# JVM 更底层实现 – 规范 和 商用实现的区别 hotspot – 官方文档

理解： JVM运行时数据区设计，多个内存区间进行交互势必会有问题

Java语言 提出 内存模型的相关规范

# 为什么不停止

1. 如何让他正确停止

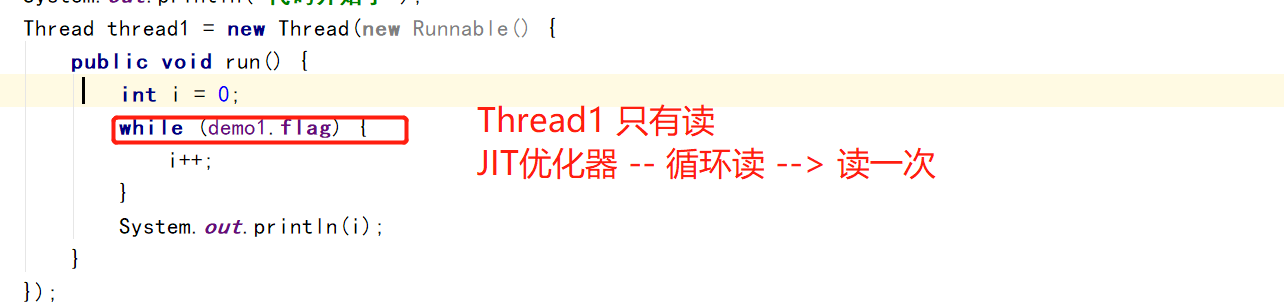
Volatile: 1. 根据内存模型的规定，保持可见性

1. ACC\_VOLATILE 访问控制，可以保证没有缓存立马可见

