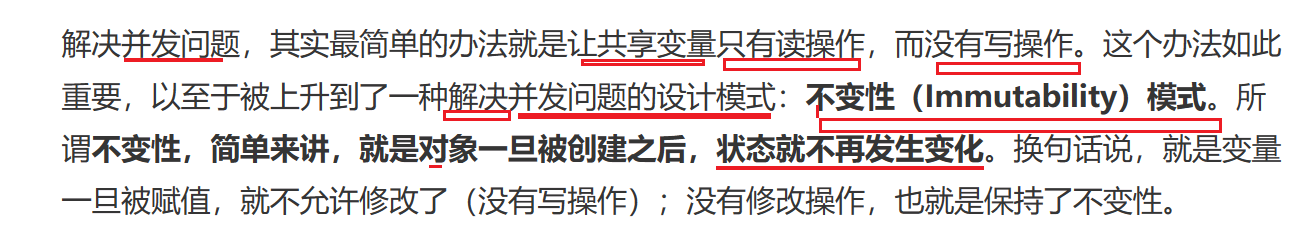
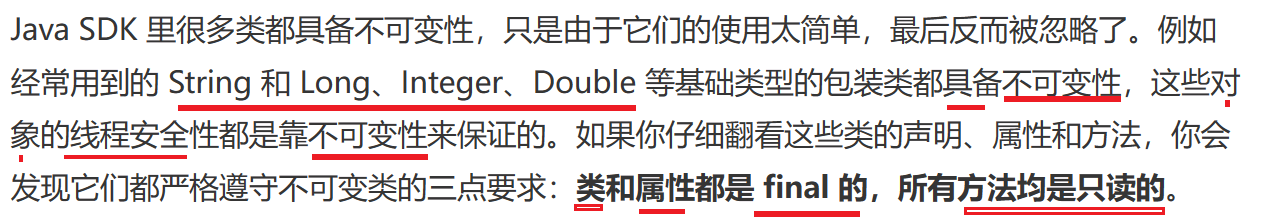
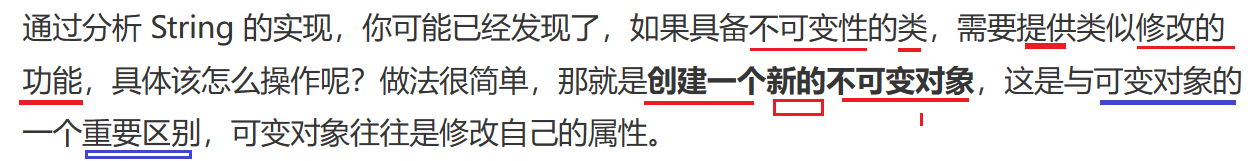
# 28.不可变模式



