**JAVA多线程与并发**

## 进程和线程的区别

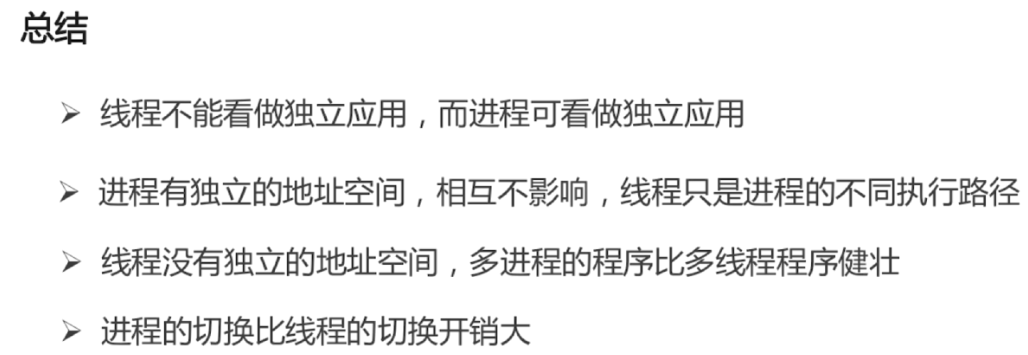
### 1.概念

进程是资源分配的最小单位，线程是cpu调度的最小单位。

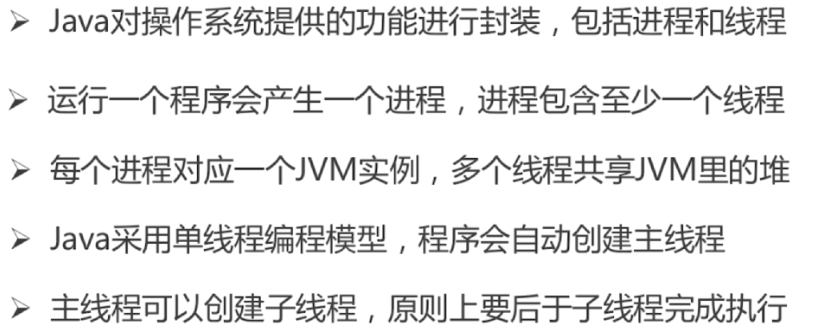
→所有与进程相关的资源，都被记录在PCB中

→进程是抢占处理机的调度单位；线程属于某个进程，共享其资源

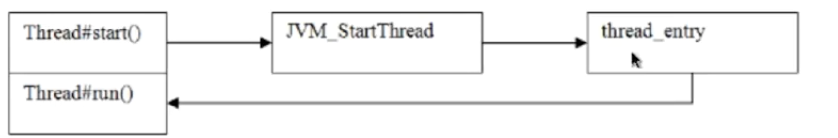
→线程只由堆栈寄存器，程序计数器和TCB组成



### java中进程和线程的关系



## thread中的start和run方法的区别



过程分析：start方法调用 JVM的startTread方法，创建一个子线程，然后这个子线程再去调用run()方法。

♦调用start()方法会创建一个新的子线程并启动

