Java内存模型

线程安全

线程安全

未实现线程安全	通过synchronized实现线程安全	concurrent包下的实现
• HashMap	• HashTable	• ConcurrentHashMap
• ArrayList	• Vector	• CopyOnWriteArrayList
• HashSet	Collections .synchronizedMap(Map<K,V>) .synchronizedList(List<T>) .synchronizedSet(Set<T>)	• CopyOnWriteArraySet

CopyOnWrite （写入时复制）：一种程序设计的优化策略。多个调用者同时请求相同资源，他们会共同获取相同的指针指向相同的资源，直到某个调用者试图修改资源的内容时，系统才会真正复制一份专用副本给该调用者，而其他调用者所见到的最初的资源仍然保持不变。此作法主要的优点是如果调用者没有修改该资源，就不会创建副本，多个调用者只是读取操作时可以共享同一份资源。

Java并发编程

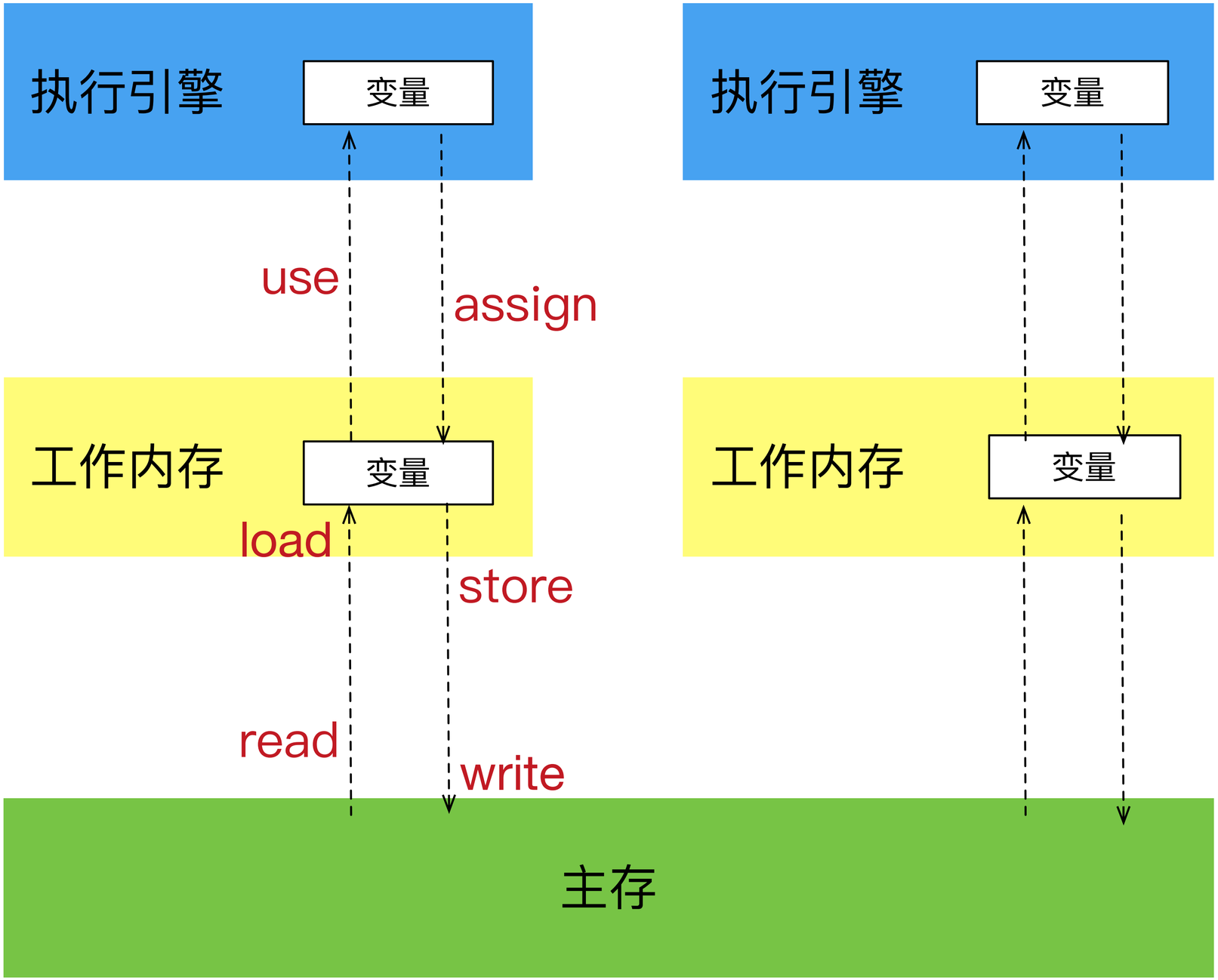
- 原子性
  - 一个操作是不可中断的。即使是在多个线程一起执行的时候，一个操作一旦开始，就不会被其它线程干扰
- 可见性
  - 当多个线程访问同一个变量时，一个线程修改了这个变量的值，其他线程能够立即看得到修改的值
- 有序性
  - 程序执行的顺序按照代码的先后顺序执行

	volatile	sychnorized	lock
原子性	✘	✔	✔
可见性	✔	✔	✔
有序性	✔	✔	✔

代码复杂度上升

8个底层内存操作

lock	主存变量	锁定	use	工作内存变量	使用
unlock	主存变量	解锁	assign	工作内存变量	赋值
read	主存变量	读取	store	工作内存变量	存储
load	工作内存变量	载入	write	主存变量	写入


执行引擎
变量
执行引擎
变量
use
assign
工作内存
变量
工作内存
变量
load
store
read
write
主存