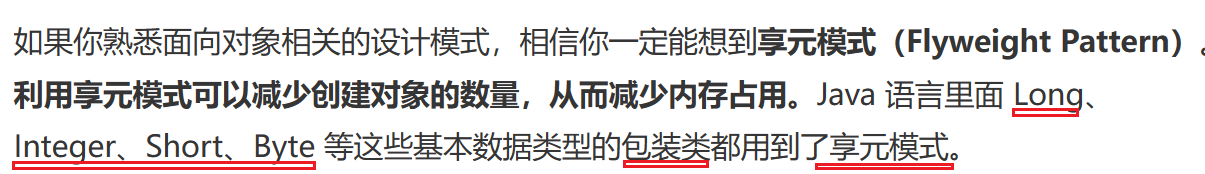
## 实现方式

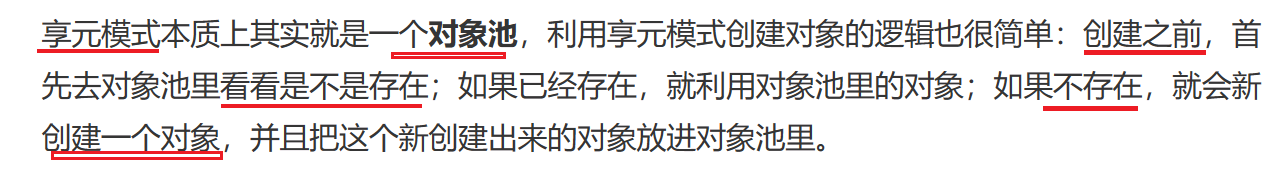


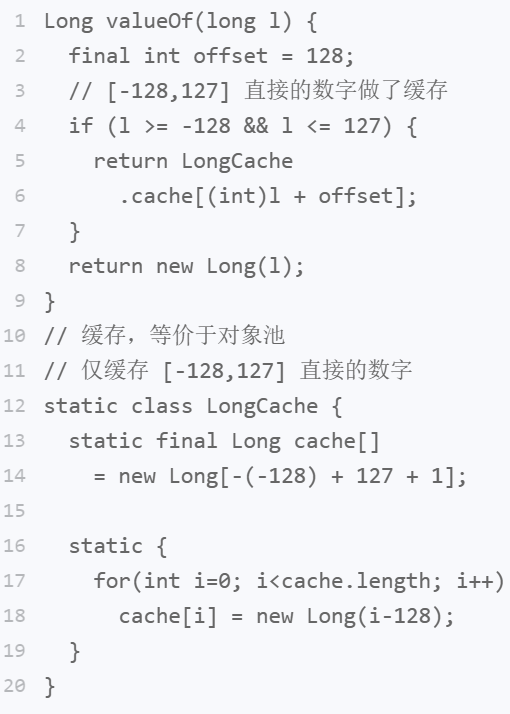
## 修改功能



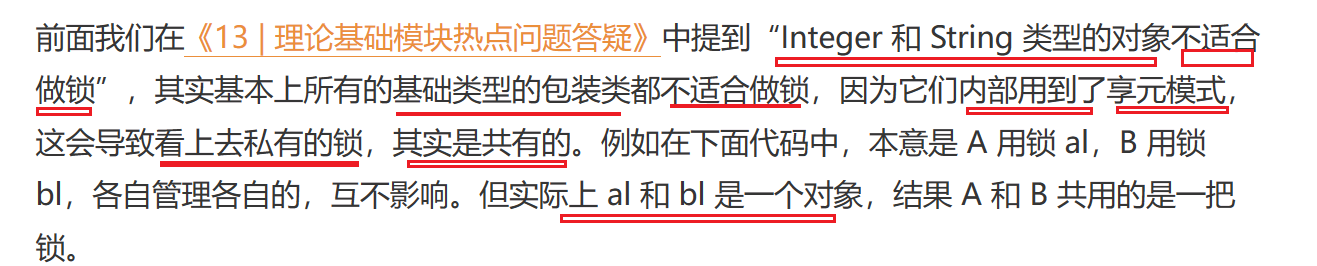
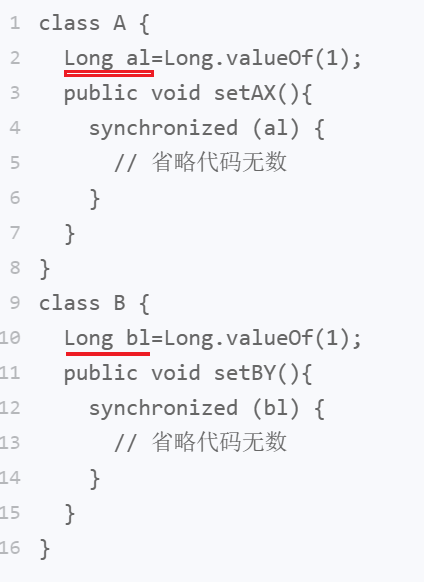
## 享元模式避免创建重复对象



