

# 并发编程

- [基础知识](#)
- - [并发编程的优缺点](#)
    - ◻ [为什么要使用并发编程（并发编程的优点）](#)
    - ◻ [并发编程有什么缺点](#)
    - ◻ [并发编程三要素是什么？在 Java 程序中怎么保证多线程的运行安全？](#)
    - ◻ [并行和并发有什么区别？](#)
    - ◻ [什么是多线程，多线程的优劣？](#)
  - [线程和进程区别](#)
  - - ◻ [什么是线程和进程？](#)
    - ◻ [进程与线程的区别](#)
    - ◻ [什么是上下文切换？](#)
    - ◻ [守护线程和用户线程有什么区别呢？](#)
    - ◻ [如何在 Windows 和 Linux 上查找哪个线程cpu利用率最高？](#)
    - ◻ [什么是线程死锁](#)
    - ◻ [形成死锁的四个必要条件是什么](#)
    - ◻ [如何避免线程死锁](#)
  - [创建线程的四种方式](#)
  - - ◻ [创建线程有哪几种方式？](#)
    - ◻ [说一下 runnable 和 callable 有什么区别？](#)
    - ◻ [线程的 run\(\)和 start\(\)有什么区别？](#)
    - ◻ [为什么我们调用 start\(\)方法时会执行 run\(\)方法，为什么我们不能直接调用 run\(\)方法？](#)
    - ◻ [什么是 Callable 和 Future？](#)
    - ◻ [什么是 FutureTask](#)
  - [线程的状态和基本操作](#)
  - - ◻ [说说线程的生命周期及五种基本状态？](#)
    - ◻ [Java 中用到的线程调度算法是什么？](#)
    - ◻ [线程的调度策略](#)
    - ◻ [什么是线程调度器\(Thread Scheduler\)和时间分片\(Time Slicing.\)？](#)
    - ◻ [请说出与线程同步以及线程调度相关的方法。](#)
    - ◻ [sleep\(\)和 wait\(\)有什么区别？](#)
    - ◻ [你是如何调用 wait\(\)方法的？使用 if 块还是循环？为什么？](#)
    - ◻ [为什么线程通信的方法 wait\(\), notify\(\)和 notifyAll\(\)被定义在 Object 类里？](#)
    - ◻ [为什么 wait\(\), notify\(\)和 notifyAll\(\)必须在同步方法或者同步块中被调用？](#)
    - ◻ [Thread 类中的 yield 方法有什么作用？](#)
    - ◻ [为什么 Thread 类的 sleep\(\)和 yield\(\)方法是静态的？](#)
    - ◻ [线程的 sleep\(\)方法和 yield\(\)方法有什么区别？](#)
    - ◻ [如何停止一个正在运行的线程？](#)
    - ◻ [Java 中 interrupted 和 isInterrupted 方法的区别？](#)
    - ◻ [什么是阻塞式方法？](#)
    - ◻ [Java 中你怎样唤醒一个阻塞的线程？](#)
    - ◻ [notify\(\)和 notifyAll\(\)有什么区别？](#)
    - ◻ [如何在两个线程间共享数据？](#)
    - ◻ [Java 如何实现多线程之间的通讯和协作？](#)
    - ◻ [同步方法和同步块，哪个是更好的选择？](#)
    - ◻ [什么是线程同步和线程互斥，有哪几种实现方式？](#)

- [在监视器\(Monitor\)内部, 是如何做线程同步的? 程序应该做哪种级别的同步?](#)
  - [如果你提交任务时, 线程池队列已满, 这时会发生什么](#)
  - [什么叫线程安全? servlet 是线程安全吗?](#)
  - [在 Java 程序中怎么保证多线程的运行安全?](#)
  - [你对线程优先级的理解是什么?](#)
  - [线程类的构造方法、静态块是被哪个线程调用的](#)
  - [Java 中怎么获取一份线程 dump 文件? 你如何在 Java 中获取线程堆栈?](#)
  - [一个线程运行时发生异常会怎样?](#)
  - [Java 线程数过多会造成什么异常?](#)
- [并发理论](#)
- [Java内存模型](#)
    - [Java中垃圾回收有什么目的? 什么时候进行垃圾回收?](#)
    - [如果对象的引用被置为null, 垃圾收集器是否会立即释放对象占用的内存?](#)
    - [finalize\(\)方法什么时候被调用? 析构函数\(finalization\)的目的是什么?](#)
  - [重排序与数据依赖性](#)
  - [为什么代码会重排序?](#)
  - [as-if-serial规则和happens-before规则的区别](#)
- [并发关键字](#)
- [synchronized](#)
    - [synchronized 的作用?](#)
    - [说说自己是怎么使用 synchronized 关键字, 在项目中用到了吗](#)
    - [说一下 synchronized 底层实现原理?](#)
    - [什么是自旋](#)
    - [多线程中 synchronized 锁升级的原理是什么?](#)
    - [线程 B 怎么知道线程 A 修改了变量](#)
    - [当一个线程进入一个对象的 synchronized 方法 A 之后, 其它线程是否可进入此对象的 synchronized 方法 B?](#)
    - [synchronized、volatile、CAS 比较](#)
    - [synchronized 和 Lock 有什么区别?](#)
    - [synchronized 和 ReentrantLock 区别是什么?](#)
  - [volatile](#)
    - [volatile 关键字的作用](#)
    - [Java 中能创建 volatile 数组吗?](#)
    - [volatile 变量和 atomic 变量有什么不同?](#)
    - [volatile 能使得一个非原子操作变成原子操作吗?](#)
    - [volatile 修饰符的有过什么实践?](#)
    - [synchronized 和 volatile 的区别是什么?](#)
  - [final](#)
    - [什么是不可变对象, 它对写并发应用有什么帮助?](#)
- [Lock体系](#)
- [Lock简介与初识AQS](#)
  - [Java Concurrency API 中的 Lock 接口\(Lock interface\)是什么? 对比同步它有什么优势?](#)
    - [乐观锁和悲观锁的理解及如何实现, 有哪些实现方式?](#)
    - [什么是 CAS](#)
    - [CAS 的会产生什么问题?](#)
    - [什么是死锁?](#)
    - [产生死锁的条件是什么? 怎么防止死锁?](#)
    - [死锁与活锁的区别, 死锁与饥饿的区别?](#)

