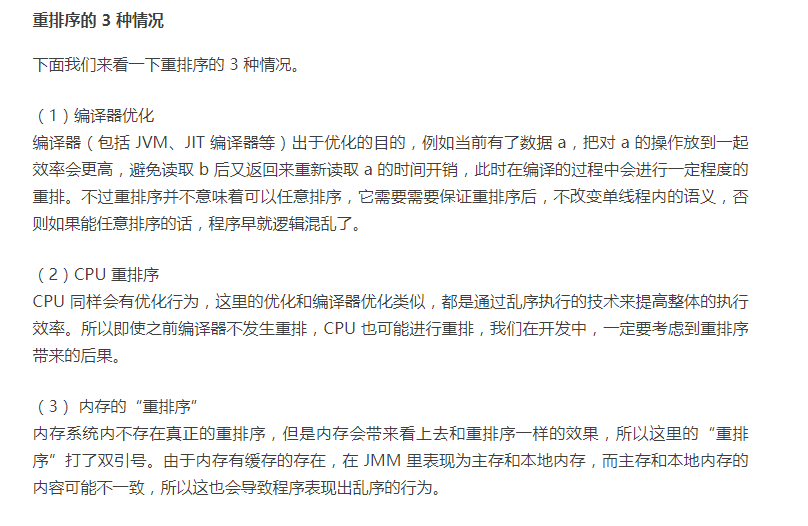
1. **JMM内存模型**

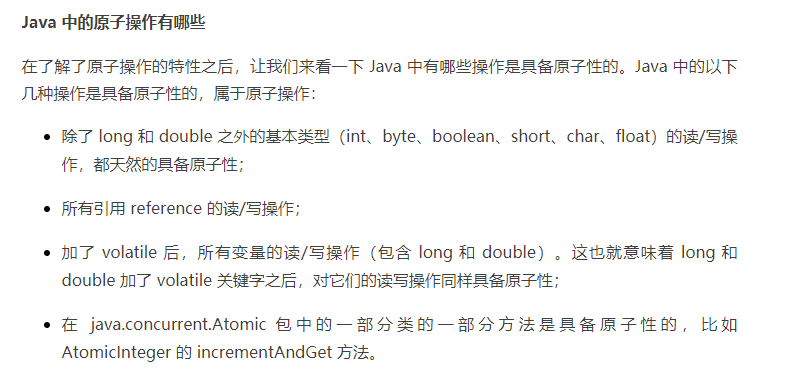
JMM 与处理器、缓存、并发、编译器有关。它解决了 CPU 多级缓存、处理器优化、指令重排等导致的结果不可预期的问题。

