

Rails之道



[Rails之道_下载链接1](#)

著者:(美)Obie Fernandez

出版者:人民邮电出版社

出版时间:2010-4

装帧:

isbn:9787115220721

《Rails之道》按照Rails的各个子系统进行组织编排，分别介绍了Rails的环境、初始过程、配置和日志记录，Rails的分配器、控制器、页面生成和路由，REST、资源和Rails，ActiveRecord的基础、关联、验证和高级技巧，ActionView的模板、缓存和帮助器，Ajax、Prototype和Scriptaculous等JavaScript代码库和RJS，Session管理、用户登录和认证系统，XML和ActiveResource，后台处理和ActionMaile，测试和specs(包括RSpec on Rails和Selenium)，安装、管理、编写插件，Rails的生产部署、配置和Capistrano等内容。

《Rails之道》详细讨论了Rails的程序代码并通过分析Rails中的代码片段来深入解释它的功能，同时，《Rails之道》部分章节也摘录了一些API文档中的内容，使读者能够快速地找到对应的API文档、相关的示例代码以及深入的解析说明。

《Rails之道》是Rails的权威参考书，适合对Rails已经有一定了解的开发人员学习和使

用。

作者介绍:

Obiec Fernandez是一位广为人知的技术行业领袖和独立咨询师。a从20世纪80年代获得第一台Commodore VIC-20开始,1他就一直在从事各种黑客工作。a20世纪90年代中期,1他终于找到了自己的位置,1成为第一代Java企业项目的程序员。a他于1998年移居到美国乔治亚州亚特兰大市,1并作为当地新兴企业Media Ocean的首席架构师而闻名。a他还成立了Extremec Programming(后改名为Agile Atlanta)用户社团,1并在该社团担任了几年的主席和组织人。a2004年,1他又回归企业,1为世界著名的咨询公司Thoughtworks处理那些具有很大发展潜力的高风险项目。

193从2005年初开始,2他就通过博客和出版物推广Ruby和Rails,2并且在Java开源社区的老朋友之间获得到了一定认可(当然也有指责)。a从那时起,2他就定期参加一些业界活动和用户会议,2有时也会为想要参与Rails开发的公司或组织提供一定培训。

194目前,3Obie致力于开发和营销大规模的基于网路的应用程序。a他几乎每天都会更新自己深受欢迎的科技博客<http://obiefernandez.2com>,3就各种话题进行讨论。

