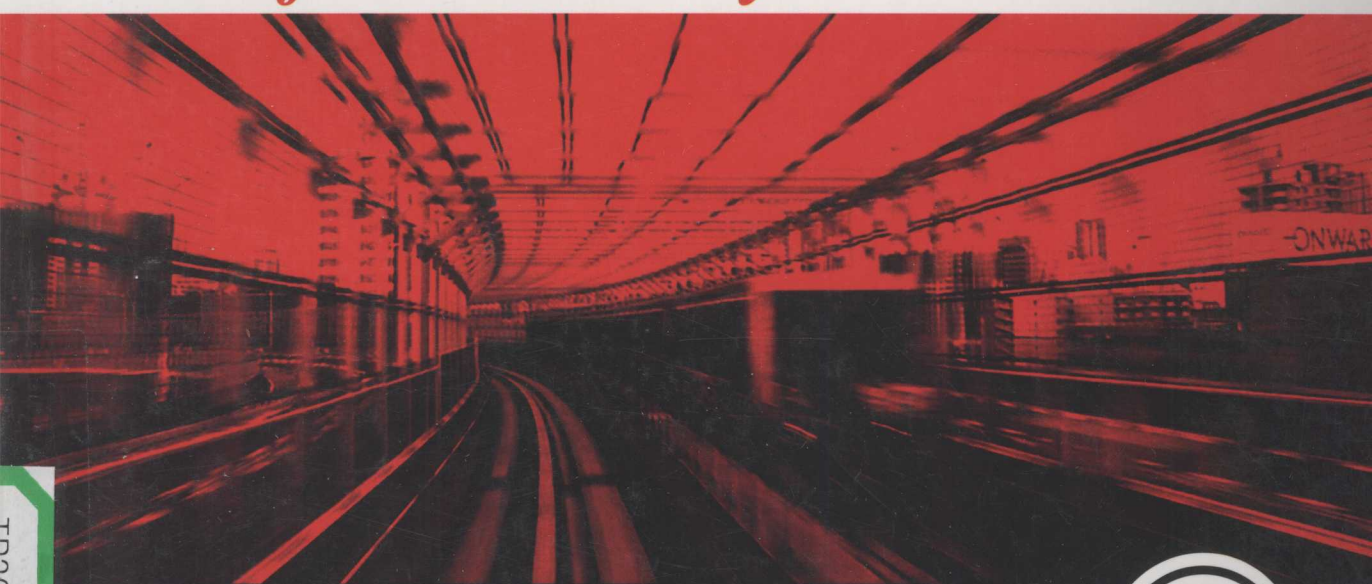
HZ BOOKS

PEARSON Addison Wesley

专家讲解，最新技术

Ruby 精粹专家编程

Professional Ruby Collection



（美） Matt Peiletier Zed Shaw David A. Black 著

洪文迅 译

Ruby 和 Rails 技术系列

机械工业出版社
China Machine Press

Ruby

精粹专家编程

Professional Ruby Collection

（美） Matt Peiletier Zed Shaw David A. Black 著

洪文迅 译

机械工业出版社
China Machine Press

本书是《Mongrel：服务、部署及扩展 Ruby 应用程序》和《Rails 路由》的合订本。第一本书介绍并指导 Mongrel（一个 Ruby 语言环境下快速、通用的 Web 服务器）的部署和使用。第二本书介绍 Rails 路由的相关内容，包括路由系统概述、编写自定义路由、REST 风格路由、具名路由以及对路由的反思等。这两本书都由行业专家和畅销书作者写就，包含了读者最需要的最新的专业技术信息。

本书适合 Ruby 开发人员和 Ruby 爱好者参考。

本书版权登记号：图字：01-2008-1788

图书在版编目（CIP）数据

Ruby 精粹专家编程/（美）皮尔特尔（Peiletier, M.），（美）肖（Shaw, Z.），（美）布莱克（Black, D. A.）著；洪文迅译. —北京：机械工业出版社，2008.12

