程序员如何理解自己的职业与发展
程序员如何看待自己的工作与生活

一本写给程序员的思考书

技匠社 TECHMASK

程序员的自我修养

陈逸鹤 著

清華大學出版社

程序员的自我修养

陈逸鹤　著

清華大學出版社
北京

内容简介

程序员作为一个职业、也作为一个群体，正逐渐从幕后走向前台，并以他们自己的能力加速改变着世界，也改变着人们生活的方方面面。然而，对于程序员，特别是年轻程序员们来说，如何理解自己的职业与发展，如何看待自己的工作与生活，这些问题往往比那些摆在面前的技术难题更让他们难以解答。

这本书从一个成熟程序员、一名 IT 管理者的角度，以杂记的形式为大家分享关于国内程序员职业生涯、个人发展、编程中的实践与认知乃至自学设计等方面的经验方法与思考感悟。其中每一篇文章都涉及一个与程序员息息相关的话题，无论你是即将走上程序员岗位的在校大学生，是刚刚成为程序员的职场新人，还是有一定经验的程序员，这本书都会给你带来启发。

图书在版编目(CIP)数据

程序员的自我修养 / 陈逸鹤著. — 北京 : 清华大学出版社, 2017
ISBN 978-7-302-46808-0

Ⅰ ①程… Ⅱ. ①陈… Ⅲ. ①程序设计 Ⅳ. ①TP311.1

中国版本图书馆CIP数据核字(2017)第052721号

责任编辑：张　敏
封面设计：杨玉兰
责任校对：徐俊伟
责任印制：沈　露
出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，　http://www.wqbook.com
地　　址：北京清华大学学研大厦A座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015, zhiliang@tup.tsinghua.edu.cn
印 装 者：北京嘉实印刷有限公司
经　　销：全国新华书店
开　　本：170mm×240mm　　**印　　张**：13　　**字　　数**：200千字
版　　次：2017年5月第1版　　**印　　次**：2017年5月第1次印刷
印　　数：1～3500
定　　价：49.00元

产品编号：072511-01

前 言
PROFACE

我从来没有想过自己会写一本关于程序员的书，正如你后面将会读到的，这本书中的大部分文章都来自于我平时的杂记（我以“技匠”为笔名在自己的博客和一些专栏中写文章）。虽然也包含一些实用的技术类文章，但大部分是我作为一名程序员，或跳出自己的职业去看待程序员这个群体时，所记录下的心得、感悟。

在我们所处的这个互联网时代，软件技术正快速地渗透到每一个行业和几乎所有的专业领域，并加速推动着社会的变革与发展。而这一切的背后正是千千万万像你我这样的普通程序员不断努力的成果。然而，我们是否已经做好准备去应对那些挑战，或者说我们是否清楚应该成为一个怎样的自己呢？这些问题曾不断困扰着我，但也正是通过对这些问题的深入思考及不断解答，才使自身获得了进步和提升。回想这十几年来的成长经历，给我最大帮助的莫过于自己对职业生涯、对个人发展、对编程本身，以及对美的本质的追求。

还是来谈谈这本书吧。由于都是闲时所记，本不成什么体系，但由于出版的需要，最终将全书归纳为五个章节，但我却从心底里建议读者朋友们按照自己的喜好与兴趣挑选阅读，因为它们之间并无关联，也不存在学习某一技术时所必须遵循的顺序，况且相似题材的文章堆在一起阅读反而不那么易于消化。

第一章　谈职业生涯：从程序员职业生涯的角度，阐述了我的一些观点和意见。其中，大部分内容是我的亲身经历，还有一些则是从他人身上获得的经验教训。经过多年积累，深深觉得，如果能让更多年轻程序员或是那些即将走上这条道路的在校大学生早些了解这些经验教训，或许能让他们在将来少走些

弯路吧。

第二章　谈编程中的实践与认知：讲的则是自己多年的编程实践，以及从中获得的感悟。既有《全栈工程师如何快速构建一个 Web 应用》《一名全栈设计师的 Mac 工具箱（设计、开发、效率）》这样的实践类文章，又包含了《突破程序员思维》《我似乎理解了编程的意义》这样看上去很“湿”，却是我真正希望能够启发年轻程序员们去思考编程本身，以及体会编程意义的文章。

第三章　谈程序员的个人发展：以程序员如何变得优秀为话题，从当下有关程序员的各个热点（包括创业、自由职业、建立个人品牌等）入手，为你分析那些优秀程序员身上的共同特质，未来的发展方向，以及值得每一名程序员去做的有益尝试。