♦run()方法只是Thread的一个普通方法的调用

## Thread和Runnable是什么关系

♦Thread是实现了Runnable接口的类，是的run支持多线程。

♦因类的单一继承原则，推荐多食用Runnable接口

## 如何给run()方法传参（了解）

实现方式主要有三种

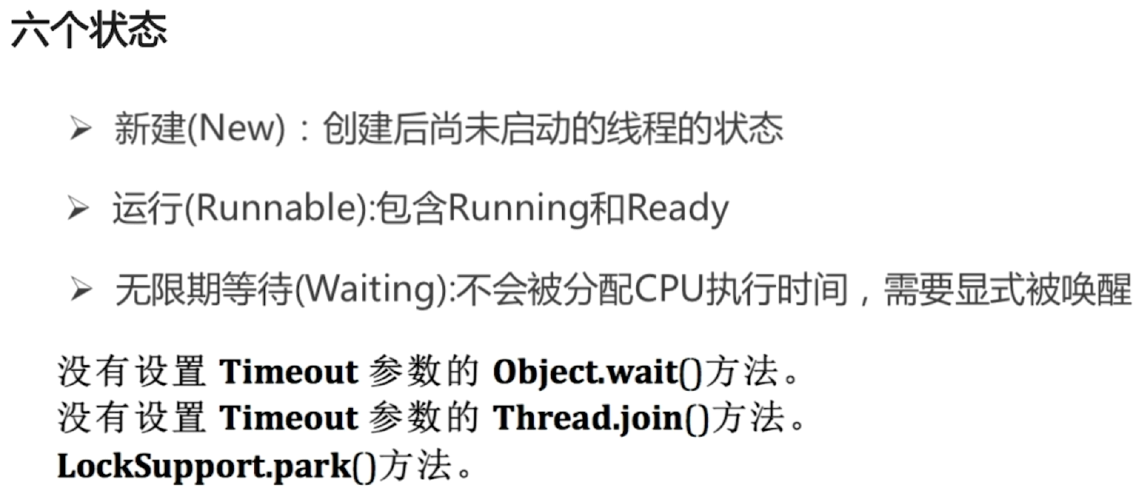
1. 构造函数传参
2. 成员变量传参
3. 回调函数传参

## 如何实现处理线程的返回值

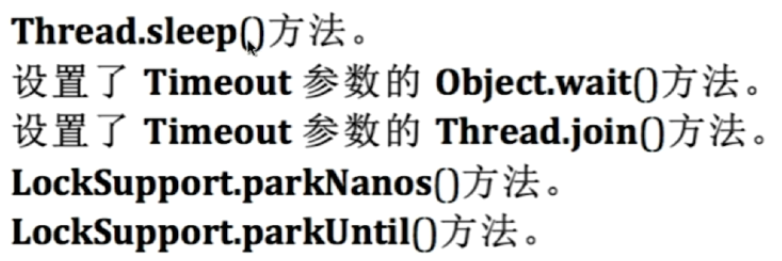
实现的方式有三种

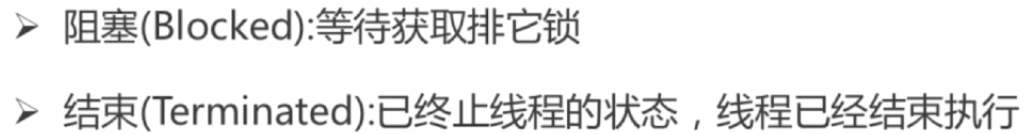
1. 主线程等待法
2. 使用Thread类中的join()方法阻塞当前的线程以等待子线程处理完毕
3. 通过callable接口实现，通过FutureTask 或者 线程池获取。

