# Java面试题

### 进程和线程

线程和进程的概念、并行和并发的概念

并行：多个线程可以同时执行，每一个时间段，可以有多个线程同时执行。

并发：多个线程同时竞争一个位置，竞争到的才可以执行，每一个时间段只有一个线程在执行。

<https://blog.csdn.net/kisscatforever/article/details/81000016>

创建线程的方式及实现

继承Thread，实现Runnable接口，实现Callable接口，使用线程池

进程间通信的方式

<https://blog.csdn.net/wh_sjc/article/details/70283843>

说说 CountDownLatch、CyclicBarrier 原理和区别

简单的意思就是CountDownLatch用于同步时一个或多个线程等待其他线程的某些操作完后才继续进行,而CyclicBarrier用于若干线程需要阻塞在一个地方,够数了然后在同时进行.

<https://blog.csdn.net/ruanjian1111ban/article/details/57150913>

说说Semaphore 原理

Semaphore经常用于限制获取某种资源的线程数量。

<https://blog.csdn.net/qq_19431333/article/details/70212663>

说说Exchanger 原理

这两个线程通过exchange方法交换数据， 如果第一个线程先执行exchange方法，它会一直等待第二个线程也执行exchange，当两个线程都到达同步点时，这两个线程就可以交换数据，将本线程生产出来的数据传递给对方。

<https://blog.csdn.net/carson0408/article/details/79477280>

ThreadLocal 原理分析，ThreadLocal为什么会出现OOM，出现的深层次原理

由于ThreadLocalMap的生命周期跟Thread一样长，如果没有手动删除对应key就会导致内存泄漏，而不是因为弱引用。

每次使用完ThreadLocal，都调用它的remove()方法，清除数据。

<https://blog.csdn.net/bntx2jsqfehy7/article/details/78315161>

讲讲线程池的实现原理

线程池的几种实现方式

线程的生命周期，状态是如何转移的

可参考：《Java多线程编程核心技术》

### 锁机制

说说线程安全问题，什么是线程安全，如何保证线程安全

重入锁的概念，重入锁为什么可以防止死锁

产生死锁的四个条件（互斥、请求与保持、不剥夺、循环等待）

如何检查死锁（通过jConsole检查死锁）

volatile 实现原理（禁止指令重排、刷新内存）

synchronized 实现原理（对象监视器）

synchronized 与 lock 的区别

AQS同步队列

CAS无锁的概念、乐观锁和悲观锁

常见的原子操作类

什么是ABA问题，出现ABA问题JDK是如何解决的

乐观锁的业务场景及实现方式

Java 8并法包下常见的并发类

偏向锁、轻量级锁、重量级锁、自旋锁的概念

可参考：《Java多线程编程核心技术》

### 多线程

AtomicInteger底层实现原理；

synchronized与ReentraLock哪个是公平锁；

CAS机制会出现什么问题；

用过并发包下边的哪些类；

一个线程连着调用start两次会出现什么情况？

wait方法能不能被重写，wait能不能被中断；

线程池的实现？四种线程池？重要参数及原理？任务拒接策略有哪几种？

线程状态以及API怎么操作会发生这种转换；

常用的避免死锁方法；

* 避免一个线程同时获取多个锁。
* 避免一个线程在索内同时占用多个资源，尽量保证每个索只占用一个资源。
* 尝试使用定时索，使用 lock.tryLock(timeout) 来替代使用内部索机制。
* 对于数据库索，加锁和解锁必须在一个数据库连接里，否则会出现解锁失败的情况。