第四章　谈编程中的教与学：鼓励大家在通过大量技术干货获取技能的同时，也关注那些优秀湿货对于自己成长的重要性。同时，我也对编程教学提出了一些自己的意见和观点。

第五章　谈自学设计：在我看来，自学设计是对“美”的意识的重新唤醒，而这一章正是我在自学设计过程中写下的心得及学习笔记。在本章中，我与读者朋友们分享的不是那些设计中的应用技巧，而是如何尝试去理解设计的本质，希望读者朋友们也能以这种态度去学习设计，你会逐渐获得感知并创造美的能力，而这反过来也会促进你编程能力的提升。

除了以上章节内容之外，我还为此书构建了一个主题网站，取名“技匠社”（jijiangshe.com），读者们（程序员或设计师朋友们）可以在此分享和推荐他们认为有价值的工具、资源和教程。目前，网站已收录了会员们分享的将近 500 个各类资源，涵盖了从前端、移动、后端、数据库、大数据开发到设计中的色彩、字体、图标、模板等各个方面。希望这个网站也能像它的名字所表达的那样，能够成为技术匠们汇聚的地方，并帮助更多的程序员或设计师获得成长。

写文章对于我来说是一件轻松愉快的事情，我从未感受到写作本身带来的压力，反而是在完成之后，发现会有一些不足之处，而无法给读者带来真正帮

助的时候，感到颇为沮丧。但我仍将这些文章收入到这本书里，因为这就是我当前所能达到的思考深度，也是一个匠人经过十几年成长后所形成的完整思想框架。这本书中的一些文章可能让你产生共鸣，一些可能对你有所帮助，而另一些你可能并不喜欢甚至感到厌恶，但这恰恰说明你在阅读的过程中进行了思考，我的想法蹦到了你的脑子里，衍生出你的新想法，这个连锁反应在你脑中反复激荡，最终让你看得更加透彻，思考得更深入，而这才应该是本书的真正目的和意义所在。

这本书能够完成，需要感谢很多人，他们在我写作的过程中给予了我极大的帮助和鼓励。首先，要感谢指导我完成这本书的清华大学出版社编辑，她在目录的修订，封面与排版的设计等方面给了我很多意见。其次，要感谢我的妻子，她时常鼓励我进行写作，也总是我文章的第一个读者。还要感谢我的儿子，由于利用业余时间写作，陪伴他的时间少了很多，但他很懂事，从不在我写作时打扰。此外，我还要特别感谢我的母亲，是她给予了我写作的力量，激励我去做一些更有意义的事情。最后，我想感谢所有我文章的读者，是你们的喜爱促使我不断思考并写出更好的文章，你们的反馈更是我不断修正思想获得提升的来源。

编　者

目 录

CONTENTS

第一章　谈职业生涯

一、写给年轻程序员的10点启示

最近，陆续在 51CTO 和 CSDN 上看到好几篇写程序员如何成长，如何拿到高薪的文章。文章都很不错，一些观点也很实用，但整篇文章读下来，我总觉得意犹未尽，感觉还应该再给年轻的程序员们说些什么似的。

其实我离开正式的程序员岗位（当时我是一名架构师）已经快 3 年了，现在我仍然在业余时间做一些感兴趣的开源或个人项目。但我发现恰恰是最近几年，让我能够从不同的角度和视野，对程序员这个职业有了更深入的理解和认识。当我成为一名 IT 管理者时，我从更多的程序员身上看到了他们身上的一些特质对其职业生涯所产生的影响；当我开始自学设计，我又从设计大师们那里得到很多极具启发性的观点，而它们对程序员们同样具有很大的价值；而当我开始写作，每一篇文章完成后与年轻的程序员读者们进行交流的过程中，我的想法和观点又能得到进一步的完善和提升。

因此，我迫不及待地将这些记录下来，希望从一些不同的角度给予年轻程序员朋友们一些启发。

1. 正确地认识自己

我听到过很多用来形容程序员的网络词汇，例如，码农、程序猿、软件工程师、张江男、屌丝程序员，等等。其中大部分都略带贬义，有些甚至是程序员们自己发明出来用于自嘲的。其实这些称呼对你来说并不重要，关键是你自己内心中用了哪一个词汇来形容自己。当你仅仅将自己定位成一个码农，那你可能就是那个整天用着相同的工具，写着相似代码的码农；当你把自己看作是一个屌丝程序员的话，那你也可能就是那个衣食不愁，但整天浑浑噩噩、无所追求的屌丝。

