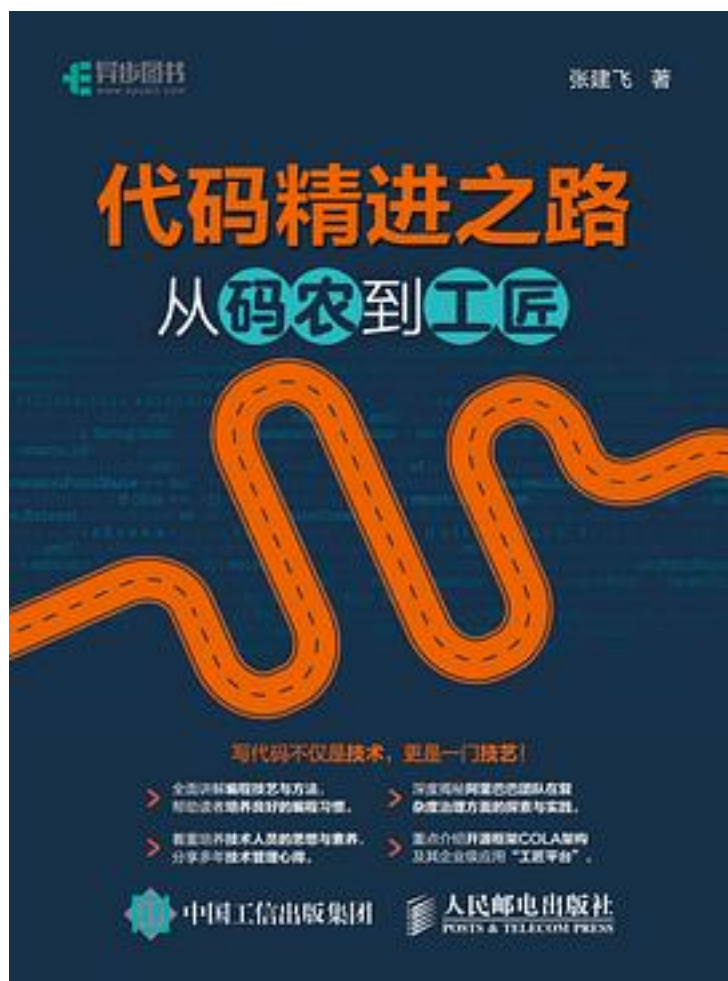


代码精进之路 从码农到工匠



[代码精进之路 从码农到工匠 下载链接1](#)

著者:张建飞

出版者:人民邮电出版社

出版时间:2020-1-1

装帧:精装

isbn:9787115521026

这是一本为专业程序员而写的书，写好代码、追求卓越和工匠精神是每个程序员都应该具备的优秀品质。

本书共有13章内容，主要分为技艺部分、思想部分和实践部分。技艺部分详细介绍了编程技巧和方法论，并配以详尽的代码案例，有助于读者提高编写代码的能力，优化代码质量。思想部分主要包括抽象能力、分治思想，以及程序员应该具备的素养等内容。实践部分主要介绍了常见的应用架构模式，以及COLA架构的设计原理。

作者介绍:

张建飞，阿里巴巴集团高级技术专家，Java全球管理组织（JCP）执行委员会正式会员（Full Member）。2007年计算机工程硕士毕业后，先后在软件公司InfoSys与互联网公司eBay担任高级研发和技术专家的职务。2014年加入阿里巴巴，先后在1688、ICBU和零售通担任技术主管。

作者精通面向对象技术，有丰富的一线编码实战和架构经验。特别是在应用架构、领域建模和复杂度治理领域，自研了COLA框架。COLA自开源以来，已经被多个技术团队使用，解决了DDD落地和应用扩展问题，受到了普遍关注和一致好评。

作者提倡“工匠精神”，对于如何打造一个追求卓越、独具匠心的技术团队，如何量化考核工程师的技术贡献，都有着非常深入的思考和实践，并探索出一套切实可行的方法论。基于该方法论打造的“工匠平台”，在阿里巴巴内部被广泛使用，“工匠平台”丰富了对技术人员考察的维度，是除业务结果之外的从技术视角给技术人员“照镜子”的有效工具。

目录: 第一部分 技 艺

第1章 命名 / 3
1.1 命名的力量 / 3
1.2 命名其实很难 / 4
1.3 有意义的命名 / 5
1.3.1 变量名 / 5
1.3.2 函数名 / 5
1.3.3 类名 / 6
1.3.4 包名 / 7
1.3.5 模块名 / 7
1.4 保持一致性 / 7
1.4.1 每个概念一个词 / 8
1.4.2 使用对仗词 / 8
1.4.3 后置限定词 / 9
1.4.4 统一业务语言 / 10
1.4.5 统一技术语言 / 10
1.5 自明的代码 / 10
1.5.1 中间变量 / 11
1.5.2 设计模式语言 / 11
1.5.3 小心注释 / 12
1.6 命名工具 / 14
1.7 本章小结 / 15
第2章 规范 / 16
2.1 认知成本 / 16
2.2 混乱的代价 / 17
2.3 代码规范 / 18
2.3.1 代码格式 / 18
2.3.2 空行规范 / 19
2.3.3 命名规范 / 21