## 线程的状态

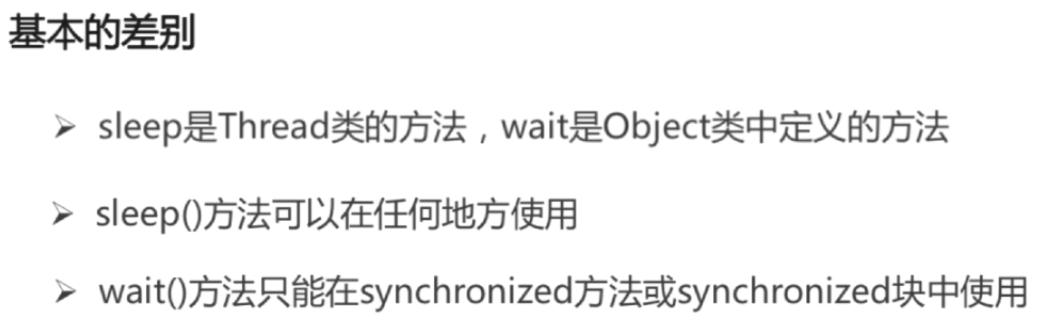


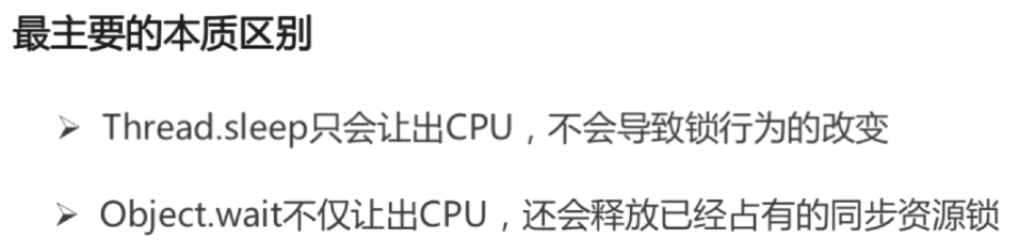






## sleep和wait的区别



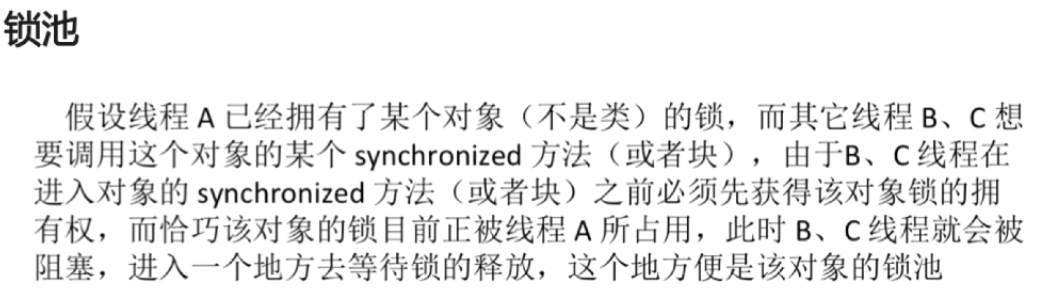


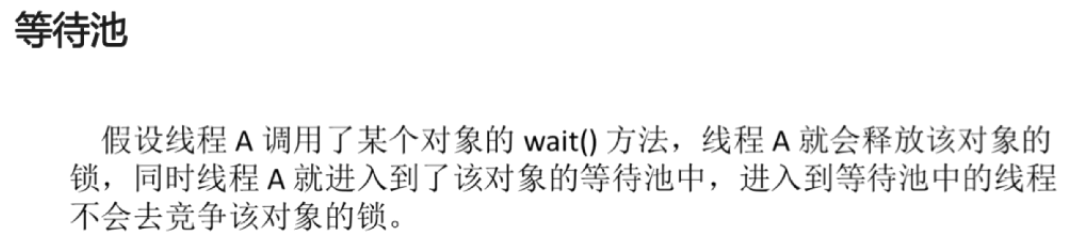
## notify和 notifyAll的区别

### 1.锁池和等待池的概念

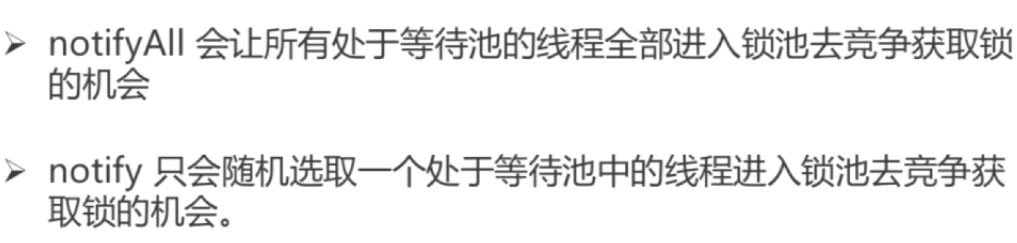
♦锁池EntryList

♦等待池WaitSet

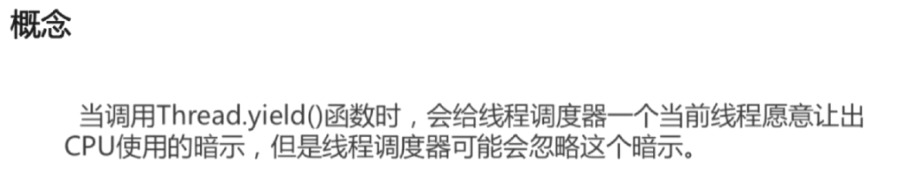