目录: 第1章 Rails环境与配置 1

1.1 启动 1

1.1.1 默认环境设置 1

1.1.2 引导 2

1.1.3 RubyGems 3

1.1.4 初始化 4

1.1.5 默认加载路径 4

1.1.6 Rails模组及代码自动加载 5

1.1.7 内置的Rails信息 5

1.1.8 配置 6

1.1.9 附加配置 8

1.2 开发模式 8

1.2.1 类文件自动化重新加载 9

1.2.2 Rails类加载器 9

1.3 测试模式 10

1.4 生产模式 11

1.5 日志器 11

1.5.1 Rails日志文件 12

1.5.2 日志分析 13

1.5.3 Syslog 15

1.6 总结 15

第2章 运用控制器 16

2.1 调度器: 从这里开始 16

2.1.1 接收请求 17

2.1.2 和调度器亲密接触 17

2.2 渲染视图 18

2.2.1 何时开始渲染 19

2.2.2 指定渲染 19

2.2.3 渲染其他动作的模板 19

2.2.4 渲染一个完全不同的模板 20

2.2.5 渲染局部模板 20

2.2.6 渲染内联模板代码 21

2.2.7 渲染文本 21

- 2.2.8 渲染其他类型的数据结构 21
- 2.2.9 什么都不渲染 21
- 2.2.10 渲染的属性 22
- 2.3 重定向 23
- 2.4 控制器和视图之间的通信 25
- 2.5 过滤器 25
 - 2.5.1 过滤器继承 26
 - 2.5.2 过滤器的类型 27
 - 2.5.3 过滤器的队列的顺序 28
 - 2.5.4 Around过滤器 28
 - 2.5.5 跳过过滤器 29
 - 2.5.6 过滤器条件 29
 - 2.5.7 过滤器挂起 30
- 2.6 流 30
 - 2.6.1 send_data(data, options = {}) 30
 - 2.6.2 send_file(path, options = {}) 31
 - 2.6.3 让web服务器发送文件 33
- 2.7 小结 33
- 第3章 路由 34
 - 3.1 路由的两个目的 35
 - 3.2 绑定参数 35
 - 3.3 使用通配符(“接收器”) 36
 - 3.4 静态字符串 37
 - 3.5 route.rb文件 38
 - 3.5.1 默认的路由信息 39
 - 3.5.2 聚焦在:id字段 40
 - 3.5.3 默认的路由生成规则 40
 - 3.5.4 修改默认的路由信息 41
 - 3.6 默认路由信息之前的信息和respond_to 41
 - 3.7 空的路由信息 42
 - 3.8 编写自定义路由规则 43
 - 3.9 使用静态字符串 43
 - 3.10 使用你自己的“接收器” 44
 - 3.11 关于路由次序的说明 45
 - 3.12 在路由信息中使用正则表达式 45
 - 3.13 默认参数和url_for方法 46
 - 3.14 使用文字化的URL 47
 - 3.15 路由中的通配字段 47
 - 3.16 通配符的键—值对 48
 - 3.17 具名路由 48
 - 3.17.1 创建具名路由 49
 - 3.17.2 比较name_path和name_url的使用 49
 - 3.17.3 请考虑 49
 - 3.18 如何命名你的路由 50
 - 3.18.1 参数糖衣 50
 - 3.18.2 更多糖衣 51
 - 3.19 特殊的范围方法with_options 51
 - 3.20 小结 52
- 第4章 REST, 资源和Rails 53
 - 4.1 REST简介 53
 - 4.2 Rails的REST 54
 - 4.3 路由选择和CRUD 54
 - 4.4 资源和表现 55
 - 4.4.1 REST资源与Rails 55

- 4.4.2 从具名路由到REST支持 55
- 4.4.3 重新认识HTTP方法 56
- 4.5 标准的REST化的控制器动作 57
 - 4.5.1 模拟PUT和DELETE操作 58
 - 4.5.2 REST化的资源的单数和复数 58
 - 4.5.3 特殊的拍档: new/create和edit/update 58
- 4.6 单数的资源路由 59
- 4.7 嵌套资源 59
 - 4.7.1 显式地设置:path_prefix 60
 - 4.7.2 显式地设置:name_prefix 61
 - 4.7.3 显式地设置REST化的控制器 61
 - 4.7.4 使用所有选项 62
 - 4.7.5 思考嵌套路由 63
 - 4.7.6 嵌套过深 63
- 4.8 自定义REST化的路由 64
 - 4.8.1 添加成员路由 65
 - 4.8.2 添加集合路由 65
 - 4.8.3 思考 65
- 4.9 仅有控制器的资源 67
- 4.10 资源的不同展现形式 68
 - 4.10.1 respond_to方法 68
 - 4.10.2 格式化具名路由 68
- 4.11 REST化的Rails动作集合 69
 - 4.11.1 Index 69
 - 4.11.2 Show 71
 - 4.11.3 Destroy 71
 - 4.11.4 New和Create 72
 - 4.11.5 Edit和Update 73
- 4.12 小结 74
- 第5章 探究路由选择 75
 - 5.1 在应用程序控制台检查路由 75
 - 5.1.1 转存路由信息 75
 - 5.1.2 剖析Route对象 76
 - 5.1.3 在控制台识别和生成路由 78
 - 5.1.4 控制台的具名路由 79
 - 5.2 测试路由 80
 - 5.3 Routing Navigator插件 80
 - 5.4 小结 81
- 第6章 运用ActiveRecord 82
 - 6.1 基础知识 82
 - 6.2 数据迁移 84
 - 6.2.1 创建迁移 84
 - 6.2.2 用于迁移的API 87
 - 6.2.3 定义列 88
 - 6.3 宏样式方法 92
 - 6.3.1 关系声明 93
 - 6.3.2 约定优于配置 93
 - 6.3.3 复数化 94
 - 6.3.4 手动设置名字 95
 - 6.3.5 遗留的命名约定 95
 - 6.4 定义属性 96
 - 6.4.1 默认属性值 96
 - 6.4.2 序列化属性 98
 - 6.5 CRUD: 创建、读取、更新和删除 98