（Ruby 和 Rails 技术系列）

ISBN 978-7-111-25351-8

Ⅰ. R…　Ⅱ. ①皮…　②肖…　③布…　④洪…　Ⅲ. 计算机网络－程序设计　Ⅳ. TP393.09

中国版本图书馆 CIP 数据核字（2008）第 161266 号

机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码　100037）

责任编辑：周茂辉

北京牛山世兴印刷厂印刷 · 新华书店北京发行所发行

2008 年 12 月第 1 版第 1 次印刷

186mm × 240mm · 9.5 印张

标准书号：ISBN：978-7-111-25351-8

定价：33.00 元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

译 者 序

若干年前，当我第一次听说 Ruby 语言时，它对于我而言只是一个符号，没有什么特殊意义。就像 Algol 60 语言，我父亲曾经在学校教过它，但对他而言，Algol 60 也只是一个符号，因为那个年代几乎没有什么机会使用它。

到 2004 年底，一位朋友给我看了一段短短的视频，这就是后来颇为流行的 DHH 对 Rails 早期版本的演示，我还记得演示主题是 15 分钟内构建一个简单的 Blog。

Cool!

是的，不可思议！当时那个视频场景深深地印在了我的脑海里。

此后，身边陆续有些朋友开始接触 Ruby 和 Rails，可惜我当时并不从事 Web 开发，只能留下羡慕和关注。直到 2006 年，Rails 已经炙手可热，学习 Ruby 的人也越来越多。我这个先知后觉的人才开始动手，通过一些试验性项目学习使用 Ruby 和 Rails。

我有个朋友是技术杂志总编，见识很广。他跟我提到 Ruby 语言是一种能够带给人快乐的语言。这可能会让许多将编程作为生计的人不太能够理解。因为大部分事情，当它成为职业后，给人的感受往往是痛苦和压力要多于快乐。可我想说的是，Ruby 带给我（以及其他许多人）的快乐却是实实在在的，因为它高效、直接、非常有趣，因此也让你觉得自己空前的强大！是的，Ruby 让我感受到前所未有的快乐（而在我曾经专注过的 Java 领域，快乐却是有点奢侈的事情）。

本书包含了业内大牛 Zed Shaw（Mongrel 的创造者）和 David Black（畅销书《Ruby for Rails》的作者）等人所撰写的精彩之作。我读后猜测，这些大概是牛人们的讨论班讲义的精华，因为其中一些内幕招式并不是网上轻易可以找到的，另一方面，写作措辞方面也不是那么严肃，而是轻松随意，或展示作者的强烈个性（尤其是 Zed Shaw）。

由于工作较忙，加上今年各种事情不断，翻译工作对我而言变成一个强度颇大的挑战，感谢广州电信国际通信中心的陈莉佳协助我的校对工作，为我节省了一些时间。还要特别感谢华章公司编辑陈冀康对我的信任和支持。限于本人的文笔和水平，翻译不当之处，请读者批评指正！

此外，需要注意的是，虽然本书书名为《Ruby 精粹专家编程》，但主要内容实际上是围绕 Rails 和 Web 应用开发展开，并不太适合纯粹只对 Ruby 语言感兴趣的读者。如果

你不做 Web 开发，只是用 Ruby 来进行一些脚本工作（例如进行系统维护管理、日常辅助工作），它也是一件利器，而且非常有趣——我推荐你读读华章引进出版的《Practical Ruby for System Administration》[⊖]，将会有另一番收获！

总之，Have Fun with Ruby！

译　者

2008 年 10 月

⊖ 其中文版由机械工业出版社出版，书名为《Ruby 系统管理实战》（书号是 978-7-111-25083-8）。——编辑注

Mongrel：服务、部署及扩展 Ruby 应用程序

(美)Matt Peiletier　Zed Shaw　著

献给我的父亲 Mike，

多年前他把我带到人间。

——M. P.

