程序员练级攻略（2018）：程序员修养

2018-06-07 陈皓

程序员练级攻略（2018）：程序员修养
陈皓
- 00:00 / 11:40

在完成上述的入门知识学习之后，我们要向专业的计算机软件开发进军了。但是在学习那些专业的知识前，我们先要抽一部分的篇幅来说一下程序员的修养。这是程序员的工程师文化，也就是程序员的价值观，因为如果你的技术修养不够的话，你学再多的知识也是没有用的。

要了解程序员的修养，你可以先从Quora上的这个贴子开始 “What are some of the most basic things every programmer should know?”，我摘录一些在这里供你参考。

- Bad architecture causes more problems than bad code.
- You will spend more time thinking than coding.
- The best programmers are always building things.
- There's always a better way.
- Code reviews by your peers will make all of you better.
- Fewer features for better code is always the right answer in the end.
- If it's not tested, it doesn't work.
- Don't reinvent the wheel, library code is there to help.
- Code that's hard to understand is hard to maintain.
- Code that's hard to maintain is next to useless.
- Always know how your business makes money, that determines who gets paid what.
- If you want to feel important as a software developer, work at a tech company.

然后是 《97 Things Every Programmer Should Know》，其中有97个非常不错的编程方面的建议。这篇文章是比较经典的，别被"97"吓住，你可以快速浏览一下，会让你有不同的感觉的。另外，在工作一段时间后再来读，你会更有感觉。

英文能力

必须指出，再往下走，有一个技能非常重要，那就是英文。如果对这个技能发怵的话，那么你可能无缘成为一个程序员高手了。因为我们所有的计算机技术全部来自于西方国家，所以如果你要想成为一个高手的话，那么必须到信息的源头去。英文的世界真是有价值的信息的集散地。你可以在那里，到官网上直接阅读手册，到StackOverflow上问问题，到YouTube上看很多演讲和教学，到GitHub上参与社区，用Google查询相关的知识，到国际名校上参加公开课.....

如果你的英文能力有问题的话，那么基本上来说，你无法成为一个高手。因此，学好英文是非常有必要的，我说的不只是读写，还有听和说。相信你在学校里学过英文，有一定的基础。所以，我给你下面的这些建议。

1. 坚持Google英文关键词，而不是在Google里搜中文。
2. 在GitHub上只用英文。用英文写代码注释，写Code Commit信息，用英文写Issue和Pull Request，以及用英文写Wiki。
3. 坚持到YouTube上每天看5分钟的视频。YouTube上有相关的机器字幕，实在不行就打开字幕。
4. 坚持用英文词典而不是中文的。比如：剑桥英语词典 或是 Dictionary.com 。你可以安装一个Chrome插件 Google Dictionary。
5. 坚持用英文的教材而不是中文的。比如：BBC 的 Learning English ，或是到一些ESL网站上看看，如 ESL: English as a Second Language 上有一些课程。
6. 花钱参加一些线上的英文课程，用视频和老外练习。

问问题的能力

提问的智慧（How To Ask Questions The Smart Way）一文最早是由Eric Steven Raymond所撰写的，详细描述了发问者事前应该做好什么，而什么又是不该做的。作者认为这样能让问题容易令人理解，而且发问者自己也能学到较多东西。

此文一经发出，就广受好评，被广泛转载并奉为经典。该文也有 简体中文翻译版 被流传着，所以在华人界也是篇很有名的文章。有两个著名的缩写STFW（Search the fxxking web）以及RTFM（Read the fxxking manual）就是出自本文。

另外，还有一个经典的问题叫 X-Y Problem。对我来说，这是一个很容易犯的错误，所以，你也要小心避免（我曾经在我的Coolshell上写过这个事《X-Y问题》）。

