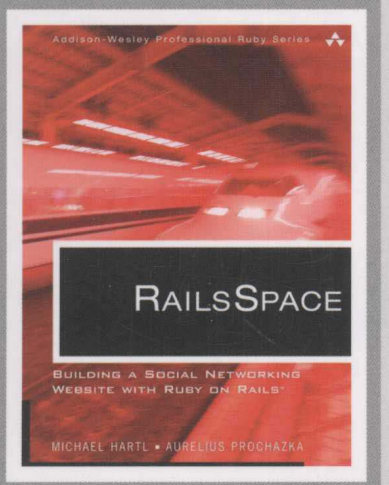
开源系列

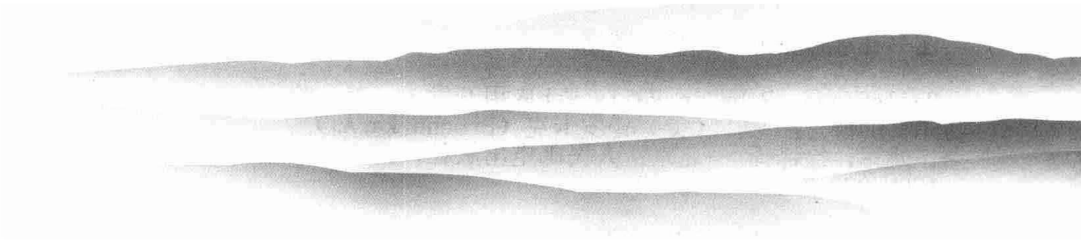
RailsSpace
—Ruby on Rails Web应用开发

[美] Michael Hartl　Aurelius Prochazka　著
姚军　徐锋　译



RailsSpace:
Building a Social Networking Website with Ruby on Rails

人民邮电出版社
POSTS & TELECOM PRESS



RailsSpace

——Ruby on Rails Web 应用开发

[美] Michael Hartl　Aurelius Prochazka　著
姚军　徐锋　译

人民邮电出版社
北京

图书在版编目（CIP）数据

RailsSpace：Ruby on Rails Web应用开发 /（美）哈特尔（Hartl, M.），（美）普罗卡克（Prochazka, A.）著；姚军，徐锋译. —北京：人民邮电出版社，2009.1
ISBN 978-7-115-19121-2

Ⅰ. R… Ⅱ. ①哈…②普…③姚…④徐… 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170033号

版 权 声 明

RailsSpace——Ruby on Rails Web 应用开发

◆ 著　[美] Michael Hartl　Aurelius Prochazka
译　姚　军　徐　锋
责任编辑　李　际
执行编辑　刘映欣

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
邮编　100061　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京顺义振华印刷厂印刷

◆ 开本：800×1000　1/16
印张：27
字数：611 千字　2009 年 1 月第 1 版
印数：1－3 000 册　2009 年 1 月北京第 1 次印刷
著作权合同登记号　图字：01-2008-2653 号

ISBN 978-7-115-19121-2/TP

定价：55.00 元

读者服务热线：(010)67132705　印装质量热线：(010)67129223
反盗版热线：(010)67171154

内 容 提 要

本书通过对一个社交网络 RailsSpace 开发过程的介绍，详细地展示了流行的 Web 应用程序开发框架 Ruby on Rails 的配置和使用方法。本书循序渐进地带领读者完成一个完整的项目，从静态的标题页开始，通过添加注册和验证功能，逐步完成一个高度动态的网站，它具备用户配置、图像上传、简单的博客、纯文本和地理位置搜索以及交友请求系统等功能。本书内容翔实，涵盖了诸如 MVC 程序架构、关系数据库和 AJAX 支持、强大的测试机制和 REST 风格等许多 Rails 的精彩特性，以及注册和登录、CSS 样式和特效、后台数据库交互、博客站点等内容。

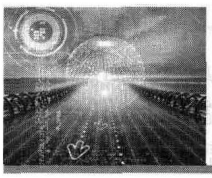
本书是以实例指南的形式组织编写的入门书籍，适合渴望了解 Ruby on Rails 的开发人员、各类 Web 开发人员以及网站建设人员。已经熟悉 Ruby 甚至已经对 Rails 有了一定了解的读者，也能够在本书中学到 Rails 更新版本的许多新特性。对于需要“Rails 百科全书”的读者，本书则提供了许多参考书籍和网站。

