**什么是线程安全：**

1. 一段代码可以在多线程中调用，且不会出现错误的交互，始终表现正确性。

2. 对于调用者来说无需采取同步操作。

**如何保证线程安全：**

1. 使用线程安全的数据结构

1. 线程不安全主要体现在对全局、静态变量的并发读写上。这些对象就是有状态的对象。应尽可能使用局部变量，减少对象可变状态能更大程度保证线程安全。

2. 注意程序中由于多个步骤的原子性，例如“读-修改-写入”和“先检查后执行”。

3. 使用并发容器，例如ConcurrentLinkedQueue，ConcurrentHashMap，ConcurrentLinkedHashMap，ConcurrentSkipListMap；使用原子计数器，例如AtomicInteger。

4. 使用同步代码块synchronized，或加重入锁、读写锁保护可变状态。

5. ArrayList，HashSet,LinkedList,HashMap等等非线程安全的类，通过[工具类Collections](http://how2j.cn/k/collection/collection-collections/369.html)转换为线程安全的，例如Collections.synchronizedList。

6. 重入锁，Lock锁机制， 通过创建Lock对象，采用lock()加锁，unlock()解锁，来保护指定的代码块

7. 使用Threadlocal 类来包装共享变量，做到每个线程有自己的copy。

J**ava的基本数据结构哪些是线程安全的：**

1. Java基本数据结构中Hashtable、Vector、Stack、ArrayBlocingQueue、DelayQueue、ThreadLocal、Atomic\*、ConcurrentHashMap、ConcurrentSkipListMap是线程安全的。