Ruby on Rails: Up and Running

Ruby on Rails

构建与运行

O'REILLY®
東南大學出版社

Bruce A. Tate & Curt Hibbs 著
O'Reilly Taiwan 编译

Ruby on Rails: 构建与运行

Bruce A.Tate & Curt Hibbs 著

O'Reilly Taiwan 公司 编译

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Paris • Sebastopol • Taipei • Tokyo

O'Reilly Media, Inc. 授权东南大学出版社出版

东南大学出版社

图书在版编目（CIP）数据

Ruby on Rails：构建与运行 /（美）泰特（Tate, B.A.），
（美）希布斯（Hibbs, C.）著；O'Reilly Taiwan 公司编译 .
南京：东南大学出版社，2007.8
书名原文：Ruby on Rails: Up and Running
ISBN 978-7-5641-0872-4

I. R... II. ①泰 ... ② O... III. 软件开发 IV. TP311.52

中国版本图书馆 CIP 数据核字（2007）第 121645 号

江苏省版权局著作权合同登记
图字：10-2006-254 号

书　　名 / Ruby on Rails：构建与运行
责任编辑 / 张烨
封面设计 / Karen Montgomery，张健
出版发行 / 东南大学出版社
地　　址 / 南京四牌楼 2 号（邮编 210096）
印　　刷 / 扬中市印刷有限公司
开　　本 / 787 毫米 × 980 毫米　16 开本　11.5 印张　193 千字
版　　次 / 2007 年 8 月第 1 版　2007 年 8 月第 1 次印刷
印　　数 / 0001-4000 册
书　　号 / ISBN 978-7-5641-0872-4/TP · 139
定　　价 / 26.00 元（册）

O'Reilly Media, Inc. 介绍

为了满足读者对网络和软件技术知识的迫切需求，世界著名计算机图书出版机构O'Reilly Media, Inc. 授权东南大学出版社，翻译出版一批该公司久负盛名的英文经典技术专著。

O'Reilly Media, Inc. 是世界上在 Unix、X、Internet 和其他开放系统图书领域具有领导地位的出版公司，同时也是联机出版的先锋。

从最畅销的《The Whole Internet User's Guide & Catalog》（被纽约公共图书馆评为二十世纪最重要的50本书之一）到GNN（最早的Internet门户和商业网站），再到WebSite（第一个桌面PC的Web服务器软件），O'Reilly Media, Inc.一直处于Internet发展的最前沿。

许多书店的反馈表明，O'Reilly Media, Inc. 是最稳定的计算机图书出版商 —— 每一本书都一版再版。与大多数计算机图书出版商相比，O'Reilly Media, Inc. 具有深厚的计算机专业背景，这使得O'Reilly Media, Inc. 形成了一个非常不同于其他出版商的出版方针。O'Reilly Media, Inc. 所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O'Reilly Media, Inc. 还有许多固定的作者群体 —— 他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O'Reilly Media, Inc. 依靠他们及时地推出图书。因为O'Reilly Media, Inc. 紧密地与计算机业界联系着，所以O'Reilly Media, Inc. 知道市场上真正需要什么图书。

作者简介

Bruce A. Tate是来自德州奥斯汀的一名独木舟、登山自行车爱好者，他也是两个孩子的爸爸。2001年，他创立J2Lite，LLC（译注）独立顾问公司，现今称为RapidRed。在那里，他主要致力于使用Ruby on Rails从事快速软件开发的培训、实现与咨询。客户涵盖FedEx、Great West Life、AutoGas、TheServerSide以及BEA。他拥有20年的丰富经验，包括在IBM的13年以及担任好几家新创公司的领导层职务的经验。他是个享誉国际的演讲者以及9本广受推崇的软件开发书籍的作者，包含脍炙人口的《超越Java》（O'Reilly出版，中文版由东南大学出版社出版）、赢得Jolt大奖的《轻快的好Java》（O'Reilly出版，影印版由东南大学出版社出版）、备受管理层瞩目的《From Java to Ruby》（Progmatic）及轰动一时的《Bitter Java》（Manning）。

