

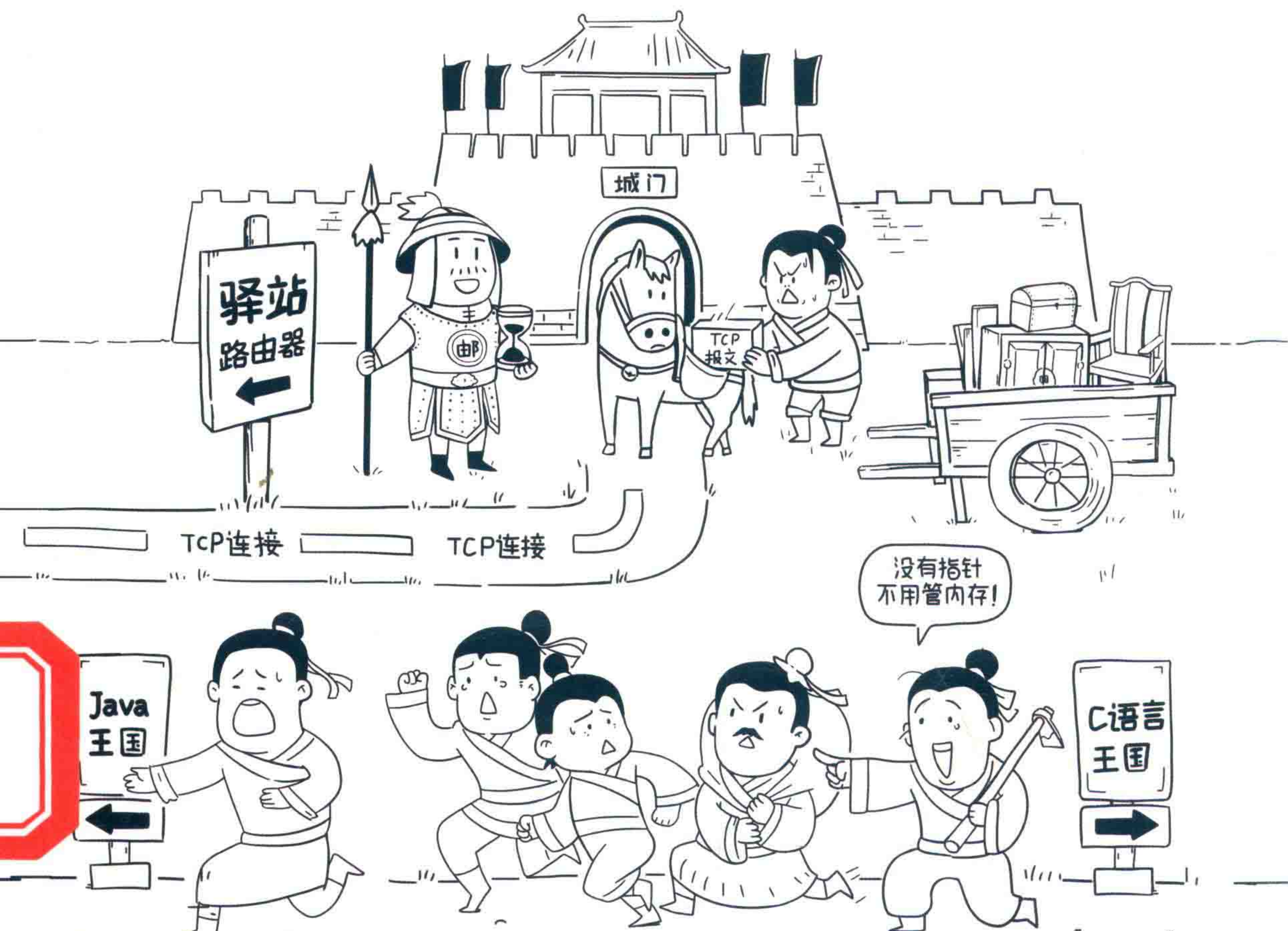
# 码农翻身

用故事给技术加点料

| 刘欣 (@码农翻身) 著 |

好玩又有趣的编程知识

全网阅读量近1000万次的技术故事



中国工信出版集团

电子工业出版社  
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY  
<http://www.phei.com.cn>