- 6.5.1 创建新的ActiveRecord实例 98
- 6.5.2 读取ActiveRecord对象 99
- 6.5.3 读写属性 100
- 6.5.4 在类型转换之前访问并且操作属性 102
- 6.5.5 重新载入 102
- 6.5.6 基于属性的动态查找器 102
- 6.5.7 自定义SQL查询 103
- 6.5.8 查询缓存 104
- 6.5.9 更新 106
- 6.5.10 基于条件的更新 107
- 6.5.11 更新特定实例 107
- 6.5.12 更新特定属性 107
- 6.5.13 便利的更新器 108
- 6.5.14 控制对属性的访问 108
- 6.5.15 删除和销毁 109
- 6.6 数据库锁定 109
 - 6.6.1 乐观锁定 110
 - 6.6.2 悲观锁定 111
 - 6.6.3 需要考虑的问题 112
- 6.7 高级查找 112
 - 6.7.1 条件 112
 - 6.7.2 查询结果集的排序 114
 - 6.7.3 限制和偏移量 114
 - 6.7.4 select选项 115
 - 6.7.5 from选项 115
 - 6.7.6 group by选项 115
 - 6.7.7 locking选项 116
 - 6.7.8 连接和包含关联 116
 - 6.7.9 read only 116
- 6.8 在不同的数据模型中连接多个数据库 116
- 6.9 直接使用数据库连接 117
 - 6.9.1 DatabaseStatements模组 118
 - 6.9.2 其他连接方法 119
- 6.10 其他配置选项 120
- 6.11 小结 121
- 第7章 ActiveRecord关联 122
 - 7.1 关联的继承关系 122
 - 7.2 一对多关联 123
 - 7.2.1 向集合添加已经关联的对象 124
 - 7.2.2 AssociationCollection的方法 125
 - 7.3 belongs_to关联 127
 - 7.3.1 重新载入关联 127
 - 7.3.2 通过关联来构建(build)或创建(create)相关对象 128
 - 7.3.3 belongs_to的选项 128
 - 7.4 has_many关联 132
 - 7.4.1 has_many的选项 133
 - 7.4.2 代理方法 137
 - 7.5 多对多关系 138
 - 7.5.1 has_and_belongs_to_many 138
 - 7.5.2 has_many:through 143
 - 7.5.3 has_many:through的选项 146
 - 7.6 一对一关系 148
 - 7.6.1 has_one 148
 - 7.6.2 has_one的选项 150