Curt Hibbs长久以来一直热衷于新技术及新趋势的发展。然而，他会告诉你之所以如此纯粹是因为他懒，他总是在寻求新方法与新技术，好让工作更轻松愉快、更具生产力。这种倾向促成他在2001年发现Ruby（当时，Ruby在日本以外相对不为人知），也让他开创了一些极为成功的Ruby开放源码项目。在他的职业生涯里的大部分时间中（开始于20世纪70年代早期），Curt曾经担任一些著名公司的顾问，如Hewlett Packard、Intuit、Corel、WordStar、Charles Schwab、Vivendi Universal等，他也曾担任一些新创公司的重要职务。Curt目前任职于在圣路易的波音公司，做资深软件工程师。

封面介绍

在《Ruy on Rails：构建与运行》一书封面上的动物是ibex（阿尔卑斯野山羊，*Capra pyrenaica*），它们分布在欧洲、中亚、北非的山区，ibex大部分时间生活在海拔7500到11500英尺（1英尺=0.3048米）的山区。ibex以其令人印象深刻的长角而闻名，雄性成羊的角长达3英尺。求偶季节里，雄性成羊为争取交配的权利在激烈的战斗中以长角相互猛力撞击。

虽然这种英勇事迹的实质意义暧昧含糊，但根据传说，ibex的角异常强壮，以至在生命受威胁时，这种动物可从断崖绝壁上一跃而下，并且凭借着它的长角安然无恙地着陆。

封面图像摘自《Riverside Natural History》一书。

目录

前言

无视于既有的编程语言、陈规惯例或商业支持，Ruby on Rails现象正风起云涌，席卷整个业界。你能从一些网络文章、出色好书甚至正式课程中得到许多关于Ruby on Rails的完整信息。不过，这里有些东西被遗漏了——一个有经验的程序人员怎样凭借些许的Ruby知识超越基础、运用Rails形成强大的生产力？

在这本《Ruby on Rails：构建与运行》中，我们不再对相关参考手册作赘述，或者取代你用Google就能找到的资料；相反，我们将努力带给你Rails应用程序怎样构建的整体图，并且告诉你在哪里可找到各章所未包含的信息。你将明白Rails怎样动态地为全部数据库数据模型添加特性功能，即所谓的Active Record对象。通过理解此整体图，你将能够仔细且充分利用最好的参考手册。

我们不会试图带你仔细领悟所有细节信息，相反，我们将基于端到端（end-to-end）应用程序的情境，为你奠定理论基础。我们将带你一步步完成简单项目的建立——一个比博客或购物车要求更高但结构简单的项目，Rails初学者将能迅速理解其内涵。

我们不会试图涵盖每个新功能特性，相反，我们将告诉你我们视之为骨干脊梁的那些部分，它们形成你所必须理解的最重要的元素。我们也将以某种程度的详述涵盖迁移（migration）与Ajax，因为你不容易找到关于这两种框架的信息。

简言之，我们不会试图建造一个全面的Rails链接库，我们将给你构建与运行Rails应用程序所需要的基础。

谁应该读这本书？

《Ruby on Rails：构建与运行》是针对不熟悉Rails（也包括Ruby）的有开发经验的人

而写，要使用这本书，你不必是个厉害的Ruby编程人员。然而，我们确实预期你是个编程人员，你应该足够了解你所选择用来写程序的平台、安装软件、使用系统控制台运行script、编辑文件、使用数据库并且了解基本的Web应用程序怎样运作。