我更愿意用技匠这个词来形容程序员，就像我给自己起的花名一样。程序员应该是那些不断追求更高技术，并有着自己产品梦的工匠。当你通过对自己技术不断打磨，一次又一次做出那些优秀产品的时候，你会发现自己不再是他人口中的码农或是屌丝，而更多被称为了大师、大神，而受到大家的尊重。

我相信没有一个真正的程序员内心里会将自己当作码农或屌丝，那么不妨也像我一样找到一个能真正反映你内心的词汇来定位你自己，通过努力，你会慢慢变成你希望的样子。

2. 比一般人更加努力

我曾看到一个关于天才与普通人的有趣漫画（很遗憾我没能找到原图，只能用文字进行描述）。

漫画描述 1：一个普通人每天都很努力，他头顶上显示的能力槽也在慢慢地增长。而图中的另一个天才，则整天不务正业，当然他头上的能力槽也几乎没有增长。

漫画描述 2: 经过了长时间的努力，普通人头顶上的能力槽慢慢接近了满格，此时那个天才醒悟并开始了努力。

漫画描述 3：又过了一段时间，普通人头顶上的能力槽终于满格了，但此时天才头顶上的能力槽却早已爆表，高出了那个普通人很大一截，那个努力的普通人只好无奈地抬头仰望着那位天才。

漫画描述 4：但当普通人默默回过头来，却惊讶地发现有更多的人正同样充满敬意地仰望着他，因为此时的他也早已成为了大多数人眼中的那个佼佼者。

这组漫画非常发人深省，优秀的程序员往往会被同天才或高智商的人联系在一起，但我想告诉你，你周围所看到的那些天才（身边的优秀程序员）只是在你没看到的时候花了更多时间工作或者学习而已，当你也坚持这么做时，你也会变得和他们一样优秀。

3. 适时建立个人权威

我以前有一个美国同事，是个东欧人，在公司里负责一个非常老旧系统的开发和维护工作，用的技术也是几乎快被淘汰的 Power Builder，因此他在公司里是个不怎么被重视的人。然而有一天，他突然拿着笔记本电脑敲开了老板办公室的门，并且给老板展示了一个非常漂亮而且易用的 Web 系统。原来他利用业余时间自学 .NET 技术，将他所负责的那个系统整个重写了一遍。这个焕然一新的系统一下子让老板和整个公司惊呆了，大家由衷地对他报以敬佩之情。不久，他便被提拔为了开发团队的负责人，而他的那套系统也很快作为公司的拳头产品推出了。

我经常听到一些程序员抱怨自己的职业生涯毫无起色，或者在工作了几年之后就担忧自己遇到了瓶颈。其实，造成这些的原因往往是他们已经习惯了听命于人，而缺少自己的观点和主张，久而久之便成了那个在他人眼中可有可无的平庸之人。你不妨也学一学我的那位美国同事，选择合适的时机去表现自己，建立个人权威，这能让其他人看到你的不同之处，并为你在公司或团队内部构建起良好的影响力。当然，这一切的前提是你通过不断努力积累了自己的实力，并在恰当的时候去展现它。

4. 遵循最佳实践

技术总是在不断发展，我们每年都能看到很多新的开发语言、工具和框架的出现，而每隔几年又会产生一些大的技术变革。那么作为程序员，如何才能适应这种变化呢？其实，就像每年都会有新的流行设计趋势，然而设计的本质

和原则却始终不变一样。作为一名程序员，你也需要尝试去理解那些软件领域最本质的东西，而我的建议就是学习那些最佳实践。

当你理解了 GOF 的那些经典设计模式，你就会知道如何使用一个单例来最有效地实现一个 Logger 组件；当你理解了 SOA 或是最新的微服务架构，你就能够通过架构使你的企业 IT 治理更加有效；当你理解了 DEVOPS 这种新的运维文化和理念，你就能使企业 IT 运维效率得到显著提升。此外，你还需要理解在网络安全、性能调优、代码优化等各个方面的最佳实践，以使你写出更高质量的代码、做出更优秀的产品。

这些最佳实践或由此衍生出来的框架、工具都是那些富有经验的程序员通过大量实践，总结出来的最优秀的软件开发思想。通过理解和对它们的有效实践，能够让你站到前人的肩膀之上，对软件开发本身获得更深入的理解和认识。