启动参数 -client -server 重大区别在于jit优化

可见性 ：

1. CPU等其他缓存，导致可见性（短时间内）
2. 重排序可能导致可见性



运行三种：编译、解释、混合

编译：字节码 --- jit提前编译 -- 汇编

解释：字节码 – 一段段编译 – 汇编

混合 ：– 运行的过程中，JIT编译器生效，针对热点代码进行优化

**while** (demo1.**flag**) {  
 i++;  
}

// 优化后的伪代码 --- 1. 旁敲侧击 2. 查看优化的汇编代码jitwatch

boolean f = demo1.**flag;**

**if(f){**

**while(true) {**

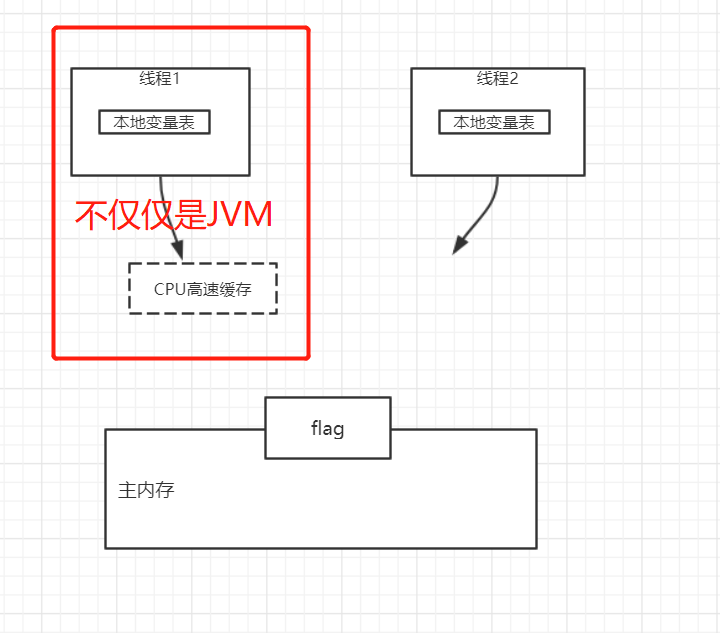
**i++;**

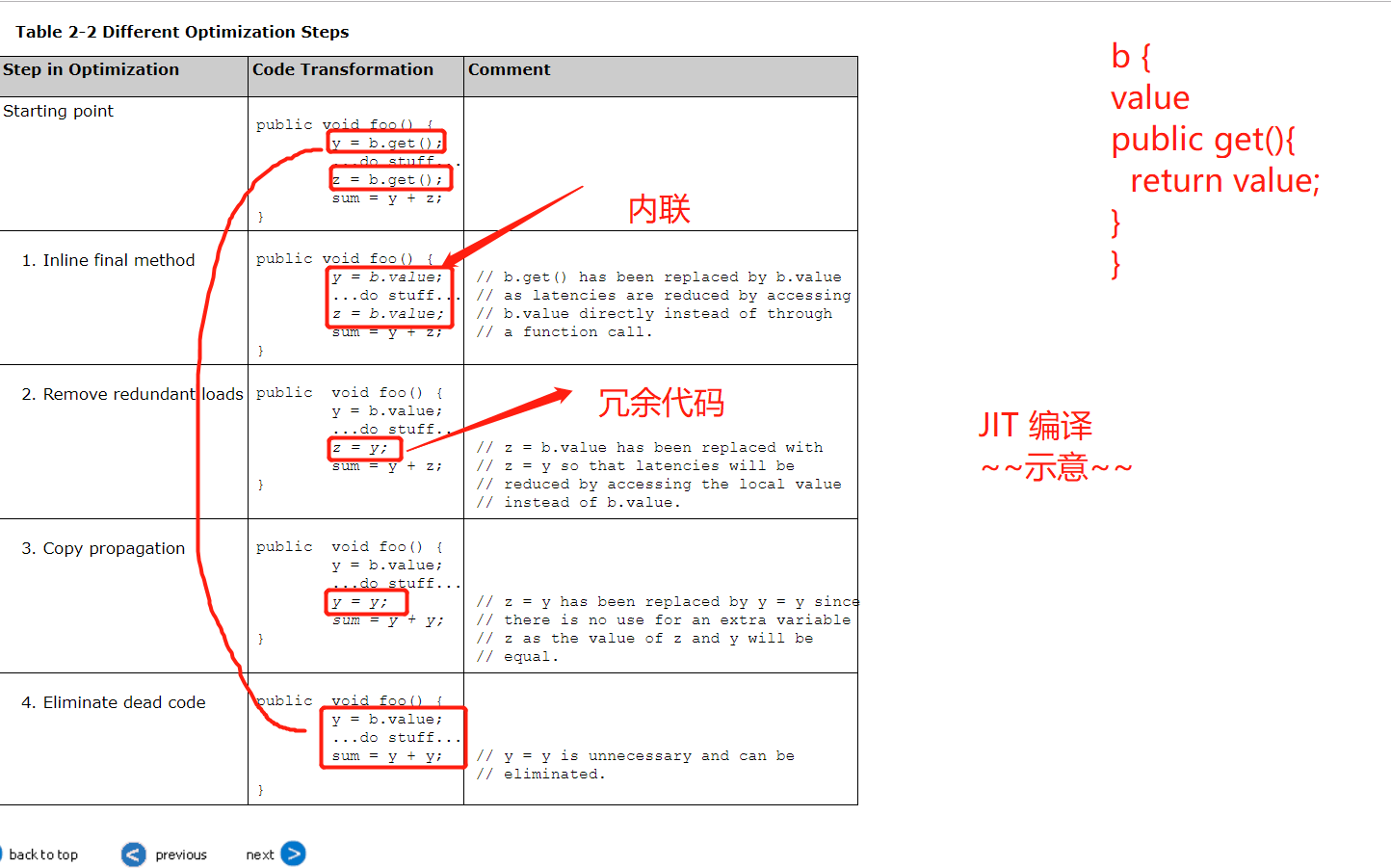
**}**

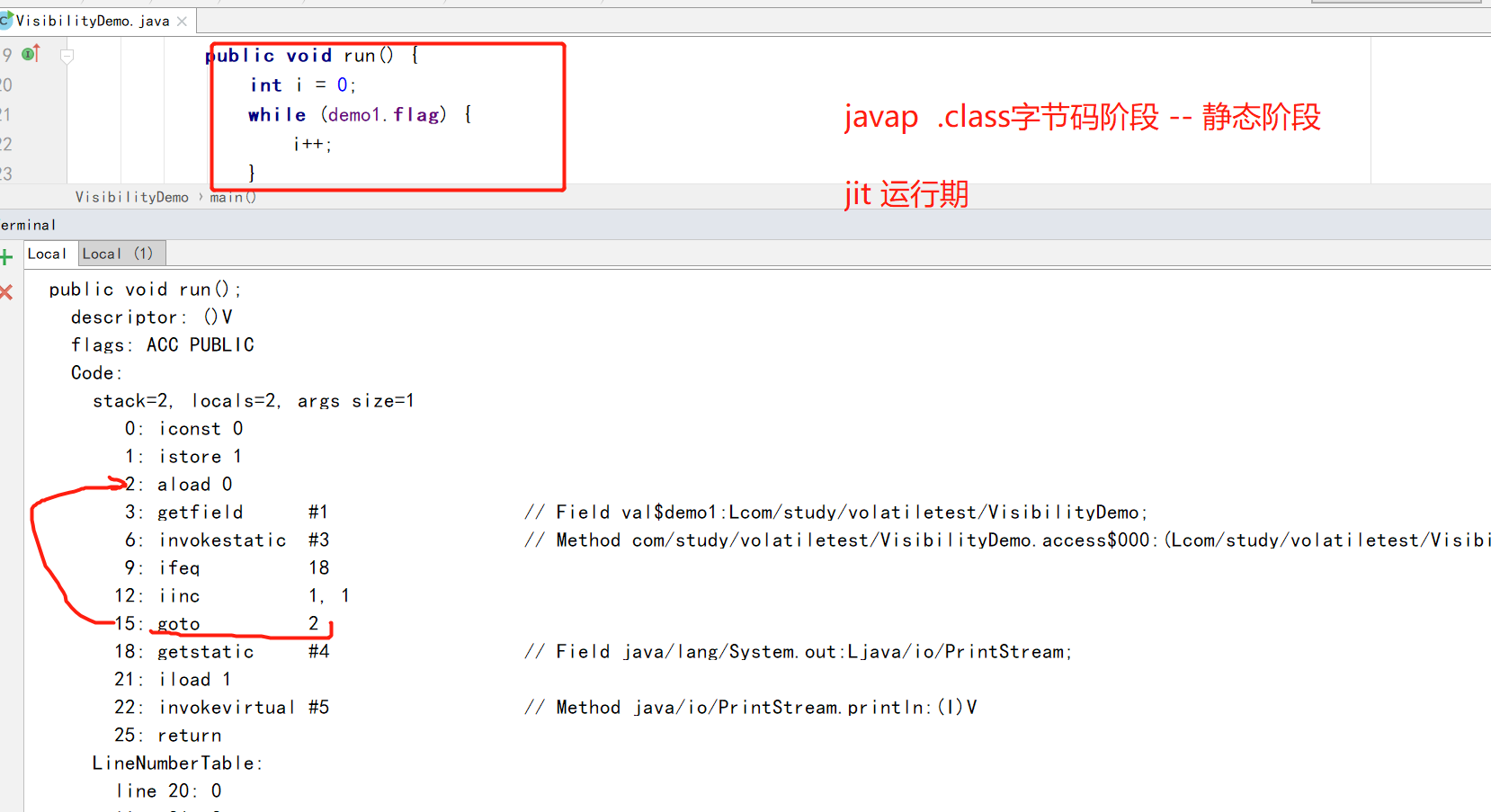