- 7.7 未保存的对象和关联 151
 - 7.7.1 一对一关联 151
 - 7.7.2 集合 152
- 7.8 关联的扩展 152
- 7.9 AssociationProxy类 153
- 7.10 小结 154
- 第8章 ActiveRecord验证 155
 - 8.1 查找错误 155
 - 8.2 简单的验证声明 155
 - 8.2.1 validates_acceptance_of 156
 - 8.2.2 validates_associated 156
 - 8.2.3 validates_confirmation_of 156
 - 8.2.4 validates_each 157
 - 8.2.5 validates_inclusion_of和validates_exclusion_of 157
 - 8.2.6 validates_existence_of 158
 - 8.2.7 validates_format_of 158
 - 8.2.8 validates_length_of 159
 - 8.2.9 validates_numericality_of 159
 - 8.2.10 validates_presence_of 159
 - 8.2.11 validates_uniqueness_of 160
 - 8.2.12 RecordInvalid 161
 - 8.3 通用验证选项 161
 - 8.3.1 :allow_nil 161
 - 8.3.2 :if 161
 - 8.3.3 :message 161
 - 8.3.4 :on 161
 - 8.4 条件认证 162
 - 8.5 使用Errors对象 163
 - 8.5.1 操作Errors集合 163
 - 8.5.2 Errors的检查 163
 - 8.6 自定义验证 164
 - 8.7 跳过验证 164
 - 8.8 小结 165
- 第9章 ActiveRecord的高级技巧 166
 - 9.1 回调方法 166
 - 9.1.1 注册回调方法 167
 - 9.1.2 匹配before/after回调方法 167
 - 9.1.3 中断执行 168
 - 9.1.4 使用回调方法 168
 - 9.1.5 特殊的回调方法：after_initialize和after_find 170
 - 9.1.6 回调方法类 171
 - 9.2 观察器 173
 - 9.2.1 命名规则 173
 - 9.2.2 注册观察器 174
 - 9.2.3 时机 174
 - 9.3 单表继承(single-table inheritance, STI) 174
 - 9.3.1 将继承映射到数据库 176
 - 9.3.2 STI值得注意的几点 177
 - 9.3.3 STI和数据模型关联 177
 - 9.4 抽象数据类型的基类 179
 - 9.5 多态的has_many关系 180
 - 9.5.1 假如数据模型带有注释功能 180
 - 9.5.2 使用has_many的注意事项 182
 - 9.6 用以重用通用行为的模块 182

- 9.6.1 回顾类的作用范围和环境 184
- 9.6.2 Include回调方法 185
- 9.7 在运行时修改ActiveRecord 186
 - 9.7.1 应用时的注意事项 187
 - 9.7.2 Ruby和域指定语言 187
- 9.8 小结 188
- 第10章 ActionView 189
 - 10.1 ERb基础 189
 - 10.1.1 ERb实践 189
 - 10.1.2 整理ERb输出 191
 - 10.1.3 ERb分隔符中的注释 191
 - 10.1.4 条件输出 191
 - 10.1.5 RHTML? RXML? RJS? 191
 - 10.2 布局和模板 192
 - 10.2.1 使用Yield生成内容 193
 - 10.2.2 模板变量 194
 - 10.2.3 针对用户提交的数据保护你的视图 196
 - 10.3 局部模板 197
 - 10.3.1 简单的使用示例 197
 - 10.3.2 重用局部模板 198
 - 10.3.3 共享局部模板 198
 - 10.3.4 给局部模板传递变量 199
 - 10.3.5 渲染数据集 200
 - 10.3.6 日志 201
 - 10.4 缓存 201
 - 10.4.1 在开发模式下的缓存 201
 - 10.4.2 页面缓存 202
 - 10.4.3 动作缓存 202
 - 10.4.4 片段缓存 203
 - 10.4.5 缓存内容的期限 205
 - 10.4.6 使用Sweeper自动过期缓存 206
 - 10.4.7 缓存日志 207
 - 10.4.8 Action Cache插件 208
 - 10.4.9 缓存的存储 208
 - 10.5 小结 209
- 第11章 辅助方法 210
 - 11.1 ActiveRecordHelper 210
 - 11.1.1 报告验证的错误 210
 - 11.1.2 自动创建表单 212
 - 11.1.3 自定义验证错误的高亮方式 213
 - 11.2 AssetTagHelper 214
 - 11.2.1 Head的辅助方法 214
 - 11.2.2 针对插件的辅助方法以设定默认的JavaScript文件 217
 - 11.3 BenchmarkHelper 217
 - 11.4 CacheHelper 217
 - 11.5 CaptureHelper 218
 - 11.6 DateHelper 218
 - 11.6.1 日期时间选择 218
 - 11.6.2 单个日期和时间选择辅助 219
 - 11.6.3 日期选择辅助方法的通用选项 220
 - 11.6.4 名字复杂的distance_in_time方法 221
 - 11.7 DebugHelper 222
 - 11.8 FormHelper 222
 - 11.8.1 通过表单创建Active Record模型 222