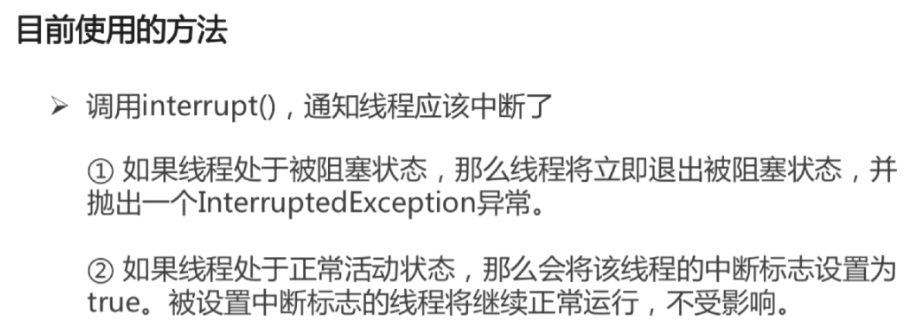
### 2.notify和 notifyAll的区别



## Yield



## 如何中断线程

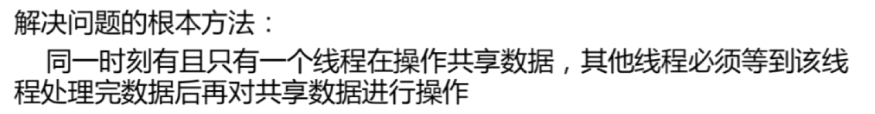


## Synchronized

### 线程安全问题的主要原因

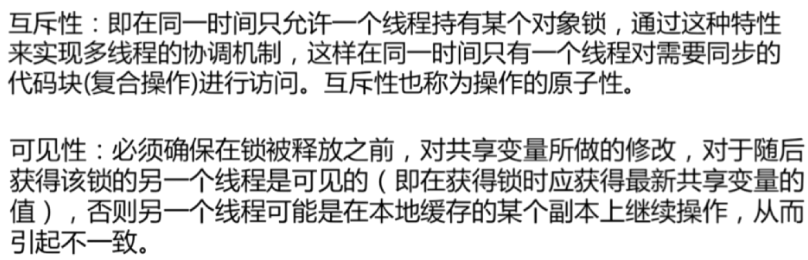
→存在共享数据（也称临界数据）

→存在多条线程共同操作这些共享数据



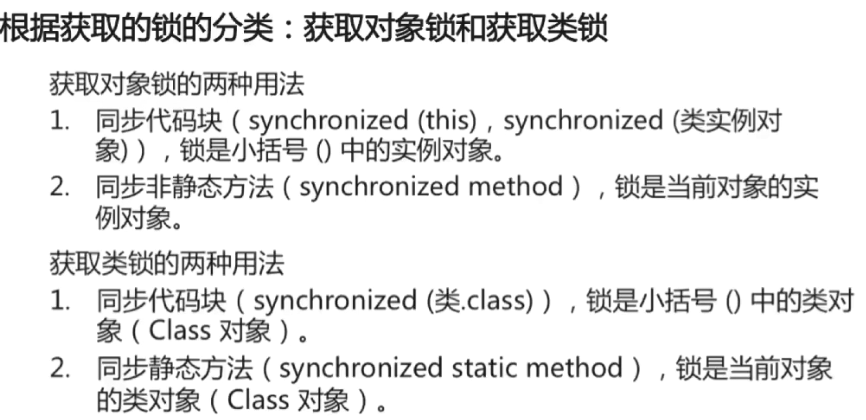
### 互斥锁

#### （1）互斥锁的特性

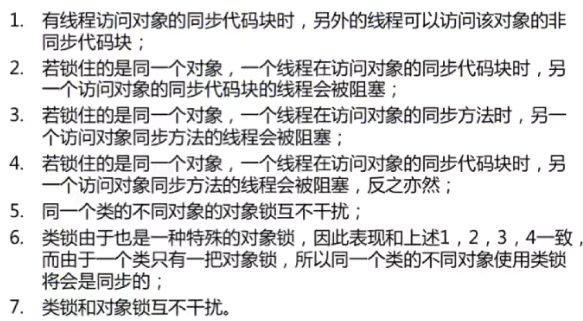


