

Broadview<sup>®</sup>  
www.broadview.com.cn



# Ruby on Rails

## 敏捷开发 最佳实践

李刚 郭秋霞 编著



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY  
<http://www.phei.com.cn>



TP393.09/171D

2008

# *Ruby on Rails*

## 敏捷开发 最佳实践

李刚 郭秋霞 编著

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

## 内 容 简 介

Ruby On Rails 框架一经推出,立即引起 B/S 结构应用开发领域革命性的变化:开发者无需理会架构,只需要按 Rails 框架的约定向应用中填充代码,一切 OK。

为了让众多 Ruby On Rails 学习、工作者,以及准备使用 Ruby On Rails 作为开发平台的开发人员能快速掌握 Ruby On Rails 的开发,作者精心编写了本书,书中既详细讲解了 Ruby 语言的基本语法,又重点介绍了 Rails 框架相关知识点的各种用法。最后给出两个综合案例,作为 Ruby On Rails 应用开发者的参考,读者可以通过这两个案例触类旁通,解决日常开发中的问题。

本书适用于正在使用 Ruby On Rails 进行应用开发的开发人员、渴望了解 Ruby On Rails 框架的开发人员,尤其适合有初步的 Java EE 开发经验,想从 Java EE 平台过渡到 Ruby On Rails 开发平台的开发者。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

## 图书在版编目(CIP)数据

Ruby on Rails 敏捷开发最佳实践 / 李刚, 郭秋霞编著. —北京: 电子工业出版社, 2008.4  
ISBN 978-7-121-05570-6

I. R… II. ①李… ②郭… 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 (2007) 第 191767 号

责任编辑: 高洪霞

印 刷: 北京东光印刷厂

装 订: 三河市皇庄路通装订厂

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1092 1/16 印张: 43.25 字数: 1075 千字

印 次: 2008 年 4 月第 1 次印刷

印 数: 5000 册 定价: 79.80 元 (含光盘 1 张)

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 [zltz@phei.com.cn](mailto:zltz@phei.com.cn), 盗版侵权举报请发邮件至 [dbqq@phei.com.cn](mailto:dbqq@phei.com.cn)。

服务热线: (010) 88258888。

# 前言

Rails 框架从诞生的第一天开始就受到了 B/S 应用开发领域的广泛关注,经过大约几年发展, Rails 框架已经逐渐成熟稳定起来,国外已有相当多的公司开始选择从 Java EE 平台过渡到 Rails 开发平台。毕竟, Rails 平台提出了一个非常具有诱惑力的口号:开发效率是 Java EE 开发平台的 10 倍,这个口号正确与否不是笔者所关心的,但 Rails 确实具有很好的开发效率,而且很难出错:在 Java EE 开发领域,你需要掌握很多框架,详细了解每个框架的各种特性,你才可以组合出一套适合自己的开发平台。但在 Rails 领域,你无需选择,你只要按 Rails 平台的约定开发,一切都会非常合理,非常优雅。

Rails 平台不仅提供了整套的 MVC 解决方案,并且提供了丰富的代码生成器。Rails 甚至不完全是一个框架,更像一个企业开发平台,它搭建了整个应用开发的骨架,并提供了大量的代码生成器来辅助开发,开发者只需在系统生成的代码骨架上进行适当修改、定制即可完成应用的开发。选择使用 Rails 作为开发平台,仿佛上了铁轨的火车一样,火车只管向前运行,铁轨将负责把火车带到最终目的地(Rails 就是铁轨的意思)。

Rails 框架对迭代式的增量开发有很好地支持,即使系统的数据库需要重新设计、需要增加系统实体, Rails 一样能够轻松面对(本书第 2 章示范了这种过程)。迭代式的增量开发正是敏捷开发的核心之一,这也是 Rails 框架被称为敏捷开发框架的原因。不仅如此,由于客观世界的复杂性,需求的变更常常是许多开发者最头疼的事情。使用了 Rails 开发平台后一切都改变了: Rails 替你做好了拥抱改变的准备。