<https://blog.csdn.net/csdnones/article/details/50371688>

一个线程连着调用start两次会出现什么情况？（由于状态只有就绪、阻塞、执行，状态是无法由执行转化为执行的，所以会报不合法的状态！）

wait方法能不能被重写？（wait是final类型的，不可以被重写，不仅如此，notify和notifyall都是final类型的），wait能不能被中断；

Java线程池的核心属性以及处理流程；

Java线程池的几个参数的意义和实现机制；

Java线程池使用无界任务队列和有界任务队列的优劣对比；

Java中有哪些同步方案（重量级锁、显式锁、并发容器、并发同步器、CAS、volatile、AQS等）

多个线程同时读写，读线程的数量远远⼤于写线程，你认为应该如何解决并发的问题？你会选择加什么样的锁？

线程池内的线程如果全部忙，提交⼀个新的任务，会发⽣什么？队列全部塞满了之后，还是忙，再提交会发⽣什么？

synchronized关键字锁住的是什么东西？在字节码中是怎么表示的？在内存中的对象上表现为什么？

wait/notify/notifyAll⽅法需不需要被包含在synchronized块中？这是为什么？

乐观悲观锁的设计，如何保证原子性，解决的问题；

sync原理详细，sync内抛异常会怎样，死锁吗？还是释放掉？怎么排查死锁？死锁会怎样？有没有什么更好的替代方案？

多线程是解决什么问题的？线程池解决什么问题？

线程池，如何设计的，里面的参数有多少种，里面的工作队列和线程队列是怎样的结构，如果给你，怎样设计线程池？

怎么认为一个类是线程安全？线程安全的定义是什么？Java有多少个关键字进行同步？为什么这样设计？（聊了一大堆，一堆为什么）；

两个线程设计题。记得一个是：t1,t2,t3，让t1，t2执行完才执行t3，原生实现。

# 总结

这里有个重要的概念：监视器（monitor）。

JAVA中的每一个对象都有一个监视器，用来检测并发时的重入问题，在非多线程情况下，

监视器不起作用，在synchronized情况下监视器才起作用。

线程开始执行synchronized之前，必须要获得对象的监视器的锁，简称监视器锁。

那么是哪个对象呢？

如果使用synchronized块就是传入参数的对象。

如果使用synchronized实例方法就是this对象。

如果使用synchronized静态方法相当于类本身，该类的所有对象共享一把锁。

## 架构师专场之享受一场并发编程带来的视觉盛宴

### 温故而知新

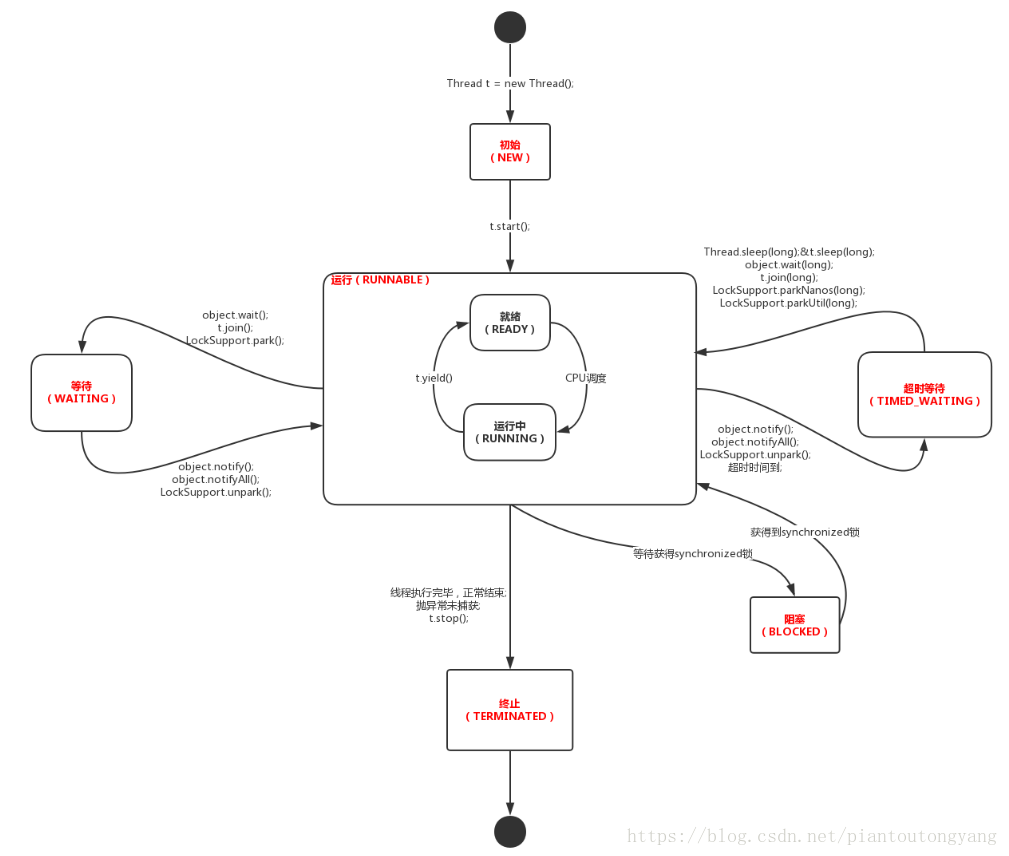
并发 🡪 你认知的并发是什么？

#### 多线程的实现方式

Thread Runnable Callable ExecutorService

#### Thread和Runnable的区别

#### 线程的5种状态，状态的转换



创建，就绪，运行，阻塞，终止

阻塞：

A．等待阻塞:wait()

B．同步阻塞:线程抢占锁的时候，锁被其他线程所占用

C．其他阻塞:sleep(),join()

#### (4)线程安全

线程安全，非线程安全，同步，异步，阻塞，非阻塞

集合：

ArrayList 线程安全的？非安全？

HashMap 🡪 ConcurrentHashMap 线程安全的解决方法

HashTable

#### (5)造成线程不安全的原因是什么

A.可见性

B.原子性

#### (6)synchronized

加锁

当前的对象锁this，自定义锁Object Lock

同类型锁之间互斥，不同类型锁之间互相不干扰

#### (7)同步和异步

#### (8)线程之间的通信

wait notify 必须要结合synchronized一起使用

#### (9)volatile关键字

保证一个可见性 不能保证原子性

#### (10)做一个小案例

利用多线程去实现一个阻塞的安全队列

阻塞：如果取不到元素，就等

非阻塞：如果取不到元素，就返回空

Server.accept() 🡪 while(true)

重点：Atomic 🡪 保证原子性

对原来的值进行一个判断

CAS Compare and Swap/Set 实际上就是一个乐观锁

加锁的做法，原子性和可见性 性能消耗太大 悲观锁

## 轻松搞定面试中各种并发问题

### 从线程的生命周期开始着手

### 2.synchronized

#### a)锁类型

#### b)对象头

#### c)源码

### 3.wait和notify

### 4.volatile

#### a)volatile的作用

#### b)CPU缓存去分析volatile的原理

#### c)volatile的汇编指令

#### d)指令重排

### 5.内存屏障

Jdk1.6以后 优化

无锁状态 -> 偏向锁 -> 轻量级锁 -> 重量级锁

UseBiasedLocking

轻量级锁

对象头 MarkWord赋值到锁记录中

CAS

偏向锁

在很多情况下，锁不仅不会存在竞争，而且很多时候都是由同一个线程获得锁

对象头、实例数据、对齐填充