然后，你可以到StackOverflow上看看如何问问题的一些提示-- "FAQ for StackExchange Site"。

作为一个程序员，不做伸手党，你必需要读一读这几篇文章，并努力践行。

写代码的修养

除了《代码大全》外，你还需要补充一些如何写好代码的知识，有以下几本书推荐。

- 《重构：改善既有代码的设计》，这本书是Martin Fowler的经典之作。这本书的意义不仅仅在于"改善既有代码的设计"，也指导了我们如何从零开始构建代码的时候避免不良的代码风格。这是一本程序员必读的书。
- 《修改代码的艺术》，这本书是继《重构》之后探讨修改代码技术的又一里程碑式的著作，而且从涵盖面和深度上都超过了前两部经典（《代码大全》和《重构》）。作者将理解、测试和修改代码的原理、技术和最新工具（自动化重构工具、单元测试框架、仿对象、集成测试框架等），与解依赖技术和大量开发和设计优秀代码的原则、最佳实践相结合，许多内容非常深入。这本书可以让你不仅能掌握最顶尖的修改代码技术，还可以大大提高对代码和软件开发的领悟力。
- 《代码整洁之道》，这本书提出一种观念：代码质量与其整洁度成正比。干净的代码，既在质量上较为可靠，也为后期维护和升级奠定了良好基础。本书作者给出了一系列行之有效的整洁代码操作实践。这些实践在本书中体现为一条条规则（或称"启示"），并辅以来自现实项目正反两面的范例。
- 《程序员的职业素养》，这本书是编程大师Bob大叔40余年编程生涯的心得体会，讲解成为真正专业的程序员需要什么样的态度、原则，需要采取什么样的行动。作者以自己以及身边的同事走过的弯路、犯过的错误为例，意在为后来人引路，助其职业生涯迈上更高台阶。

另外，作为一个程序员，Code Review是非常重要的程序员修养。 Code Review对我的成长非常有帮助，我认为没有Code Review的公司都没有必要呆（因为不做Code Review的公司一定是不尊重技术的）。下面有几篇我觉得还不错的Code Review的文章，供你参考。

- Code Review Best Practices
- How Google Does Code Review
- LinkedIn's Tips for Highly Effective Code Review

除了Code Review之外，Unit Test也是程序员的一个很重要的修养。写Unit Test的框架一般来说都是从JUnit衍生出来的，比如CppUnit之类的。学习JUnit使用的最好方式就是到其官网上看 JUnit User Guide（中文版）。然后，有几篇文章你可以看看（也可以自行Google）：

- You Still Don't Know How to Do Unit Testing
- Unit Testing Best Practices: JUnit Reference Guide
- JUnit Best Practices

安全防范

在代码中没有最基本的安全漏洞问题，也是我们程序员必需要保证的重要大事，尤其是对外暴露Web服务的软件，其安全性就更为重要了。对于在Web上经常出现的安全问题，有必要介绍一下 OWASP - Open Web Application Security Project。

OWASP是一个开源的、非盈利的全球性安全组织，致力于应用软件的安全研究。其被视为Web应用安全领域的权威参考。2009年，国际信用卡数据安全技术PCI标准将其列为必要组件，美国国防信息系统局、欧洲网络与信息安全局、美国国家安全局等政府机构所发布的美国国家和国际立法、标准、准则和行业实务守则参考引用了OWASP。

美国联邦贸易委员会（FTC）强烈建议所有企业需遵循OWASP十大Web弱点防护守则。所以，对于https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project">OWASP Top 10 项目 是程序员非常需要关注的最基本的也是最严重的安全问题，现在其已经成了一种标准，这里是其中文版《OWASP Top 10 2017 PDF 中文版》。

下面是安全编程方面的一些Guideline。

- 伯克立大学的Secure Coding Practice Guidelines。
- 卡内基梅隆大学的 SEI CERT Coding Standards。

此外，有一篇和HTTP相关的安全文章也是每个程序员必需要读的——《Hardening Your HTTP Security Headers》。

最后想说的是"防御性编程"，英文叫Defensive Programming，它是为了保证对程序的不可预见的使用，不会造成程序功能上的损坏。它可以被看作是为了减少或消除墨菲定律效力的想法。防御式编程主要用于可能被滥用，恶作剧或无意地造成灾难性影响的程序上。下面是一些文章。