献给 Mike 和 Roxanne，

他们给予一位穷苦孩子最需要的帮助。

——Z. S.

致　谢

Zed：我要向所有为 Mongrel 项目工作过的人们表示感谢，不管是贡献了代码、思路、在邮件列表提供帮助、编写文档的，还是当我故意搞笑的时候取笑过我的。为避免遗漏了谁，我会运用 Internet 的力量来告诉正在阅读本书的人，去看看 Mongrel 的网站（http://mongrel.rubyforge.org），上面可以找到帮助过我们的人员名单。

Matt：我要感谢 Wilson Bilkovich，Trotter Cashion，Jeff Ng，Lee Nussbaum，Josh Viney，Ezra Zygmuntowicz 对本书的帮助，以及所有帮忙审阅和改进它的人。Zed 和我还要感谢我们的编辑 Debra Williams Cauley，她很早就意识到 Mongrel（和 Ruby）的潜力。

关于作者

Matt Pelletier 是 EastMedia 的合伙人，这是一家位于纽约市的软件、移动、商业开发公司。通过与技术伙伴 VeriSign 合作，EastMedia 为 Apache Heraldry 项目中的 Mongrel 提供了商业赞助。在该项目中，Mongrel 作为 Identity 2.0 软件层次中的关键组成部分之一。Matt 也是 NYC. rb 组织——纽约市 Ruby 用户组的联合创始人。

Zed Shaw 是 Mongrel（以及其他一些 Ruby 项目）的创始人和主要作者。他在日渐成长的 Mongrel 领域中表现非常活跃，回答问题、响应功能需求和故障报告、帮着调试排错、支持不断增长的需求和使用。

目　录

第1章　本书涵盖内容

这份“快速指南”是关于 Mongrel(一种 Ruby 语言环境下快速、通用的 Web 服务器)的介绍和使用指南。如果你需要构建或管理 Web 应用程序，那么以下内容对于在开发环境和生产环境下架设使用 Mongrel，是非常有用的参考资料。同时它也可以作为一本手册，指导你根据自己的需要对 Mongrel 进行扩展。

除了介绍如何使用和扩展 Mongrel 之外，我们还参考了许多自认为是“最佳实践”的主题，包括现代软件开发、部署和性能测试等方面。虽然本书的讨论主要是围绕 Mongrel 的使用场景，但这些内容也适用于任何软件项目。我们自己也常常通过观察其他技术实践而汲取经验。因此，希望本书关于 Mongrel 设计和开发的经验方法和哲学的分享是有意思的，并有助于达到你自己的目标。

1.1　本书的格式

本书使用通用的格式惯例表示代码、文件名等内容。如果你以前从没读过软件方面的书籍，请参考以下最基本的描述。

缩进的代码块或 shell 命令都将使用等宽的字体，如：

```
uri'/', :handler => DirHandler.new('/var/www/opinions/')
```

在句子中对文件、函数或类进行引用时，将使用斜体字，例如：*HttpHandler*、*mongrel. log*、*some_ function*(*param*)。

1.2　Zed 说(Zed Sez)

书中加入了称为“Zed 说”(Zed Sez)的特色专栏。Mongrel 是一个“有思想主见的软件”，这种思想来自 Zed Shaw——Mongrel 的创始人，他也是本书合作者之一、一位有思想主见的开发者。从某种意义上说，Mongrel 是 Zed 长期以来对自己工作中的实践和技巧反复锤炼后的“最佳作品”。对于运用他那套讲究实际的方法，Zed 决不退缩或迟疑(Mongrel 邮件列表的用户都可以证明这点)。所以我们认为“Zed 说”专栏是最适合 Zed 发表个人演说的空间，由他来告诉你关于 Mongrel 的一切。你在书中会不时看到穿插进来的“Zed 说”专栏，这是发自 Zed 内心的文字(他甚至还给自己画了个头像)。你不一定完全同意 Zed 所说的，不过，可能还是应该认同。

