**并发编程**

# ReentrantReadWriteLock

## 概述

　ReentrantReadWriteLock是Lock的另一种实现方式，我们已经知道了ReentrantLock是一个排他锁，同一时间只允许一个线程访问，而ReentrantReadWriteLock允许多个读线程同时访问，但不允许写线程和读线程、写线程和写线程同时访问。相对于排他锁，提高了并发性。在实际应用中，大部分情况下对共享数据（如缓存）的访问都是读操作远多于写操作，这时ReentrantReadWriteLock能够提供比排他锁更好的并发性和吞吐量。

　　读写锁内部维护了两个锁，一个用于读操作，一个用于写操作。所有 ReadWriteLock实现都必须保证 writeLock操作的内存同步效果也要保持与相关 readLock的联系。也就是说，成功获取读锁的线程会看到写入锁之前版本所做的所有更新。

　　ReentrantReadWriteLock支持以下功能：

　　　　1）支持公平和非公平的获取锁的方式；

　　　　2）支持可重入。读线程在获取了读锁后还可以获取读锁；写线程在获取了写锁之后既可以再次获取写锁又可以获取读锁；

　　　　3）还允许从写入锁降级为读取锁，其实现方式是：先获取写入锁，然后获取读取锁，最后释放写入锁。但是，从读取锁升级到写入锁是不允许的；

　　　　4）读取锁和写入锁都支持锁获取期间的中断；

　　　　5）Condition支持。仅写入锁提供了一个 Conditon 实现；读取锁不支持 Conditon ，readLock().newCondition() 会抛出 UnsupportedOperationException。