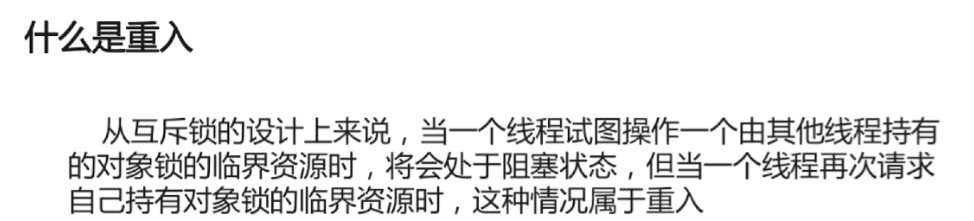
**Synchronized锁的不是代码，锁的是对象**

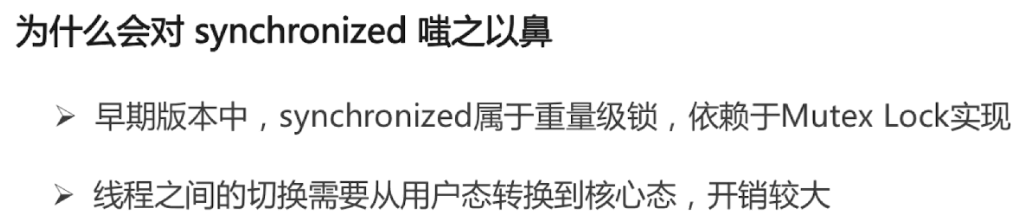
#### 锁的分类



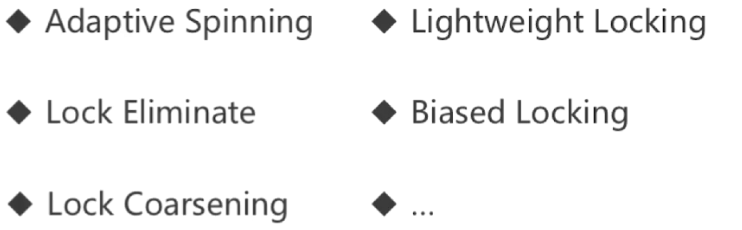
#### 对象锁和类锁的总结



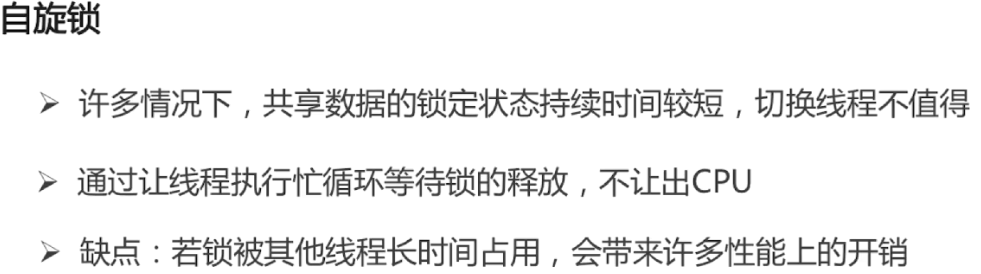




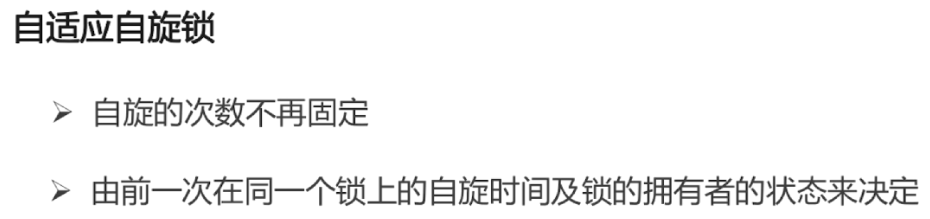




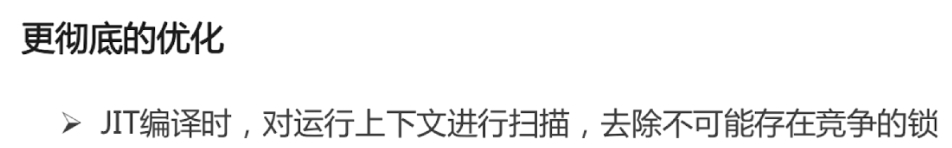