鸣谢

感谢 Debra Williams Cauley 在本书出版过程中对我们的指导。还要感谢技术审校 Francis Hwang 和 Michael Vanier 的认真阅读与评论。

目录

第 2 部分 创建一个社交网络



第 1 章

导　言

本书通过开发一款名为 RailsSpace 的真实应用程序来讲授 Ruby on Rails 的相关知识。RailsSpace 是针对 Rails 社区的社交网站。我们将带领你从几乎是静态的标题页开始，通过添加注册和验证功能，逐步完成一个高度动态的具备用户配置、图像上传、简单的博客、纯文本和地理位置搜索以及交友请求系统的站点。RailsSpace 当然并不打算与重量级的社交网站竞争，但它也不是一个玩具，它是用来展示如何使用 Rails 构建适合于部署到生产环境中的 Web 应用程序。

1.1　使用 Rails 的理由

Ruby on Rails 是用于 Web 应用程序开发的工具，它确实也非常擅长于这项工作。你可能会对是什么使 Rails 如此特别而感到惊奇。当你向一位已经爱上 Rails 的编程人员询问这个问题时，他可能会告诉你因为 Rails 是“敏捷”的，你必须深入了解一下这对于编程人员而言意味着什么。你也可能会淹没在一系列缩写词中，诸如 ORM、MVC 或者 DRY；虽然这些都是 Ruby on Rails 的精彩特性，但 Rails 流行的真正原因并不是获益于这些缩写词，而是与其高效的设计理念有关。

“设计”这个词本身就是一个意思含糊的概念，但是也不至于将它理解错误。好的设计就像文学作品一样，只要看到就知道好不好；Rails 就是一个能让你感到非常棒的设计。要总结 Rails 的设计为何如此之好是件困难的事，但我们会尽力去做，我们认为核心的原因是“自由的 free[1]生产力”。

1.1.1　生产力趋于自由

自由的（free）生产力的本质是 Rails 具有预见 Web 编程人员需求的奇特能力。这样说并

[1] 译者注：在此 free 是一语多关，含义包括自由软件、免费交流以及非常大的生产力。

不是非常具体，下面通过一系列实例来说明 Rails 所提供的自由生产力。所有这些特性并非现在就对你有意义，但是能帮助你了解 Rails 所擅长的领域。

- Ruby　Rails 应用程序以及 Rails 本身都是使用 Ruby 编写的，Ruby 是一种动态的、面向对象的编程语言。Ruby 源于 Perl，它的创始人松本行弘称它为“比 Perl 更好的 Perl”。根据经验，大部分熟悉这两种语言的编程人员都同意这一说法。我们还要加上一句，嵌入式 Ruby（ERb）是比 PHP 更好的 PHP。Rails 的主要优势是能够充分利用 Ruby 的功能和简练性。
- 数据库表到 Ruby 对象的映射　在大部分的 Rails 应用程序中没有杂乱无章的 SQL 调用，而是 Ruby 的对象和方法。Rails 在后台完成这些与数据库相关的操作。（当然 Rails 也支持执行最原始的 SQL[1]语句。）
- 自动的数据模型与 HTML 转换　在系统中，对象的模型化代码和负责向用户显示这些对象的代码之间是无缝集成的。这在数据验证中更为明显，例如，如果数据模型要求用户提交电子邮件地址，但是用户提交的表单中没有该地址，Rails 将能够自动捕捉这个错误并把它放在一个用于回显给客户的变量中。
- 内建的数据模型和 Web 页面自动测试　Rails 使编写用来验证数据模型和网站上网页正确性的测试套件变得更加容易，让你相信对代码的修改将不会破坏应用程序。
- 独立于数据库的表的创建和修改　Rails 的 migrations（移植）功能能使数据创建、数据模型修改以及必要的回滚操作变得更加容易，在某种程度上，这使得在不同数据库上使用相同的数据模型成为可能。

