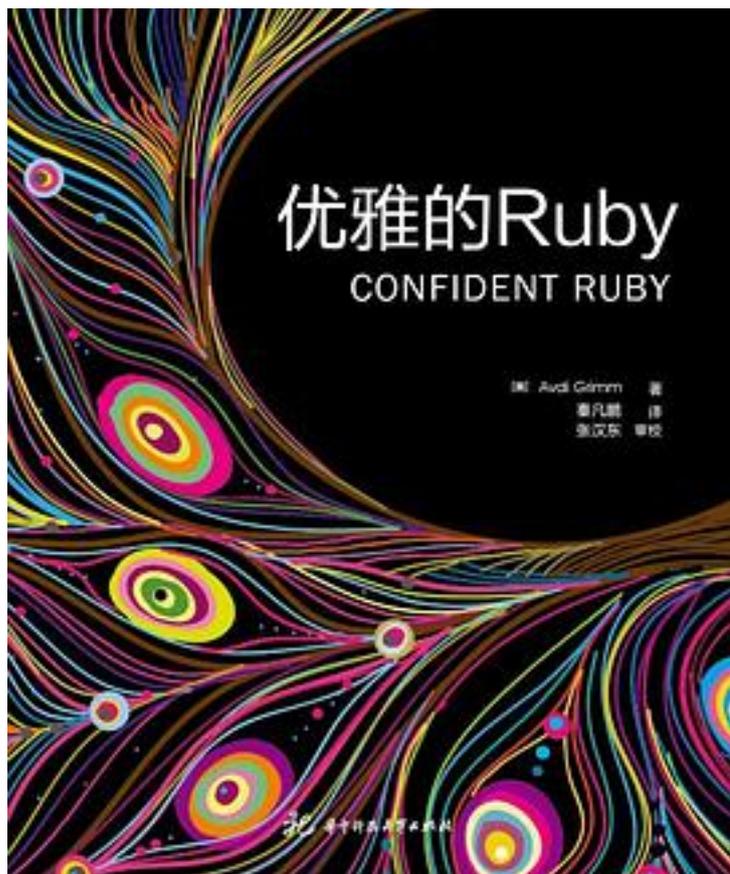


优雅的Ruby



[优雅的Ruby_下载链接1](#)

著者:Avdi Grimm

出版者:华中科技大学出版社

出版时间:2017-4-1

装帧:平装

isbn:9787568024891

《优雅的Ruby》总结了32条Ruby编程技巧，帮助读者写出清晰、优雅、稳定的Ruby代码。作者Avdi Grimm主张Ruby方法应该像故事一样易于阅读。他将Ruby方法分成输入处理（Collecting Input）、功能实现（Performing Work）、输出处理（Delivering Output）、失败处理（Handling

Failures) 四个部分，针对每个部分的特点归纳实用的编程模式，并配合丰富的实例讲解，让读者写出优雅实用的Ruby代码，找回Ruby编程的乐趣。

·
《优雅的Ruby》让复杂的代码变得容易编写了。

——Kevin Sjöberg

·
如果你想找回Ruby编程的乐趣，这是最棒的一本书。

——Matt Rogers

·
我写了30年代码，《优雅的Ruby》仍然让我受益匪浅。

——Jeff Dickey

·
全书可以分成六个部分。

首先讨论用消息和角色的思想来实现方法。

第2章讨论“实现功能”。虽然这看起来不符合前文提到的“方法组成顺序”，但是通过这一章的学习，你将学会思考如何设计方法，以便为后面的模式学习打下基础。

第3章到第5章是本书最核心的模式部分，每个模式又由五个部分组成：

1. 适用场景。就像药品包装上写有适用症状，这部分内容简要地介绍了模式的适用场景，比如用来解决特定问题，或者修正编写代码的不良习惯。
2. 摘要。当你尝试回忆某个模式，但又不记得名字时，摘要能够给你莫大的帮助。
3. 基本原理，阐述为何要用这个模式。
4. 示例。借助一两个具体的例子阐述选择该模式的原因及实现方法。
5. 小结。总结模式的优点、潜在的陷阱和不足。

