< / > 在这里 / 有面试笔试常见技巧的提炼与总结；
< / > 在这里 / 有面试笔试高频前端知识点的整理与剖析；
< / > 在这里 / 有面试笔试历年前端真题的解答与拓展。

PROGRAMMER

Interview and written examination

前端程序员 面试笔试宝典

猿媛之家 / 组编　平文 等 / 编著

98%
高频知识点

本书覆盖了近3年程序员面试笔试中超过98%的前端高频知识点

当你细细品读完本书后，各类企业的offer将任由你挑选

一书在手 / 工作不愁 > .

机械工业出版社
CHINA MACHINE PRESS

前端程序员面试笔试宝典

猿媛之家　组编
平　文　等编著

机械工业出版社

本书是一本前端程序员面试笔试的应试类用书，在内容上，除了讲解如何解答前端程序员面试笔试问题，还引入了相关知识点辅以说明，让读者能够更加容易理解。

本书包括前端程序员面试笔试过程中各类知识点，在题目的广度上，搜集了近 3 年来多家 IT 企业针对前端岗位的笔试面试真题。在讲解的深度上，本书由浅入深，庖丁解牛式地分析每一个知识点，并提炼归纳，同时，引入相关知识点，并对其进行深度剖析，让读者不仅能够理解这个知识点，还能在遇到相似问题的时候，也能游刃有余地解决。本书根据知识点进行分类，结构合理，条理清晰，便于读者学习与查阅。

图书在版编目（CIP）数据

前端程序员面试笔试宝典 / 猿媛之家组编；平文等编著.—北京：机械工业出版社，2018.9
ISBN 978-7-111-60747-2

Ⅰ. ①前… Ⅱ. ①猿… ②平… Ⅲ. ①程序设计－资格考试－题解
Ⅳ. ①TP311.1-44

中国版本图书馆 CIP 数据核字（2018）第 194494 号

机械工业出版社（北京市百万庄大街 22 号　邮政编码 100037）
策划编辑：时　静　　责任编辑：时　静
责任校对：张艳霞　　责任印制：张　博

三河市国英印务有限公司印刷

2018 年 9 月第 1 版 • 第 1 次印刷
184mm×260mm • 19.75 印张 • 484 千字
0001－3000 册
标准书号：ISBN 978-7-111-60747-2
定价：69.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务
服务咨询热线：（010）88361066
读者购书热线：（010）68326294
　　（010）88379203

网络服务
机 工 官 网：www.cmpbook.com
机 工 官 博：weibo.com/cmp1952
教育服务网：www.cmpedu.com
金　书　网：www.golden-book.com

前　言

计算机技术博大精深，日新月异。Hadoop、GPU 计算、移动互联网、模式匹配、图像识别、神经网络、蚁群算法、大数据、机器学习、人工智能、深度学习等新技术让人眼花缭乱，稍有不慎，就会被时代抛弃。于是，很多 IT 从业者就开始困惑了，不知道从何学起，到底什么才是计算机技术的基石。其实，究其本质，还是最基础的计算机语言知识、数据结构与算法知识。所以，无论是世界级的大型企业还是几个人的小公司，在面试求职者的时候，往往都会考察这些最基础的知识，无论你的研究方向是什么，这些基础知识都是应该熟练掌握的。

本书在写作风格上，推陈出新，对于前端知识点的讲解，不仅有文字描述，更以示例佐证（示例代码可以从 https://github.com/pwstrick/FrondEndInterviewCode 下载）。为了能够写出精品书籍，我对每一个技术问题，都反复推敲，与技术大牛一起反复论证其可行性；对每一句话，都咬文嚼字，字斟句酌，所有这些付出，只为让读者能够对书中技术点放心，文字描述舒心。