1.1.2 不自由的生产力

我们已经对自由的生产力有了直观的认识，那么还需要解释一下什么不是自由的生产力。许多框架都带有大量内建的功能：漂亮的管理界面、奇特的基于角色的验证和授权机制，或者无需编码就能生成应用程序的能力。基于称为“脚手架”（scaffolding）的特性，能够使“表单—数据库交互”的开发变得不值一提，当然这同时也导致了声名狼藉的“15 分钟博客引擎”（http://media.rubyonrails.org/video/rails_take2_with_sound.mov）的出现。

这些生产力都不是自由的。内建的验证或者 15 分钟博客引擎可能会对创建更大的市场有利，但是我们可能会同意这种观点：不管框架多么杰出，任何重要的软件应用开发都应该花费比 15 分钟更多的时间。如果应用程序不知什么原因被已有的框架做成了这种普通的样式，那么尽管使用它。但是你可能更想要建立新的、一个在你需要的时候能帮助你的框架，而不仅仅是在你不需要的时候不妨碍你的框架。你需要一个能帮助自己认识愿景（vision）的工具，一个足够灵活可以随着愿景的变化而变化的工具，最后，这个工具将转变为一个新的框架，它让人感觉像是为你的应用程序而专门定制的。

[1] 原文注：SQL 全称为 Structured Query Language（结构化查询语言），是关系型数据库的语言。

这就是 Rails 的过人之处。Rails 非常擅长制作定制的应用程序，帮助你制作你自己想要的应用程序。如果不准备转用 Rails，可能是因为有许多可用的、精彩的公告板和博客软件。而转用 Rails 是因为使用 Rails 创建自己定制的应用程序要容易得多。

1.2 选择本书的理由

任何有教育意义的书籍都需要在纯粹的指南和纯粹的参考手册这两种极端之间寻求某种平衡。我们坚定地倾向于指南这一方面，这有许多好处。因为应用程序是真实的，可以像真正使用时那样学习 Rails，从而看到为了简化 Web 应用程序使用 Rails 编写的各种主要方法，从优秀的模型—视图—控制器（MVC）架构到杰出的嵌入式 Ruby。从模仿中还将获得好处：了解大部分最重要的思想。这是渗透性学习的结果。

在创建 RailsSpace 的过程中，我们还将看到 Rails 是如何简化用来确认应用程序完成预期任务的自动化测试的编写工作的。因为我们使用了指南的写作方法，所以能够展现如何与应用程序一起更好地开发测试，就像在实际中所做的（或者说应该做的）那样。随着应用程序的演化，可以发现拥有一个测试套件非常好，无论什么时候，只要修改或者增加一个特性，就可以运行测试以确认我们没有对应用程序的其他部分造成破坏。

当然，在有助于学习的同时，以指南形式写作的书对于查阅有关知识的帮助并不大。作为对本书的补充，我们推荐 Dave Thomas 和 David Heinemeier Hansson 所著的 *Agile Web Development with Rails*（Pragmatic Programmers 出版，中译版名为《使用 Rails 进行敏捷 Web 开发》），这是介绍 Rails 的最早的一本书。如果要学习 Ruby，我们推荐 Dave Thomas 等人所著的 *Programming Ruby, Second Edition*（Pragmatic Programmers 出版，中译版名为《Programming Ruby 中文版》）和 *The Ruby Way, Second Edition*（Hal Fulton 著，Addison-Wesley 出版，中译版名为《The Ruby Way（第二版，中文版）》）。

1.3 本书读者

本书是值得阅读的。由于本书是一本入门书籍，因此我们不假设读者拥有任何 Ruby 编程语言或者 Rails 框架的相关知识。当然，如果有这两个方面的基础知识会更好。即使你有一些 Rails 的使用经验，我们也希望你能从书中学习到一些东西。特别是因为我们使用了在 Rails 更新版本中引入的多种特性，诸如 migrations、表单生成器、REST 和 AJAX 支持。

为了从本书里获得更多有益的东西，你或许应该至少了解一种编程语言，如果熟悉面向对象编程就更有帮助了。如果有 Java 的背景可能就足够了[1]，但如果你熟悉诸如 Perl、Python 或者 PHP（还有 Ruby）当然更好，如果使用过其中一种语言制作动态网站那就再好不过了。对 JavaScript 和层叠样式单（CSS）有一定了解也不错，至少你应该熟悉使用 HTML 制作静