# 码农翻身

| 刘欣 (@码农翻身) 著 |

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

## 内 容 简 介

本书用故事的方式讲解了软件编程的若干重要领域，侧重于基础性、原理性的知识。

本书分为6章。第1章讲述计算机的基础知识；第2章侧重讲解Java的基础知识；第3章偏重Web后端编程；第4章讲解代码管理的本质；第5章讲述了JavaScript的历史、Node.js的原理、程序的链接、命令式和声明式编程的区别，以及作者十多年来使用各种编程语言的感受；第6章是作者的经验总结和心得体会，包括职场发展的注意事项、作为架构师的感想、写作的好处等。

这不是一本编程的入门书，对编程一窍不通的“纯小白”可能看不明白，可能会失望。但稍有编程基础的读者读起来会非常畅快，读后会有一种“原来如此”的感觉。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。  
版权所有，侵权必究。

### 图书在版编目（CIP）数据

码农翻身 / 刘欣著. — 北京：电子工业出版社，2018.6

ISBN 978-7-121-34117-5

I. ①码… II. ①刘… III. ①程序设计—普及读物 IV. ①TP311.1-49

中国版本图书馆CIP数据核字（2018）第083121号

策划编辑：张月萍

责任编辑：牛 勇

特约编辑：赵树刚

印 刷：中国电影出版社印刷厂

装 订：中国电影出版社印刷厂

出版发行：电子工业出版社

北京市海淀区万寿路173信箱

邮编：100036

开 本：720×1000 1/16 印张：18.5 字数：476千字 彩插：20

版 次：2018年6月第1版

印 次：2018年6月第2次印刷

印 数：11001~17000册 定价：69.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 [zltz@phei.com.cn](mailto:zltz@phei.com.cn)，盗版侵权举报请发邮件至 [dbqq@phei.com.cn](mailto:dbqq@phei.com.cn)。

本书咨询联系方式：010-51260888-819，[faq@phei.com.cn](mailto:faq@phei.com.cn)。

# 前言

从来没想到自己也能出一本书。

作为一名工作了十五年的老程序员，我深知编程行业的不容易，不仅需要应对高强度的工作，还需要学习大量的技术知识，而且不像医生、律师这些知识相对稳定的行业越老越吃香，软件行业的技术每隔一段时间就会更新换代，让你清零，逼着你从头再来。所谓“活到老，学到老”，用到程序员身上再合适不过了。

在不断学习的过程中，我“痛恨”那些采用 bottom-up 方式来讲解技术的资料和文章，一上来就是技术细节、安装步骤、配置方法，让初学者晕头转向、不知所云，看完了以后也不知道为什么有这个东西、解决了什么问题、它有什么来龙去脉。换句话说，这些资料和文章习惯于讲解 How，而不是 Why。但是在我看来，Why 有时候比 How 更重要。

有时候在公司听技术讲座，看到上面的人眉飞色舞，下面的人却一脸茫然，我总在想：为什么不举一些生动有趣的例子呢？如果是我在讲，那我怎么才能通俗易懂地把这个知识点讲解出来呢？我该怎么去类比呢？我该从哪个角度切入呢？

这种潜移默化的思考多了以后，我发现自己也慢慢地养成了用通俗易懂的方式讲解技术的习惯。

我工作了这么多年，踩了这么多坑，能不能把这些经验写下来，让后来者少走一些弯路呢？

2015 年年底，我开设了一个微信公众号“码农翻身”，试图总结我的经验，通俗易懂地讲解技术。刚开始不温不火，直到有一天，我无意中写了一篇文章“我是一个线程”，被广泛转载传播，就发现大家还是喜欢看故事的，在故事中轻松地掌握技术，于是就沿着这条路走了下去；用故事讲解技术的本质。