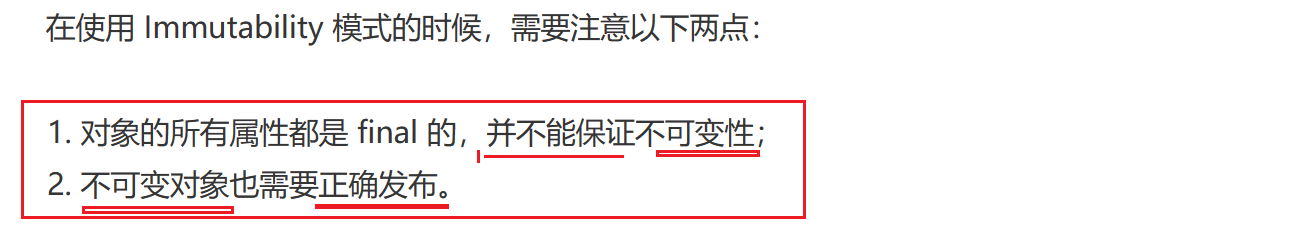


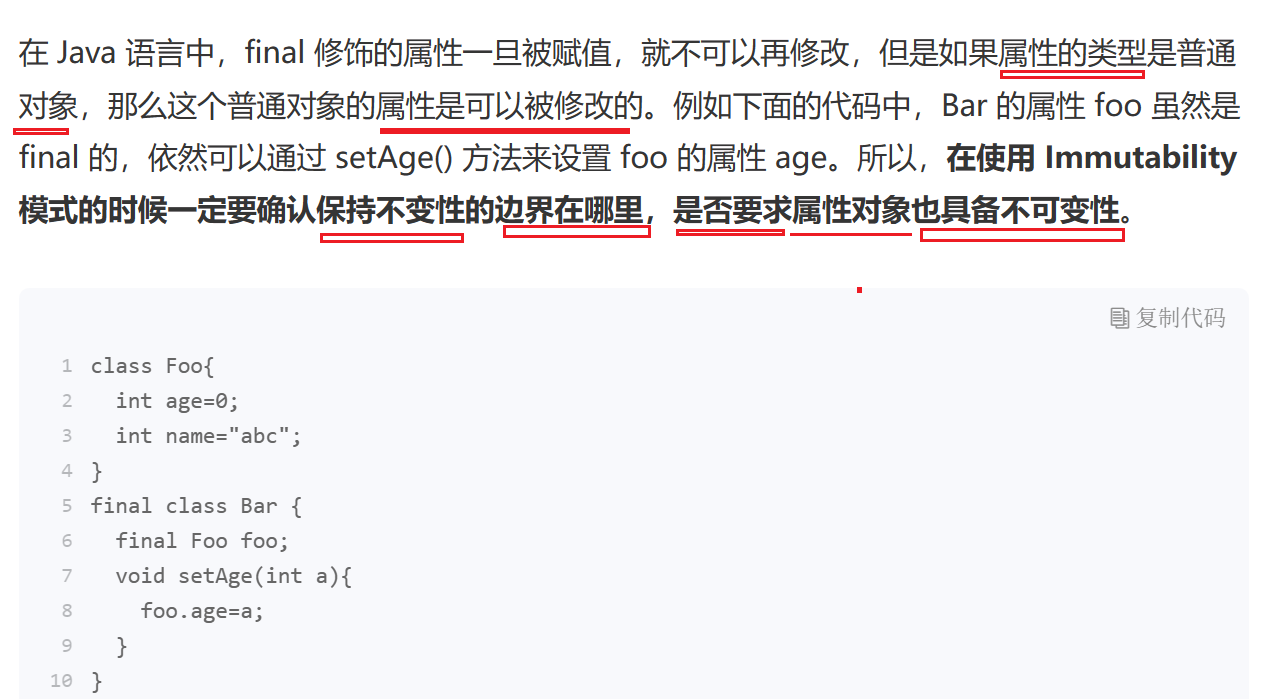


## 基本类型的包装类不适合做锁