[1] 原文注：如果 Java 是你所了解的唯一语言，那么可能你会因为 Ruby 中缺乏静态类型而遇到困难。如果觉得自己对此感到陌生，只要做个深呼吸并且试着相信动态类型也能工作就行了。

态网页的方法。最后，我们假定读者有一定的计算机技巧，这样当我们说“下载 MySQL 并安装”时你就可以自己完成这项工作（当然需要有足够的时间和咖啡），即使你之前从未安装过一种数据库。

这些假定的必备条件实际上都不是最重要的，最重要的因素是你对学习如何使用 Rails 制作应用程序的热情。如果你对将一些绝妙的东西放到网上感到非常兴奋，那么不管你的背景如何，这本书都是为你而准备的。

1.3.1 阅读本书的方法

本书是按从头到尾的阅读顺序而设计的，如果你跟着本书的进程一起进行编码，那么本书将会发挥最大的效用，但是即使你没有坐在计算机前，这本书也会是有益和有趣的。还有，耐心是很重要的。如果你没有马上获得想学的知识，那么应继续下去，花一点时间来完全了解这些思想。你将会吃惊地看到，在某个章节开始时还令你困惑的东西，在读完这一章时对你来说就已经不在话下了。要想将自己的神经网络组织训练成 Ruby on Rails 的模式，除了练习和坚持没有什么可以替代的方法。

1.3.2 跟踪本书动态

本书相应的网站是 http://RailsSpace.com/book，在那里可以找到勘误表、示例数据以及能够下载的 RailsSpace 源代码。你还能找到针对每一章的录像，即解说本书源代码的视频。

1.4 两个 Rails 的故事

我们已经对 Rails 流行的原因做了一些分析，但是没有什么东西可以替代两个优秀的、以传统方式描述的见证性故事。这些都是我们在通往 Rails 道路上的亲身经历。

1.4.1 Aure

那是 1994 年的事情了，我和加州理工学院研究生航空实验室的朋友发现，我们拥有一个在/~user name/URLs 上提供自己内容的部门级 Web 服务器。我们开始用与自己喜爱的音乐家和飞机相关的信息替代个人网站上的内容，并相互之间开始竞争点击率。我们中的一位是该实验室的助教，他所拥有的账户是~ta。我们将这个缩写重新解释为 The Asylum（庇护所），并且停止了竞争，改成将所有的资源汇聚于此，建立一个 Web 上任何人都可以登录并且免费提供信息的疯狂地带。我们创建的应用程序中有一个公共的书签知识库（del.icio.us[1]的先驱）、

[1] 译者注：del.icio.us 是目前网络上最大的书签库网站。

一个公共的写作论坛（wiki[1]的先驱）以及一个基于 Web 的 Lite-Brite[2]模拟器和图库（一定程度上是 Flickr[3]的远祖）。

我们给予了这些网站很多的关注，很快电影制片厂、录音公司都纷纷致电实验室，雇佣我们为他们创建动态网站。当博士生导师在实验室的时候，接到环球电影公司的电话实在是件令人尴尬的事情，因此我们开展了相应的业务，并且设立了一个拥有自己电话号码的办公室。

我们花费了一点时间在用 Perl/CGI 构建定制网站的同时获到了学位（我们都设法拿到了，但花费了 Yahoo 的富翁从斯坦福退学时省下的时间[4]）。慢慢地，使用 Perl 从头开始构建网站的压力使我们感到厌倦，公司也解散了。我回顾这些网站时，意识到许多网站都非常相似，于是我开始向客户提供标准的 Web 组件。同时，Philip Greenspun 开始了 Photo.net 网站的运作，有许多人希望得到这个网站的可执行代码。Philip 召集他的朋友们来增强 Photo.net 的代码，我们就一起创立了 ArsDigita，并开放了我们的源代码，这就是 ACS（ArsDigita 社区系统），它是由标准的 Web 组件组成的，并且大部分都是使用 TCL 编写的。我们借助 ACS 从低级的开发人员中脱颖而出，并且使想要定制这个工具包的、财力充盈的公司找到了我们。遗憾的是，风险投资也看上了 ArsDigita，一旦我们收了他们的钱，他们也就具有了影响力，在 fuckedcompany.com 上可以看到后来所发生的事情。