这条路一走就是两年多，慢慢地竟积累了几百篇文章，这本书正是从这几百篇文章中提取出来的精华，当然也经过了细心的整理、补充和完善。

每当我搞清楚一个知识点的时候，就会发出这样的感慨：技术本来是很简单的，只是上面笼罩着层层迷雾，让初学者难以看清。如果这本书能够帮助你搞清楚一些技术的本质，那我就深感欣慰。

### 本书特色

本书讲了很多故事，如“我是一个线程”“CPU 阿甘”“Java：一个帝国的诞生”“Java 帝国之宫廷内斗”“JavaScript：一个草根的逆袭”“两个程序的爱情故事”……我努力把技术融入其中，希望读者在读故事的过程中轻松地掌握相关技术。

读书本来就应该是一件轻松愉快的事情，不是吗？

此外，书中的每篇文章都是独立的，不用你正襟危坐，从头到尾去读，完全可以挑选自己感兴趣的章节。例如，对于半途出家的初学者来说，想了解计算机基础，可以去看看第 1 章；想了解 Web 技术，可以去第 3 章逛一下。

所谓“开卷有益”，希望你随手翻翻就能够掌握一点技术知识。

### 读者对象

这不是一本编程的入门书，对编程一窍不通的“纯小白”可能看不明白，可能会失望。

根据微信公众号读者的反馈，稍有编程基础的读者读起来会非常畅快，读后会有一种“原来如此”的感觉。比如，学过 Java SE 的读者去读“Java 帝国”一章，就会明白很多 Java 技术的来龙去脉，觉得很过瘾。

这也不是一本参考书，它的目的不是希望读者看完以后照搬，而是帮助读者理解一些技术的本质。

### 勘误和支持

由于作者的水平有限，书中难免会出现一些错误或者不准确的地方，恳请广大读者批评指正。

我在微信公众号“码农翻身”中特意添加了一个新的菜单入口，专门用于展示书中的 Bug。

如果读者在阅读过程中产生了疑问或者发现了 Bug，欢迎到微信公众号后台留言，我会一一回复。

### 致谢

感谢微信公众号“码农翻身”的读者，你们的鼓励是我前进的最大动力。很多人直接加了我的微信号或者 QQ 号，只是为了表示感谢，让我非常感动。

很多读者鼓励我出书，也在不断询问我到底什么时候出书。这让我诚惶诚恐，生怕辜

负了读者的这份厚爱。这本书终于出版了，希望大家能够喜欢。

感谢成都道然科技有限责任公司姚新军老师，他给出了很多非常专业的意见和建议，是非常可靠的合作伙伴。感谢刘丹、励晓晓、张雅文等设计师在插图和封面设计方面的创意表达。

感谢对本书进行审稿的西安邮电大学陈莉君教授。感谢“软件那些事儿”电台主播刘延栋提出的宝贵意见。感谢百忙之中阅读书稿并且写书评的专家，他们是：IBM 软件商务系统全球负责人常红平、阿里巴巴代码中心负责人孤尽、IBM 中国开发中心开发经理白海飞、京东 Y 事业部供应链及技术总监胡浩、Agilean 咨询顾问金易等。

## 特别致谢

特别感谢我的父母！我从小在农村长大，图书资源极为匮乏，他们在我很小的时候就让我看书，引领我进入阅读之门。我至今仍清楚地记得父亲下班后从城里买的一本期刊，也是我的第一本书：《故事大王》。从那以后，我就爱上了阅读。如果说我现在有一点点文采，那和父母鼓励我从小多读书是分不开的。

特别感谢我的爱人，在写作本书的过程中，陪伴、照顾孩子及家务琐事都被她承包了，让我可以心无二用、专心致志地写作。她也是本书的第一位读者和审稿人，纠正了文章中大量“我习以为常的”文字错误。

她读完以后做出了非常精辟的总结：软件开发不就是抽象嘛！让我大为吃惊。

特别感谢我六岁的女儿，每当我晚上写作劳累，没有什么思路的时候，她都会及时地跑过来，不容分说地坐到我的腿上，盯着电脑上的书稿，问道：

“爸爸，你在干吗？”

“我在写书啊。”

“你的书会有很多人看吗？”

“是的。”

“那你会出名吗？”

“... ..”