## 本书内容结构

本书分成 4 篇:

第 1 篇(第 1~2 章)主要介绍了 Ruby on Rails 的基本知识,包括搭建 Ruby on Rails 开发平台,第 2 章则介绍了一个 Ruby on Rails 的入门项目,让读者体验 Rails 敏捷开发,从而感受 Rails 平台的魅力。

第 2 篇(第 3~10 章)主要介绍 Ruby 语言基本知识, Rails 平台选择 Ruby 作为开发语言,而且 Rails 框架本身也是使用 Ruby 语言开发的,因此 Ruby 语言基本知识也是本书介绍的重要内容。本书详细介绍了 Ruby 语言的相关语法,包括 Ruby 面向对象特征, Ruby 的迭代器、正则表达式支持、异常处理等基础知识,也包括 Ruby 的多线程、IO 和网络通信等高级内容。

第 3 篇(第 11~16 章)主要介绍 Rails 框架,详细介绍了 Rails 的模型(M)、视图(V)和控制器(C)各部分的解决方案,也讲解了 Rails 的邮件支持和 Ajax 支持,以期读者能充分掌握利用 Ruby on Rails 开发 B/S 结构应用的相关知识。

第 4 篇(第 17~18 章)是案例部分,介绍了两个案例:在线考试系统和电子拍卖系统,两个系统分别属于电子学习领域和电子商务领域,极具代表性。案例的实体包括复杂的 1-N

多重关联映射，对于实际企业开发有很好的示范意义。不仅如此，案例还充分利用了 Rails 提供的 Ajax 支持来改善用户体验，具有很好的指导作用。

## 本书有什么特点

当《Struts 2 权威指南——基于 WebWork 核心的 MVC 开发》一书问世以来，笔者收到很多读者来信，他们告诉笔者，非常喜欢书中介绍知识的方式：简短的文字介绍知识点的原理、小巧的案例介绍知识点的用法，最后指出知识点的实际应用场景。本书也保持了这一风格，依然坚持用案例来说话，希望带给读者 Ruby on Rails 如此简单的感觉。

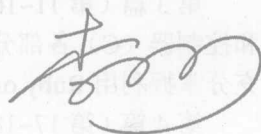
虽然笔者现在不再是一个全职开发者，但笔者一直以程序员自居：程序员喜欢简单的东西，程序员的职责就是把复杂的事情简单化，然后交给计算机去干。所以笔者坚持认为：真正属于程序员的书应该是简单的，而不是难以阅读的。如果读者阅读本书时，觉得书里介绍的知识很简单，那就可以按书里的介绍进行 Ruby on Rails 开发，那笔者撰写本书的目的也达到了。

本书由两位作者合作完成，两位作者有极好的互补关系。其中郭秋霞女士是美国 Reactrix 公司 Rails 项目组成员，具有丰富的 Ruby on Rails 实际开发经验。本书之所以得以出版，也得益于她的多次敦促。笔者也从她的工作中看到 Ruby on Rails 在国外的广泛应用，从而感受到 Ruby on Rails 在国内“山雨欲来风满楼”的气氛。郭秋霞女士以她丰富的经验为基础，完成本书的全部案例部分，希望带给读者真实企业开发的感受，丰富的实际开发经验将让本书更加实用。

笔者在 Rails 流行之初就对其抱以极大的热情（笔者曾模仿 Rails 框架设计出 Java 领域的“Rails”框架，希望有机会与读者分享）。笔者目前是一个软件设计的布道者，教过的学生有应届毕业生、有刚工作的程序员、也有具有多年开发经验的技术经理、技术总监，从而能感受到各种层次读者的学习痛苦，因此写作过程中针对性很强，希望通过本书能减少各种读者在学习过程中的痛苦。