排版约定

下面是本书中用到的一些排版约定：

一般文字

用以表示菜单标题、菜单选项、菜单按钮以及键盘快捷键（例如Alt与Ctrl）。

斜体字（*Italic*）

用以表示新术语、URL、电子邮件地址、文件名、文件扩展名、目录、路径名称以及Unix工具。

等宽字（`Constant Width`）

用以表示指令、文件内容、指令的输出。

等宽斜体字（*`Constant Width Italic`*）

用以表示要以用户所提供的值替换的文字。

等宽黑体字（**`Constant Width Bold`**）

用以表示要以用户所提供的值替换的指令或其他文字。

使用范例程序代码

这本书的目的是协助你完成工作。一般而言，你可以在自己的程序与文件里使用这本书的范例程序代码。除非重新制作并散布大部分程序代码，否则不需要与我们联系以取得授权。例如，开发程序时使用书中一些范例程序代码并不需要取得授权，然而贩卖或散布O'Reilly书籍的范例程序光盘就需要取得授权；为了回答问题引用这本书的内容或程序代码并不需要取得授权，把书中大量范例放到你自己的产品文件中就需要取得授权。

你可以在《Ruby on Rails：构建与运行》的主页上取得示例程序代码：*http://www.oreilly.com/catalog/rubyrails/*。你将找到一些ZIP文件，它们包含每章的示例项目，按章编号的应用程序样本实例。若你想要略过某一章，只要下载正确的ZIP文件即可。

我们会很感谢你在使用范例程序代码时注明其归属，但这并非必要。归属说明通常包括书名、作者、出版社以及ISBN。例如“《Ruby on Rails：构建与运行》，Bruce A. Tate与Curt Hibbs合著。版权为2006 O'Reilly Media, Inc.所有，978-0-596-10132-9”。

如果你感到对范例程序代码的使用超出公平使用或上述的许可范围，请通过 *permissions@oreilly.com* 与我们联系。

平台

Ruby on Rails 是跨平台的，但是 Unix 和 Windows 的 shell 在行为上略有不同。为求一致，我们在全书中使用 Windows。当然，你也可以轻易在 Unix 或 Mac OS X 操作系统上运行这些例子。下面是几个微小的差异：

- 在 Windows 上，你可以用斜线（/）或反斜线（\）指定路径。我们将尽量保持一致，使用斜线（/）指定所有路径。
- 在 Windows 上，为运行组成 Rails 的各种 Ruby script，必须明确输入 `ruby`。在 Unix 环境中则不必。如果你正在运行 Unix 并且被指示输入指令 `ruby script/server`，则可放心把 `ruby` 省略。
- 在 Windows 上，为了在一个独立的 shell 里运行程序，就要在指令前加上 `start`。在 Unix 和 Mac OS X 上，则在指令之后加上 & 字符，让指令在后台运行。

建议与评论

本书的内容都经过测试，尽管我们做了最大的努力，但错误和疏忽仍然是在所难免的。如果你发现有什么错误，或者是对将来的版本有什么建议，请通过下面的地址告诉我们：

美国：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472

中国：

100080 北京市海淀区知春路 49 号希格玛公寓 B 座 809 室
奥莱理软件（北京）有限公司

与本书有关的在线信息（包括勘误表、范例程序、相关链接）：

http://www.oreilly.com.cn/book.php?bn=978-7-5641-0872-4

http://www.oreilly.com/catalog/rubyrails

最后，您可以在以下站点找到我们：

http://www.oreilly.com
http://www.oreilly.com.cn

致谢

写书是一种高要求的活动，需要热情、承诺和坚持。名义上的作者将获得所有光环（也包括责备），而很多人对本书有贡献，我们想要在此表达由衷的感谢，是他们让本书的编写成为一个完整的经验。

Curt和Bruce要感谢杰出的审阅团队，他们提供了许多宝贵的意见，包括David Mabelle、Mauro Cicio、Brooke Hedrick、Faisal Jawdat、Shane Claussen、Leo de Blaauw、Anne Bowman、Seth Havermann、Dave Hastings与Randy Hanford。还要感谢David Geary，他充实了Photo Share（译注1）的一些早期想法。