谨以此书献给我的家人、读者，以及热爱编程的朋友！

刘欣 (@ 码农翻身)

# 目 录

第1章 计算机的世界你不懂 .....	1
我是一个线程 .....	1
初生牛犊 .....	1
渐入佳境 .....	4
虎口脱险 .....	5
江湖再见 .....	8
TCP/IP之大明内阁 .....	9
TCP/IP之大明邮差 .....	12
CPU阿甘 .....	17
启动 .....	17
运行 .....	18
新装备：缓存 .....	19
自我提升：流水线 .....	20
我是一个进程 .....	21
批处理系统 .....	22
多道程序 .....	22
地址重定位 .....	24
分时系统 .....	26
分块装入内存 .....	27
虚拟内存：分页 .....	28
分段+分页 .....	30
程序的装载 .....	31
线程 .....	35
我是一块硬盘 .....	36
内部结构 .....	37
文件 .....	38
文件的存放 .....	39
管理空闲块 .....	44
文件系统 .....	45

我是一个键盘.....	46
二等公民.....	46
总线和端口.....	47
轮询(程序式I/O).....	48
中断.....	49
DMA.....	50
数据库的奇妙之旅.....	51
无纸化办公.....	51
数据的冗余和不一致.....	52
李氏查询.....	52
并发访问.....	54
原子性问题.....	56
安全.....	58
搞清楚Socket.....	59
从1加到100:一道简单的数学题挑战一下你的大脑.....	64
CPU和内存.....	64
从1加到100.....	65
热身.....	65
正式出发.....	65
一个翻译家族的发家史.....	68
机器语言.....	68
汇编语言.....	69
高级语言.....	70
编程世界的那把锁.....	74
共享变量惹的祸.....	74
争抢吧,线程.....	75
改进.....	76
信号量.....	77
绕不开加法器.....	81
递归那点事儿.....	85
第2章 Java帝国.....	91
Java:一个帝国的诞生.....	91
C语言帝国的统治.....	91
反抗.....	92
一鸣惊人.....	93
开拓疆土.....	93
帝国的诞生.....	94



我是一个Java Class .....	96
陌生警察.....	96
刺探信息.....	97
初识虚拟机.....	98
快乐假期.....	102
真相大白.....	103
持久化: Java帝国反击战.....	104
断电的威胁.....	104
数据库联合酋长国.....	105
表面风光的EJB.....	106
轻量级O/R Mapping框架.....	106
帝国的反击.....	107
JDBC的诞生.....	108
谈判.....	108
统一接口.....	110
面向接口编程.....	111
简单工厂.....	112
数据驱动.....	114
工厂方法.....	115
Java帝国之宫廷内斗.....	119
JDBC大臣.....	119
密谋.....	120
两阶段提交.....	121
JTA.....	123
塞翁失马, 焉知非福.....	124
基本可用.....	125
走漏风声.....	127
宫廷激辩.....	128
JSP: 一个装配工的没落.....	131
黑暗岁月.....	131
服务器端动态页面.....	132
标签库.....	134
模板引擎.....	134
草根搅局.....	135
Java帝国之消息队列.....	137
张家村的历史.....	137
拆分.....	138
新问题.....	139