本书在选择题目时也下了巨大的工夫：首先，搜集近 3 年来多家 IT 企业的面试笔试真题，包括已经出版的其他著作、技术博客、在线编码平台、刷题网站等，保证所选样本足够大。其次，在选择题目的时候，尽可能不选择那种一眼就能知道结果的简单题，也不选择怪题、偏题、难题。选题原则是选择难度适中或者看上去简单但实际容易被坑的题目，力求遴选出来的真题能够最大限度地帮助读者。在真题的解析上，采用层层递进的写法，先易后难，层层深入，将问题抽丝剥茧，使读者能够跟随书中的思路，一步步找到问题的最优解。

写作的过程是一个自我提高、自我认识的过程，很多知识，只有你深入理解与剖析后，才能领悟其中的精髓，掌握其中的技巧，程序员求职面试也不例外。本书不仅具备了其他书籍分析透彻，代码清晰合理等优点，还具备以下两个方面的优势：

第一，知识覆盖面全。本书涉及知识点覆盖了前端程序员面试笔试中至少 98%的基础知识点。

第二，讲解详细，剖析深入。本书对每个知识点都进行了详细阐述，知识点介绍完以后还会针对部分真题进行深度分析与讲解，图文并茂，生动形象。

我在相关技术岗位上已经有多年的经验。一直想把自己的心得分享给大家，机缘巧合促成了这本书的撰写。在此，要感谢与出版本书有关的人，因为有你们，我才能坚持下来，完成整本书的编写。

首先，感谢机械工业出版社时静老师给我写作的机会。其次，要感谢的是楚秦，他不但让我加入到这项工作中，还审阅了整本书，并对文字和代码进行了修改和润色，为改进本书提供了许多宝贵的建设性意见，这些建议与意见极大地提高了本书的质量。

除此之外，也感谢那些给予我热情帮助的人，从他们那里亦得到了很多非常好的建议，这些人是（按姓氏拼音排列）：陈安阳、陈曼杰、陈涛、潘义璠、沈哲俊、王春明、王汝婷、夏丽、赵茹林、周捷、周山。还要特别感谢周晶撰写序。

最后，我要感谢我的家人，他们是我生命中最重要的人，感谢他们对我的理解和鼓励。尤其要感谢我的爱妻，一直陪伴在我身边，在我感到困难的时候支持我、鼓励我，为我营造了一个安心、舒适的写作环境，让我有信心完成整本书的写作。

其实，本书不仅可以作为程序员求职的应试类书籍，还可以作为前端程序员的教辅书籍进行学习。

书中的很多思想、方法对于提高对前端知识的理解大有裨益，无论你是本科生还是研究生，无论你是低年级学生还是高年级学生，无论你对计算机底层知识或是当前的计算机前沿知识是否了解，都不影响你学好本书。

本书中有部分思想来源于网络上的无名英雄，无法追踪到原始出处，在此对这些幕后英雄致以最崇高的敬意。没有学不好的学生，只有教不好的老师，希望无论是什么层次的学生，都能毫无障碍地看懂书中所讲内容。如果读者存在求职困惑或是对书中的内容存在异议，都可以通过 yuancoder@foxmail.com 联系作者。

平　文

于上海松江

目　录

面试笔试经验技巧篇

面试笔试技术攻克篇

面试笔试经验技巧篇

想找到一份程序员的工作，一点技术都没有显然是不行的，但是，只有技术也是不够的。面试笔试经验技巧篇主要针对程序员面试笔试中遇到的18个常见问题进行深度解析，并且结合实际情景，给出了较为合理的参考答案以供读者学习与应用。掌握这18个问题的解答精髓，对求职者大有裨益。

经验技巧 1 如何巧妙地回答面试官的问题?

