

课程学习须知

并发编程课程大纲

- 并发编程的发展以及价值 (5月15号)
- 并发编程带来的挑战之同步锁 (5月16号)
- 并发编程带来的挑战之可见性 (5月19号)
- 并发安全性之Lock锁及原理分析 (5月22号)
- **线程阻塞唤醒wait,notify以及condition,死锁等原理分析 (5月23号)**
- **J.U.C并发工具集场景及原理分析 (5月26号)**
- **随便聊聊ThreadLocal&ForkJoin (5月29号)**
- **深度剖析阻塞队列的设计原理及实现 (5月30号)**
- 并发安全的集合ConcurrentHashMap (6月2号)
- 站在架构的角度思考线程池的设计和原理 (6月5号)
- Java8新的异步编程方式 (6月6号)

注意：随着课程的推进，有可能会增加课时来讲解，所以如果遇到课程大纲的变化，我会及时通知！

新同学注意事项

并发编程是一个完整的体系，到目前为止已经讲了5节课了，如果前面部分内容没有听的同学，一定要记得补一下，不然越到后面越听不懂。

另外，有些同学基础比较薄弱点，对并发这块完全不熟悉，我强烈建议这些同学至少听2遍以上。因为并发编程整个体系是偏基础和底层的，和我们平时开发用的应用框架的理解完全不一样。

这就需要大家逐步去转变一些学习思维，从而更好的理解。

有部分学员反馈有问题找不到我，记得加我微信 **mic4096**，qq目前登录较少，有些消息回应不及时。

本阶段课程变化

- 新增了部分内容，如异步编程、ForkJoin等。
- 内容更加细致。
- 讲课节奏会稍微放缓，帮助大家更好的吸收。

本阶段内容的学习目标

并发编程是很多大公司面试重点考察的范围，据部分拿到高薪Offer的学员反馈，大部分面试中，并发编程很好的帮助他们拿到Offer。

所以今年，对于并发编程的内容做了调整，更加细致和全面。

大家学完之后要达到的目标。

- 很好的理解线程的本质。
- 能够灵活运用线程。
- 从原理层面理解并发编程。
- **在并发编程领域击败99%的程序员**

本周预习资料

本周的三次可，应该会讲以下内容，condition部分因为没讲完成，所以会放在本周三讲剩下部分。

- 线程阻塞唤醒wait,notify以及condition,死锁等原理分析 (5月23号)
- J.U.C并发工具集场景及原理分析 (5月26号)
- ThreadLocal&ForkJoin (5月29号)
- 深度剖析阻塞队列的设计原理及实现 (5月30号)

阻塞队列及condition (5月26号)

- condition源码分析
- 基于condition实现一个阻塞队列
- 理解什么是阻塞队列
- 分析阻塞队列的原理
- Java中常见阻塞队列的应用

J.U.C并发工具集场景及原理分析 (5月29号)

- CountDownLatch

countdownlatch是一个同步工具类，它允许一个或多个线程一直等待，直到其他线程的操作执行完毕再执行。从命名可以解读到countdown是倒数的意思，类似于我们倒计时的概念。

countdownlatch提供了两个方法，

- countDown
- await

CountDownLatch初始化的时候需要传入一个整数，在这个整数倒数到0之前，调用了await方法的程序都必须等待，然后通过countDown来倒数。

- Semaphore

semaphore也就是我们常说的信号灯，semaphore可以控制同时访问的线程个数，通过acquire获取一个许可，如果没有就等待，通过release释放一个许可。有点类似限流的作用。叫信号灯的原因也和他的用处有关，比如某商场就5个停车位，每个停车位只能停一辆车，如果这个时候来了10辆车，必须要等前面有空的车位才能进入。


- CyclicBarrier

CyclicBarrier的字面意思是可循环（Cyclic）使用的屏障(Barrier)，“可重复使用的栅栏”，它要做的事情是，让一组线程到达一个屏障(也可以叫同步点)时被阻塞，直到最后一个线程到达屏障时，屏障才会开门，所有被屏障拦截的线程才会继续干活，线程进入屏障通过CyclicBarrier的await()方法。

ThreadLocal&ForkJoin (5月29号)

- ThreadLocal的作用及原理
- ForkJoin的作用
- ForkJoin原理分析
- ForkJoin应用实战

预习须知



同学们，预习很重要，但是预习不是给大家一堆资料去看，而是自己先去根据我要讲的内容查询一下相关资料，然后在查询过程中会产生很多疑问，而我在课堂上，会帮助大家解决这些疑问，这样的学习效果是更好的。

如果一味的去被动接受这些知识，没有思考。那么对大家来说，几遍是吸收了一些东西，但是最根本的学习能力还是不具备的。

技术的迭代速度非常快，我们需要有一些自我学习能力！！