内存模型三剑客：重排序、原子性、内存可见性

1. **重排序**



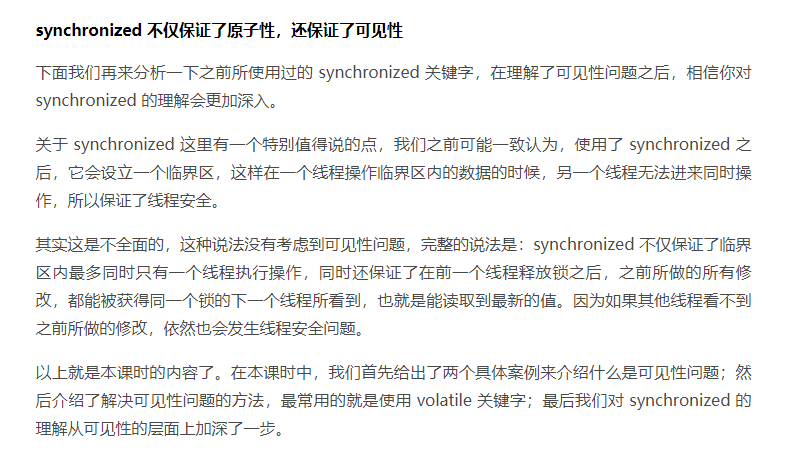
1. **原子操作（原子性）**



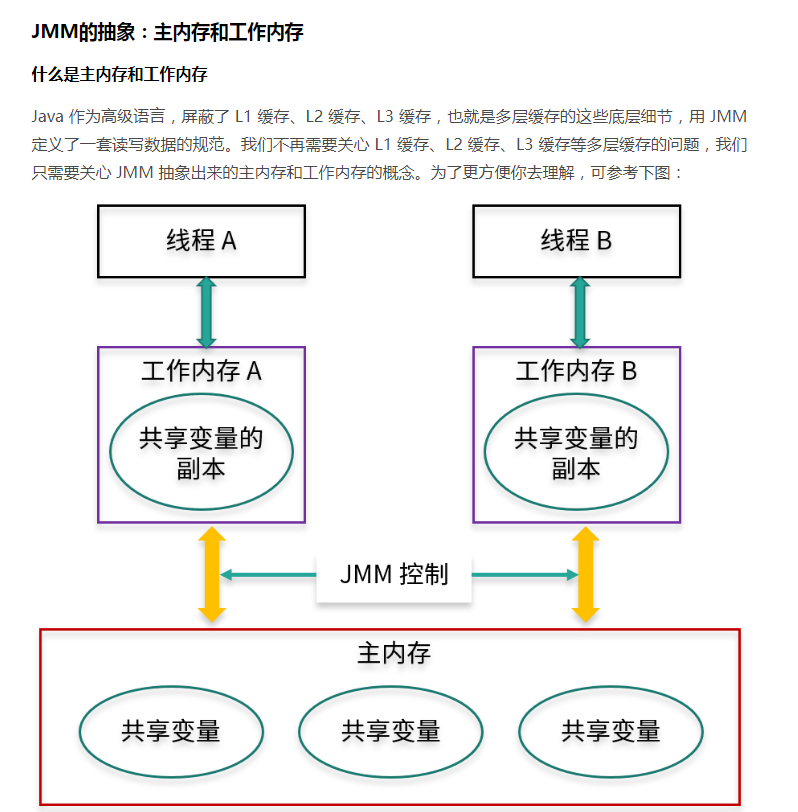
注意：原子操作 + 原子操作 != 原子操作

1. **什么是“内存可见性”问题？**

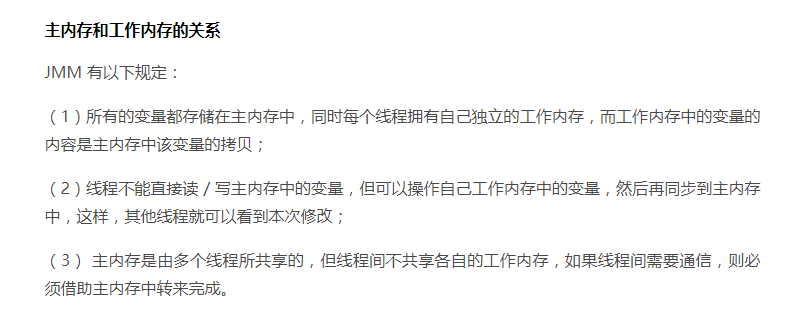
能够保证可见性的措施：volatile 、synchronized、Lock、并发集合、happens-before 原则



1. **主内存和工作内存的关系？**
2. 什么是主内存和工作内存



1. 关系



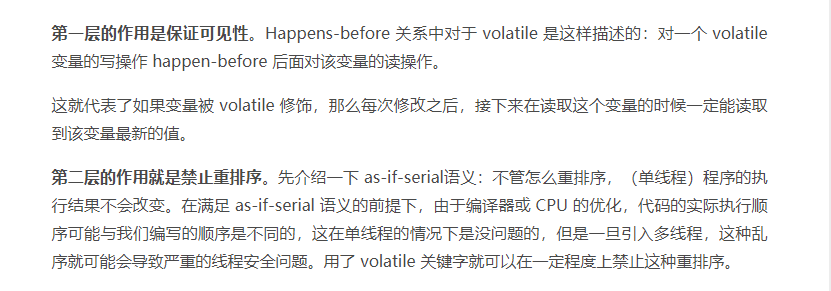
1. **什么是 happens-before 关系？**

Happens-before 关系是用来描述和可见性相关问题的：如果第一个操作 happens-before 第二个操作（也可以描述为，第一个操作和第二个操作之间满足 happens-before 关系），那么我们就说第一个操作对于第二个操作一定是可见的，也就是第二个操作在执行时就一定能保证看见第一个操作执行的结果。

如：单线程规则、锁操作规则（synchronized 和 Lock 接口等）、volatile 变量规则、线程启动规则、线程 join 规则、中断规则、并发工具类的规则等。

1. **volatile作用**

适用场景：布尔标记位、作为触发器



1. **volatile 和 synchronized 的关系**



1. **单例模式的双重检查锁模式为什么必须加 volatile？**

