

《Ruby原理剖析》

图书基本信息

书名：《Ruby原理剖析》

13位ISBN编号：9787568022625

出版时间：2016-12-1

作者：[美] Patrick Shaughnessy

页数：380

译者：张汉东,秦凡鹏

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu111.com

《Ruby原理剖析》

内容概要

《Ruby原理剖析》解开Ruby编程语言的魔法面纱。全书图文并茂、深入浅出地剖析了Ruby编程语言的核心工作原理。作者本着科学实证的精神，设计了一系列实验，帮助读者轻松了解这门编程语言的工作奥秘，包括Ruby如何用虚拟机执行代码，Ruby的垃圾回收算法，以及类和模块在Ruby内部的关系等。

《Ruby原理剖析》

作者简介

Patrick Shaughnessy是著名的Ruby开发者，目前在麦肯锡管理咨询公司（McKinsey & Co.）从事开发工作。Patrick有20多年软件开发工作经验，精通C、Java、PHP、Ruby等多种编程语言。他是Ruby Conference Circuit的主持人，定期在Ruby Weekly电子报、Ruby5 Podcast和The Ruby Show上发表文章和演讲。他的博客地址：<http://patshaughnessy.net>

书籍目录

- 1 分词与语法解析 3
 - 1.1 词条：构成Ruby语言的单词 5
 - 1.2 语法解析：Ruby如何理解代码 13
 - 1.2.1 理解LALR解析算法 14
 - 1.2.2 真实的Ruby语法规则 21
 - 1.3 总结 31
- 2 编译 33
 - 2.1 Ruby 1.8没有编译器 34
 - 2.2 Ruby 1.9和Ruby 2.0引入了编译器 35
 - 2.3 Ruby如何编译简单脚本 37
 - 2.4 编译块调用 41
 - 2.5 本地表 49
 - 2.5.1 编译可选参数 52
 - 2.5.2 编译关键字参数 53
 - 2.6 总结 57
- 3 Ruby如何执行代码 59
 - 3.1 YARV内部栈和Ruby调用栈 60
 - 3.1.1 逐句查看Ruby如何执行简单脚本 62
 - 3.1.2 执行块调用 65
 - 3.2 访问Ruby变量的两种方式 72
 - 3.2.1 本地变量访问 72
 - 3.2.2 方法参数被看成本地变量 75
 - 3.2.3 动态变量访问 76
 - 3.3 总结 86
- 4 控制结构与方法调度 89
 - 4.1 Ruby如何执行if语句 90
 - 4.2 作用域之间的跳转 93
 - 4.2.1 捕获表 94
 - 4.2.2 捕获表的其他用途 96
 - 4.3 send指令：Ruby最复杂的控制结构 99
 - 4.3.1 方法查找和方法调度 99
 - 4.3.2 Ruby方法的11种类型 100
 - 4.4 调用普通Ruby方法 102
 - 4.4.1 为普通Ruby方法准备参数 103
 - 4.5 调用内建的Ruby方法 104
 - 4.5.1 调用attr_reader和attr_writer 105
 - 4.5.2 方法调度优化attr_reader和attr_writer 106
 - 4.6 总结 110
- 5 对象与类 113
 - 5.1 Ruby对象内部 114
 - 5.1.1 检验klass和ivptr 115
 - 5.1.2 观察同一个类的两个实例 117
 - 5.1.3 基本类型对象 118
 - 5.1.4 简单立即值完全不需要结构体 119
 - 5.1.5 基本类型对象有实例变量吗 120
 - 5.1.6 基本类型对象的实例变量保存在哪里 122
 - 5.2 RClass结构体内部有什么 125

- 5.2.1 继承 128
- 5.2.2 类实例变量vs类变量 129
- 5.2.3 存取类变量 131
- 5.2.4 常量 134
- 5.2.5 真实的RClass结构体 135
- 5.3 总结 140
- 6 方法查找和常量查找 143
- 6.1 Ruby如何实现模块 145
- 6.1.1 模块是类 145
- 6.1.2 将模块include到类中 147
- 6.2 Ruby的方法查找算法 148
- 6.2.1 方法查找示例 149
- 6.2.2 方法查找算法实践 151
- 6.2.3 Ruby中的多继承 152
- 6.2.4 全局方法缓存 153
- 6.2.5 内联方法缓存 154
- 6.2.6 清空Ruby的方法缓存 155
- 6.2.7 在同一个类中include两个模块 155
- 6.2.8 在模块中include模块 157
- 6.2.9 Module#prepend 示例 158
- 6.2.10 Ruby如何实现Module#prepend 161
- 6.2.11 在已被include的模块中增加方法 164
- 6.2.12 在已被include的模块中include其他模块 164
- 6.2.13 “被include的类”与原始模块共享方法表 166
- 6.3 常量查找 168
- 6.3.1 在超类中查找常量 169
- 6.3.2 Ruby如何在父级命名空间中查找常量 170
- 6.4 Ruby中的词法作用域 171
- 6.4.1 为新类或模块创建常量 172
- 6.4.2 在父命名空间中使用词法作用域查找常量 173
- 6.4.3 Ruby的常量查找算法 175
- 6.4.4 Ruby真实的常量查找算法 177
- 6.5 总结 178
- 7 散列表：Ruby内部的主力军 181
- 7.1 Ruby中的散列表 182
- 7.1.1 在散列表中保存值 183
- 7.1.2 从散列表中检索值 185
- 7.2 散列表如何扩展以容纳更多的值 188
- 7.2.1 散列冲突 188
- 7.2.2 重新散列条目 189
- 7.3 Ruby如何实现散列函数 195
- 7.3.1 Ruby 2.0中的散列优化 202
- 7.4 总结 203
- 8 Ruby如何借鉴Lisp几十年前的理念 207
- 8.1 块：Ruby中的闭包 208
- 8.1.1 Ruby如何调用块 210
- 8.1.2 借用1975年的理念 212
- 8.2 Lambda和Proc：把函数当做一等公民 219
- 8.2.1 栈内存vs堆内存 220