## 本书写给谁看

读者阅读本书之前，应有一定的 B/S 应用开发基础。本书适用于正在使用 Ruby on Rails 进行应用开发的开发人员，渴望了解 Ruby on Rails 框架的开发人员。尤其适合有初步的 Java EE 开发经验，想从 Java EE 平台过渡到 Ruby on Rails 开发平台的开发者。



2008-1-1

# 目 录

## 第 1 篇 Ruby on Rails 入门

|                                   |    |
|-----------------------------------|----|
| 第 1 章 Ruby on Rails 和企业级应用开发      | 2  |
| 1.1 企业级应用简介                       | 3  |
| 1.1.1 企业级应用简介                     | 3  |
| 1.1.2 开发企业级应用面临的挑战                | 4  |
| 1.1.3 Java EE 开发平台简介              | 5  |
| 1.1.4 .NET 开发平台简介                 | 7  |
| 1.1.5 Ruby on Rails 开发平台简介        | 8  |
| 1.2 Ruby on Rails 概述              | 9  |
| 1.2.1 什么是 Ruby                    | 9  |
| 1.2.2 什么是 Rails                   | 10 |
| 1.2.3 Ruby on Rails 的发展现状         | 10 |
| 1.2.4 Ruby on Rails 与现存企业级应用平台的对比 | 11 |
| 1.3 Ruby on Rails 体系介绍            | 12 |
| 1.3.1 Ruby on Rails 架构模型          | 12 |
| 1.3.2 富领域对象                       | 14 |
| 1.3.3 持久层解决方案: ActiveRecord       | 15 |
| 1.3.4 控制器解决方案: ActionController   | 16 |
| 1.3.5 表现层技术: RHTML 页面             | 17 |
| 1.4 搭建 Ruby on Rails 开发和运行环境      | 17 |
| 1.4.1 安装 Ruby 解释器                 | 17 |
| 1.4.2 安装 Rails                    | 20 |
| 1.4.3 安装 MySQL 数据库                | 22 |
| 1.4.4 Ruby on Rails 相关开发工具        | 25 |
| 1.5 本章小结                          | 32 |
| 第 2 章 体验 Ruby on Rails 敏捷开发       | 33 |
| 2.1 创建第一个 Web 应用                  | 34 |
| 2.1.1 生成 Rails 应用及启动应用            | 34 |
| 2.1.2 配置数据库                       | 37 |
| 2.1.3 使用代码生成器生成支架                 | 38 |
| 2.2 完善第一个 Web 应用                  | 40 |

|       |          |    |
|-------|----------|----|
| 2.2.1 | 使用中文响应   | 40 |
| 2.2.2 | 改进用户注册   | 41 |
| 2.2.3 | 实现用户登录   | 45 |
| 2.3   | 应对新需求    | 49 |
| 2.3.1 | 添加 Model | 49 |
| 2.3.2 | 改进“浏览留言” | 51 |
| 2.3.3 | 改进“新增留言” | 53 |
| 2.3.4 | 改进“查看留言” | 57 |
| 2.3.5 | 改进“删除留言” | 58 |
| 2.4   | 本章小结     | 59 |

## 第 2 篇 Ruby 基本语法

|       |                 |    |
|-------|-----------------|----|
| 第 3 章 | Ruby 编程基础       | 62 |
| 3.1   | Ruby 代码基本格式     | 63 |
| 3.1.1 | 使用 Ruby 解释器     | 63 |
| 3.1.2 | Ruby 代码落脚点和程序入口 | 64 |
| 3.1.3 | Ruby 区分大小写      | 65 |
| 3.1.4 | Ruby 程序的注释      | 66 |
| 3.1.5 | Ruby 中的标识符      | 68 |
| 3.1.6 | Ruby 的垃圾回收      | 69 |
| 3.2   | 常量              | 69 |
| 3.2.1 | Ruby 是弱类型语言     | 69 |
| 3.2.2 | 普通常量            | 69 |
| 3.2.3 | 常量的范围           | 71 |
| 3.2.4 | 类名和模块名常量        | 73 |
| 3.3   | 变量及变量的作用域       | 74 |
| 3.3.1 | 变量的概念           | 74 |
| 3.3.2 | 局部变量            | 74 |
| 3.3.3 | 全局变量            | 77 |
| 3.3.4 | 实例变量            | 79 |
| 3.3.5 | 类变量             | 83 |
| 3.3.6 | 伪变量             | 85 |
| 3.4   | 基本数据类型          | 85 |
| 3.4.1 | 弱类型语言           | 86 |
| 3.4.2 | 数值型             | 86 |
| 3.4.3 | 字符串             | 89 |