- [多线程锁的升级原理是什么？](#)
  - [AQS\(AbstractQueuedSynchronizer\)详解与源码分析](#)
  - - [AQS 介绍](#)
    - [AQS 原理分析](#)
  - [ReentrantLock\(重入锁\)实现原理与公平锁非公平锁区别](#)
  - - [什么是可重入锁（ReentrantLock）？](#)
  - [读写锁ReentrantReadWriteLock源码分析](#)
  - - [ReadWriteLock 是什么](#)
  - [Condition源码分析与等待通知机制](#)
  - [LockSupport详解](#)
- [并发容器](#)
- - [并发容器之ConcurrentHashMap详解\(JDK1.8版本\)与源码分析](#)
  - - [什么是ConcurrentHashMap？](#)
    - [Java 中 ConcurrentHashMap 的并发度是什么？](#)
    - [什么是并发容器的实现？](#)
    - [Java 中的同步集合与并发集合有什么区别？](#)
    - [SynchronizedMap 和 ConcurrentHashMap 有什么区别？](#)
  - [并发容器之CopyOnWriteArrayList详解](#)
  - - [CopyOnWriteArrayList 是什么，可以用于什么应用场景？有哪些优缺点？](#)
  - [并发容器之ThreadLocal详解](#)
  - - [ThreadLocal 是什么？有哪些使用场景？](#)
    - [什么是线程局部变量？](#)
  - [ThreadLocal内存泄漏分析与解决方案](#)
  - - [ThreadLocal造成内存泄漏的原因？](#)
    - [ThreadLocal内存泄漏解决方案？](#)
  - [并发容器之BlockingQueue详解](#)
  - - [什么是阻塞队列？阻塞队列的实现原理是什么？如何使用阻塞队列来实现生产者-消费者模型？](#)
  - [并发容器之ConcurrentLinkedQueue详解与源码分析](#)
  - [并发容器之ArrayBlockingQueue与LinkedBlockingQueue详解](#)
- [线程池](#)
- - [Executors类创建四种常见线程池](#)
  - - [什么是线程池？有哪几种创建方式？](#)
    - [线程池有什么优点？](#)
    - [线程池都有哪些状态？](#)
    - [什么是 Executor 框架？为什么使用 Executor 框架？](#)
    - [在 Java 中 Executor 和 Executors 的区别？](#)
    - [线程池中 submit\(\) 和 execute\(\) 方法有什么区别？](#)
    - [什么是线程组，为什么在 Java 中不推荐使用？](#)
  - [线程池之ThreadPoolExecutor详解](#)
  - - [Executors和ThreaPoolExecutor创建线程池的区别](#)
    - [你知道怎么创建线程池吗？](#)
    - [ThreadPoolExecutor构造函数重要参数分析](#)
    - [ThreadPoolExecutor饱和策略](#)
    - [一个简单的线程池Demo: Runnable + ThreadPoolExecutor](#)
  - [线程池之ScheduledThreadPoolExecutor详解](#)

- [FutureTask详解](#)
- [原子操作类](#)
- ◦
  - [什么是原子操作? 在 Java Concurrency API 中有哪些原子类\(atomic classes\)?](#)
  - [说一下 atomic 的原理?](#)
- [并发工具](#)
- ◦ [并发工具之CountDownLatch与CyclicBarrier](#)
- - [在 Java 中 CyclicBarrier 和 CountdownLatch 有什么区别?](#)
- [并发工具之Semaphore与Exchanger](#)
- - [Semaphore 有什么作用](#)
  - [什么是线程间交换数据的工具Exchanger](#)
  - [常用的并发工具类有哪些?](#)