5. 保持好奇心并乐于探索新的事物

好奇心可能是那些优秀程序员与普通程序员之间最显著的区别之一。优秀程序员们往往不会满足于对手头工作的认识和理解，他们有很强的意愿去了解那些更深入的东西。比如，他们会通过研究公司的框架（甚至是更底层的框架）源代码，去了解它们具体的实现原理和设计思想。这对提升程序员的技术深度是非常有帮助的。

另外，优秀的程序员们也往往非常乐于探索那些看似与工作无关的技术。比如，做后端的程序员去学习前端的技能，前端工程师则去学习 UI 设计，等等，这些虽不会让他们成为那一个领域的专家，但技术往往是相通的，当你在探索这些新鲜事物的同时，你会发现你原有的技能也得到了提升。（最近，我在自学一些简单的机器语言，从中就受到了一些很有价值的启发，我也希望能通过几篇文章来分享给大家）

6. 抛开代码与人沟通

“紧盯着电脑屏幕，不断敲击键盘，目光有些呆滞。”这可能是程序员给人的印象。而我所看到的那些优秀的程序员却都不是这样的，他们往往兴趣广

泛，并且都非常乐于与人沟通交流。

程序员们很容易会忽视与人的沟通，这其实对他们的职业生涯发展是不利的。我很支持那些年轻的程序员们坚持去走技术路线，但这不应该成为你排斥与人沟通的理由。你需要与人沟通来获得他人的帮助；你需要与人沟通来建立良好的工作关系；当你的能力不断提升，被赋予更多职责时，你更需要与人沟通来管理好自己的团队，以及与老板或客户进行有效的沟通等。

尝试离开你的电脑桌去与人沟通，相信我，这会给你的工作和生活带来积极的变化。

7. 要为优秀的人工作

我曾面试过一位优秀的 UI 设计师（“技匠”公众号中《给年轻程序员的职涯建议》一文中提到的），当问到他的离职原因时，他告诉我，当他每天听到他的老板和同事们聊天内容的大多是股票、育儿、游戏时，他就下定决心要离职了。

环境对一个人的影响是巨大的，而最可怕的是当你身处其中时，很难意识到你正在变得越来越糟。我之前带过一些不错的程序员，他们中的一些人去了一些整体氛围或环境不是太好的公司。过了一段时间后，再次与他们碰面聊天时，我发现他们的思维、观点相较之前并没有提升，有些甚至反而退步了。

其实，我并不鼓励程序员盲目跳槽，但当你发现所在的环境和周围的人已经无法让你获得提升时，不妨学一下我提到的那位设计师，选择一个更好的环境，尽可能与那些优秀的人一起工作。

8. 生活、睡眠、旅行

我们时常听到关于程序员因疲劳过度而猝死的新闻，我非常为他们感到惋惜。但这里所反映的一个问题却值得深思，那便是程序员不懂得生活。我希望每个程序员都能明白工作、技术、写代码这些并不是你存在的意义，而生活才是，你需要懂得生活，并且学会生活。

- 生活：尝试有节制和有规律的生活，程序员生涯绝不是一次冲刺跑，而

更像是一场马拉松。你需要合理规划自己的时间分配（学习、阅读、写代码）并持之以恒地去做。另外，不要忽视你的家庭，扮演好你在家庭中的角色，无论你是与父母还是与妻儿住在一起，不要将自己隔离起来，尝试融入其中，做一个快乐有爱的程序员。

- 睡眠：有很多人觉得睡眠是弱者的表现，他们往往会长时间熬夜，其实我觉得这是一种恶性循环，反而会使你的工作效率变得更低。充足的睡眠往往能使你更有效率地投入到新一天的工作中，你也会变得更有创造力。请记住，保证充足的睡眠将使你变得更加强大。
- 旅行：去任何一个新的地方都可以称之为旅行，旅行不是为了放松，因为你需要坐飞机、开车、走路，这些都会使你产生压力，但你仍需要去旅行，因为这能让你发现和感受新的东西，而这些是从电脑屏幕上无法获得。

9. 相信自己的天赋和创造力

做到以上这些，你已经是一名优秀的程序员了，但你离杰出还差了一点。你需要依靠你的天赋和创造力，让你更进一步。天赋和创造力绝不是那些天才的专利，每个人的身上都有属于他自己的天赋和创造力，但它们也绝不是与生俱来的，你需要在生活中不断地培养和发掘它们。下面是我觉得一些行之有效的方法。

- 阅读优秀的书籍：好的想法绝不是凭空产生的，尝试从书中寻找那些能激发你创意和灵感的优秀内容。
- 记录和收集：尝试用一个小本子，将你转瞬即逝的好想法记录下来，它们可能并不直接有效，但下一个更好的想法可能就是从这些你记录下来的想法中产生的。
- 尝试动手：光有好的想法是不够的，你需要成为一个有工匠精神的人，通过亲自动手去尝试和实践，你会不断从中得到新的创造力。