- 8.2.2 深入探索Ruby如何保存字符串的值 220
- 8.2.3 Ruby如何创建Lambda 223
- 8.2.4 Ruby如何调用Lambda 226
- 8.2.5 Proc对象 227
- 8.2.6 在同一个作用域中多次调用lambda 232
- 8.3 总结 234
- 9 元编程 237
 - 9.1 定义方法的多种方式 239
 - 9.1.1 Ruby的普通方法定义过程 239
 - 9.1.2 使用对象前缀定义类方法 241
 - 9.1.3 使用新的词法作用域定义类方法 242
 - 9.1.4 使用单类定义方法 244
 - 9.1.5 在单类的词法作用域中定义方法 245
 - 9.1.6 创建Refinement 246
 - 9.1.7 使用Refinement 248
 - 9.1.8 顶级作用域中的self 250
 - 9.1.9 类作用域中的self 251
 - 9.1.10 元类作用域中的self 252
 - 9.1.11 类方法中的self 253
 - 9.2 元编程与闭包：eval、instance_eval和binding 255
 - 9.2.1 能写代码的代码 255
 - 9.2.2 使用binding参数调用eval 257
 - 9.2.3 instance_eval示例 259
 - 9.2.4 Ruby闭包的另一个重点 260
 - 9.2.5 instance_eval改变接收者的self 262
 - 9.2.6 instance_eval为新的词法作用域创建单类 262
 - 9.2.7 使用define_method 266
 - 9.2.8 充当闭包的方法 266
 - 9.3 总结 268
- 10 JRuby：基于JVM的Ruby 271
 - 10.1 使用MRI和JRuby运行程序 272
 - 10.1.1 JRuby如何解析和编译代码 274
 - 10.1.2 JRuby如何执行代码 276
 - 10.1.3 用Java类实现Ruby类 278
 - 10.1.4 使用-J-XX:+PrintCompilation选项 281
 - 10.1.5 JIT是否提升了JRuby程序的性能 283
 - 10.2 JRuby和MRI中的字符串 284
 - 10.2.1 JRuby和MRI如何保存字符串数据 284
 - 10.2.2 写时复制 286
 - 10.2.3 创建唯一且非共享的字符串 288
 - 10.2.4 可视化写时复制 290
 - 10.2.5 修改共享字符串更慢 291
 - 10.3 总结 293
- 11 Rubinius：用Ruby实现的Ruby 295
 - 11.1 Rubinius内核和虚拟机 296
 - 11.1.1 词法分析和解析 298
 - 11.1.2 使用Ruby编译Ruby 299
 - 11.1.3 Rubinius字节码指令 300
 - 11.1.4 Ruby和C++一起工作 302

- 11.1.5 使用C++对象实现Ruby对象 303
- 11.1.6 Rubinius中的（栈）回溯 305
- 11.2 Rubinius和MRI中的数组 307
 - 11.2.1 MRI中的数组 307
 - 11.2.2 Rubinius中的数组 309
 - 11.2.3 阅读Array#shift源码 311
 - 11.2.4 修改Array#shift方法 312
- 11.3 总结 315
- 12 MRI、JRuby、Rubinius垃圾回收 317
 - 12.1 垃圾回收器解决三个问题 319
 - 12.2 MRI中的垃圾回收: 标记与清除 320
 - 12.2.1 空闲列表 320
 - 12.2.2 标记 321
 - 12.2.3 MRI如何标记存活对象 323
 - 12.2.4 清除 323
 - 12.2.5 延迟清除 324
 - 12.2.6 标记-清除的缺点 325
 - 12.2.7 观察MRI执行延迟清除 327
 - 12.2.8 观察MRI执行全回收 328
 - 12.2.9 解读GC分析报告 329
 - 12.3 JRuby和Rubinius中的垃圾回收 332
 - 12.4 复制垃圾回收 333
 - 12.4.1 碰撞分配 333
 - 12.4.2 半空间算法 334
 - 12.4.3 伊甸堆 336
 - 12.5 分代垃圾回收 337
 - 12.5.1 弱代假说 337
 - 12.5.2 为新生代使用半空间算法 338
 - 12.5.3 晋升对象 338
 - 12.5.4 成熟代对象垃圾回收 339
 - 12.6 并发垃圾回收 341
 - 12.6.1 当对象图改变时进行标记 341
 - 12.6.2 三色标记 343
 - 12.6.3 JVM中的三种垃圾收集器 344
 - 12.6.4 触发主收集 347
 - 12.7 延伸阅读 348
 - 12.8 总结 349
- 索引 351

《Ruby原理剖析》

精彩短评

- 1、非常好的一本原理书，深入浅出，看起来其乐无穷
- 2、清晰透彻，力荐
- 3、书很值得一读，不管你懂不懂Ruby，这本书会告诉你像ruby这样的语言的内部的工作机制，会提升你编译原理的知识。翻译的不能再好了，看不懂的部分我去看原文，也有理解难度。其他部分读起来都是很流畅的，全书应该经过细致的审稿，极少错误。
- 4、翻译不错。英文书的问题是：元编程及以后的章节写得不够深入。可读性不如metaprogramming ruby
- 5、把Ruby原理解析的不错
- 6、Ruby又一经典力作，帮助你从底层重新思考Ruby。

《Ruby原理剖析》

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:www.tushu111.com