第2章 入门简介

本章将向你介绍 Mongrel，解释它的来由和历史，以及如何使用。由于我们发现关于 Mongrel 如何工作、能做些什么，有哪些普遍的错误看法。所以，需要澄清这些误解，以便能顺利开始。即使你认为自己了解 Mongrel 是什么，还是应该先读一下这部分内容。

2.1 Mongrel 是什么

Mongrel 是一个小型、快速、几乎完全由 Ruby 开发的 Web 服务器[⊖]。它的设计目标就是只做很少的几件事情，但一定把这些事情做好：包括让 Ruby 应用的开发、部署、扩展变得极为简易——主要就是这些！

Mongrel 的简短历史

Zed Shaw 在 2005 年 12 月底的时候开始了 Mongrel 这个项目。他之所以编写 Mongrel，是因为在开发和部署 Ruby 的 Web 应用程序时，对当时的解决方案很失望：FastCGI 方式问题百出，而 WEBrick 则慢得难以忍受。同年早些时候，Zed 曾写过一个 SCGI 方式的 Rails 运行环境（Runner），试图作为 FastCGI 的替代方案，但他很快遇到了阻碍，因为这个新方案也只是个折中方式。这一次，Zed 也感到厌烦了，但他仍然满怀信心，所以亲手编写了 Mongrel 来解决眼前的棘手问题。事实上，许多人跟 Zed 一样在这个问题上碰壁，所以 Zed 的解决方案对许多 Ruby 开发者和系统管理员大有裨益。Mongrel 的成功很大程度上基于 Zed 对软件架构的思路，而在 Ruby 开发的世界里，开发思想与实践和代码本身是同等重要的事情。因此，Zed 按照自己的思路打造了 Mongrel——它是一个 Web 服务器，而不是一个通用的适配器（adapter）；它紧凑、小巧并且安全。Zed 很乐意向你解释为何如此设计。

Mongrel 已经包含在许多 Linux 版本的发行包里，此外还将在 Apple 公司的 OSX 的下一个发布版（命名为“Leopard”，详情可参考 http://www.apple.com/server/macosx/leopard/more.html）中出现。

Zed 说（Zed Sez）

在我开始 Mongrel 项目的时候，Rails 开发方面有许多混乱状况。当时看来，似乎只

⊖ 开发中使用了 Ruby 的 C 语言扩展来处理 URI 解析，此外全部使用 Ruby 语言编写。

有37signals公司[1]才知道应该如何部署大规模的Rails应用程序。我猜这是因为他们在问题发作的时候去祭神保佑了[2]。而我也曾经尝试过SCGI方式的Rails运行环境，但这并没有什么改善。其实我们需要的只是一个Web服务器，运行Rails像FastCGI运行Rails一样快，使用方面像WEBrick一样简单。我也经常想搞清楚Mongrel是不是那么容易安装搭建，同时是不是灵活适用于尽量多的场合。

2.2 Mongrel是如何工作的

Web服务器的基本功能就是监听HTTP请求并传回响应。请求通常只是一个URL，一些信息头(header)，有时还有信息体(body)，例如表单参数。如果是请求一张图片，那么Web服务器通常会从文件系统读出文件，然后将它传送回去。如果请求需要送达一个Web应用，那么Web服务器一般会把请求转交给该应用来处理，由它产生一个包含信息头和HTML的响应结果，然后再由Web服务器传送给请求者。大多数Web服务器都逐渐发展成为庞然大物，因为它们被当作万用服务器来使用——需要处理种种想象得到的选项、模块和功能特性。Mongrel并不是取代这类服务器，而只是能够很好地处理HTTP请求，特别是面向Ruby应用的请求。可以说，Mongrel从一开始就是为非常简单的目标而设计的——就是易于使用、易于扩展并且非常安全。