- The Art of Defensive Programming。
- 当然，也别太过渡了，这篇文章可以看看，Overly defensive programming。

软件工程和上线

系统上线是一件比较严肃的事，这表明你写的软件不是跑在自己的机器上的玩具，或是实验室里的实验品，而是交互给用户使用的，甚至是用户付费的软件。对于这样的软件或系统，我们需要遵守一些上线规范，比如，需要认真测试，并做上线前检查，以及上线后监控。下面是几个简单的规范，供你参考。

- 关于测试，推荐两本书。
  - 《完美软件：对软件测试的各种幻想》，这本书重点讨论了与软件测试有关的各种心理问题及其表现与应对方法。作者首先阐述软件测试之所以如此困难的原因–人的思维不是完美的，而软件测试的最终目的就是发现对改善软件产品和软件开发过程有益的信息，故软件测试是一个信息获取的过程。
  - 《Google软件测试之道》，描述了测试解决方案，揭示了测试架构是如何设计、实现和运行的，介绍了软件测试工程师的角色；讲解了技术测试人员应该具有的技术技能；阐述了测试工程师在产品生命周期中的职责；讲述了测试管理，并对在Google的测试历史上或者主要产品上发挥了重要作用的工程师的访谈，这令那些试图建立类似Google的测试流程或团队的人受益很大。
- 当你的系统要上线时，你是不是已经做好上线的准备了？这里有两个Checklist供你做上线前的一些检查。
  - Server Side checklist
  - Single Page App Checklist
- 《Monitoring 101》这是一篇运维方面的入门文章，告诉你最基本的监控线上运行软件的方法和实践。

小结

好了，总结一下今天分享的主要内容。程序员修养看似与程序员练级关系不大，实际上却能反映出程序员的工程师特质和价值观，决定了这条路你到底能走多远。有修养的程序员才可能成长为真正的工程师和架构师，而没有修养的程序员只能沦为码农。

因此，在这篇文章中，我指出了我认为比较重要的几个方面：英文能力、问问题的能力、写代码的修养、安全防范意识、软件工程和上线规范等。这些能力的训练和培养将为后续的学习和发展夯实基础。

附录：编程规范

我们在写代码时，最好参考一些已有的最佳实践。为什么要有编程规范和最佳实践，要让所有人按一定的规范来编程呢？有下面几个主要原因。

- 可以让你的代码很规整，这有利于代码易读性，从而可以更容易地维护。
- 提升开发效率，我们知道，效率来自于结构化，而不是杂乱。
- 可以让你的软件避免一些容易掉坑的陷阱，也让Bug更少，质量更高。
- 可以让团队成员更高效率地协作。

如果一个程序员没有这类规范和最佳实践的沉淀，那么是很难成为真正的程序员的，只能沦为码农。

当然，对于一些代码风格方面的东西，比如左大括号是否要换行，缩进是用tab还是空格等等，我觉得没有对错，只要团队统一就好了。

下面，我罗列了一堆各种语言的编程规范，供你参考。

编程语言相关

C语言

- NASA C Style。
- C Coding Standard。
- C Programming/Structure and style。
- Linux kernel coding style。
- GNU Coding Standard，GNU的编码规范。

C++语言

- C++ Core Guidelines，这个文档是各种C++的大拿包括原作者在内在持续讨论更新的和C++语言相关的各种最佳实践。
- Google C++ Style Guide。

Go语言

- Effective Go ，Go的语法不复杂，所以，Go语言的最佳实践只需要看这篇官方文档就够了。

Java语言

- Code Conventions for the Java™ Programming Language ，Java官方的编程规范。
- Google Java Style Guide，Google的Java编码规范。