没有Ruby on Rails核心团队的卓越贡献，就没有《Ruby on Rails：构建与运行》。我们要感谢David Heinemeier Hansson（Rails的创造者）、Florian Weber、Jamis Buck、Jeremy Kemper、Leon Breedt、Marcel Molina, Jr.、Michael Koziarski、Nicholas Seckar、Sam Stephenson、Scott Barron、Thomas Fuchs以及Tobias Luetke。Ruby是一种了不起的程序语言，我们想要感谢许多对它有贡献的人，特别感谢Yukihiro Matsumoto（亦称为“Matz”，Ruby的创造者）、Dave Thomas以及Andy Hun，没有他们，Ruby可能在日本之外几乎无人知晓。

Bruce要特别感谢Curt，在此项目似乎终结时加入我们。此外，感谢AutoGas人员的鼎力相助，他们在真实的应用程序产品中尝试这项新技术，特别是核心开发团队，包括Mathew Varghese、Karl Hoenshel、Cheri Byerley、Chris Gindorf以及Colby Blaisdell，他们的团队经验让这本书超越你将知道的。再次感谢我的荷兰朋友Leo，以及他对本书的影响。虽然他基本上是个Java开发人员，但他对本书的影响可能超乎你的想象。最要感谢是我的家人Kayla与Julia，你们是我灵魂里的火花，让我的创作之火永燃。Maggie，你是我的灵感，对你的爱超过所有言语。

Curt要感谢他的妻子Wasana，允许他隐身于计算机屏幕后，直到深夜（有时到第二天），从无抱怨。还要感谢O'Reilly的朋友给他一个论坛，传达Ruby on Rails难以置信的生产力优势的信息。特别要感谢chromatic（译注2）发布我的ONLamp.com文章；还有Mike Loukides，在我一直告诉他不想再写书时仍然坚持不放弃。

译注1：Photo Share：贯穿全书的范例应用程序。

译注2：chromatic：O'Reilly Network的技术编辑，请参考*http://www.oreillynet.com/pub/au/176*。

第一章

从0到60：Rails简介

Rails可能是过去10年来最重要的开放源码项目，它被推崇为历史上最具生产力的Web开发框架之一，并且是基于日益重要的Ruby编程语言。迄今，成果如何？

- 到2006年12月前，你很可能看见Rails相关的书籍出版量将超越任何Java的单一旗舰级框架相关的书籍出版量，包括JSF、Spring或Hibernate。
- Rails框架仅从它诞生的第二年至2006年5月止，已经被下载至少500000次。这个数字超越任何语言中最受欢迎的开放源码框架（注1）。
- 相较于其他语言中最受欢迎的Web开发框架每天收到数十封邮件，Rails社群的邮件论坛每天获得数百封的信件。
- Rails框架已经引起Ruby程序语言在使用上的迅速膨胀，在此之前，情况都还是相对模糊不清。
- Rails的蓬勃发展对聚焦于其他程序语言的门户网站产生了日益激烈的论辩，Java社群尤其强烈。

不必费心找寻优良的Rails概述，你可以看几个由Rails创造者David Heinemeier Hansson叙述的，显示Rails运作情形的教学影片。你可以看到他在不到10分钟的时间里构建出简单的应用程序，完成数据库与验证工作的支持。但不同于许多你看过的quick-and-dirty环境，Rails让你quick，但却不dirty。它让你基于model-view-controller（数据模型—视图—控制器）的开发哲学，构建干净的应用程序。Rails真的是一种特别的框架。

注1：500000这个数字实际上是保守估计，此数字是针对广受欢迎的下载工具*gems*所下载的分布版本统计而来，但还有许多其他分布版本存在，例如Mac OS X上的Locomotive分布版本。实际上的数字可能会轻易超过这个数字的两倍。

当然，Rails有它的限制。Ruby对传统schema所提供的对象—关系映射（object-relational mapping，ORM）的支持较差（注2）。这方面，Rails的做法比Java逊色。而且Ruby尚未有非常出色的集成开发环境（IDE）。每种框架都有其限制，Rails也一样。但对绝大范围内的应用程序来说，Rails的弱点相对于其强大力量而言可谓瑕不掩瑜。