- 11.8.2 表单辅助方法是怎样取值的? 227
- 11.9 FormOptionsHelpers 228
 - 11.9.1 Select标签的辅助器 228
 - 11.9.2 Option标签的辅助器 229
- 11.10 FormTagHelper 232
- 11.11 JavaScriptHelper 234
- 11.12 NumberHelper 235
- 11.13 PaginationHelper 236
 - 11.13.1 will_paginate 236
 - 11.13.2 paginator 237
 - 11.13.3 Paginating Find 237
- 11.14 RecordIdentification 237
- 11.15 RecordTagHelper 239
- 11.16 TagHelper 239
- 11.17 TextHelper 240
- 11.18 UrlHelper 245
- 11.19 编写你自己的辅助方法 249
 - 11.19.1 小小的优化: Title辅助方法 249
 - 11.19.2 封装视图的逻辑: photo_for辅助方法 250
 - 11.19.3 智能视图: breadcrumbs辅助方法 250
- 11.20 包装并生成局部视图模板 251
 - 11.20.1 一个tiles辅助器 251
 - 11.20.2 生成局部模板 253
- 11.21 小结 255
- 第12章 Ajax on Rails 256
 - 12.1 Prototype 257
 - 12.1.1 FireBug 257
 - 12.1.2 Prototype API 257
 - 12.1.3 顶层函数 258
 - 12.1.4 类 259
 - 12.1.5 JavaScript的对象类的扩展 259
 - 12.1.6 扩展JavaScript的Array类 260
 - 12.1.7 扩展document对象 261
 - 12.1.8 扩展Event类 261
 - 12.1.9 扩展JavaScript的Function类 263
 - 12.1.10 扩展JavaScript的Number类 263
 - 12.1.11 扩展JavaScript String类 264
 - 12.1.12 Ajax对象 266
 - 12.1.13 Ajax.Responders 266
 - 12.1.14 Enumerable 267
 - 12.1.15 Hash 270
 - 12.1.16 ObjectRange 271
 - 12.1.17 Prototype 对象 271
 - 12.2 PrototypeHelper模组 271
 - 12.2.1 link_to_remote 271
 - 12.2.2 remote_form_for 274
 - 12.2.3 periodically_call_remote 275
 - 12.2.4 observe_field 276
 - 12.2.5 observe_form 277
 - 12.3 RJS——在Ruby中编写 Javascript 277
 - 12.3.1 RJS模板 278
 - 12.3.2 (javascript) 279
 - 12.3.3 [](id) 279
 - 12.3.4 alert(message) 279

- 12.3.5 call(function, *arguments, &block) 279
- 12.3.6 delay(seconds = 1){ ... } 280
- 12.3.7 draggable(id, options = {}) 280
- 12.3.8 drop_receiving(id, options = {}) 280
- 12.3.9 hide(*ids) 280
- 12.3.10 insert_html(position, id, *options_for_render) 280
- 12.3.11 literal(code) 281
- 12.3.12 redirect_to(location) 281
- 12.3.13 remove(*ids) 281
- 12.3.14 replace(id, *options_for_render) 281
- 12.3.15 replace_html(id, *options_for_render) 281
- 12.3.16 select(pattern) 281
- 12.3.17 show(*ids) 281
- 12.3.18 sortable(id, options = {}) 282
- 12.3.19 toggle(*ids) 282
- 12.3.20 visual_effect(name, id = nil, options = {}) 282
- 12.4 JSON 282
- 12.5 Drag and Drop 283
- 12.6 Sortable 285
- 12.7 自动完成 285
- 12.8 可切换的编辑输入框 286
- 12.9 小结 287
- 第13章 Session管理 288
 - 13.1 该把什么放在session中 288
 - 13.1.1 当前用户 288
 - 13.1.2 Session使用规则 289
 - 13.2 Session属性 289
 - 13.2.1 针对机器人关闭session功能 289
 - 13.2.2 选择性开启session 290
 - 13.2.3 安全的session 290
 - 13.3 存储机制 290
 - 13.3.1 ActiveRecord存储机制 291
 - 13.3.2 PStore(基于文件的方式) 291
 - 13.3.3 DRb Session存储机制 291
 - 13.3.4 memcach存储机制 292
 - 13.3.5 关于CookieStore的争议 292
 - 13.4 超时机制和session生命周期 293
 - 13.4.1 Session超时插件 294
 - 13.4.2 跟踪活跃的session 294
 - 13.4.3 增强安全性 295
 - 13.4.4 清除陈旧的session 295
 - 13.5 Cookies 295
 - 13.6 小结 296
- 第14章 登录与认证 297
 - 14.1 Acts as Authenticated 297
 - 14.1.1 安装与设置 298
 - 14.1.2 User数据模型 298
 - 14.1.3 Account控制器 304
 - 14.1.4 从Cookie中登录 306
 - 14.1.5 当前用户 307
 - 14.2 在登录期间测试 308
 - 14.3 小结 309
- 第15章 XML和ActiveResource 310
 - 15.1 to_xml方法 310