所谓“来者不善，善者不来”，在程序员面试中，求职者不可避免地需要回答面试官各种刁钻、犀利的问题。回答面试官的问题千万不能简单地回答“是”或者“不是”，而应该具体分析“是”或者“不是”的原因。

回答面试官的问题是一门很深的学问。那么，面对面试官提出的各类问题，如何才能条理清晰地回答呢？如何才能让自己的回答不至于撞上枪口呢？如何才能让自己的回答结果令面试官满意呢？

谈话是一门艺术，回答问题也是一门艺术，同样的话，不同的回答方式，往往会产生不同的效果，甚至是截然相反的效果。在此，编者提出以下几点建议，供读者参考。

首先回答问题务必谦虚谨慎。既不能让面试官觉得自己很自卑，唯唯诺诺，也不能让面试官觉得自己自负，应该通过问题的回答表现出自己自信从容、不卑不亢的一面。例如，当面试官提出“你在项目中起到了什么作用”的问题时，如果求职者回答：我完成了团队中最难的工作，此时就会给面试官一种居功自傲的感觉，而如果回答：我完成了文件系统的构建工作，这个工作被认为是整个项目中最具有挑战性的一部分内容，因为它几乎无法重用以前的框架，需要重新设计。这种回答不仅不傲慢，反而有理有据，更能打动面试官。

其次，回答面试官的问题时，不要什么都说，要适当地留有悬念。人一般都有猎奇的心理，面试官自然也不例外，而且，人们往往对好奇的事情更有兴趣、更加偏爱，也更加记忆深刻。所以，在回答面试官问题时，切记说关键点而非细节，说重点而非和盘托出，通过关键点，吸引面试官的注意力，等待他们继续“刨根问底”。例如，当面试官对你的简历中一个算法问题有兴趣，希望了解时，可以如下回答：我设计的这种查找算法，对于 80%以上的情况，都可以将时间复杂度从 O(n)降低到 O(log n)，如果您有兴趣，我可以详细给您分析具体的细节。

最后，回答问题要条理清晰、简单明了，最好使用“三段式”方式。所谓“三段式”，有点类似于中学作文中的写作风格，包括“场景/任务”“行动”“结果”三部分内容。以面试官提的问题“你在团队建设中，遇到的最大挑战是什么”为例，第一步，分析场景/任务：在我参与的一个 ERP 项目中，我们团队一共四个人，其他三个人中，两个人能力很强，人也比较好相处，但有一个人却不太好相处，每次我们小组讨论问题的时候，他都不太爱说话，也很少发言，分配给他的任务也很难完成。第二步，分析行动：为了提高团队的综合实力，我决定找个时间和他好好单独谈一谈。于是我利用周末时间，约他一起吃饭。吃饭的时候，顺便讨论了一下我们的项目。我询问了一些项目中他遇到的问题，通过他的回答，我发现他并不懒，也不糊涂，只是对项目不太了解，缺乏经验，缺乏自信，所以越来越孤立，越来越不愿意讨论问题。为了解决这个问题，我尝试着把问题细化到他可以完成的程度，从而建立起他的自信心。第三步，分析结果：他是小组中水平最弱的人，但是，慢慢地，他的技术变得越来越厉害了，也能够按时完成安排给他的工作了，人也越来越自信了，也越来越喜欢参与我们的讨论，并发表自己的看法，我们也都愿意与他一起合作了。“三段式”回答的一个最明显的好处就是条理清晰，既有描述，也有结果，有根有据，让面试官一目了然。

回答问题的技巧是一门大学问。求职者完全可以在平时的生活中加以练习，提高自己与人沟通的技能，等到面试时，自然就得心应手了。

经验技巧 2 如何回答技术性的问题?

程序员面试中，面试官会经常询问一些技术性的问题，有的问题可能比较简单，都是历年的笔试面试真题，求职者在平时的复习中会经常遇到，应对自然不在话下。但有的题目可能比较难，来源于 Google、Microsoft 等大企业的题库或是企业自己为了招聘需要设计的题库，求职者可能从来没见过或者从来都不能完整地、独立地想到解决方案，而这些题目往往又是企业比较关注的。