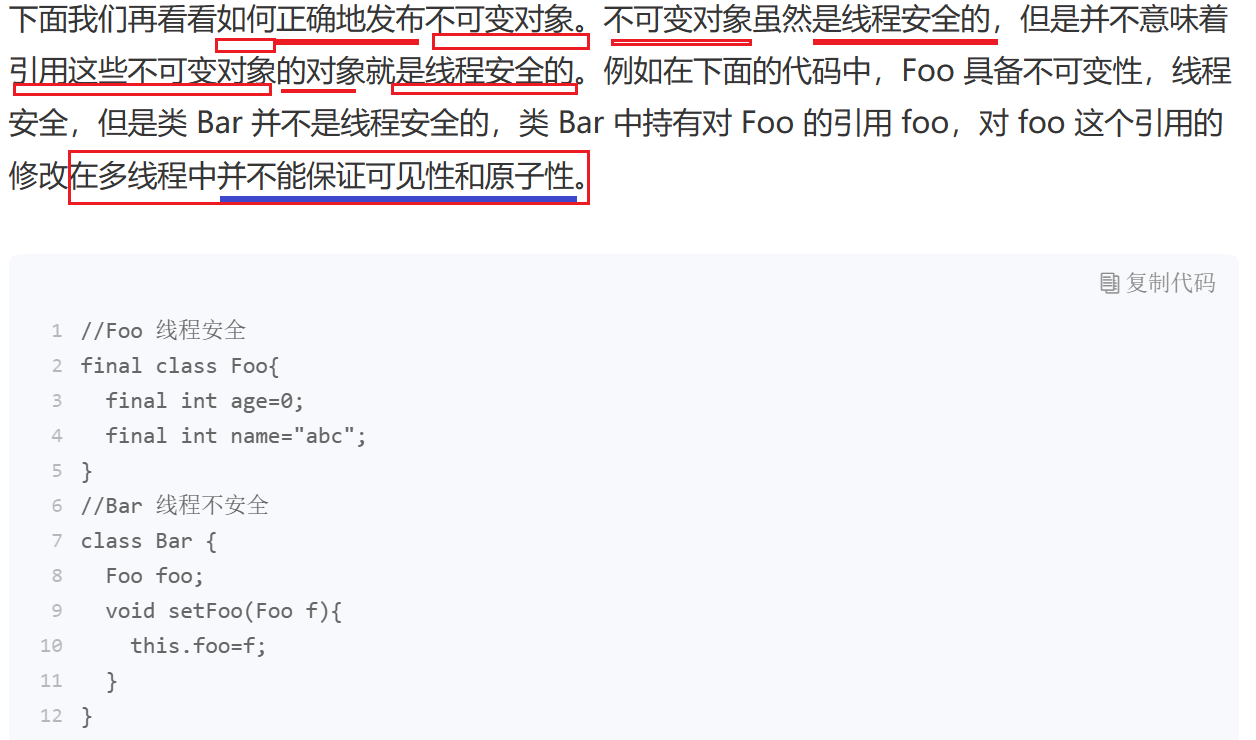


## 使用 Immutability 模式的注意事项



1. 

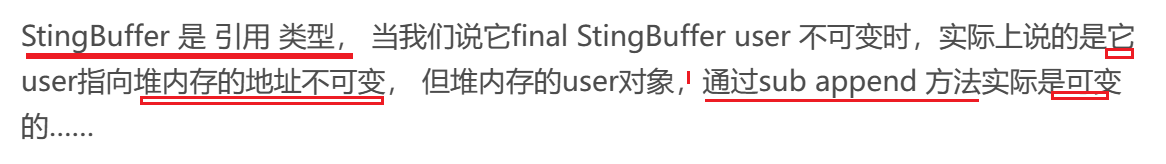
2.