2.3.4 日志规范 / 22
2.3.5 异常规范 / 25
2.4 埋点规范 / 28
2.5 架构规范 / 30
2.6 防止破窗 / 30
2.7 本章小结 / 31
第3章 函数 / 32
3.1 什么是函数 / 32
3.2 软件中的函数 / 33
3.3 封装判断 / 33
3.4 函数参数 / 34
3.5 短小的函数 / 35
3.6 职责单一 / 36
3.7 精简辅助代码 / 37
3.7.1 优化判空 / 37
3.7.2 优化缓存判断 / 38
3.7.3 优雅降级 / 39
3.8 组合函数模式 / 40
3.9 SLAP / 43
3.10 函数式编程 / 48
3.11 本章小结 / 49
第4章 设计原则 / 51
4.1 SOLID概览 / 51
4.2 SRP / 52
4.3 OCP / 53
4.4 LSP / 54
4.4.1 警惕instanceof / 55
4.4.2 子类覆盖父类函数 / 55
4.5 ISP / 57
4.6 DIP / 58
4.7 DRY / 61
4.8 YAGNI / 61
4.9 Rule of Three / 62
4.10 KISS原则 / 62
4.11 POLA原则 / 63
4.12 本章小结 / 63
第5章 设计模式 / 64
5.1 模式 / 64
5.2 GoF / 65
5.3 拦截器模式 / 69
5.4 插件模式 / 73
5.5 管道模式 / 75
5.5.1 链式管道 / 75
5.5.2 流处理 / 78
5.6 本章小结 / 79
第6章 模型 / 81
6.1 什么是模型 / 81
6.1.1 物理模型 / 82
6.1.2 数学模型 / 82
6.1.3 概念模型 / 82
6.1.4 思维模型 / 83
6.1.5 模型不能代替现实 / 83
6.2 UML / 84
6.3 类图 / 85

6.3.1 类的UML表示法 / 86	
6.3.2 类的关联关系 / 87	
6.3.3 类的依赖关系 / 92	
6.3.4 类的泛化关系 / 93	
6.3.5 接口与实现关系 / 94	
6.4 领域模型 / 95	
6.5 敏捷建模 / 96	
6.6 广义模型 / 97	
6.6.1 C4模型 / 97	
6.6.2 UI流程图 / 97	
6.6.3 业务模型 / 98	
6.7 本章小结 / 99	
第7章 DDD的精髓 / 101	
7.1 什么是DDD / 101	
7.2 初步体验DDD / 102	
7.3 数据驱动和领域驱动 / 104	
7.3.1 数据驱动 / 104	
7.3.2 领域驱动 / 106	
7.3.3 ORM / 108	
7.4 DDD的优势 / 109	
7.4.1 统一语言 / 110	
7.4.2 面向对象 / 110	
7.4.3 业务语义显性化 / 111	
7.4.4 分离业务逻辑和技术细节 / 111	
7.5 DDD的核心概念 / 112	
7.5.1 领域实体 / 112	
7.5.2 聚合根 / 114	
7.5.3 领域服务 / 115	
7.5.4 领域事件 / 116	
7.5.5 边界上下文 / 117	
7.6 领域建模方法 / 118	
7.6.1 用例分析法 / 118	
7.6.2 四色建模法 / 121	
7.7 模型演化 / 127	
7.8 为什么DDD饱受争议 / 127	
7.8.1 照搬概念 / 128	
7.8.2 抽象的灵活性 / 128	
7.8.3 领域层的边界 / 128	
7.9 本章小结 / 130	
第二部分 思想	
第8章 抽象 / 133	
8.1 伟大的抽象 / 133	
8.2 到底什么是抽象 / 134	
8.3 抽象是OO的基础 / 135	
8.4 抽象的层次性 / 136	
8.5 如何进行抽象 / 137	
8.5.1 寻找共性 / 137	
8.5.2 提升抽象层次 / 139	
8.5.3 构筑金字塔 / 142	
8.6 如何提升抽象思维 / 143	
8.6.1 多阅读 / 144	
8.6.2 多总结 / 144	
8.6.3 领域建模训练 / 145	
8.7 本章小结 / 145	