如何能够回答好这些技术性的问题呢？编者建议：会做的一定要拿满分，不会做的一定要拿部分分。即对于简单的题目，求职者要努力做到完全正确，毕竟这些题目，只要复习得当，完全回答正确一点问题都没有（编者认识的一个朋友据说把《编程之美》《编程珠玑》《程序员面试笔试宝典》上面的技术性题目与答案全都背得滚瓜烂熟了，后来找工作简直成了“offer 杀器”，完全就是一个 Bug，无解了）；对于难度比较大的题目，不要惊慌，也不要害怕，即使无法完全做出来，也要努力思考问题，哪怕是半成品也要写出来，至少要把自己的思路表达给面试官，让面试官知道你的想法，而不是完全回答不会或者放弃，因为面试官很多时候除了关注你的独立思考问题的能力以外，还会关注你技术能力的可塑性，观察求职者是否能够在别人的引导下去正确地解决问题，所以，对于你不会的问题，他们很有可能会循序渐进地启发你去思考，通过这个过程，让他们更加了解你。

一般而言，在回答技术性问题时，求职者大可不必胆战心惊，除非是没学过的新知识，否则，一般都可以采用以下六个步骤来分析解决。

（1）勇于提问

面试官提出的问题，有时候可能过于抽象，让求职者不知所措，或者无从下手。所以，对于面试中的疑惑，求职者要勇敢地提出来，多向面试官提问，把不明确或二义性的情况都问清楚。不用担心你的问题会让面试官烦恼，影响你的面试成绩，相反，这样做还会对面试结果产生积极影响：一方面，提问可以让面试官知道你在思考，也可以给面试官一个心思缜密的好印象；另一方面，方便自己对问题的解答。

例如，面试官提出一个问题：设计一个高效的排序算法。求职者可能丈二和尚摸不到头脑，排序对象是链表还是数组？数据类型是整型、浮点型、字符型还是结构体类型？数据基本有序还是杂乱无序？数据量有多大，1000 以内还是百万以上个数？此时，求职者大可以将自己的疑问提出来，问题清楚了，解决方案自然也就出来了。

（2）高效设计

对于技术性问题，如何才能打动面试官？完成基本功能是必需的，仅此而已吗？显然不是，完成基本功能顶多算及格水平，要想达到优秀水平，还应该考虑更多的内容，以排序算法为例：时间是否高效？空间是否高效？数据量不大时也许没有问题，如果是海量数据呢？是否考虑了相关环节，例如数据的“增删改查”？是否考虑了代码的可扩展性、安全性、完整性以及鲁棒性？如果是网站设计，是否考虑了大规模数据访问的情况？是否需要考虑分布式系统架构？是否考虑了开源框架的使用？

（3）伪代码先行

有时候实际代码会比较复杂，上手就写很有可能会漏洞百出、条理混乱，所以，求职者可以首先征得面试官的同意，在编写实际代码前，写一段伪代码或者画好流程图，这样做往往会让思路更加清晰明了。

切记在写伪代码前要告诉面试官，否则他们很有可能对你产生误解，认为你只会纸上谈兵，实际编码能力却不行。只有征得了他们的允许，方可先写伪代码。

（4）控制节奏

如果是算法设计题，面试官都会给求职者一个时间限制用以完成设计，一般为 20min 左右。完成得太慢，会给面试官留下能力不行的印象，但完成得太快，如果不能保证百分百正确，也会给面试官留下毛手毛脚的印象，速度快当然是好事情，但只有速度，没有质量，速度快根本不会给面试加分。所以，编者建议，回答问题的节奏最好不要太慢，也不要太快，如果实在是完成得比较快，也不要急于提交给面试官，最好能够利用剩余的时间，认真检查一些边界情况、异常情况及极性情况等，看是否也能满足要求。

（5）规范编码

回答技术性问题时，多数都是纸上写代码，离开了编译器的帮助，求职者要想让面试官对自己的代码一看即懂，除了字迹要工整，不能龙飞凤舞以外，最好严格遵循编码规范，注意函数变量命名、换行缩进、语句嵌套和代码布局等，同时，代码设计应该具有完整性，保证代码能够完成基本功能、输入边界值能够得到正确输出、对各种不合规范的非法输入能够做出合理的错误处理，否则，写出的代码即使