## 不可变对象原子性问题

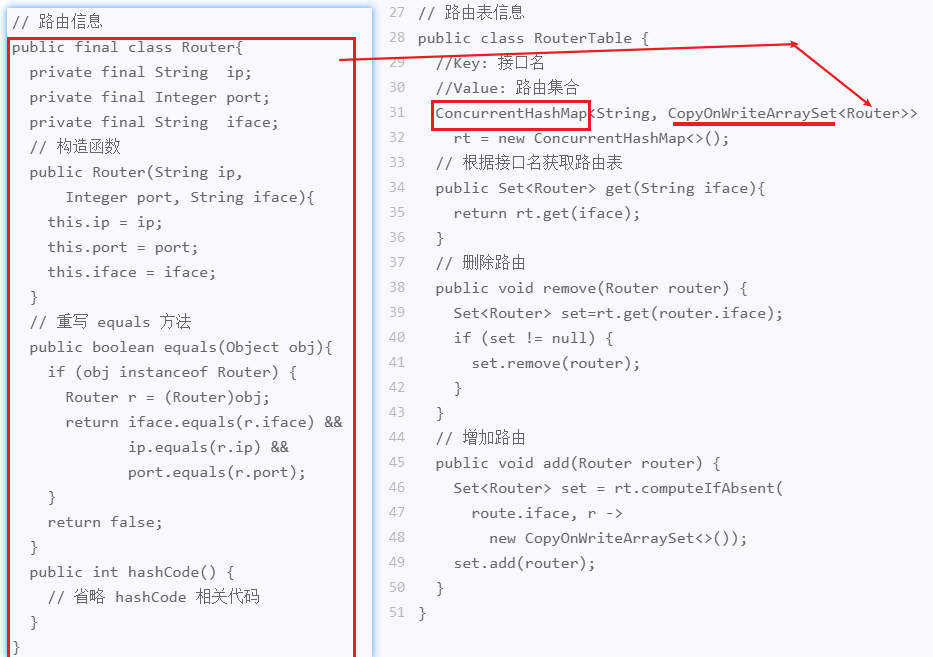






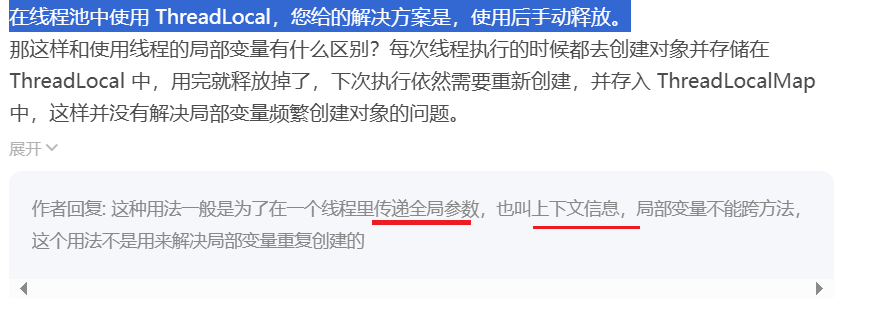
# 29.Copy-on-Write模式

你会发现它本质上是一种**Copy-on-Write 方法**。所谓 Copy-on-Write，经常被缩写为 COW 或者 CoW，顾名思义就是**写时复制**。



# 30. 线程本地存储模式

**ThreadLocal**

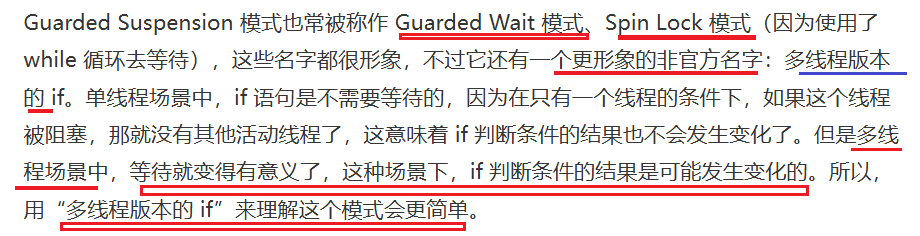


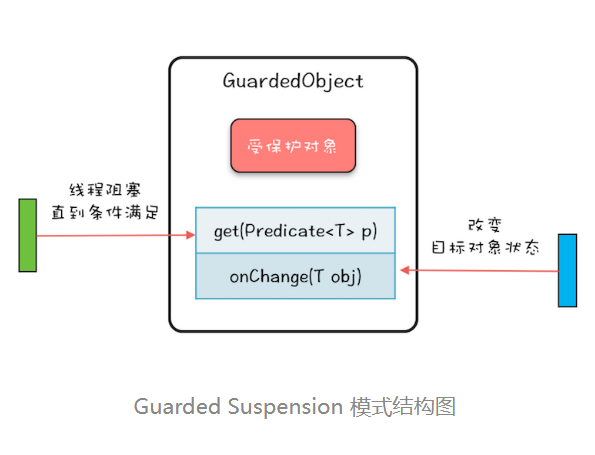
<https://juejin.im/post/5ce7e0596fb9a07ee742ba79>