根据我提出的组成方法的原则，这些模式被分为以下三大系列。

输入处理的模式。

输出处理的模式，让方法调用者优雅地调用方法。

异常处理模式，保障方法井然有序。

第6章将讨论一些实际的Ruby开源项目示例，并把本书中的模式应用到它们身上。

作者介绍:

Avdi

Grimm是ShipRise.com和RubyTapas.com的创始人，有着十几年Ruby编程经验，是Ruby程序界的领军人物。他目前与妻子居住在美国宾夕法尼亚南部。

目录: 第1章 引言 1

1.1 当Ruby遭遇现实 2

1.2 自信优雅的代码 2

1.3 好的故事，糟糕的讲述 3

1.4 像写故事一样写代码 4

1.5 方法的四个部分 4

1.6 本书组织结构 8

第2章 功能实现 11

2.1 发送有效的消息 12

2.2 导入交易记录 13

2.3 识别消息 14

2.4 识别角色 14

2.5 避免马盖先主义 17

2.6 让语言为系统服务 17

2.7 像鸭子一样叫 18

2.8 驯养鸭群 19

第3章 收集输入 21

3.1 输入处理概述 21

3.1.1 间接输入 23

3.1.2 从角色到对象 26

3.1.3 保护边界而非内部 27

3.2 使用内置的类型转换协议 28

3.2.1 适用场景 28

3.2.2 摘要 28

3.2.3 基本原理 28

3.2.4 示例：宣布获奖结果 28

3.2.5 示例：Emacs配置文件 30

3.2.6 标准类型转换方法列表 32

3.2.7 显式转换和隐式转换 33

3.2.8 明确提出参数要求 37

3.2.9 小结 39

3.3 有条件地使用类型转换方法 40

3.3.1 使用场景 40

3.3.2 摘要 40

3.3.3 基本原理 40

3.3.4 示例：打开文件 40

3.3.5 违反鸭子类型的唯一特例 42

3.3.6 小结 45

3.4 自定义类型转换协议 46

3.4.1 使用场景 46

3.4.2 摘要 46

3.4.3 基本原理 46

3.4.4 示例：接收一个点或一对坐标 46

3.4.5 小结 48

3.5 定义自定义类型的转换协议 49

- 3.5.1 使用场景 49
- 3.5.2 摘要 49
- 3.5.3 基本原理 49
- 3.5.4 示例：将英尺转换为米 49
- 3.5.5 小结 52
- 3.6 利用内置强制类型转换方法 53
 - 3.6.1 使用场景 53
 - 3.6.2 摘要 53
 - 3.6.3 基本原理 53
 - 3.6.4 示例：格式化打印数字 53
 - 3.6.5 Hash.[] 57
 - 3.6.6 小结 57
- 3.7 用Array()将输入数组化 58
 - 3.7.1 使用场景 58
 - 3.7.2 摘要 58
 - 3.7.3 基本原理 58
 - 3.7.4 示例：可变参数 58
 - 3.7.5 小结 60
- 3.8 自定义强制类型转换方法 61
 - 3.8.1 使用场景 61
 - 3.8.2 摘要 61
 - 3.8.3 基本原理 61
 - 3.8.4 示例：应用于2D图形中的强制类型转换方法 62
 - 3.8.5 关于module_function 63
 - 3.8.6 结合类型转换协议和强制类型转换方法 64
 - 3.8.7 用Lambdas表达式作case分支 66
 - 3.8.8 小结 67
- 3.9 用自定义类替换类字符串类型 68
 - 3.9.1 使用场景 68
 - 3.9.2 摘要 68
 - 3.9.3 基本原理 68
 - 3.9.4 示例：红绿灯的状态问题 69
 - 3.9.5 小结 77
- 3.10 用适配器装饰输入 78
 - 3.10.1 使用场景 78
 - 3.10.2 摘要 78
 - 3.10.3 基本原理 78
 - 3.10.4 示例：将日志写进IRC 78
 - 3.10.5 小结 82
- 3.11 利用透明适配器逐步消除类型依赖 83
 - 3.11.1 适用场景 83
 - 3.11.2 摘要 83
 - 3.11.3 基本原理 83
 - 3.11.4 示例：再探将日志写进IRC的示例 83
 - 3.11.5 小结 86
- 3.12 利用先决条件排除非法输入 87
 - 3.12.1 使用场景 87
 - 3.12.2 摘要 87
 - 3.12.3 基本原理 87
 - 3.12.4 示例：员工入职日期 87
 - 3.12.5 “可执行文档” 91
 - 3.12.6 小结 91
- 3.13 利用#fetch确保Hash键的存在性 92
 - 3.13.1 使用场景 92