2.3 Mongrel能做些什么

这一节将介绍作为开发者、系统管理员或经理，你能通过Mongrel得到什么。

2.3.1 对于开发者

Mongrel在本地开发时极为出色

如果你正在进行Ruby开发，那么选用Mongrel是完全不需要考虑的事情。它比WEBrick快得多，并且能轻松地处理大负荷的开发工作(包括开启日志和应用装载)。你只需要使用一个命令行来安装它，以及另一个命令行来配置它，安装运行甚至不超过一分钟。

它与Rails完美集成(也适用于其他Ruby框架)

Mongrel从一开始就是为了简化Rails应用的开发和运行而创建的，所以Mongrel的主要用户也是Rails的开发人员。他们已经为Mongrel贡献了一些出色的插件来处理一些常见任务，包括群集(clustering)、文件上传、DRb进程处理等，这些有助于学习如何

[1] 发明Rails框架的公司，DHH是其合伙伙伴。

[2] 原文：and I think that's only because they sacrificed chickens to Ninhursag between bouts of Touretts-inspired screaming。Ninhursag是苏美尔神话中的众神之母。——译者注

编写你自己的插件。早在 2006 年 6 月，Rails 的核心开发团队就已将 Mongrel 集成到 Rails 的“*script/server*”命令中了，如果这正是你喜欢的，那么剩下需要做的只是安装好 Mongrel 就行了。

易于扩展你的应用

当初 Mongrel 的设计就是为了让你易于扩展（或者说，扩展 Web 应用）。你可以把 Mongrel 想象成一个紧凑的 Ruby 程序库，专门用来处理繁琐的 HTTP 机制，让你很方便地处理请求，而其他细节则替你隔离起来。用于扩展的接口都非常简洁也易于理解，即使你对 Rails 世界以外的 Ruby 开发接触不多，这些接口也值得看看。你经常能够让 Mongrel 来替你处理某些任务，从而提高应用程序的运行速度。如果手头有自己的 Ruby 程序库，但不是 Mongrel 所支持的框架之一（参考 3.4 节），你可以编写一个处理程序（handler），从而让 Mongrel 载入你所需要的特殊选项（具体可参考第 6 章）。

Mongrel 包含了优秀的开发调试工具

Mongrel 为你调试程序提供了各种工具。你可以查看整个请求和响应处理过程，如果需要，还可以编写你自己的处理程序和插件（plugin）来深入测试你的应用程序和环境。

Mongrel 在生产环境中表现一致

在开发、构建、测试和部署应用程序后，你可以确信 Mongrel 将以一致的方式在生产环境中工作。特别是你需要运行测试、与其他系统或服务集成、使用缓存等场合下，这一特性非常有用。关于如何在本地开发机器上和生产环境下同时使用 Mongrel，可参考第 4 章。

2.3.2　对于系统/网络管理员

对于如何将 Mongrel 纳入你的系统架构，如果你还心存疑虑，那么你真的不必担心。因为 Zed 做过许多年的系统管理工作，曾花费大量时间来处理那些设计不良的软件包——所以，他知道怎样才能让 Mongrel 和常见的服务配置环境一起很好地工作。

Mongrel 是个真正的 Web 服务器

Mongrel 是个完整遵循 HTTP 协议的服务器（相对大多数其他服务器而言），它能够通过 HTTP 协议与常见软硬件配置环境的网络设备直接对话。这使得你不必再将请求翻译为 CGI，或使用问题多、速度又慢的模块（例如 mod_ruby 或 FastCGI 之类）来连接你的 Ruby 应用。

Mongrel 本身也是个 Ruby 应用

因为 Mongrel 是用 Ruby 编写的，所以它自然能够装载、运行其他 Ruby 程序代码，这就好像 Tomcat 能够处理 Java 应用程序一样（不一样的是，Mongrel 并不需要吃掉 120MB 内存）。正因为如此，你不再需要 FCGI 方式、SCGI 方式的 Rails 运行环境，或者任何其他模块把 HTTP 请求传递到 Rails 应用框架。