# 31.Guarded Suspension模式

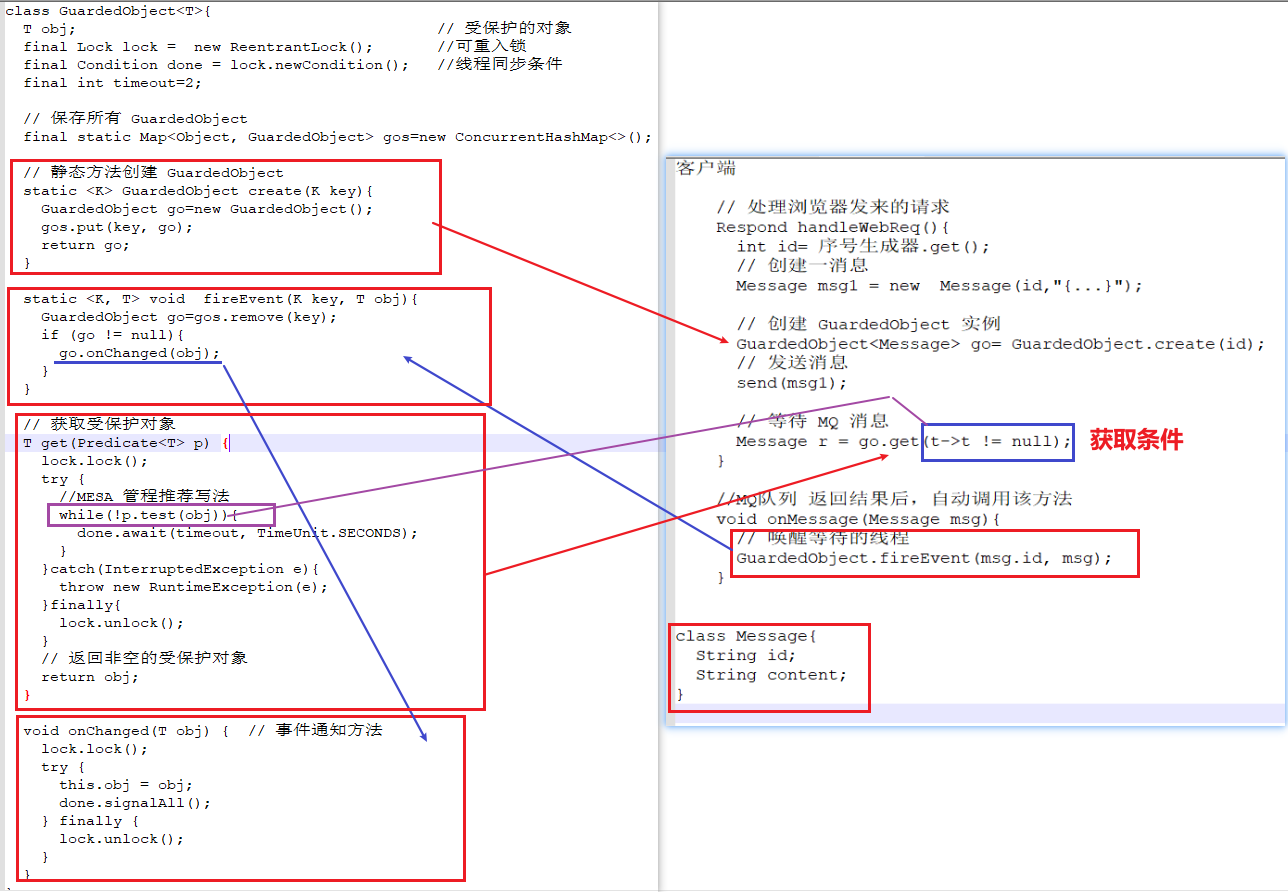
Guarded Suspension 模式本质上是一种等待唤醒机制的实现，通过让线程的等待，来保证受保护对象的安全。直译过来就是 “**保护性地暂停**“模式,主要解决异步信息同步问题