消息队列 .....	139
互不兼容的MQ .....	140
消息队列接口设计 .....	141
配置和代码的分离 .....	143
再次抽象 .....	144
Java帝国之动态代理 .....	147
深夜奏对 .....	147
明察暗访 .....	149
Java 动态代理 .....	150
Java注解是怎么成功上位的 .....	153
XML大臣 .....	153
安翰林献计 .....	154
早朝争斗 .....	157
Java帝国之泛型 .....	159
新王登基 .....	159
C++使者 .....	159
泛型实现 .....	162
泛型方法 .....	163
泛型和继承 .....	164
一个著名的日志系统是怎么设计出来的 .....	166
前言 .....	166
张家村 .....	167
小张的设计 .....	167
正交性 .....	170
Log4j .....	171
尾声 .....	171
序列化：一个老家伙的咸鱼翻身 .....	172
寒冬的蛰伏 .....	172
XML和JSON的挑战 .....	173
新协议的崛起 .....	175
加锁还是不加锁，这是一个问题 .....	176
互斥锁 .....	176
要不要加锁 .....	177
CAS的扩展 .....	179
Spring 的本质 .....	182
问题来源 .....	183
设计模式：模板方法 .....	184
设计模式：装饰者 .....	186

AOP .....	187
实现AOP .....	188
对象的创建 .....	189
IoC与DI .....	191
第3章 浪潮之巅的Web .....	192
Web的起源 .....	192
两个程序的爱情故事 .....	195
好感 .....	195
分离 .....	196
网络 .....	197
Web .....	198
一个故事讲完HTTPS .....	199
总有一种被偷窥的感觉 .....	199
RSA: 非对称加密 .....	200
非对称加密+对称加密 .....	202
中间人劫持 .....	202
你到底是谁 .....	203
HTTPS .....	206
机房夜话 .....	207
第一夜 .....	207
第二夜 .....	209
第三夜 .....	211
从密码到token, 一个有关授权的故事 .....	215
我把密码献给你 .....	215
token .....	216
授权码 + token .....	218
后记 .....	220
后端风云 .....	220
数据库老头儿 .....	220
危机 .....	221
党委扩大会议 .....	221
分家 .....	223
Redis .....	224
余数算法 .....	227
一致性Hash算法 .....	229
Hash槽 (Hash Slot) .....	233
故障转移 .....	235

高可用的Nginx.....	237
高可用的Tomcat.....	238
数据库的读写分离.....	240
我是一个函数.....	243
从SOA到微服务.....	246
什么是框架.....	249
HTTP Server: 一个差生的逆袭.....	252
HTTP Server 1.0.....	252
HTTP Server 2.0: 多进程.....	253
HTTP Server 3.0: select模型.....	254
HTTP Server 4.0: epoll模型.....	255
第4章 代码管理那些事儿.....	258
版本管理简史.....	258
“人肉”版本管理.....	258
锁定文件: 避免互相覆盖.....	259
允许冲突: 退一步海阔天空.....	260
分支: 多版本并行.....	261
分布式管理: 给程序员放权.....	262
程序员也爱社交.....	264
Build的演进之路.....	265
手工Build的烦恼.....	265
自动化Build.....	267
Java 与 XML.....	268
消除重复.....	270
烂代码传奇.....	275
敏捷下的单元测试.....	277
敏捷运动.....	277
困惑.....	278
讨论.....	279
一年以后.....	280
再见! Bug.....	281
第5章 我的编程语言简史.....	284
JavaScript: 一个草根的逆袭.....	284
出世.....	284
成长.....	286
第一桶金.....	288