Mongrel 很容易进行集群和部署

Mongrel 被设计用来在生产环境下极容易地安装部署。具体来说，一些主要关心的

问题如集群和部署都考虑到了，并且采用了最优实践方案。你可以参考 4.2 节以了解更多关于 mongrel_cluster 的内容，还可以参考第 5 章，看大家是如何通过 Capistrano 来部署应用的。

2.3.3　对于开发经理

有眼光者选择了 Mongrel

在 Rails 应用的开发和运行的可用环境中，Mongrel 被广泛认为是一个明智的选择。其实对于 Ruby 应用的开发和运行也是如此，但目前大部分 Ruby 开发人员只是使用 Mongrel 来运行他们开发的 Rails 网站。Mongrel 并不是一堆庞大的代码，事实上，它的诞生还不到一年[⊖]。但出于各种原因，就在短短 6 个月内，它成为开发环境和生产环境下运行 Web 应用的首选。

Mongrel 的安全性真的很好

对外公布软件的安全性也是个惯例要求，对此我们要说的是“Mongrel 的安全性真的很好”！Zed 将在第 9 章为你具体解说。

Mongrel 的授权方式适于商用

Mongrel 以及它的子项目 GemPlugin 使用的是 Ruby 的授权方式，具体内容你可参考：http://mongrel.rubyforge.org/license.html。有些人害怕使用开源软件，因为他们觉得这可能使得自己也被迫开源自己的代码（在某些特定场合，如 GPL 和 LGPL 授权下确实如此）。但 Ruby 的授权协议并没有那么严格的限制，这已足以支持 Apple 公司在下个版本的 OSX 中将 Mongrel 加入进去。

⊖ 这是相对于本文作者的写作时间而言。——译者注

第3章　Mongrel的安装和使用初步

你已经了解了Mongrel的一切，现在让我们开始使用它吧。安装和配置Mongrel仅仅需要一个简单的命令。本章内容将介绍Mongrel的基本使用方法，而更深入的配置方面将在第4章介绍。

下文中的例子假定你是在Rails应用中使用Mongrel。如果并非如此，你使用的是其他的支持框架，或者在特定的应用中使用Mongrel，你仍然可以跟随我们一起学习下面的内容。虽然Mongrel是框架无关的，现在大部分社区都使用它运行Rails应用，所以最经得住考验和最成熟的命令、操作、过滤器和模式都是面向Rails的。其他框架也是按类似的方法运用Mongrel的。如果你想了解Mongrel扩展的内容，请看第6章。

我们的例子假定你使用UNIX操作系统。Mongrel在Windows下是完全支持的，所以大部分例子不需要修改也能在Windows下正常运行。有关Win32的特殊限制和差异的详细内容请参考第3.3节和Mongrel的网站(http://mongrel.rubyforge.org)。

3.1　安装Mongrel

Mongrel的安装非常简单，唯一的先决条件是安装Ruby 1.8.4或以上版本，以及RubyGems。如果还没有安装这些，你将不能运行Ruby。关于Ruby和RubyGems的安装帮助请参考Ruby的网站(http://www.ruby-lang.org/en/)。大多数情况下，你是需要安装这两项的。

你可以通过命令行，一步完成Mongrel的安装：

```
$ sudo gem install mongrel --include-dependencies
```

该命令将根据你的平台类型(Daemon或者Win32服务式)安装Mongrel gem和gem附带的相关程序。如果不是在Windows环境下安装gem，你可以看到在安装过程中Mongrel在编译C。如果在安装过程中遇到问题，你可以发邮件或者在官方网站上查找原因。当然也有可能是Ruby安装的问题引起的。

Mongrel的预发布版本

如果你是冒险主义者(毕竟你正在使用Ruby，不是吗?)，可以从Mongrel预发布共享资源中安装和测试预发布版本gem。跟大部分预发布版本代码一样，这些gem包含新的功能和(或)需要一些平台测试的问题修正，总体上它们还是比较稳定的。Ruby倾向于引入一些比较有趣的系统平台差异，所以在不同的平台上进行测试尤为重要。