#### 自旋锁与自适应自旋锁



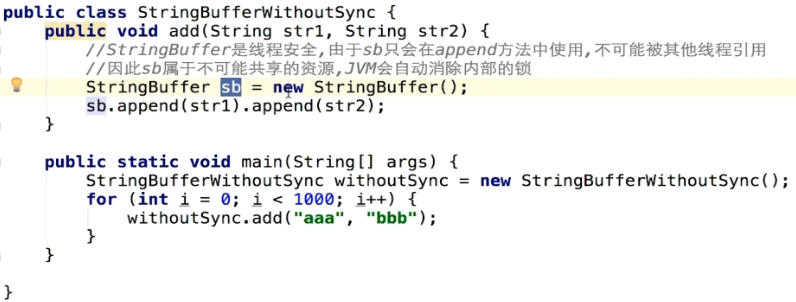
Java4时已存在，但是默认关闭，java6时默认开启。



#### 锁消除

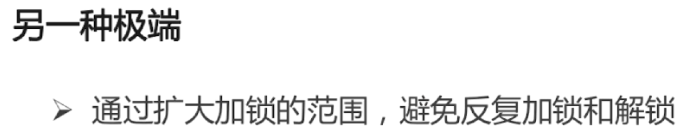


实例：

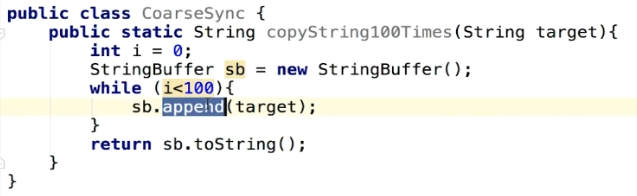


|  |
| --- |
| append()底层,使用了synchronized锁 |

#### 锁粗化



实例：一连串的append（）操作

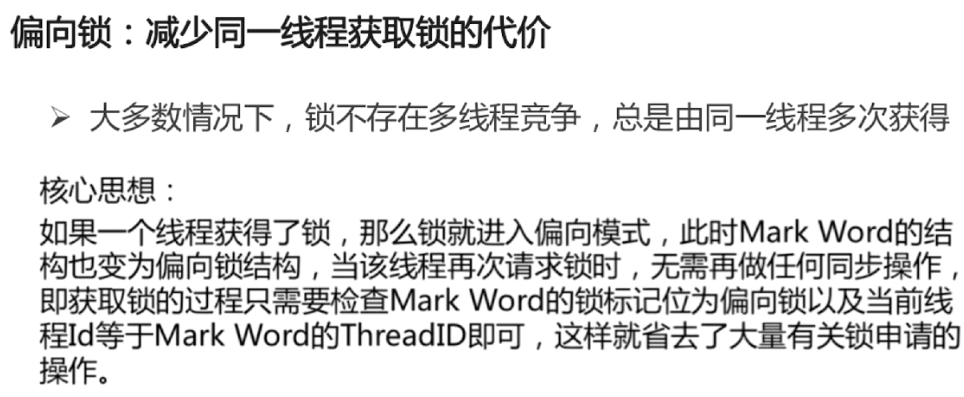


#### synchronized的四种状态

Synchronized属于悲观锁。