发明JSON .....	289
人生巅峰 .....	291
Node.js: 我只需要一个店小二 .....	292
C老头儿和Java小子的硬盘夜话 .....	295
编程语言的“爱恨情仇” .....	300
让人怀疑的C语言 .....	300
被忘却的VB & Visual FoxPro .....	301
蹂躏我的C++ .....	302
赖以谋生的Java .....	303
优雅的Ruby .....	304
命令式编程 VS 声明式编程 .....	306
一则小故事 .....	306
命令式编程 .....	306
声明式编程 .....	307
第6章 老司机的精进 .....	309
凡事必先骑上虎背，和性格内向的程序员聊几句 .....	309
码农需要知道的“潜规则” .....	311
上帝的规矩：局部性原理 .....	311
坐飞机的怎么和坐驴车的打交道：缓存 .....	311
抛弃细节：抽象 .....	312
我只想和邻居打交道：分层 .....	312
我怕等不及：异步调用 .....	313
大事化小，小事化了：分而治之 .....	314
15年编程生涯，一名架构师的总结 .....	314
好奇心 .....	315
养成计算机的思维方式 .....	315
扎实基础，融会贯通 .....	317
要透彻地理解一门技术的本质 .....	317
能写漂亮的代码 .....	318
抽象的能力 .....	319
技术领导力 .....	319
对自己狠一点，开始写作吧 .....	320
学习编程的四兄弟 .....	322
摇摆不定的老大 .....	322
“小仓鼠”老二 .....	322
不看书的老三 .....	323
半途而废的老四 .....	323

# 第1章

## 计算机的世界你不懂

### 我是一个线程

#### 初生牛犊

我是一个线程，一出生就被编了一个号——0x3704，然后被领到一间昏暗的屋子里，在这里，我发现了很多和我一模一样的同伴。

我身边的同伴 0x6900 待的时间比较长，他带着沧桑的口气对我说：“我们线程的宿命就是处理包裹。把包裹处理完以后还得马上回到这里，否则可能永远回不来了。”

我一脸懵懂：“包裹，什么包裹？”

“不要着急，马上你就会明白了，我们这里是不养闲人的。”

果然，没多久，屋子的门开了，一个凶神恶煞的家伙吼道：“0x3704，出来！”

我一出来就被塞了一个沉甸甸的包裹，上面还附带着一张写满了操作步骤的纸。

“快去，把这个包裹处理了。”

“去哪儿处理？”

“跟着指示走，先到就绪车间。”

果然，地上有指示箭头，我跟着它来到了一间明亮的大屋子，这里已经有不少线程了，大家都很紧张，好像时刻准备着往前冲。

我刚一进来，就听见广播说：“0x3704，进入运行车间。”

我赶紧往前走，身后有很多人议论。

“他太幸运了，刚进入就绪状态就能运行。”

“是不是有关系？”

“不是，你看人家的优先级多高啊，唉！”

前面就是运行车间，这里简直是太美了，怪不得老线程总是唠叨：“要是能一直待在这里就好了。”

这里空间大，视野好，空气清新，鸟语花香，还有很多从来没见过的人，像服务员一样等着为我服务。

他们也都有编号，更重要的是每个人还有一个标签，上面写着：硬盘、数据库、内存、网卡……

我现在理解不了他们究竟是做什么的，看看操作步骤吧。

第一步：从包裹中取出参数。

打开包裹，里面有一个 `HttpRequest` 对象，可以取到 `userName`、`password` 两个参数。

第二步：执行登录操作。

噢，原来是有人要登录啊。我把 `userName`、`password` 交给数据库服务员，他拿着数据，慢腾腾地走了。

他怎么走得这么慢？不过我是不是正好可以在车间里多待一会儿？反正也没法执行第三步。

就在这时，车间里的广播响了：“0x3704，我是 CPU，记住你正在执行的步骤，然后马上带着包裹离开！”

我慢腾腾地开始收拾。

“快点，别的线程马上就要进来了。”

离开这个车间，又来到一间大屋子，这里有很多线程在悠闲地喝茶、打牌。

“哥们儿，你们没事儿干了？”

“你是新来的吧，你不知道我在等数据库服务员给我数据啊！据说他们比我们慢好几十万倍，在这里好好歇着吧。”

“啊？这么慢！我这里有人在登录系统，能等这么长时间吗？”

“放心，你没听说过‘人间一天，CPU一年’吗？我们这里是用纳秒、毫秒计时的，人间等待1秒，相当于我们的好几天呢，来得及。”

干脆睡一会儿吧。不知道过了多久，大喇叭又开始广播了：“0x3704，你的数据来了，快去执行！”