JavaScript语言

- JavaScript The Right Way ，一个相对比较容读的JavaScript编程规范，其中不但有代码规范，还有设计模式，测试工具，编程框架，游戏引擎.....
- Google JavaScript Style Guide，Google公司的JavaScript的编码规范，一个非常大而全的编程规范。
- Airbnb JavaScript Style Guide，Airbnb的JavaScript编程规范。没Google的这么大而全，但是也很丰富了。
- jQuery Core Style Guide，jQuery的代码规范。
- JavaScript Clean Code，前面推荐过的《代码整洁之道》一书中的JavaScript的实践 。

还有一些其它相对比较简单的JavaScript编程规范。

- JavaScript Style Guides And Beautifiers ，这是一篇推荐JavaScript编程规范的文章，你可以看看。
- JavaScript Style Guide and Coding Conventions，这是W3Schools的JavaScript。
- Code Conventions for the JavaScript。

PHP语言

- PHP FIG，PHP编码规范及标准推荐。
- PHP The Right Way，除了编码规范之外的各种PHP的最佳实践，还包括一些设计模式，安全问题，以及服务部署，Docker虚拟化以及各种资源。
- Clean Code PHP，《代码整洁之道》的PHP实践。

Python语言

- Style Guide for Python Code，Python官方的编程码规范。
- Google Python Style Guide，Google公司的Python编码规范。
- The Hitchhiker's Guide to Python，这不只是Python的编程规范，还是Python资源的集散地，强烈推荐。

Ruby语言

- Ruby Style Guide，Airbnb公司的Ruby编程规范。
- Ruby Style Guide 。

Rust语言

- Rust Style Guide。
- Rust Guidelines 开源社区里最好的Rust编程规范。

Scala语言

- Scala Style Guide，Scala官方的编程规范。
- Databricks Scala Guide - Databricks的Scala编程规范。
- Scala Best Practices。

Shell语言

- Google Shell Style Guide，Google的Shell脚本编程规范。

Node.js相关

- npm-coding-style。
- Microsoft + Node.js Guidelines。

Node.js Style Guide。

Mozilla的编程规范

- Mozilla Coding Style Guide，其中包括C、C++、Java、Python、JavaScript、Makefile和SVG等编程规范。

前端开发相关

- CSS Guidelines，CSS容易学，但是不好写，这篇规范会教你如何写出一个健全的、可管理的，并可以扩展的CSS。
- Scalable and Modular Architecture for CSS，这是一本教你如何写出可扩展和模块化的CSS的电子书，非常不错。
- Frontend Guidelines，一些和HTML、CSS、JavaScript相关的最佳实践。
- Sass Guidelines，Sass作为CSS的补充，其要让CSS变得更容易扩展。然而，也变得更灵活，这意味着可以被更容易滥用。这里这篇"富有主见"的规范值得你一读。
- Airbnb CSS / Sass Styleguide， Airbnb的CSS/Sass规范。
- 说了Sass就不得不说LESS，这里有几篇和LESS相关的：LESS Coding Guidelines、LESS Coding Guidelines、LESS coding standard。
- HTML Style Guide，一个教你如何写出性能更高，结构更好，容易编程和扩展的HTML的规范。
- HTML + CSS Code Guide，如何写出比较灵活、耐用、可持续改进的HTML和CSS的规范。
- CoffeeScript Style Guide，CoffeeScript的最佳实践和编程规范。
- Google HTML/CSS Style Guide，Google的HTML/CSS的编程规范。
- Guidelines for Responsive Web Design ，响应式Web设计的规范和最佳实践。
- U.S. Web Design Standards，这是美国政府网端要求的一些UI交互可视化的一些规范。