- 3.13.2 摘要 92
- 3.13.3 基本原理 92
- 3.13.4 示例：useradd(8)包装器 92
- 3.13.5 尝试#fetch 95
- 3.13.6 自定义#fetch 98
- 3.13.7 小结 99
- 3.14 利用#fetch提供默认参数 100
- 3.14.1 使用场景 100
- 3.14.2 摘要 100
- 3.14.3 基本原理 100
- 3.14.4 示例：可选的logger参数 100
- 3.14.5 可重用的#fetch代码块 104
- 3.14.6 双参数#fetch 106
- 3.14.7 小结 107
- 3.15 用断言验证假设 108
- 3.15.1 使用场景 108
- 3.15.2 摘要 108
- 3.15.3 基本原理 108
- 3.15.4 示例：导入银行记录 108
- 3.15.5 小结 113
- 3.16 用卫语句来处理特殊场景 114
- 3.16.1 使用场景 114
- 3.16.2 摘要 114
- 3.16.3 基本原理 114
- 3.16.4 示例：“静音模式”标志 114
- 3.16.5 提前返回 116
- 3.16.6 小结 117
- 3.17 用对象表示特殊场景 118
- 3.17.1 使用场景 118
- 3.17.2 摘要 118
- 3.17.3 基本原理 118
- 3.17.4 示例：游客用户 118
- 3.17.5 用特例对象来表示当前用户 121
- 3.17.6 小步改进 126
- 3.17.7 保持特例对象和普通对象的同步 128
- 3.17.8 小结 129
- 3.18 用空对象表示不做事的情况 130
- 3.18.1 使用场景 130
- 3.18.2 摘要 130
- 3.18.3 基本原理 130
- 3.18.4 示例：输出日志到shell命令行 131
- 3.18.5 通用空对象 133
- 3.18.6 穿越事界 134
- 3.18.7 让空对象返回false 138
- 3.18.8 小结 140
- 3.19 用良性值替代nil 142
- 3.19.1 使用场景 142
- 3.19.2 摘要 142
- 3.19.3 基本原理 142
- 3.19.4 示例：显示会员地理位置信息 142
- 3.19.5 无害就好 145
- 3.19.6 小结 146
- 3.20 用symbols做占位符 147
- 3.20.1 使用场景 147

- 3.20.2 摘要 147
- 3.20.3 基本原理 147
- 3.20.4 示例：web service可选认证 147
- 3.20.5 都是nil惹的祸 149
- 3.20.6 带语义的占位符 152
- 3.20.7 小结 154
- 3.21 将参数封装到参数对象中 155
 - 3.21.1 使用场景 155
 - 3.21.2 摘要 155
 - 3.21.3 基本原理 155
 - 3.21.4 参数对象回顾 155
 - 3.21.5 添加可选参数 159
 - 3.21.6 小结 163
- 3.22 提取参数构建器 164
 - 3.22.1 使用场景 164
 - 3.22.2 摘要 164
 - 3.22.3 基本原理 164
 - 3.22.4 示例：方便的绘点API 164
 - 3.22.5 Net/HTTP vs. Faraday 168
 - 3.22.6 提取参数Builder 170
 - 3.22.7 小结 172
- 第4章 输出处理 173
 - 4.1 用全函数作为方法返回值 174
 - 4.1.1 使用场景 174
 - 4.1.2 摘要 174
 - 4.1.3 基本原理 174
 - 4.1.4 示例：单词搜索 174
 - 4.1.5 小结 178
 - 4.2 执行回调而非返回状态 179
 - 4.2.1 使用场景 179
 - 4.2.2 摘要 179
 - 4.2.3 基本原理 179
 - 4.2.4 示例 179
 - 4.2.5 小结 182
 - 4.3 用良性值表示失败 183
 - 4.3.1 使用场景 183
 - 4.3.2 摘要 183
 - 4.3.3 基本原理 183
 - 4.3.4 示例：在侧边栏上显示推文 183
 - 4.3.5 小结 185
 - 4.4 用特例对象表示失败 186
 - 4.4.1 使用场景 186
 - 4.4.2 摘要 186
 - 4.4.3 基本原理 186
 - 4.4.4 示例：游客用户 186
 - 4.4.5 小结 187
 - 4.5 返回状态对象 188
 - 4.5.1 使用场景 188
 - 4.5.2 摘要 188
 - 4.5.3 基本原理 188
 - 4.5.4 示例：记录导入结果 188
 - 4.5.5 小结 192
 - 4.6 将状态对象传给回调 193
 - 4.6.1 使用场景 193

