了满足事务的**隔离**，通过版本号的方式，避免同一数据不同事务间的竞争，所说的乐观锁只在事务级别为读未提交读提交，才会生效，*

眼睛男：具体**mvcc**机制有什么？（穷追不舍）

多版本并发控制，保证数据操作在多线程过程中，保证事务隔离的机制，可以降低锁竞争的压力，保证比较高并发量，这个过程。在每开启一个事务时，会生成一个事务的版本号，被操作的数据会生成一条新的数据行（临时），但是在提交前对其他事务是不可见的，对于数据的更新操作成功，会将这个版本号更新到数据的行中,事务提交成功，将

新的版本号，更新到此数据行（永久）中，这样保证了每个事务操作的数据，都是相互不影响的，也不存在锁的问题；

眼睛男：那么在多个事务（操作同一条数据）并发过程中，谁先成功？

mysql判断，其实就是谁先提交成功算谁的

眼睛男：说到事务了，聊聊事务，

*事务常说一系列操作作为一个整体要么都成功要么都失败，主要特性acid，事务的的实现主要依赖两个**log redo-log,undo-log**,每次事务都会记录数据修改前的数据undo-log，修改后的数据放入redo-log,提出成功则使用**redo-log** 更新到磁盘，失败则使用**undo-log**将数据恢复到事务之前的数据*

眼镜男，嗯，再说说独占锁，共享锁吧（转移阵地）

（嗯，独占，共享，公平，非公平，自旋锁这些都是广泛的概念，很多语言都有，包括操作系统，js的同学请回避）

独占锁很明显就是持锁的线程只能有一个，共享锁则可以有多个

眼睛男：独占可以理解，**共享**的意义在哪里？

共享锁是为了提高程序的效率，举个例子数据的操作有读写之分，对于写的操作加锁，保证数据正确性，而对于读的操作如果不加锁，在写读操作同时进行时，读的数据有可能不是最新数据，如果对读操作加独占锁，面对读多写少的程序肯定效率很低，所有就出现了共享锁，对于读的的操作就使用共享的概念，但是对于写的操作则是互斥的，保证了读写的数据操作都一致，在java中上述的锁叫读写锁

眼睛男：读写锁的机制是什么呢？（佯攻）

*在java中**读写锁（ReadWritelock）**的机制是基于**AQS**的一种实现，保证读读共享，读写互斥，写写互斥，如果要说机制的话，还要从**AQS**说起，这是java实现的一种锁机制，互斥锁，读者写锁，条件产量，信号量，栅栏的都是它的衍生物，主要工作基于**CHL队列**，**voliate关键字修饰的状态符stat**，线程去修改状态符成功了就是获取成功，失败了就进队列等待，等待唤醒，AQS中还有很重要的一个概念是**自旋**，在等待唤醒的时候，很多时候会使用自旋（while（!cas()））的方式，不停的尝试获取锁，直到被其他线程获取成功*

*共享与独占的区别就在于，**CHL队列**中的节点的模式是EXCLUSIVE还是SHARED，当一个线程成功修改了**stat**状态，表示获取了锁，如果线程所在的节点为SHARED，将开始一个读锁传递的过程，从头结点，向队列后续节点传递唤醒，直到队列结束或者遇到了EXCLUSIVE的节点，等待所有激活的读操作完成，然后进入到独享模式（这部分尽力了，大家还是看源码）*

公平与非公平的区别就在于线程第一次获取锁时，也就是执行修改stat操作时，是进队列还是直接修改状态，这是基本的工作机制，详细的估计可以再聊好几集

眼睛男：java 除了AQS 还有其他的锁支持么（佯攻未遂，寻找突破）

*在java中，**synchronized**关键字，是语言自带的，也叫内置锁，**synchronized**关键字，我们都知道被**synchronized**修饰的方法或者代码块，在同一时间内，只允许一个线程执行，是明显的独享锁，**synchronized的实现机制？**可以参考AQS的实现方式，只是AQS使用显示的用lock.lock()调用，而sync作为关键字修饰，你可以认为在synchronized修饰的地方，自动添加了lock方法，结束的地方进行了unlock释放锁的方法，只是被隐藏了，我们看不到。*

*它本身实现有两部分：monitor对象，线程，工作机制还是线程抢占对象使用权，对象都有自己的对象头，存储了对象的很多信息，其中有一个是标识被哪个线程持有，对比AQS，线程从修改stat，变为修改**monitor**的对象头，线程的等待区域动 AQS中的队列，变为monitor对象中的某个区域，*