|        |               |     |
|--------|---------------|-----|
| 3.4.4  | 字符串的%表示法      | 91  |
| 3.4.5  | 多行字符串的字面值     | 91  |
| 3.5    | 范围            | 94  |
| 3.6    | 正则表达式         | 97  |
| 3.6.1  | 正则表达式概述       | 97  |
| 3.6.2  | 创建正则表达式       | 97  |
| 3.6.3  | 匹配基本字符        | 99  |
| 3.6.4  | 重复            | 99  |
| 3.6.5  | 成组            | 100 |
| 3.7    | 数组            | 100 |
| 3.7.1  | 定义数组          | 100 |
| 3.7.2  | 数组的%表示方法      | 102 |
| 3.7.3  | 数组的常用方法       | 102 |
| 3.8    | Hash 对象       | 104 |
| 3.8.1  | 定义 Hash 对象    | 104 |
| 3.8.2  | 操作 Hash 对象    | 105 |
| 3.9    | 符号对象          | 107 |
| 3.10   | 运算符           | 109 |
| 3.10.1 | 算术运算符         | 110 |
| 3.10.2 | 赋值运算符         | 111 |
| 3.10.3 | 位运算符          | 111 |
| 3.10.4 | 比较运算符         | 113 |
| 3.10.5 | 逻辑运算符         | 114 |
| 3.10.6 | 三目运算符         | 116 |
| 3.10.7 | Ruby 运算符的含义   | 116 |
| 3.11   | 本章小结          | 117 |
| 第 4 章  | 表达式和语句        | 118 |
| 4.1    | 表达式           | 119 |
| 4.1.1  | 返回值的表达式       | 119 |
| 4.1.2  | 没有返回值的表达式     | 120 |
| 4.1.3  | 并行赋值          | 120 |
| 4.1.4  | 嵌套赋值          | 121 |
| 4.1.5  | BEGIN 和 END 块 | 121 |
| 4.1.6  | at_exit 块     | 122 |
| 4.2    | 顺序结构          | 123 |
| 4.3    | 分支结构          | 123 |
| 4.3.1  | if 表达式        | 123 |

|       |                |     |
|-------|----------------|-----|
| 4.3.2 | if 修饰句         | 127 |
| 4.3.3 | unless 表达式     | 127 |
| 4.3.4 | unless 修饰句     | 128 |
| 4.3.5 | case 表达式       | 129 |
| 4.4   | 循环结构           | 131 |
| 4.4.1 | while 表达式      | 131 |
| 4.4.2 | while 修饰句      | 133 |
| 4.4.3 | until 表达式      | 133 |
| 4.4.4 | until 修饰句      | 134 |
| 4.4.5 | for...in 表达式   | 135 |
| 4.4.6 | loop 表达式       | 137 |
| 4.4.7 | 使用迭代器          | 137 |
| 4.5   | 控制循环结构         | 138 |
| 4.5.1 | 使用 break 结束循环  | 138 |
| 4.5.2 | 使用 next 结束本次循环 | 139 |
| 4.5.3 | 使用 redo 重新开始循环 | 140 |
| 4.5.4 | 使用 retry 重启迭代器 | 141 |
| 4.5.5 | 使用 return 结束方法 | 142 |
| 4.6   | 深入迭代器和块        | 142 |
| 4.6.1 | 块的概述           | 143 |
| 4.6.2 | 使用块            | 143 |
| 4.6.3 | 实现迭代器          | 145 |
| 4.6.4 | 块和闭包           | 148 |
| 4.7   | 本章小结           | 150 |
| 第 5 章 | 面向对象 (上)       | 151 |
| 5.1   | 面向对象的概念        | 152 |
| 5.1.1 | 面向过程           | 152 |
| 5.1.2 | 面向对象           | 153 |
| 5.2   | 类与对象           | 153 |
| 5.2.1 | 类定义            | 153 |
| 5.2.2 | 方法定义           | 155 |
| 5.2.3 | 对象的产生与使用       | 155 |
| 5.2.4 | 类和对象的关系        | 157 |
| 5.2.5 | 动态追加类定义        | 158 |
| 5.2.6 | 内部类定义          | 158 |
| 5.2.7 | 特殊类定义          | 159 |
| 5.3   | 构造器            | 161 |

