一道题决定去留：为什么synchronized无法禁止指令重排，却能保证有序性？

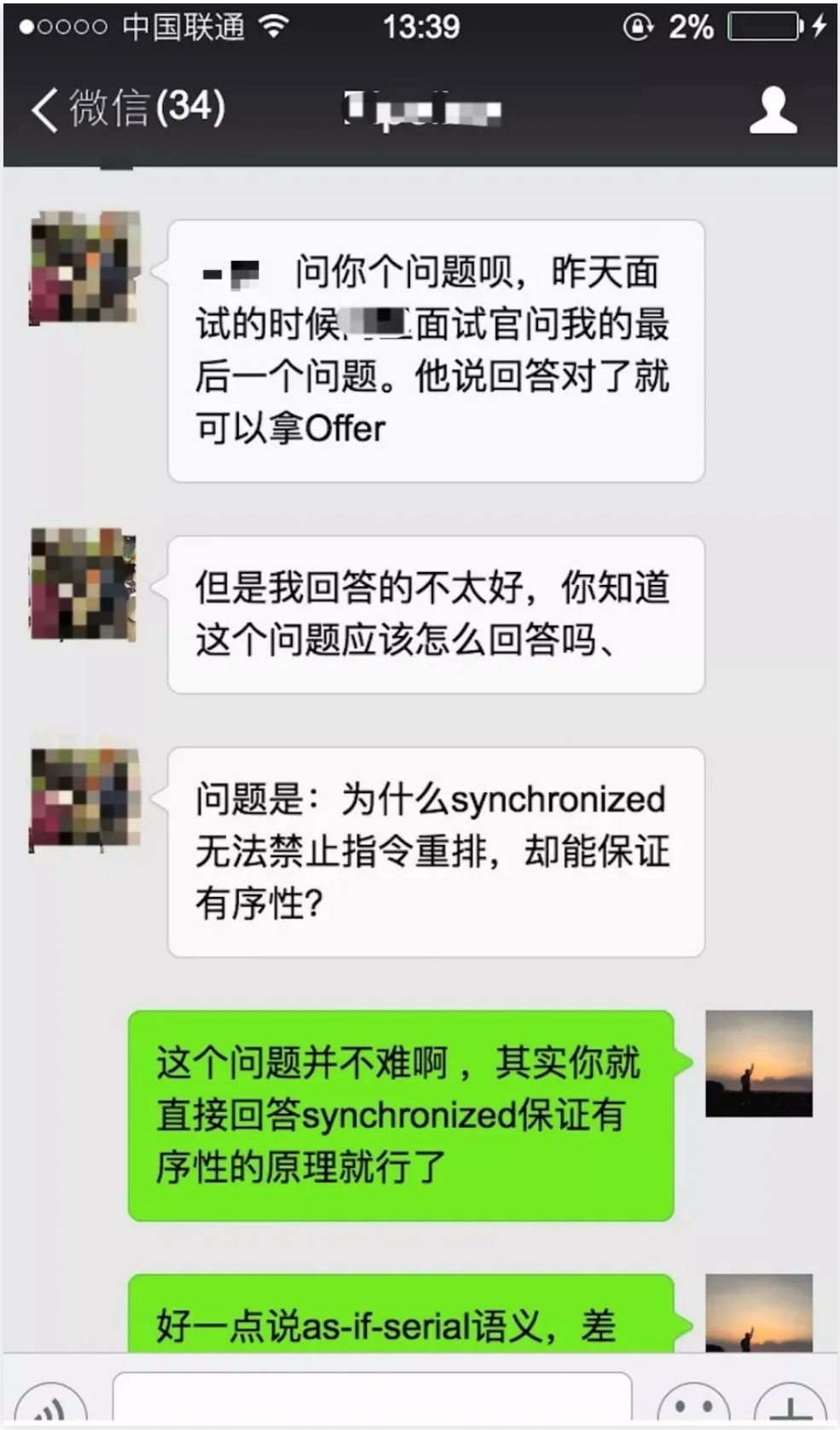
Java中文社群 5月9日

以下文章来源于Java之道 ， 作者Hollis

Java之道

有道无术，术可成；有术无道，止于道；「Java之道」致力于为广大Javaer传道授业解...

前几天有一位读者找我问一个问题，说是这道题可能影响了他接下来3年的技术成长。



据说这位读者前面的很多问题会的都还可以，属于那种可过可不过的类型的，面试官出了最后一道题，就是回答的满意就可以给Offer，回答的不好就不让过的意思。

那么这道题到底应该如何回答呢?

首先我们要分析下这道题，不得不说这个面试官还是有一定的水平的，这简单的一个问题，其实里面还是包含了很多信息的，要想回答好这个问题，面试者至少要知道一下概念：

Java内存模型、并发编程有序性问题、指令重排、synchronized锁、可重入锁、排它锁、as-if-serial语义、单线程&多线程

所以，这道题的正确回答姿势是怎样的呢？

标准答案如下：

这是个好问题，这个问题我曾经也思考过，也查阅过很多资料，甚至还去看过hotsopt的源码。

> 不管三七二十一，上来先舔一波，然后表示下自己求知好学的态度。

为了进一步提升计算机各方面能力，在硬件层面做了很多优化，如处理器优化和指令重排等，但是这些技术的引入就会导致有序性问题。

> 先告诉面试官你知道什么是有序性问题，也知道是什么原因导致的有序性问题

我们也知道，最好的解决有序性问题的办法，就是禁止处理器优化和指令重排，就像volatile中使用内存屏障一样。

> 表明你知道啥是指令重排，也知道他的实现原理

但是，虽然很多硬件都会为了优化做一些重排，但是在Java中，不管怎么排序，都不能影响单线程程序的执行结果。这就是as-if-serial语义，所有硬件优化的前提都是必须遵守as-if-serial语义。

> 重点！解释下什么是as-if-serial语义，因为这是这道题的第一个关键词，答上来就对了一半了

再说下synchronized，他是Java提供的锁，可以通过他对Java中的对象加锁，并且他是一种排他的、可重入的锁。

> 装X项，不留痕迹的展示自己对锁了解的比较多

所以，当某个线程执行到一段被synchronized修饰的代码之前，会先进行加锁，执行完之后再进行解锁。在加锁之后，解锁之前，其他线程是无法再次获得锁的，只有这条加锁线程可以重复获得该锁。

> 介绍synchronized的原理，这是本题的第二个关键点，到这里基本就可以拿满分了。

synchronized通过排他锁的方式就保证了同一时间内，被synchronized修饰的代码是单线程执行的。所以呢，这就满足了as-if-serial语义的一个关键前提，那就是**单线程**，因为有as-if-serial语义保证，单线程的有序性就天然存在了。

> 最后总结一下问题，给面试官致命一击！Offer到手！~

if快还是switch快？解密switch背后的秘密

HashMap 的 7 种遍历方式与性能分析！「修正篇」

关注公众号发送”进群“，老王拉你进读者群。