眼睛男：能细说么？（贴脸）

导演，他自己加戏

导演：按照剧本来

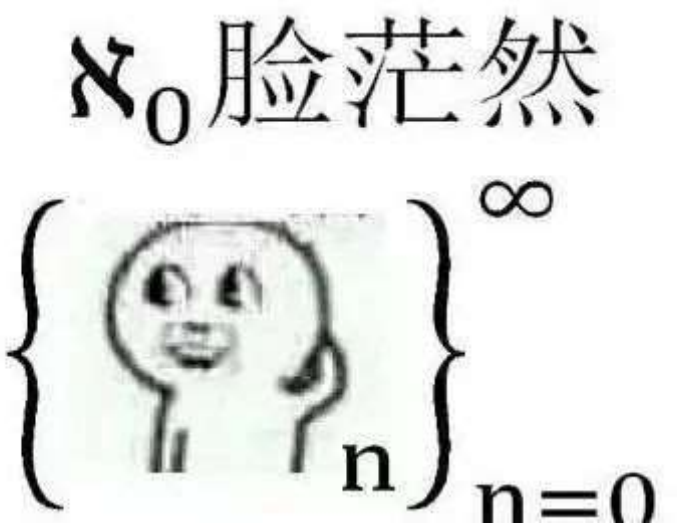
锁一直是围绕线程安全来实现的，比如独占锁，它在内存里面的操作是怎么样的

*这个地方涉及到一个概念，内存模型（这个和jvm不要混淆，**The Java memory model used internally in the JVM divides memory between thread stacks and the heap. This diagram illustrates the Java memory model from a logic perspective**），是JVM用来区别线程栈和堆的内存方式，每个线程在运行的时候，所操作的数据存储空间有两个，一个是**主内存** 一个是**工作内存**，主内存其实就是jvm中堆，工作内存就是线程的栈，每次的数据操作，都是从主内存中把数据读到工作内存中，然后在工作内存中进行各种处理，如果进行了修改，会把数据回写到主内存，然后其他线程又进行同样的操作，就这样数据在工作内存和主内存，进进出出，不亦乐乎，在多次的情况下，就是因为进进出出的顺序乱了，不是按照线程预期的访问顺序，就出现了数据不一致的问题，导致了多线程的不安全性；整个操作过程还牵涉到**CPU，高速缓存**等概念，略过。。。。*

眼睛男：内存模型 还有哪些可以聊聊的（我们是抱着学习的心态）

happen-befor** 原则，**Volatile** 关键字（线程的可见性），**内存屏障

眼睛男：哦（怂了怂了）



happen-befor原则定义了内存模型执行过程中的定律，就像1+1 = 2，不可能被打破的jvm的运行机制都依赖于这个原则，是jvm的宪法！！！

Volatile关键字就有点叼了，Volatile修饰的数据，在被某个线程修改后，会被及时的回写到主内存，然后其他线程再获取时，就是新的数据，听起来很美好，但是Volatile没有办法控制线程的顺序，当一个数据（新数据）即将被修改到主内存时，刚好，另外一个线程从主内存读了数据（老数据），并又进行了一波操作，又将数据(更新的数据)回写到了主内存，整个过程（新数据）完全没有起到一毛钱作用，最终导致了数据的错误，呼呼打完收工！！！！