**}**

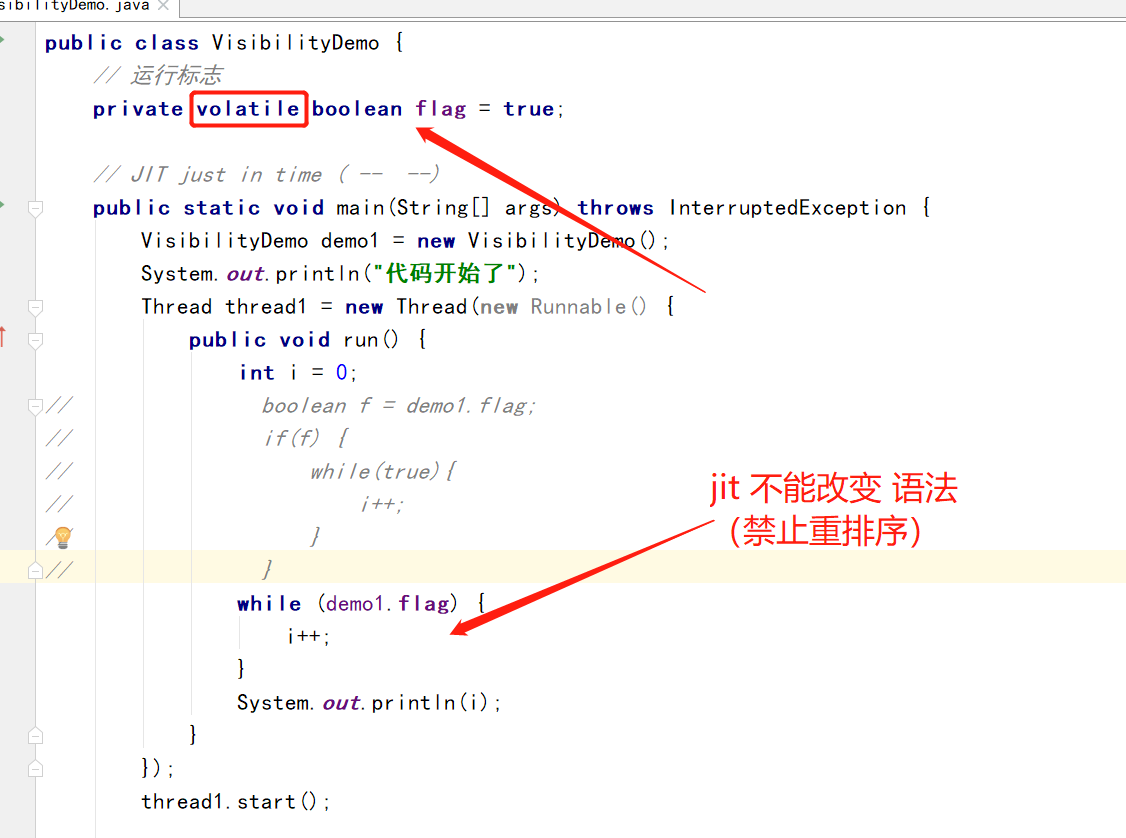
Java如何解决

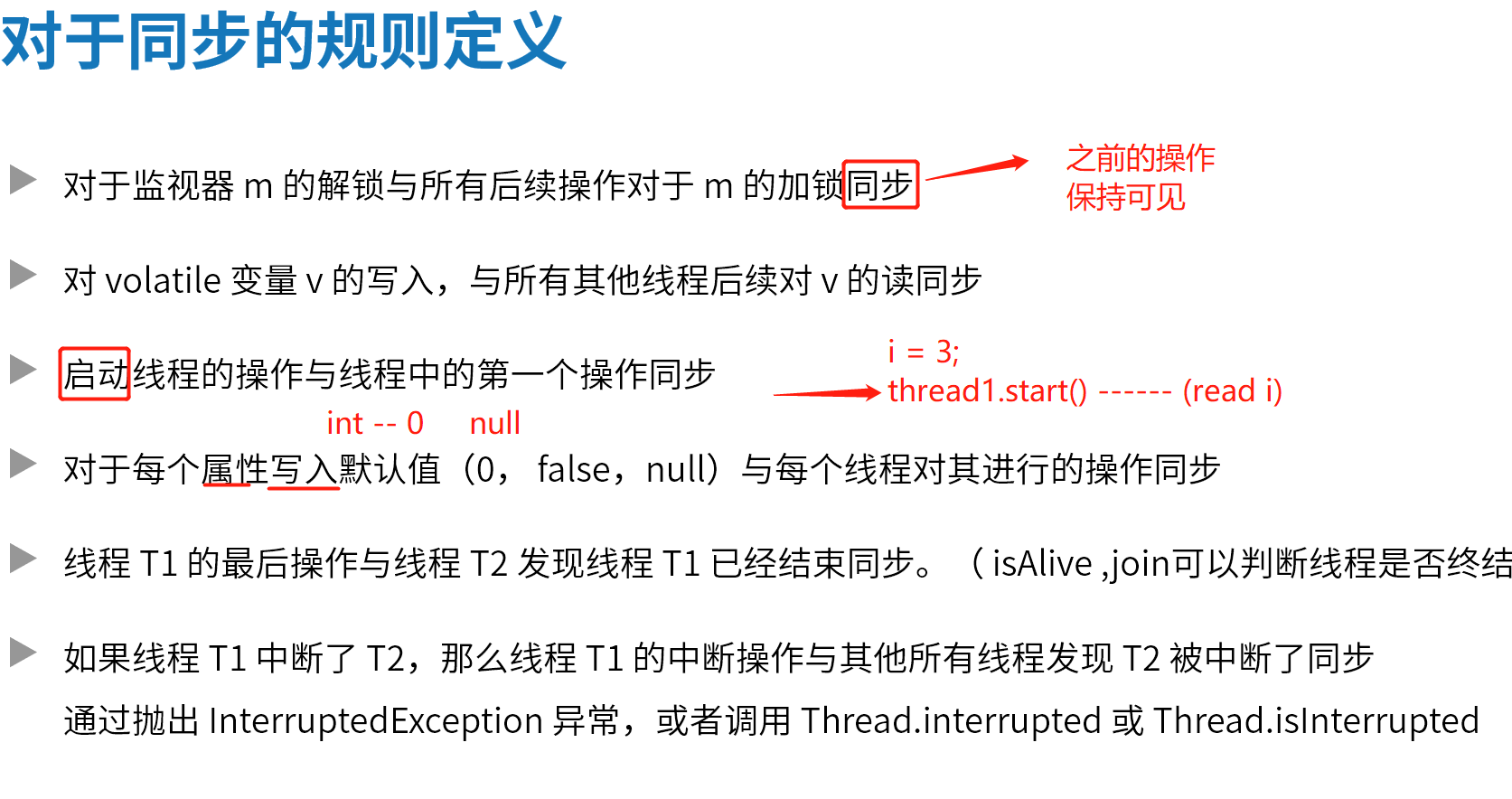
工作内存和主内存概念

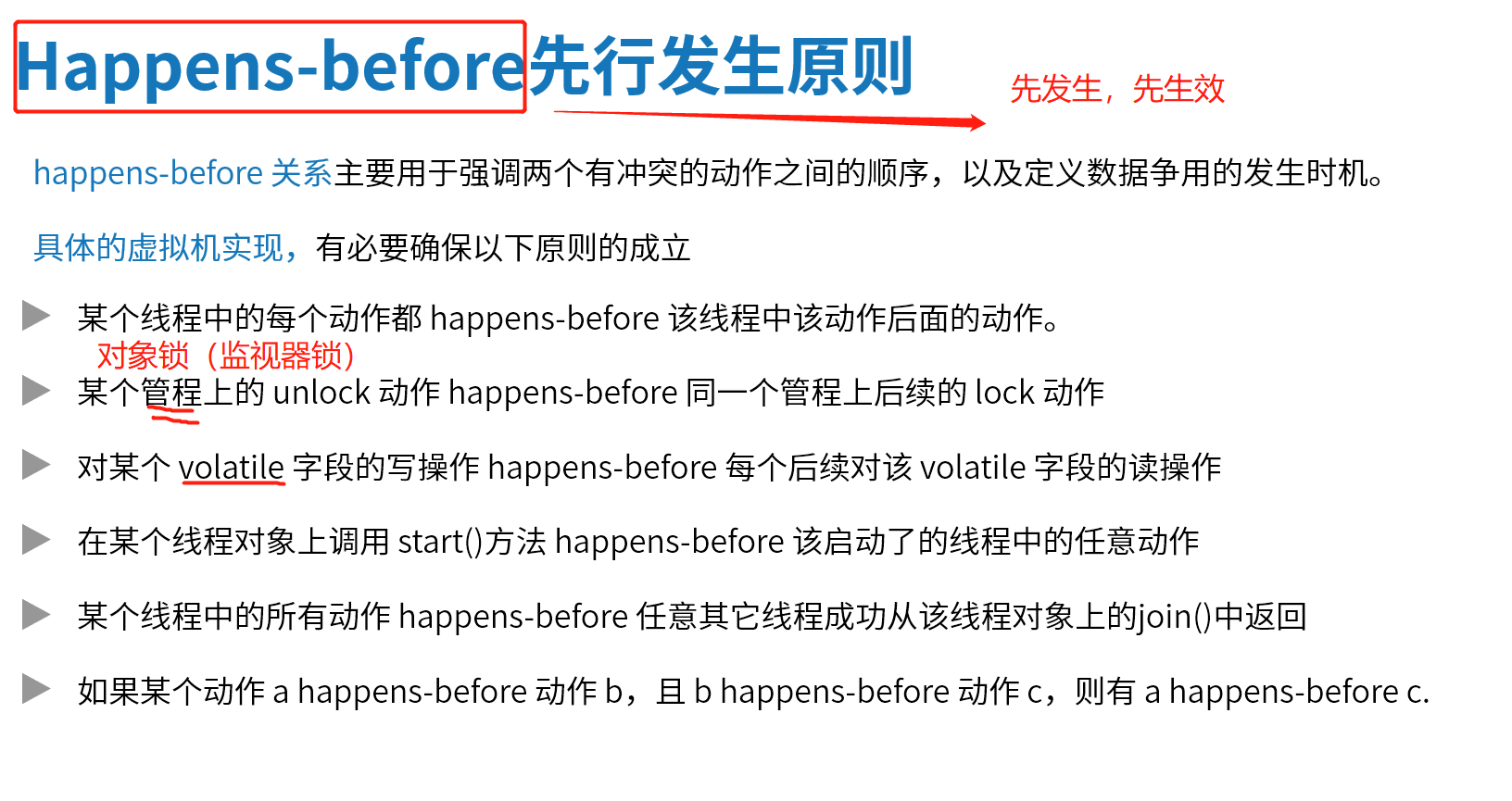












外部资料地址：

<https://docs.oracle.com/cd/E13150_01/jrockit_jvm/jrockit/geninfo/diagnos/underst_jit.html>

<https://docs.oracle.com/javase/specs/jls/se8/html/jls-17.html#jls-17.4>

<https://docs.oracle.com/javase/specs/jvms/se8/html/jvms-4.html>

https://docs.oracle.com