Rails的力量

阅读本书时，你将了解Rails怎样在没有其他语言所需的广泛链接库的情况下却能够繁荣。Ruby的灵活性让你能够以过去对你可能无效的方式来扩展你的应用程序。你将可以运用一种叫作*scaffolding*的Rails功能，迅速将数据库支持的用户界面呈现在客户面前。然后，随着你改进程序代码，scaffolding逐渐消失。你将只用几行程序代码就能够构建数据库支持的数据模型（model）对象，Rails将替你填补冗长乏味的细节。

现今典型开发项目最常见的编程问题牵涉到建立基于Web的用户界面来管理关系型数据库。对这类问题来说，Rails比任何我们曾用过的其他Web开发框架都更具生产力。其力量不局限于单一的开创性发明；相反，Rails包裹了一些让你更具生产力的功能，当中的许多功能彼此相辅相成：

元编程（Metaprogramming）

元编程（metaprogramming）技术使用程序写程序。其他框架广泛使用程序代码生成（code generation），它赋予用户一次的生产力推进（one-time productivity boost），但后续工作便帮不上忙，自定义script只能让用户把自定义程序代码加到少数精心挑选的地方。元编程取代这两种原始的技术并且消除它们的缺点。Ruby是元编程的最佳语言之一，而Rails善用这项能力（注3）。

Active Record

Rails引进Active Record框架，它将对象存储到数据库。根据Martin Fowler的设计模式的分类，Rails版本的Active Record找到数据库schema里的字段，并使用元编程自动将它们指派给你的领域对象（domain object）。这种封装数据表的方法既简单、优雅且强大。

配置的约定

.NET或Java的大多数Web开发框架强迫你编写一页又一页的配置代码（configuration code）。如果你遵循建议的命名约定，则Rails不需要多少配置。事

注2：例如，Hibernate支持三种继承映射（inheritance mapping），但Rails仅支持单一数据表的继承。Hibernate支持组合键（composite key），但Rails则有诸多限制。

注3：Rails也使用程序代码生成，但对吃力的工作主要还是依赖元编程。

实上，只要遵循命名约定，往往能削减 80% 以上的配置代码，即使遵循相同命名约定的类似 Java 框架也无法有此效率。

Scaffolding

在开发的早期，你往往会创建临时性程序代码来帮助迅速建立应用程序并看看主要组件如何一同工作。Rails 自动建立你将需要的大部分 scaffolding（临时性程序代码）。

内置测试

Rails 创建你能加以扩展的简单自动化测试。Rails 也提供称为 *harness* 与 *fixture* 的支持程序代码，让测试案例更容易编写及运行。然后，Ruby 能利用 `rake` 公用程序执行全部的自动化测试。

三种环境：开发、测试及产品

Rails 给你三种默认环境：开发、测试及产品。每一种环境在行为上略有不同，让你的整个软件开发循环更容易。例如，Rails 为每次测试的执行创建未经加工的测试数据库。

还有更多其他功能，包括支持丰富用户界面的 Ajax、用于重利用视图（view）程序代码的 partial 视图与辅助器（helper）、内置缓存、邮件功能框架以及 web service。我们无法在本书中涵盖 Rails 的所有功能，不过我们会让你知道哪里可找到更多信息。但是领会 Rails 的最好方式是从实现中理解它，因此，让我们就这么做吧。

动手做做看

你能够手动安装所有的 Rails 组件，但 Ruby 有个称为 *gems* 的东西。`gem` 安装程序（installer）访问 Ruby Forge 网站，下载应用程序单元（称为一个 gem）以及它的所有依赖物（dependency）。你能通过 gems 安装 Rails 并取得所有依赖物。可使用下列指令：（注 4）（译注 1）

```
gem install rails --include-dependencies
```