如果上面提到的其他建议都对你无效的话，那么就请坚持第2点建议吧“比一般人更加努力”，因为它将成为你最大的竞争优势。

二、那些程序员们后知后觉的职涯经验

在那些流行的博客平台上，你每天都能看到很多关于职场的文章，而这些文章的作者大多是各个行业的所谓职场达人。但我却发现，程序员作为一个很大的职业群体，却极少有人写这类文章。回想自己的职业生涯，在担任程序员或架构师的十年时间里，很少会去思考或总结那些职场经验，或许大多数年轻程序员也和当年的我一样一门心思只关注于技术，而对这些所谓的职场法则后知后觉吧！

这几年，当我开始从事 IT 管理工作后，反而有了更多时间，回顾自己，以及从身边更多年轻程序员的身上看到并体会到一些东西。我觉得让更多程序员能够尽早明白这些，或许对他们未来的职业生涯发展会有所帮助。

1. 你的薪酬与工作量无关

这第一条或许就会让你感觉有些沮丧，但仔细想想这是不是一个普遍的事实呢？当你大学毕业后进入一家公司，每天非常努力地工作，还时常加班，而同一个团队中的一些老员工看上去却一点也不忙，更可气的是，他们的工资可能还比你高出几倍。此时，你的内心是否会有些许不平衡，甚至心生不满呢？

你的薪酬其实取决于很多因素，技术能力、经验资历、工作量等，但最本质的却是，你对公司是否重要，换句话说你是否容易被取代。公司很容易找到一个和你差不多的应届毕业生，而那些对公司产品非常熟悉，并且起到关键作用的老员工，要想替代他们，公司所需要付出的代价及需要承担的风险就会高得多。

所以，不妨摆正心态，正确认识到自己在公司中的位置，努力修炼内功，让自己变得越来越重要，相信你的薪资也会随之提升的。

2. 尽可能持续做一件事

既然你对于公司的价值来自于你的不可替代性，那又该如何有效提升它呢？我的建议是尽可能持续做一件事。这既是指技术上的积累，也是指你能完整或较长时间参与同一个项目或开发一个产品。虽然，有时候你所做的工作并

非你个人所能决定，但你仍需要有意识地去主动把握那些能让你持续积累技术或项目经验的机会。

经常有年轻程序员朋友向我诉说他们所遇到的一些困惑，比如，觉得自己在工作中用到的技术太旧了，询问是否应该转向其他技术甚至转行；或是对公司里做的项目不感兴趣，觉得没有前途，是否应该跳槽，等等。我当然会鼓励他们去学习更多不同的东西，但同时我也会提醒他们，技术深度及完整项目经验的重要性，如果你总是在跟随那些新出现的技术和框架，那你很难在某一项技术上达到理想的深度；同样，在一个公司里，如果你总是在更换项目，那你也很难提升自己的价值。参与 10 个项目，不如完整参与一个项目。持续做一件事是要你把每一件事做透、做好，而不是蜻蜓点水，浅尝辄止。

3. 唯一不变的就是变化本身

在这十几年的工作时间里，我唯一看到不变的恰恰是变化本身。我们使用的技术在变，软件领域的实践方法在变，我们所做的项目在变，公司的组织架构在变，我们自己的职位和角色在变，当然还有我们的老板也如走马灯般换来换去。

作为程序员，我们又应该如何应对这些变化呢？我想说，你很难去改变所处的环境，或是阻挡那些变化的大趋势。你所能做的恰恰是培养自己持续学习的能力。在我之前的文章里，曾多次提到过 10 000 小时定律 —— 要成为一个领域的专家一定要花费 10 000 小时以上的时间，而对于程序员来说，所谓的 100 小时定律同样重要 —— 花 100 小时学习或修炼一门新的技术，往往就能超过一般人很多。你需要提升自己快速学习的能力，当你学得越多，往往就能够学得越快，因为知识之间总是存在关联性。慢慢地，你会发现自己能够非常从容地面对那些不断出现的变化，甚至可以提前预判趋势，当机会来临时，总能成为那个有准备的人。

4. 你的声誉非常重要

声誉对于一名程序员来说非常重要。我看到的那些优秀的程序员都有一个共同点，那就是他们在团队和公司内部都有着良好的声誉，而这反过来帮助他