- 15.1.1 定制to_xml输出 311
- 15.1.2 to_xml和关联性 312
- 15.1.3 高级to_xml 313
- 15.1.4 运行时的动态属性 314
- 15.1.5 重载to_xml 315
- 15.1.6 由数组的to_xml方法所学到的 315
- 15.2 XML Builder 316
- 15.3 解析XML 318
 - 15.3.1 将XML转换成散列 318
 - 15.3.2 XmlSimple 319
 - 15.3.3 类型转换 320
- 15.4 ActiveResource 320
 - 15.4.1 Find 321
 - 15.4.2 Create 322
 - 15.4.3 Update 324
 - 15.4.4 Delete 324
 - 15.4.5 HTTP头信息 325
 - 15.4.6 自定义 326
 - 15.4.7 散列形式 327
- 15.5 小结 327
- 第16章 ActionMailer 328
 - 16.1 安装 328
 - 16.2 Mailer模型 328
 - 16.2.1 准备要发出的邮件消息 329
 - 16.2.2 HTML格式的邮件信息 331
 - 16.2.3 复合格式的信息 331
 - 16.2.4 文件附件 332
 - 16.2.5 发送邮件 333
 - 16.3 接收邮件 333
 - 16.3.1 TMail::Mail API参考 334
 - 16.3.2 处理附件 334
 - 16.4 配置 335
 - 16.5 小结 335
- 第17章 测试 336
 - 17.1 Rails中测试的专用术语 337
 - 17.1.1 关于独立性… 337
 - 17.1.2 Rails的数据模拟 337
 - 17.1.3 真正的数据模拟和占位代码 338
 - 17.1.4 整合测试 339
 - 17.1.5 避免概念混淆 339
 - 17.2 Test::Unit 340
 - 17.3 数据装置 341
 - 17.3.1 CSV格式的数据装置 342
 - 17.3.2 在测试中访问数据装置中的记录 342
 - 17.3.3 动态的数据装置的数据 343
 - 17.3.4 在开发模式下使用数据装置中的数据 344
 - 17.3.5 从开发数据中生成数据装置 344
 - 17.3.6 数据装置的选项 345
 - 17.3.7 大家都不喜欢数据装置 345
 - 17.3.8 数据装置其实没那么糟糕 346
 - 17.4 断言 347
 - 17.4.1 基本的断言 347
 - 17.4.2 Rails的断言 349
 - 17.4.3 一个断言配一个测试方法 349