。例如：给MQ发送消息的线程是处理 Web 请求的线程 T1，但消费 MQ 结果的线程并不是线程 T1，那线程 T1 如何等待 MQ 的返回结果呢？





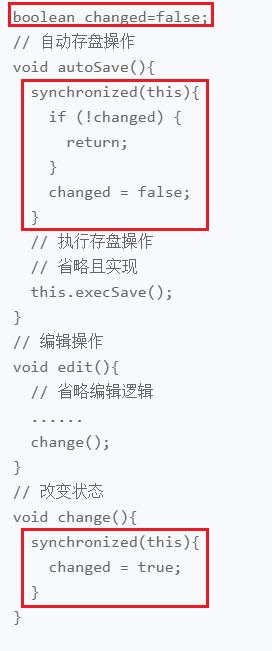
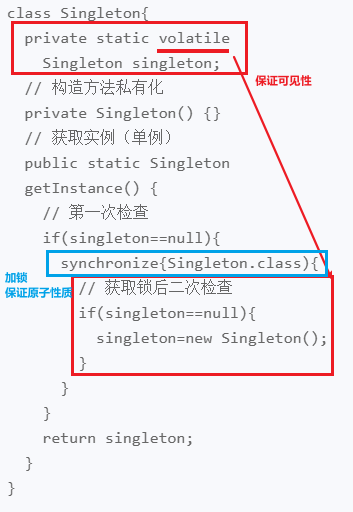
代码实现