最后是一个前端开发的各种注意事项列表，非常有用。

- Front-End Checklist ，一个前端开发的Checklist，其中包括HTML、CSS和JavaScript，还和图片、字体、SEO、性能相关，还包括关一些和安全相关的事项，这个列表真的是太好了。

移动端相关

Kotlin

- Coding Conventions。

Objective-C语言

- Objective-C Style guide，Style guide & coding conventions for Objective-C projects。
- Google Objective-C Style Guide。
- NYTimes Objective-C Style Guide ，The Objective-C Style Guide used by The New York Times。

Swift语言

- API Design Guidelines。
- Swift - 一个Swift的相关编程规范的教程。
- Swift style guide。
- Swift Style Guide - LinkedIn的官方 Swift编程规范。
- Metova's Swift style guide。
- Xmartlabs Swift Style Guide，Xmartlabs的 Swift编程规范。

API相关

- HAL，一个简单的API规范教程。
- Microsoft REST API Guidelines，微软软的Rest API规范。
- API Design Guide。
- RESTful API Designing guidelines - The best practices。
- JSON API - Recommendations，JSON相关的API的一些推荐实践。
- API Security Checklist ，API的安全问题的检查列表。

开发工具相关

Markdown相关

- Google Markdown Style Guide。
- Markdown Style Guide。

JSON

- Google JSON Style Guide。
- JSON Style Guide。

Git相关

- Git Style Guide。
- Few Rules from Git Documentation。

正则表达式相关

- RegexHQ。
- Learn regex the easy way。

下面是《程序员练级攻略（2018）》系列文章的目录（持续更新中）。

ZoriChen

2018-06-09

```
# 程序员练级攻略
## 入门教程
### 入门教程1：体会编程是什么
* 与孩子一起学编程
* Codecademy: Learn Python
* People Can Program

### 入门教程2：做一个网页
* MDN Web开发入门


## Python &
* Python编程快速上手
* Python编程：从入门到实践（优先）


## JavaScript
* MDN JavaScript教程
* W3Schools JavaScript教程
* JavaScript全栈教程（廖雪峰）


## Linux
* W3Schools Linux教程


## Visual Studio Code
* Visual Studio Code中文手册


## Web编程入门
### 前端基础
* MDN CSS文档
* MDN HTML文档
* W3Schools JavaScript HTML DOM文档

### 后端基础
* Python
* Node.js
* PHP (W3Schools PHP教程)

### 学习要点
*       HTML
```

学习　　基本语法
* 学习CSS如何选中HTML元素并应用基本样式
* 学习查看网页结构，并动态修改
* 配置LEMP-Ubuntu/Nginx/PHP/MySQL环境
* PHP连接本地/远程数据库MySQL

实践项目
* 做一个Blog或者BBS系统

编程技能Java
编程技巧
* 代码大全

编程语言
* Java核心技术（卷1）
* Head First Java
* Spring in Action
* Spring Boot实战

操作系统
* 鸟哥的Linux私房菜

网络协议
* MDN HTTP文档

数据库设计
* 慕课网 数据库设计的那些事
* MySQL官方文档
* MySQL必知必会

前端方面
* 与JavaScript相关的jQuery
* 与CSS相关的Bootstrap
* JavaScript的Ajax请求后端的API接口方式
* JavaScript的Promise模式