- 17.5 使用单元测试来测试数据模型 350
 - 17.5.1 数据模型测试基础 350
 - 17.5.2 决定测试什么 352
- 17.6 使用功能测试来测试控制器 352
 - 17.6.1 结构和setup 352
 - 17.6.2 功能测试方法 353
 - 17.6.3 通用断言 353
- 17.7 使用功能测试来测试视图 356
 - 17.7.1 测试RJS的行为 359
 - 17.7.2 其他选择方法 359
 - 17.7.3 测试路由规则 359
- 17.8 Rails整合测试 360
 - 17.8.1 基础 361
 - 17.8.2 整合测试的API 361
 - 17.8.3 使用session 362
- 17.9 和测试相关的Rake任务 362
- 17.10 验收测试 362
- 17.11 Selenium 363
 - 17.11.1 Selenium的基本概念 363
 - 17.11.2 开始使用Selenium 364
 - 17.11.3 RSelenese 365
- 17.12 小结 366
- 第18章 RSpec on Rails 367
 - 18.1 介绍RSpec 367
 - 18.1.1 Should和预期情况 368
 - 18.1.2 结果预测 369
 - 18.1.3 自定义预期情况匹配器 369
 - 18.1.4 包含多个例子的行为 371
 - 18.1.5 共享的行为 372
 - 18.1.6 RSpec的数据模拟和占位代码 374
 - 18.1.7 运行spec 376
 - 18.1.8 安装RSpec和RSpec on Rails插件 378
 - 18.2 RSpec on Rails插件 378
 - 18.2.1 代码生成器 378
 - 18.2.2 数据模型spec 378
 - 18.2.3 控制器的spec 380
 - 18.2.4 视图的spec 383
 - 18.2.5 辅助器的spec 384
 - 18.2.6 Scaffolding 385
 - 18.3 RSpec工具 385
 - 18.3.1 Autotest 385
 - 18.3.2 RCov 385
 - 18.4 小结 386
- 第19章 用插件扩展Rails 387
 - 19.1 管理插件 387
 - 19.1.1 重用代码 387
 - 19.1.2 插件脚本 388
 - 19.1.3 Subversion和script/plugin 391
 - 19.2 使用Piston 392
 - 19.2.1 安装 392
 - 19.2.2 导入Vendor库 393
 - 19.2.3 转换现有的Vendor库 393
 - 19.2.4 更新 394
 - 19.2.5 锁定和解锁版本 394

- 19.2.6 Piston属性 394
- 19.3 编写自己的插件 394
 - 19.3.1 init.rb钩子 395
 - 19.3.2 lib目录 396
 - 19.3.3 扩展Rails类 396
 - 19.3.4 README和MIT-LICENSE文件 397
 - 19.3.5 install.rb和uninstall.rb文件 398
 - 19.3.6 自定义Rake任务 399
 - 19.3.7 插件的Rakefile 399
 - 19.3.8 测试插件 400
- 19.4 小结 400
- 第20章 Rails生产环境配置 401
 - 20.1 生产环境Rails的简史 401
 - 20.2 一些基本的先决条件 402
 - 20.3 软件集清单 403
 - 20.3.1 服务器和网络环境 403
 - 20.3.2 Web层 404
 - 20.3.3 应用程序层 404
 - 20.3.4 数据库层 404
 - 20.3.5 监控 405
 - 20.3.6 版本控制 405
 - 20.4 安装 405
 - 20.4.1 Ruby 405
 - 20.4.2 RubyGems 405
 - 20.4.3 Rails 406
 - 20.4.4 Mongrel 406
 - 20.4.5 Mongrel Cluster 406
 - 20.4.6 Nginx 406
 - 20.4.7 Subversion 407
 - 20.4.8 MySQL 407
 - 20.4.9 Monit 407
 - 20.4.10 Capistrano 407
 - 20.5 配置 408
 - 20.5.1 配置Mongrel Cluster 408
 - 20.5.2 配置Nginx 408
 - 20.5.3 配置Monit 412
 - 20.5.4 配置Capistrano 414
 - 20.6 配置init脚本 414
 - 20.6.1 Nginx init脚本 414
 - 20.6.2 Mongrel init脚本 415
 - 20.6.3 Monit配置 416
 - 20.7 部署和发布 418
 - 20.8 有关生产环境软件集的其他注意点 418
 - 20.8.1 冗余和失效转移 418
 - 20.8.2 缓存 418
 - 20.8.3 性能和可扩展性 418
 - 20.8.4 安全 419
 - 20.8.5 可维护性 419
 - 20.9 结论 419
- 第21章 Capistrano 421
 - 21.1 Capistrano 概述 421
 - 21.1.1 术语 421
 - 21.1.2 基础知识 422
 - 21.1.3 Capistrano做了什么，没做什么 423