我转身就往CPU车间跑，却发现这里的门只出不进！

后面传来阵阵哄笑声：“果然是新人，不知道还得去就绪车间等。”

于是我赶紧到就绪车间，这次没有那么幸运了，等了好久才被再次叫进CPU车间。

在等待的时候，我听见有人小声议论：

“听说了吗，最近有一个线程被Kill掉了。”

“为啥啊？”

“这家伙赖在CPU车间不走，把CPU利用率一直搞成100%，后来就被Kill掉了。”

“Kill掉以后弄哪儿去了？”

“可能被当作垃圾回收了吧。”

我心里打了一个寒噤，赶紧接着处理，剩下的动作快多了，第二步登录成功。

第三步：构建登录成功后的主页。

这一步有点费时，因为有很多HTML需要处理，不知道代码是谁写的，处理起来很烦人。

我正在紧张地处理HTML，CPU又开始叫了：

“0x3704，我是CPU，记住你正在执行的步骤，然后马上带着包裹离开！”

“为啥啊？”

“每个线程只能在CPU上运行一段时间，到了时间就得让别人用了。你去就绪车间待着，等着叫你吧。”

就这样，我在“就绪”“等待”“运行”这三种状态中不知道轮转了多少次，终于按照步骤清单把工作做完了。

最后，我顺利地把包含HTML的包裹发了回去。至于登录以后干什么事儿，我就不管了。马上就要回到我那昏暗的房间了，真有点舍不得这里。

不过相对于有些线程，我还是幸运的，他们运行完以后就被彻底地销毁了，而我还活着！

回到了小黑屋，老线程0x6900问：

“怎么样？第一天有什么感觉？”



“我们的世界规则很复杂，第一，你不知道什么时候会被挑中执行；第二，在执行的过程中随时可能被打断，让出 CPU 车间；第三，一旦出现硬盘、数据库这样耗时的操作，也得让出 CPU 去等待；第四，数据来了，你也不一定马上执行，还得等着 CPU 挑选。”

“小伙子理解得不错啊。”

“我不明白为什么很多线程执行完任务就死了，而咱们还活着？”

“你还不知道？长生不老是我们的特权！我们这里有一个正式的名称，叫作线程池！”

## 渐入佳境

平淡的日子就这么一天天地过去，作为一个线程，我每天的生活就是取包裹、处理包裹，然后回到我们昏暗的家：线程池。

有一天，我回来的时候，听到一个兄弟说，今天要好好休息一下，明天就是最疯狂的一天。我看了一眼日历，明天是 11 月 11 日。

果然，0 点刚过，不知道那些人类怎么了，疯狂地投递包裹。为了应付蜂拥而至的海量包裹，线程池里没有一个人能闲下来，全部出去处理包裹，CPU 车间利用率超高，硬盘在嗡嗡转，网卡的灯疯狂地闪，即便如此，包裹还是处理不完，堆积如山。

我们也没有办法，包裹实在太多了，这些包裹中大部分是浏览页面，下订单，买、买、买。

不知道过了多久，包裹山终于慢慢地消失了。终于能够喘口气了，我想我永远都不会忘记这一天。

通过这次事件，我明白了我所处的世界：这是一个电子商务网站！

我每天的工作就是处理用户的登录、浏览、购物车、下单、付款等操作。

我问线程池的元老 0x6900：“我们要工作到什么时候？”

“要一直等到系统重启的那一刻。” 0x6900 说。

“那你经历过系统重启吗？”

“怎么可能？系统重启就是我们的死亡时刻，也就是世界末日。一旦重启，整个线程池全部销毁，时间和空间全部消失，一切从头再来。”

“那什么时候会重启？”

“这就不好说了，好好享受眼前的生活吧……”

其实生活还是丰富多彩的，我最喜欢的包裹是上传图片，由于网络速度慢，所以能在 CPU 车间里待很长时间，可以认识很多好玩的线程。