尽管 ArsDigita 从人们的视线中消失了，但这个工具包仍然以 OpenACS 的名字在 openacs.org 中继续存在。但是作为一个自由职业者，我真的有办法让客户相信一个由已经停业的公司开发的工具包能成为长期解决方案吗？还有，我对于 TCL 编程已经相当厌恶了，所以转向了我所熟悉的最美丽的语言 Python。这也几乎意味着必须使用 Zope，而且起初 Zope 也是很杰出的。确实，Zope 仍然是开发某些应用程序的好选择。但是 Zope 也有许多不好的地方，并且我的客户渴望拥有标准的版本控制、文件系统访问以及更好的关系数据库支持等功能。

我最喜欢的一位前任雇员现在就职于加州理工学院，有一天她发给我一封电子邮件，内容是关于她考虑在下一个项目中将选择一些替代 Zope 的新兴框架。列表中包括了 Django 和 Ruby on Rails。经过几个小时的研究后，我发现 Zope 已经成为历史。开始我倾向于 Django，因为它是基于 Python 的，但是 Ruby on Rails 更进一步，而且它拥有 Django 永远不会有的东西，那就是 Rails 的创始人 David Heinemeier Hansson。根据我和 ArsDigita、Philip Greenspun 打交道的经验，我知道一个具有说服力、聪明而且坚持个人看法的中心人物对于一个框架而言有多重要。此后，在本书写作期间，DHH 对 Rails 是什么和不是什么规定了一些基本原则。

虽然 DHH 是它的“创始人”，但是 Rails 显然是 Web 应用开发革命的产物。以下是 Rails

---

[1] 译者注：wiki，中文译作“维客”、“维基百科”，是一种超文本系统。这种超文本系统支持面向社群的协作式写作，同时也包括一组支持这种写作的辅助工具。

[2] 译者注：Lite-Brite 是美国 20 世纪 80 年代流行的一种玩具，通过组合 LED 灯管来进行绘图练习。

[3] 译者注：Flickr 是一种网络相册服务。

[4] 译者注：这里指 Yahoo 创始人杨致远，他在 1995 年放弃即将完成的博士学业。

给我留下深刻印象的部分。

- 最主要的是 Ruby 开发 Web 应用程序总是需要掌握许多领域的知识，至少必须知道一些关于模板语言、数据库访问以及过程化编程的知识，但是并没有有效避免开发人员经常将精力花费在不同语言的来回切换中，特别是当多种语言混合在一个文件中时。而 Rails 将这种耗费减到最小，模板语言是 Ruby，数据访问是 Ruby（主要），过程化编程还是 Ruby。
- 改进的数据模型 我以前启动项目的方法是对于所有可能需要的东西通过头脑风暴获得，并编写一个保存数据最终状态的 SQL 数据模型。然后我必须通过编写代码来构造数据库表，并且编写相应的错误检测代码以避免无效数据进入数据库。而使用 Rails 的 migrations 机制后，可以在开发过程中逐步扩充数据模型并使用验证，还能编写几乎可以自动在 Web 应用程序里作为错误报告显示的规则。migrations 是与数据库平台无关的，所以不需要为各种可用的数据库编写新的 SQL 文件。

我还从我所喜爱的 Rails 学到了许多，但是老实说，你不需要通过一个长长的、关于 Rails 精彩特性的列表来确信它是你所需要的。实际上，如果你需要借助一个长列表，说明你可能还没有建立足够的信心。这就像我购买 2001 款的 Nissan Frontier 卡车的时候，我只是确认它可以运载我的狗，速度很快，看上去像宇宙飞船，于是马上就做出了决定：成交。

使用 Ruby on Rails 可以为客户提供完全按他们需求定制的网站，同时仍然能够满足我编程的兴趣。所有人都很高兴！

1.4.2 Michael

当我在加州理工学院研究所工作时，我的朋友 Sumit Daftura 和我一起运营了本地的 NCAA 棒球联赛信息库，这涉及一个虽然很小但是很重要的 Web 组件。Sumit 使用了一个简单的 Perl 脚本在每轮联赛后生成将在网站上显示的比赛结果。尽管这简单得有点令人可笑，但是人们都很喜欢它。2001 年，我决定进一步使用 PHP 编写一个完整的 Web 界面，完善分组输入和成绩报告功能。我判断这将迫使自己学习 Web 开发，经过很多个每天工作 20 小时以上的日子，我终于学到了。