字符编码
* ASCII（关于字符编码，你所需要知道的）
* Unicode

编程工具
* 编程IDE
* 版本管理工具Git
* 调试前端工具
* 数据库设计工具

实践项目
* 投票系统

程序员修养
英文能力

提问能力

代码修养
* 重构：改善既有代码的设计
* 修改代码的艺术
* 代码整洁之道
* 程序员的职业素养
* Code Review
* Unit Test

安全防范
软件工程&上线
* 完美软件：对软件测试的各种幻想
* Google软件测试之道
编程规范

CoolSummer

2018-06-15

我就问一句：罗列的这些书您都都看完了吗？

作者回复

2018-06-15

是的，我都看过的。

Maiza

2018-06-07

木有 .net 不开心。。。。

郑然

2018-06-07

一直想问耗子哥一个问题，这么丰富的引用材料耗子哥是如何获取并积累的？尤其是大量的英文材料。

小鱼儿

2018-06-07

居然没有C#

云学

2018-06-15

2016年带领团队重构了核心资源调度模块，涉及1万行代码，调试了大量case，过程艰辛，收获颇深，再看作者推荐的那些书就很有共鸣

google666s

2018-06-13

订阅专栏最大受益就是各种学习资源汇总，评论区也有很多给力的资源推荐，师傅领进门，修行靠个人，加油��

Silence

2018-06-07

信息量太大了。。。

石头

2018-06-14

程序员的软能力：英文、问问题、写代码、安全规范、软件工程与测试、编程规范

作者回复

2018-06-15

总结的不错

---

qq779527421

2018-06-11

耗子叔，我现在嵌入式开发，你的文章每一篇我都要读好多遍才能记住结构，彻底读懂。这属于正常现象吗？这不能说明我很菜吧。

宋桓公

2018-06-10

能说说Kotlin和Java吗？��

雷霹雳的爸爸

2018-06-08

对耗子叔这篇没写成英文这事，感到五味杂陈

D瓜哥

2018-06-07

耗子哥，阿里的Java代码规范再配上IDEA插件也非常棒！

笨笨熊

2018-06-07

非常期待专业基础篇

---

opsSkateCo

2018-06-28

真得是用年做单位去学习

---

metalmac.kyle

2018-06-26

皓哥，请问下《修改代码的艺术》在哪儿还能买到新书呢？我搜了下目前人邮的07版和机械的14版都卖断货了呢（亚马逊，京东，当当包括淘宝均没货），亚马逊有英文原版实在太贵如果有影印版就好了。

Fishpro

2018-06-19

这篇文章功底太深厚，我几乎每周都要看几遍，提醒我需要学习的东西实在太多，学习知识的过程无疑是充实快乐的，尽管很多到最后可能用不到。

Hillbilly

2018-06-14

大神就是大神阅读量，知识面这么阔，成一个体系

鲸息

2018-06-14

Java 编程规范中，是不是少了阿里巴巴编程规约？

作者回复

2018-06-14

阿里的还是算了，我还是给前沿的吧

---

neilyu

2018-06-10

感觉还有沟通能力和文档能力

---

pomysky

2018-06-09

有修养的程序员才可能成长为真正的工程师和架构师，而没有修养的程序员只能沦为码农。

陈敬秀

2018-06-09

努力做一个合格的工程师！这篇文章信息量大，又是满满的干货啊

名贤集

2018-06-08

写的相当好

云飞扬

2018-06-08

练级系列文章，篇篇价值千金！

云学

2018-06-07

最近2年都在研究clean code

---

cai

2018-06-07

资源很好，谢谢分享

王机智

2018-06-07

如何ke xue shang wang?

---

macworks

2018-06-07

虽然耗子哥下一篇才写编程语言，想知道从你了解下来，哪门语言从设计上就注重代码可读性？也就是说程序员用这种语言很难写出可读性差的代码。我最近看R语言的tidyverse库，好像就有这种特性，当然它并不图灵完备。

---

bluze

2018-06-07

收藏

孙悟空

2018-06-07

请教下code review的问题，有时改动特别小，是否要code review呢?

superryanguo

2018-06-07

推荐的都不错��，但不实践就忘了

Geek_aa9c4c

2018-06-07

读你的文章总是让我收获多多，有种相见恨晚的感觉，同时也感叹自己这几年可怜的知识库，之后这是我学习的目标，一点一点去完成。

到道可道

2018-06-07

这就是大神呀，涉猎好广泛，感觉老师的成长速度像坐了火箭��

ichiro

2018-06-07

学也无涯 知也无涯