- 4.6.2 摘要 193
- 4.6.3 基本原理 193
- 4.6.4 示例：将导入结果传给回调 193
- 4.6.5 测试状态对象 198
- 4.6.6 小结 199
- 4.7 用throw提前终止执行 200
 - 4.7.1 使用场景 200
 - 4.7.2 摘要 200
 - 4.7.3 示例：提前终止HTML文档解析 200
 - 4.7.4 小结 205
- 第5章 失败处理 207
 - 5.1 优先使用顶层异常捕获 208
 - 5.1.1 使用场景 208
 - 5.1.2 摘要 208
 - 5.1.3 基本原理 208
 - 5.1.4 示例 208
 - 5.1.5 小结 209
 - 5.2 用受检方法封装危险操作 210
 - 5.2.1 使用场景 210
 - 5.2.2 摘要 210
 - 5.2.3 基本原理 210
 - 5.2.4 示例 210
 - 5.2.5 演进成Adapters 212
 - 5.2.6 小结 212
 - 5.3 使用护卫方法 213
 - 5.3.1 使用场景 213
 - 5.3.2 摘要 213
 - 5.3.3 基本原理 213
 - 5.3.4 示例：子进程状态检测 213
 - 5.3.5 小结 216
- 第6章 为了优雅重构 217
 - 6.1 MetricFu 218
 - 6.1.1 Location 218
 - 6.1.2 HotspotAnalyzedProblems 222
 - 6.1.3 排名 225
 - 6.2 Stringer 227
- 第7章 后记 231
 - • • • • [\(收起\)](#)

[优雅的Ruby_下载链接1](#)

标签

Ruby

编程

程序设计

模式

重构

计算机

程度设计

动态语言

评论

内容的密度不高，很多都是平常写代码的时候用到过的，半天可看完。学到了一些之前没注意到的技巧，更重要的是作者这种总结重构模式的工作方法。得吐槽的是第三章就占了大半本书的篇幅，其它几章有点形同虚设

感觉被坑了，不值这个价啊

比较浅显，没读一些优秀的源码好

本书书名其实不叫《优雅的Ruby》，应该叫《编写健壮的Ruby方法》。全书的主旨就是在讲述在Ruby中如何编写方法：处理输入，处理输出，功能实现和异常处理。后两者几乎用了很短的篇幅略过。而占据了全书一半篇幅的处理输入在我看来基本上就是在为动态语言埋单。只能说也许Ruby是happy for programming? 但在真实世界的项目里绝对不是happy for engineering! 另外，这个定价真是宰大头

有所获

本书写的不错，翻译也可以。书的内容属于Ruby最佳实践，教你如何更加优雅地使用Ruby。但实际上这里「优雅」的意思更倾向于「健壮」。也就是说，教你如何构建更加健壮的Ruby项目，但是不会失去Ruby的优雅。

[优雅的Ruby_下载链接1](#)

书评

昨天拿到书了，确实好书，很实用。内容类似Ruby重构那本书，可读性更好些，讲了为什么。
重点讲了输入处理，目的是代码保护，包括方法保护和系统保护。减少nil的判断、类型检测、异常处理。代码分两层，目标是让上层代码简单清晰，复杂性转移到下层。
值得反复学习。值得...

[优雅的Ruby_下载链接1](#)