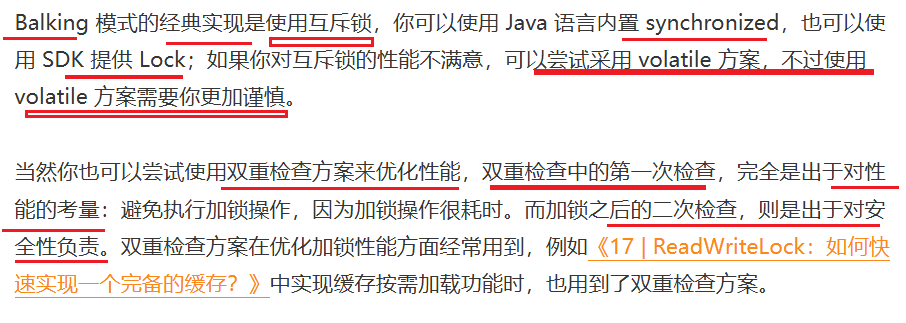


锁可以保证内存可见性，sleep无法保证变量在多线程的可见性，因为sleep不会失去当前对象的监控。而锁在等待后会失去当前的监控，进入等待队列，重新获取控制权。

# 32.Balking模式

Balking模式（也叫犹豫模式），用在一个线程 发现 另一个线程或本线程已经做了某一件相同的事，那么本线程就无需再做了，直接结束返回。

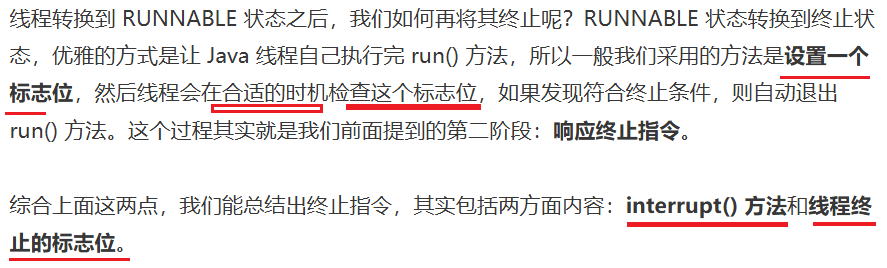
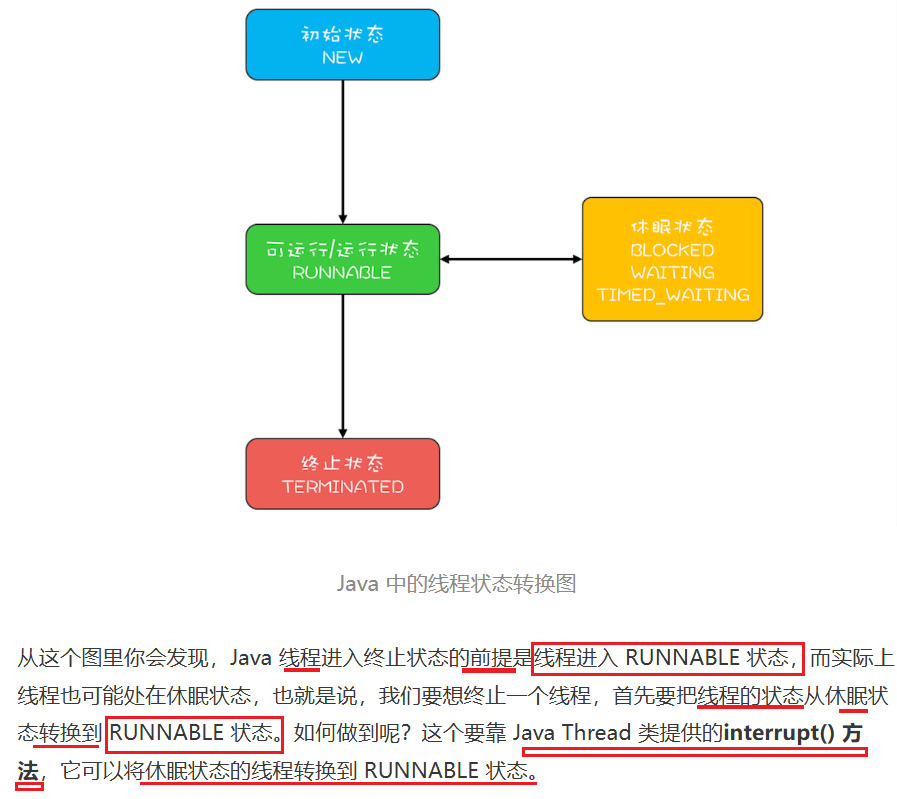
 

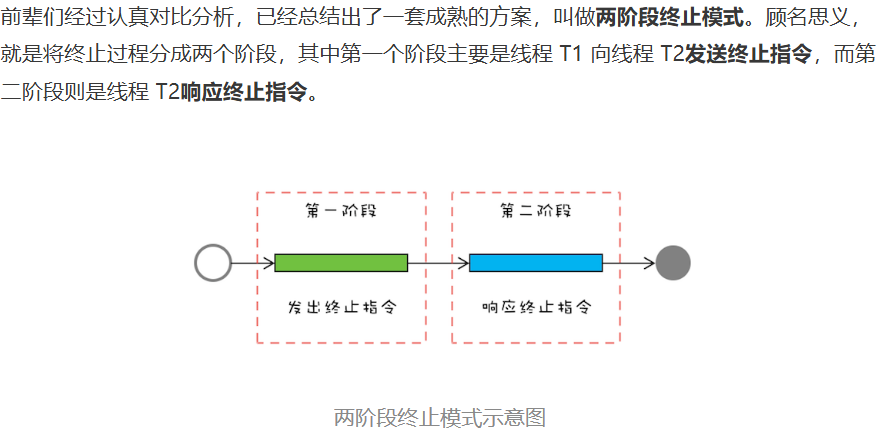


volatile关键字只能保证可见性，无法保证原子性和互斥性

# 35 | 两阶段终止模式

## 线程中断原理





## 实现案例