眼镜男：你为啥知道这么多

因为我帅啊，

眼镜男：有多帅？

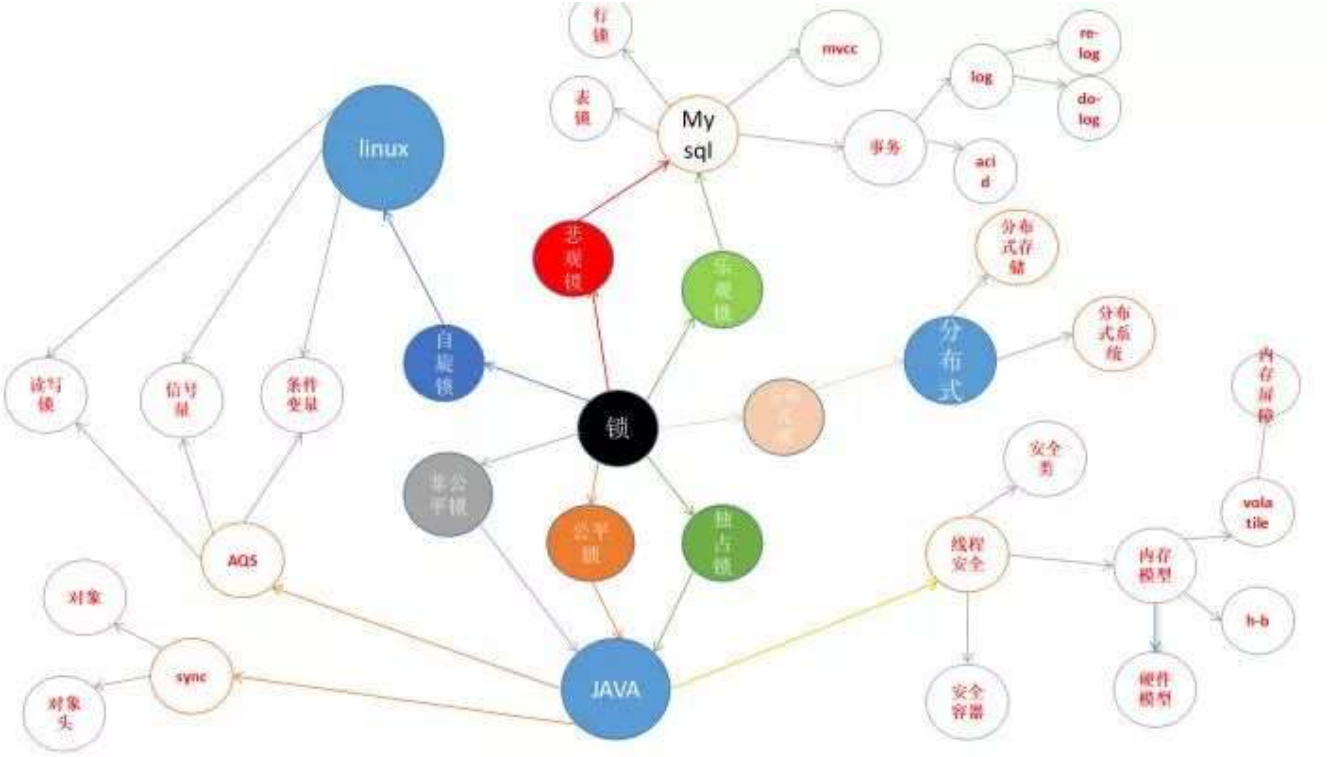
可以用微笑杀死你

眼镜男：来啊！

眼睛男猝 享年28岁！！！

。。。。。end。。。。。。

整个文章以“锁”为半径，构建了一个简单的知识体系，直观的感受一下



更详细的请打开 treenpool.com/html/inde

本文旨在为大家扩宽思路，认识一下不太常见的概念，不要太纠结于细节！！！（怂了怂了）

这是以点及面比较好的例子，从一个点不断深入，你会发现很多新的东西，而这些就是需要一点一滴咀嚼的，看博客或者视频都是经过别人嚼烂消化过的东西，然后再吐出来展示给大家，大家看看他吃的啥就行，千万不要再吃下去，因为真正的养分已经不多

了，还是需要从书啃起，一口口嚼出来，经过味蕾，大脑，胃，大肠。。。。一点点吸收，才是自己的，这也是我个人平时学习的方法，也祝愿大家可以用自己的方法，不断提升自己，扩展自己的知识体系；；；

本文很多描述的不是很完全，毕竟每个点随随便便就是几千字的，本人能力有限，只能把最近吃的吐出来给大家看看，不知道是否符合大家的胃口，

好书推荐，很多同学希望可以推荐一下书，我整理了一下，以下几类

入门级

java核心编程卷1：url.cn/5XFpwDI，很多入门的同学，想要推荐一下书，之前想推荐thinking in java，不过对比下，这本书更加实用，可以作为入门的工具书

java核心编程卷2：url.cn/51VMnA6 目前只有英文版的，篇幅比较长，对java的一些高级特效比如线程 网络之类的感兴趣可以看下，里面的AWT就不要看了，基本已经绝迹江湖了

python核心编程：url.cn/5eMWKpL ， Python的 入门工具书

进阶级

java 设计模式 url.cn/5w5BSV2，我买的第一本正儿八经的书，当时经常拿来吹牛

effective java: url.cn/5qNRfaC,这是英文版的，和我买的一样，每章都很独立，有很多奇思妙想（奇技淫巧）

深入理解Java虚拟机-JVM高级特性与最佳实践：url.cn/5Jwsuf6 ，这本书我读了差不多5编，很多jvm的点都能体现到，想要深入了解jvm的同学不要错过

Java并发编程： url.cn/5xyxETT 应该也有5遍，里面对于线程安全的东西讲的很透彻，如果想结合JMM和jvm了解并发的同学，深入理解jvm和java并发编程是绝配，

内核型书籍：

redis设计与实现：url.cn/5TpE4n9，之前对redis只是会用，甚至一直认为，redis的存储也是以树为基础，结果，被打脸，适合使用过redis的同学

linux内核设计与实现： url.cn/55hIhMB，适合所有人，你会发现很多语言性的概念都是抄袭自linux

http权威指南： url.cn/5Y731VY,一本手册，这么厚，现在依然觉得，谁读完谁傻逼，不过里面的一些章节需要好好琢磨，比如第三章 HTTP报文 第四章连接管理 第七章缓存 第九章， 第十一章 客户端识别与cookie机制 第十二章基本认证机制 第十二章。。。算了，还是尽量读完吧

高性能mysql; url.cn/512yVHi 应该有三遍，里面的对于mysql的数据结构讲述的比较清楚，但是不是很全，在锁和索引的描述篇幅比较多，以及一些高级特性 分区，事物，视图，值得一看

体系型：

大型网站系统与Java中间件实践： url.cn/5jvxxfQ

大型分布式网站架构设计与实践 ： url.cn/51XEcwk

这两本书对于工作了4-5年的同学会有很大的启发，是突破瓶颈的好书

最后！！！高能预警

欢迎大家加微信骚扰：treenpool

关注微信公众号：treenpool，更多好文在里面

请持续关注，m

文章已于2022-03-23修改