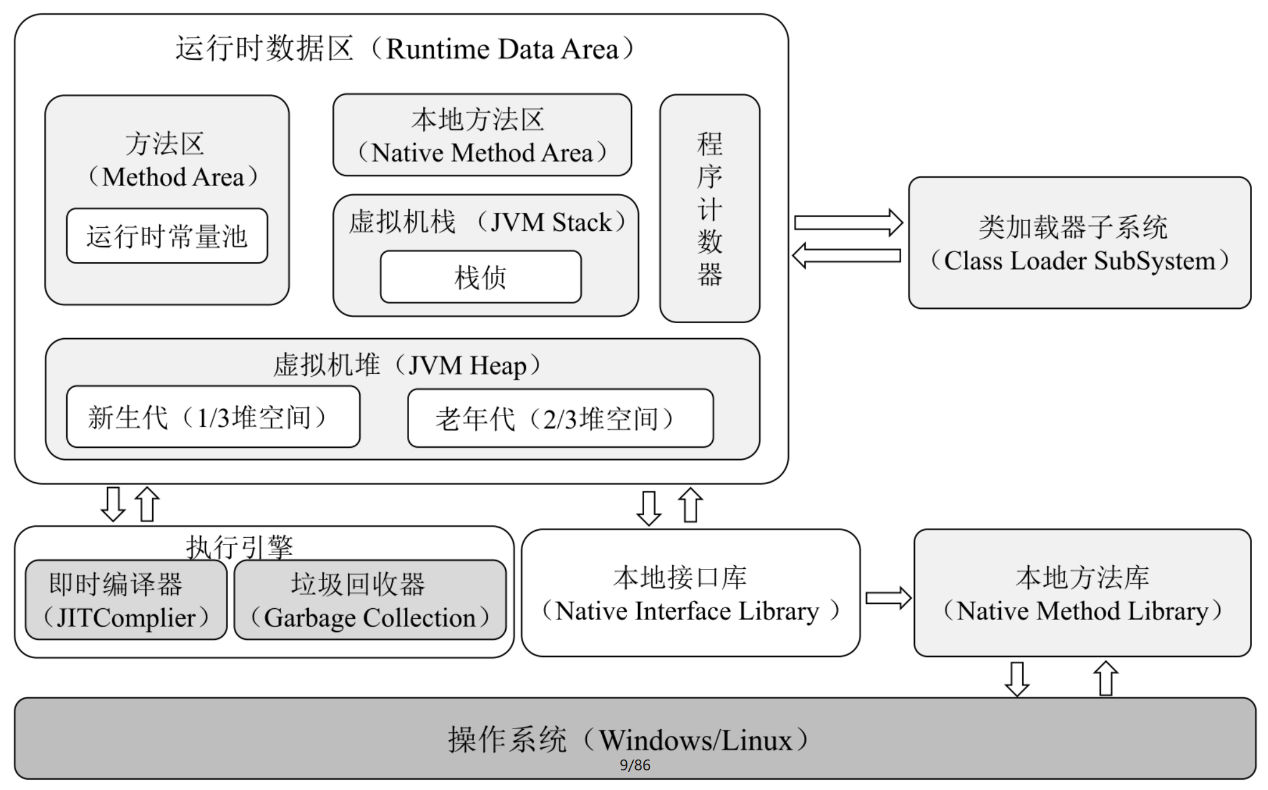
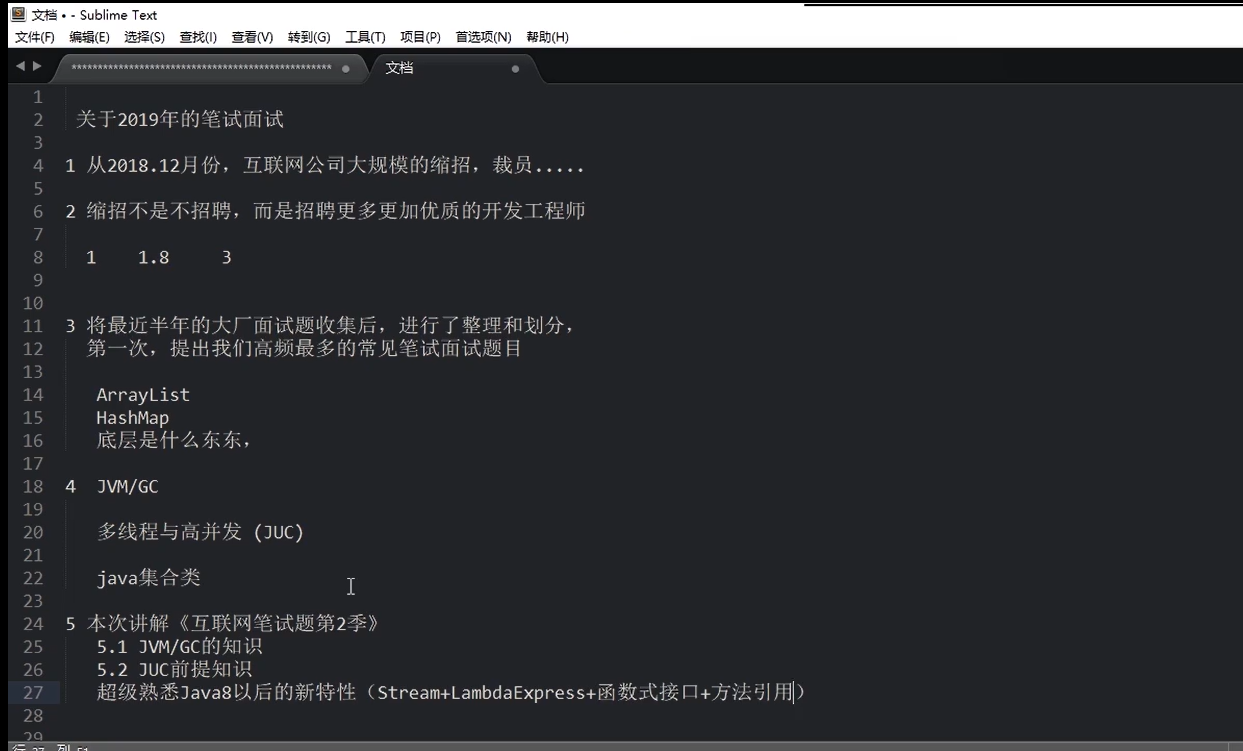
Java虚拟机包括一个**类加载器子系统**（Class Loader SubSystem）、**运行时数据区**（Runtime Data Area）、**执行引擎**和**本地接口库**（Native Interface Library）。**本地接口库通过调用本地方法库（Native Method Library）与操作系统交互**，如图1-1所示。