- 21.2 入门 423
 - 21.2.1 安装 423
 - 21.2.2 “Capify” 你的Rails应用程序 423
 - 21.2.3 配置部署 425
 - 21.2.4 一些茧合脚本 425
 - 21.2.5 设置部署目标服务器 426
 - 21.2.6 部署! 427
 - 21.3 重载Capistrano的默认假定 427
 - 21.3.1 使用远程用户账号 428
 - 21.3.2 定制Capistrano使用的源代码管理系统(SCM) 428
 - 21.3.3 部署目标服务器不能访问源代码管理系统(SCM) 428
 - 21.3.4 如果我不把database.yml放在源代码库中 428
 - 21.3.5 如果我的数据迁移不能从0运行到100 430
 - 21.4 实用的Capistrano策略 431
 - 21.4.1 变量和它们的有效范围 431
 - 21.4.2 练习#1: Staging 432
 - 21.4.3 练习#2: 管理其他服务 434
 - 21.5 多服务器部署 435
 - 21.6 事务处理 436
 - 21.7 代理访问部署目标服务器 437
 - 21.8 小结 438
- 第22章 后台进程 439
 - 22.1 script/runner 439
 - 22.1.1 入门 440
 - 22.1.2 用法说明 440
 - 22.1.3 script/runner的考虑 441
 - 22.2 DRb 441
 - 22.2.1 一个简单的DRb服务 441
 - 22.2.2 在Rails中使用DRb 442
 - 22.2.3 DRb的考虑 442
 - 22.2.4 资源 442
 - 22.3 BackgroundDRb 443
 - 22.3.1 入门 443
 - 22.3.2 配置 443
 - 22.3.3 理解BackgroundDRb 444
 - 22.3.4 使用中间人 444
 - 22.3.5 警告 445
 - 22.3.6 BackgroundDRb的考虑 446
 - 22.4 Daemons 446
 - 22.4.1 用法 446
 - 22.4.2 线程介绍 447
 - 22.4.3 Daemon的考虑 448
 - 22.5 小结 449
- 附录A ActiveSupport API参考 450
- 附录B Rails概要 498
 - • • • • ([收起](#))

[Rails之道_下载链接1](#)

标签

rails

ruby

Web开发

编程

Programming

软件开发

Ruby

ROR

评论

绝对的RAILS开发必备案头书。

rails版本有点旧

翻译的不是很好，建议看英文版

期初刚开始学的时候还是很有用的，但是到了实践的是候为题就来了，毕竟是10年前出的，14年Rails都4.0了，很多特性都发生了变化，照搬的话会出现很多情况..

强烈推荐

版本有点老

[Rails之道 下载链接1](#)

书评

非常艰难的学习着rails。
整本书并不适合入门者的阅读，如果你是一个突然想做网站心血来潮的人，你可以尝试
ror 但是建议先学好你的英文阅读能力。然后看看下面这篇文章
<http://readful.com/post/12322300571/0-ruby-on-rails> 《rails不是新手的玩具》
然后你确定不是文中那...

The Rails Way这本书的质量和AWDWR(Agile.Web.Development.with.Rails)还有The
Ruby way质量一样好。从内容上来看， Rails
way和AWDWR都是讲rails，它们的内容有很多也是重复，但是Rails
way更像一本工具书，对于rails的每个方面都做了一定程度的说明。另外，在AWDWR
针对的是Rai...

我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了
我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了
我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了
我看过了 我看...

[Rails之道 下载链接1](#)