注 4：若你想要与我们一起编写程序，请确定你已经安装了 Ruby 与 gems。附录一包含安装的详细信息。

译注 1：在附录一中，你可以使用 Instant Rails（Windows 版）或 Locomotive（OS X 版）一次完成所有软件的安装，包括 Ruby、Rails、Web 服务器、数据库等。

MVC 与 Model2

在20世纪70年代中期，MVC（model-view-controller，模式—视图—控制器）策略在Smalltalk社群里逐步演进，意在减少商业逻辑和表示层逻辑之间的耦合。通过MVC，你把商业逻辑放进独立的domain object（领域对象）并将表示层逻辑隔离在view（视图）里，view呈现来自domain object的数据。controller（控制器）管理view之间的浏览（navigation），处理用户输入并在model（模式）和view之间安排正确的domain object。好的程序人员已经使用MVC，运用许多不同语言所写成的框架（包括Ruby）实现MVC应用程序。

Web开发者使用略有不同的MVC变形，称作Model2。Model2使用相同的MVC原则，但略调整为无状态（stateless）Web应用程序。在Model2应用程序中，浏览器通过Web标准调用controller。controller与model交互以获取数据并验证用户的输入，然后准备domain object供view用于显示。接下来，controller根据验证结果或者取回的数据，调用正确的view生成器。view分层使用controller的数据产生网页，然后，框架将网页返回给用户。在Rails社群内，当某人提到MVC时，他指的是Model2。

Model2已经跨越多种编程语言，被使用在很多成功的项目里。在Java社群里，Struts是最普遍的Model2框架。在Python里，名为Zope的旗舰级Web开发框架使用Model2。你可以在*http://en.wikipedia.org/wiki/Model-view-controller*读到更多有关MVC策略的信息。

就这样，Rails安装完成了。注意：你也需要安装数据库相关的支持。如果你已安装MySQL，便已完成；如果没有，到*http://rubyonrails.org*，参考Rails安装的更多细节。接下来，看看怎样创建Rails项目：（译注2）

```
> rails chapter-1
      create
      create  app/controllers
      create  app/helpers
      create  app/models
```

译注2：这是你在本书所见的第一个Rails指令，当然，这表示你已经安装好Ruby与Rails（甚至以后会用到的数据库、Web服务器等）。因此，先翻到附录一，下载及安装一下吧。安装后，建议你将*C:\InstantRails\ruby\bin*（在此以Windows为例，并假设安装目录是*C:\InstantRails*）加到环境变量PATH里，以方便你在任何应用程序所在目录使用rails、rake、ruby等指令。另外，若你未指定应用程序所在目录（如本例），应用程序将于当前目录产生。

```
      create  app/views/layouts
      create  config/environments
      create  components
      create  db
      create  doc
      create  lib
...
      create  test/mocks/development
      create  test/mocks/test
      create  test/unit
      create  vendor
...
      create  app/controllers/application.rb
      create  app/helpers/application_helper.rb
      create  test/test_helper.rb
      create  config/database.yml
...
```

我们截短了此列表，但你可以了解整个状况。

组织

安装期间所创建的目录提供了一些位置供你放置程序代码、script（帮你管理及构建应用程序）以及很多其他必需的信息。稍后，我们将详细查看一些最有趣的目录。现在，让我们花点时间，迅速浏览所创建项目里的目录树：

app

此目录组织你的应用程序组件，内含几个子目录，用于存放view（*views*和*helpers*）、controller（*controllers*）以及后台商业逻辑（*models*）。

components

此目录存放组件——捆绑了model、view及controller的独立的（self-contained）小应用程序，。

config

此目录包含应用程序所需的少量配置代码，包括数据库配置（在*database.yml*）、Rails环境结构（*environment.rb*）、进来的Web请求的路由选择（*routes.rb*）。你也可以使用*environments*目录里的文件，略为修改用于测试、开发和部署的三种Rails环境的行为。

db

通常，你的Rails应用程序会有访问关系型数据库数据表的数据模型对象（model object）。通过创建并放置于此目录里的script，你能管理关系型数据库。