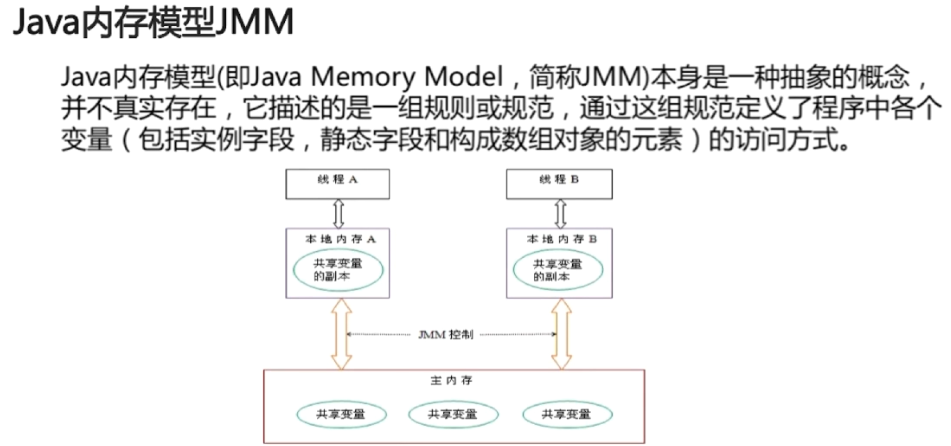


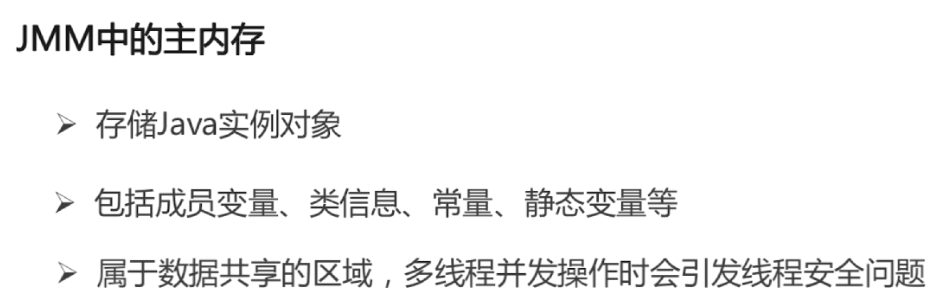


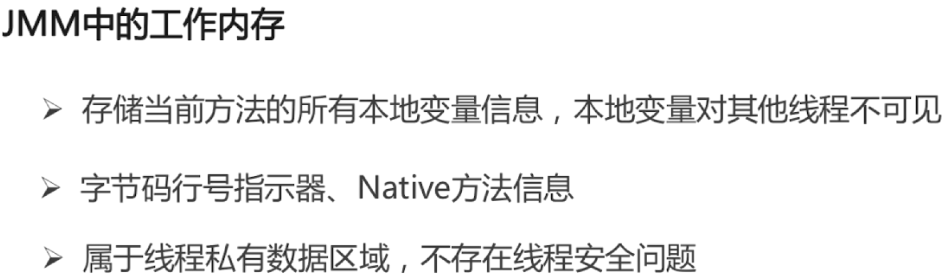


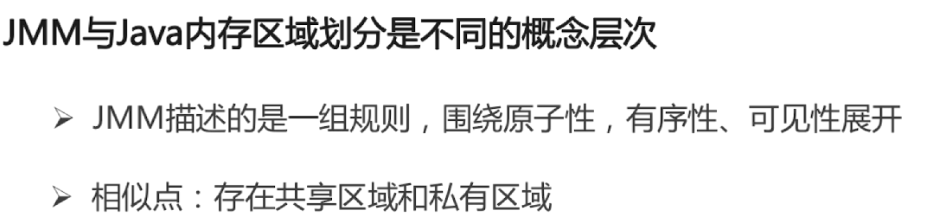


### JMM中的主内存和工作内存





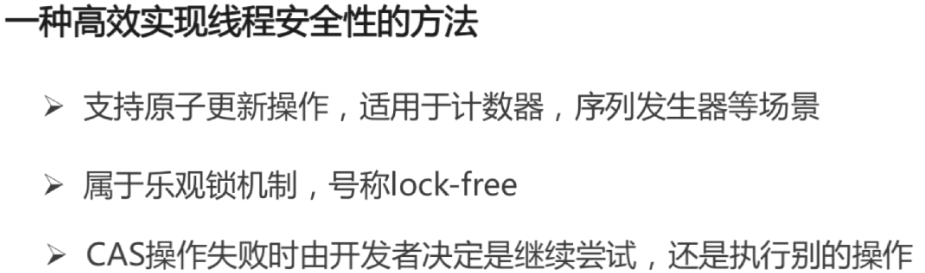


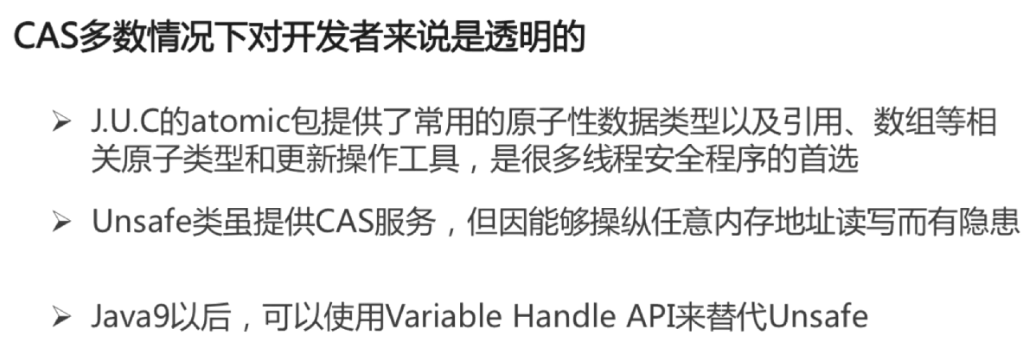


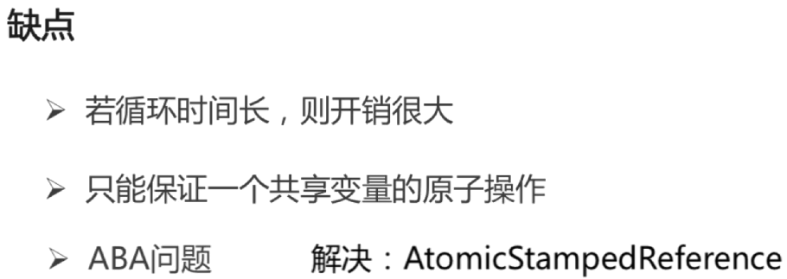
## 十二、JMM如何解决可见性问题

## 十三、CAS

|  |
| --- |
| Synchronized显然是一个悲观锁，因为它的并发策略是悲观的：不管是否会产生竞争，任何的数据操作都必须要加锁、用户态核心态转换、维护锁计数器和检查是否有被阻塞的线程需要被唤醒等操作。  乐观锁的核心算法是CAS（CompareandSwap，比较并交换），它涉及到三个操作数：内存值、预期值、新值。当且仅当预期值和内存值相等时才将内存值修改为新值。这样处理的逻辑是，首先检查某块内存的值是否跟之前我读取时的一样，如不一样则表示期间此内存值已经被别的线程更改过，舍弃本次操作，否则说明期间没有其他线程对此内存值操作，可以把新值设置给此块内存。 |







## 十四、java线程池