PHP 最初看上去很不错，但是随着我的项目日益增大，PHP 也日益变得笨重。毕业之后，Sumit 和我一起开办了公司，运营了一个独特的、每周一次的虚拟体育游戏，并制作了被称为 BracketManager 的 NCAA 网站改进版。我不想在公司网站的开发上使用 PHP，所以开始到处寻找新的产品。我在博士课程中学习过 Perl 和 Python，并且对这些语言的多种框架也有所了解。

我最终决定选择用基于 Python 的 Zope，尽管它并不特别适合于我最初的目标，也就是编写大型的定制 Web 应用程序。和许多框架一样，Zope 使用缩水（watered-down）的模板语言来生成 HTML，但是许多 Web 应用都需要（包括我们的）很复杂的 HTML，因此确实需要一种功能完善的语言。我们最终选择纯粹使用 Python 来编写大部分 HTML，并使用“外部方

法”实现与 Zope 的结合；任务虽然完成了，但是却相当麻烦。我还发现 Zope 的文档参差不齐，部分原因是 Zope 的客户群相对较小。另外，Zope 对关系型数据库的支持很弱，特别是 Zope 的 MySQL 数据库适配器不支持冗长 SQL 查询的自动生成，这使我不得不用 Python 编写一个自定义的 MySQL 程序库。

尽管 Zope 有这些问题，但是 Python 远强于 PHP，所以这些麻烦还是值得的。遗憾的是，我们的梦想比赛还未进入最关键的阶段，最后一根稻草到来了，NFL 运动员联合会开始起诉（同时 MLB 运动员联合会也开始威胁）未经许可使用运动员姓名的虚拟体育游戏公司。由于无法获得许可，我们决定关闭公司。这是不幸的事情，但是这也给了我们一次放弃 Zope 代码库，转而关注曾经听说过的、名为 Ruby on Rails 的新框架的机会。我有一个心目中理想 Web 开发框架的特性清单，并且对 Rails 是否能够满足这些要求感到非常好奇。

在这段时间里，我始终保持着与 Aure Prochazka 的联系，他是个高大、安静、不修边幅的人，我们是在加州理工合唱团中相识的。他看起来对 Web 开发所知甚多，所以在我使用 PHP 的时期曾和他谈论过那种语言，他当时认为 PHP 短小精悍。一年多以后，在我们的交谈中他告诉我，他已经仔细研究了 PHP 并决定放弃它，并已经使用 Python 和 Zope 作为替代。我告诉他我也已经转到 Zope 了。大约又过了一年，我提到了对 Zope 的不满意，并且开始关注 Ruby on Rails 了。他告诉我他最近也刚刚从 Zope 转到 Rails 了，并且很喜爱这个新的框架。

接着我就开始为自己寻找与 Rails 相关的培训，后来选择了 Dave Thomas 和 Mike Clark 主办的 Pragmatic Studio 课程，尽管 Mike 努力让我相信自己并不需要这个课程，但是我最终还是花掉了自己辛苦赚来的钱。对于 Mike 不擅长说服我感到很高兴，因为我惊奇地发现心目中那个清单里列出的项目能够一一被满足：

- 精巧的编程语言，具有灵活的数据结构和强大的抽象能力：√
- 良好的文档支持，拥有相对较大（并且快速成长）的用户基础：√
- 成熟的关系数据库支持，具备较好的对象—关系映射程序库：√
- HTML 嵌入模板，希望它是基于一种全能的编程语言：√

使用 Rails 开发了几个个人项目之后，我又为朋友的公司制作了一个 Rails 演示网站，以此来证明 Rails 可以代替他们原来的 Perl 系统。现在我确信 Rails 就是我要找的框架。当 Aure 询问我是否有兴趣和他一起编写一本关于 Rails 的书籍时，我欣然接受了。

我现在不会再走回头路了，而是坚持使用 Ruby on Rails。