第9章 分治 / 146	
9.1 分治算法 / 146	
9.1.1 归并排序 / 147	
9.1.2 二分搜索 / 148	
9.1.3 K选择问题 / 149	
9.2 函数分解 / 150	
9.3 写代码的两次创造 / 150	
9.3.1 第一遍实现功能 / 150	
9.3.2 第二遍重构优化 / 151	
9.4 分治模式 / 151	
9.5 分层设计 / 152	
9.5.1 分层网络模型 / 152	
9.5.2 分层架构 / 153	
9.6 横切和竖切 / 154	
9.7 本章小结 / 155	
第10章 技术人的素养 / 156	
10.1 不教条 / 156	
10.1.1 瀑布还是敏捷 / 157	
10.1.2 贫血还是充血 / 158	
10.1.3 单体还是分布式 / 159	
10.2 批判性思维 / 161	
10.3 成长型思维 / 162	
10.4 结构化思维 / 163	
10.4.1 如何落地新团队 / 165	
10.4.2 如何做晋升述职 / 166	
10.5 工具化思维 / 167	
10.6 好奇心 / 169	
10.7 记笔记 / 170	
10.8 有目标 / 171	
10.9 选择的自由 / 172	
10.10 平和的心态 / 173	
10.11 精进 / 174	
10.12 本章小结 / 174	
第11章 技术Leader的修养 / 175	
11.1 技术氛围 / 175	
11.1.1 代码好坏味道 / 176	
11.1.2 技术分享 / 176	
11.1.3 CR周报 / 177	
11.1.4 读书会 / 178	
11.2 目标管理 / 179	
11.2.1 什么是OKR / 179	
11.2.2 SMART原则 / 180	
11.2.3 OKR设定 / 181	
11.3 技术规划 / 182	
11.3.1 当前问题 / 182	
11.3.2 技术领域 / 183	
11.3.3 业务领域 / 183	
11.3.4 团队特色 / 183	
11.4 推理阶梯 / 184	
11.5 Leader和Manager的区别 / 185	
11.6 视人为人 / 186	
11.7 本章小结 / 187	
第三部分 实践	
第12章 COLA架构 / 191	

12.1 软件架构 / 191
12.2 典型的应用架构 / 193
12.2.1 分层架构 / 193
12.2.2 CQRS / 195
12.2.3 六边形架构 / 196
12.2.4 洋葱架构 / 198
12.2.5 DDD / 199
12.3 COLA架构设计 / 200
12.3.1 分层设计 / 200
12.3.2 扩展设计 / 201
12.3.3 规范设计 / 205
12.3.4 COLA Archetype / 208
12.4 COLA测试 / 209
12.4.1 单元测试 / 209
12.4.2 集成测试 / 210
12.4.3 ColaMock / 210
12.5 COLA架构总览 / 212
12.6 本章小结 / 214
第13章 工匠平台 / 215
13.1 项目背景 / 215
13.2 整理需求 / 216
13.3 工匠Demo / 217
13.4 使用COLA / 218
13.4.1 安装COLA / 218
13.4.2 搭建应用 / 218
13.5 领域模型 / 219
13.5.1 领域建模 / 219
13.5.2 领域词汇表 / 221
13.6 核心业务逻辑 / 222
13.7 实现技术细节 / 227
13.7.1 数据存储 / 227
13.7.2 控制器 / 228
13.8 测试 / 229
13.8.1 单元测试 / 229
13.8.2 集成测试 / 230
13.8.3 回归测试 / 231
13.9 本章小结 / 232
• • • • • (收起)

[代码精进之路 从码农到工匠_下载链接1](#)

标签

计算机

编程

程序设计

架构

思维

方法论

互联网

管理

评论

老张的心得，也算是比较完整概括新人编程应该关注的地方，还有思考后给出自己的答案，诚意满满。

印象最深的一句话——要记住，留给公司一个方便维护，整洁优雅的代码库，是我们技术人员的最好技术使命，也是我们对公司做出的最大技术贡献。

不得不说,很有收获.看书时,字里行间一阵亲切感扑面而来,如同跟作者面对面交流,不少内容是看进心里去了.
我建议有几年经验的工程师阅读.边看边思考的话,通过本书可以在编程艺术和思维素养方面得到一定的总结提炼的反馈,了解到的新知识也可以为下一阶段的学习工作有个好的指引.

目录更有用

作者语言精炼流畅。除了第7,12,13章介绍具体细节以外，本书介绍的方法论和思想都很通用有用。技艺部分的内容与《编写可读代码的艺术》大致相同。
特别值得一提的是，8-11章总结的经验感悟，在做事为人方面都于我颇有启发。

“愿天下没有烂代码”，这本书以此为切入点，首先描述了一些编程技巧和方法论，然后从更高层次的思想指导，来统领方法论和技巧，特别是对于技术Leader的职责给出了切实有效的建议，最后则是结合了一些实践，包括他们自研的COLA架构和落地。全书之中，对我最有价值的部分莫过于技术Leader这个章节了。因为我本身也是突然变成了这样一个定位，挺迷茫自己究竟要做什么，而作者对于技术Leader和Manager的分析，让人恍然大悟，作者还辅以OKR, CodeReivew, 技术规划等等一系列切实有效的技术Leader工作，受益匪浅，待开年后心里就有谱了。并且这本书也可以作为一个技术目录使用，因为很多点其实作者只是浅尝辄止，并未有特别深入去说明，完全可以在后续进行

[代码精进之路 从码农到工匠 下载链接1](#)

书评

[代码精进之路 从码农到工匠 下载链接1](#)