**尚硅谷：**



**Package java.util.concurrent**



# 1.请谈谈你对volatile的理解

## 1.volatile是Java虚拟机提供的轻量级的同步机制

## volatile三大特性：

1.1.保证可见性

1.2.不保证原子性

1.3.禁止指令重排

## 2. JMM特性你谈谈：线程安全性获得保

2.1.可见性

2.2.原子性

2.3.有序性

2.4.VolatileDemo代码演示可见性+原

JVM:java虚拟机

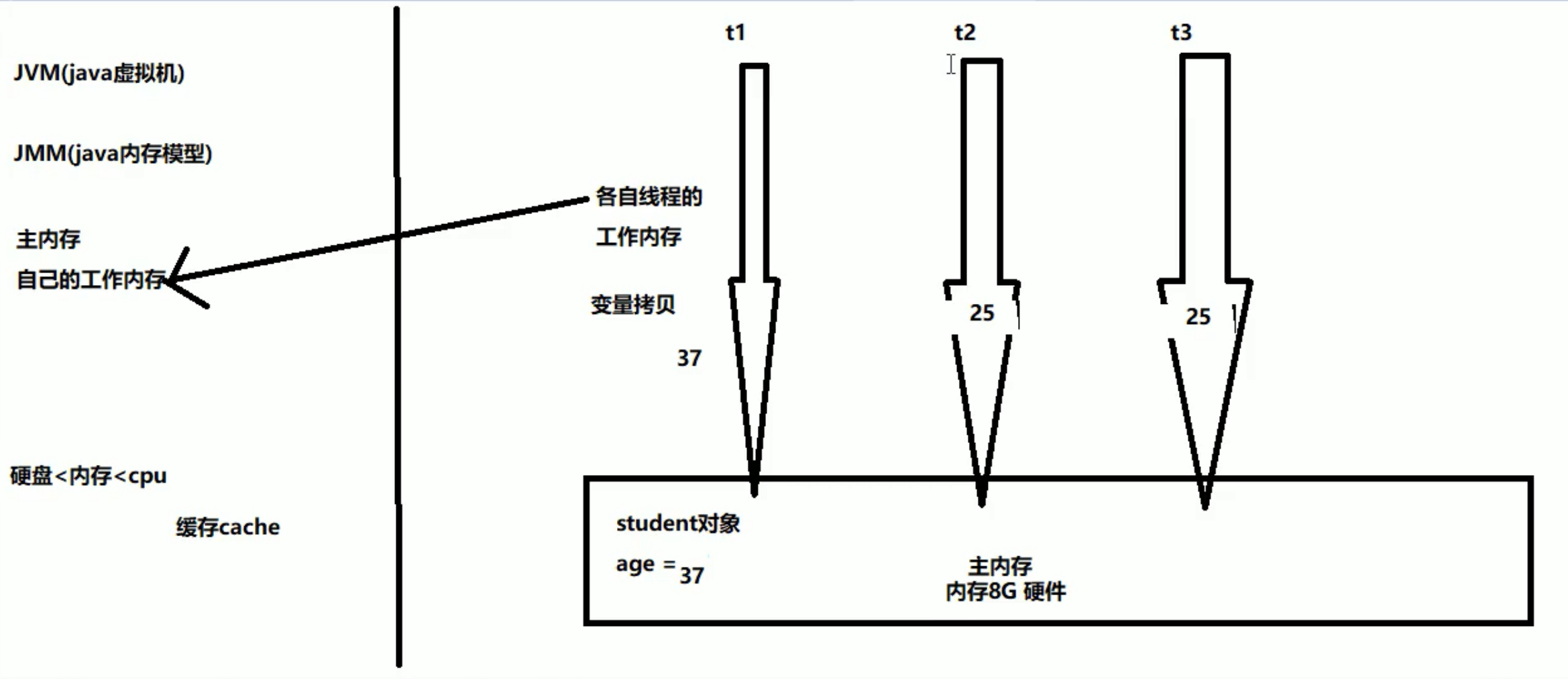
JMM:java内存模型

**JMM你谈谈：**

JMM(Java内存模型Java Memory Model， 简称JMM)本身是一-种抽象的概念并不真实存在，它描述的是一组规则或规范，通过这组规范定义了程序中各个变量(包括实例字段，静态字段和构成数组对象的元素)的访问方式。

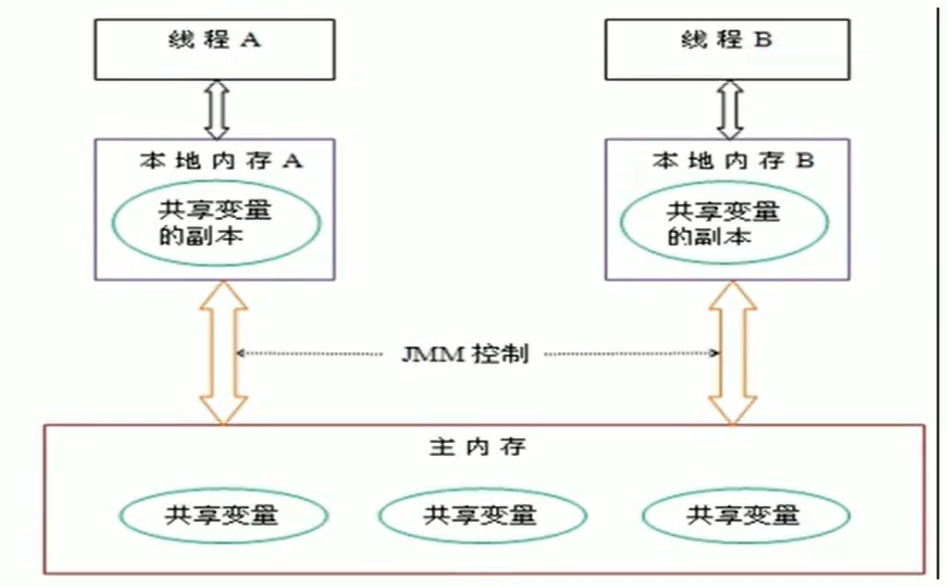
### JMM关于同步的规定:

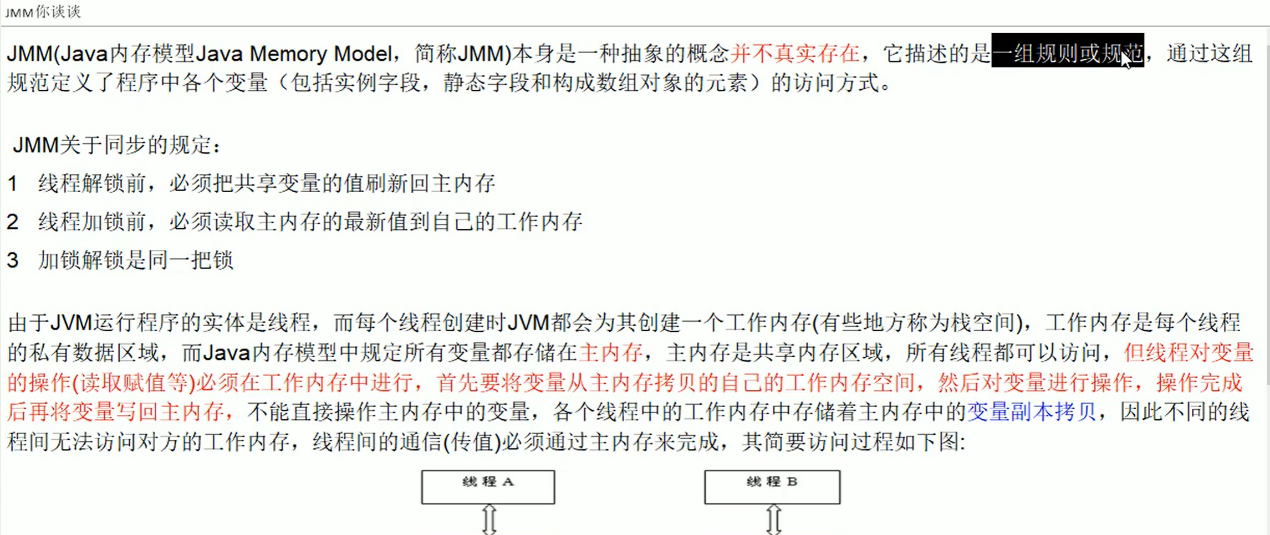
* **线程解锁前，必须把共享变量的值刷新回主内存**
* **线程加锁前，必须读取主内存的最新值到自己的工作内存**
* **加锁解锁是同一把锁**



### JMM内存模型之可见性：

由于JVM运行程序的实体是线程，而每个线程创建时JVM都会为其创建一个工作内存(有些地方称为栈空间)，工作内存是每个线程的私有数据区域，而Java内 存模型中规定所有变量都存储在主内存，主内存是共享内存区域，所有线程都可以访问，但线程对变量的操作(读取赋值等)必须在工作内存中进行，首先要将变量从主内存拷贝的自己的工作内存空间，然后对变量进行操作，操作完成后再将变量写回主内存，不能直接操作主内存中的变量，各个线程中的工作内存中存储着主内存中的变量副本拷贝，因此不同的线程间无法访问对方的工作内存，线程间的通信(传值)必须通过主内存来完成，其简要访问过程如下图:





## 3.你在哪些地方用到过volatile?

# 2.CAS你知道吗？

# 3.原子类AtomicInteger的ABA问题谈谈？原子更新引用知道吗？

# 4.我们知道ArrayList是线程不安全，请编码写一个不安全的案例并给出解决方案.

# 5.公平锁/非公平锁/可重入锁/递归锁/自旋锁谈谈你的理解？请手写一个自旋锁

# 6.CountDownLatch/CyclicBarrier/信号量使用过吗？

# 7.阻塞队列知道吗？

# 8.线程池用过吗？线程执行器谈谈你的理解？

# 9.线程池用过吗？生产上你如何设置合理参数

# 10.死锁编码及定位分析