|       |                      |     |
|-------|----------------------|-----|
| 5.4   | 类的继承                 | 162 |
| 5.4.1 | 继承的特点和语法             | 162 |
| 5.4.2 | 调用父类构造器              | 165 |
| 5.4.3 | 覆盖父类的方法              | 166 |
| 5.5   | 深入方法                 | 168 |
| 5.5.1 | 方法的定义                | 168 |
| 5.5.2 | 调用方法                 | 170 |
| 5.5.3 | 方法的参数传递              | 171 |
| 5.5.4 | 使用 Hash 对象作为参数       | 174 |
| 5.5.5 | 可变长度形参的方法定义          | 175 |
| 5.5.6 | 使用数组参数               | 176 |
| 5.5.7 | 特殊方法                 | 176 |
| 5.6   | 在方法中使用块              | 177 |
| 5.6.1 | 使用 yield 调用块         | 177 |
| 5.6.2 | Proc 参数              | 179 |
| 5.6.3 | 动态执行代码               | 181 |
| 5.7   | 本章小结                 | 182 |
| 第 6 章 | 面向对象 (下)             | 183 |
| 6.1   | 类方法                  | 184 |
| 6.1.1 | 类方法的定义和使用            | 184 |
| 6.1.2 | 单例模式                 | 187 |
| 6.1.3 | 定义 method_missing 方法 | 188 |
| 6.2   | 成员变量和属性              | 189 |
| 6.2.1 | 定义实例变量               | 189 |
| 6.2.2 | 定义类变量                | 191 |
| 6.2.3 | 定义属性                 | 193 |
| 6.2.4 | 存取器                  | 196 |
| 6.2.5 | 虚拟属性                 | 198 |
| 6.3   | 弱类型语言的多态             | 198 |
| 6.3.1 | 为什么 Ruby 没有接口        | 198 |
| 6.3.2 | 鸭子类型和多态              | 199 |
| 6.4   | 访问控制                 | 203 |
| 6.5   | 模块                   | 206 |
| 6.5.1 | 定义模块                 | 206 |
| 6.5.2 | 命名空间                 | 207 |
| 6.5.3 | 混入                   | 209 |
| 6.5.4 | 混入的深入讨论              | 211 |

|       |                      |     |
|-------|----------------------|-----|
| 6.5.5 | 导入其他文件               | 213 |
| 6.6   | Ruby 的命名习惯           | 214 |
| 6.7   | 关于定义的操作              | 214 |
| 6.7.1 | 使用 alias 为定义命名别名     | 214 |
| 6.7.2 | 使用 undef 释放定义        | 215 |
| 6.7.3 | 使用 defined? 判断是否完成定义 | 216 |
| 6.8   | 本章小结                 | 217 |
| 第 7 章 | 异常处理                 | 218 |
| 7.1   | 异常概述                 | 219 |
| 7.1.1 | 了解异常                 | 219 |
| 7.1.2 | 异常机制的优势              | 220 |
| 7.2   | 异常处理流程               | 221 |
| 7.2.1 | 使用 rescue 捕捉异常       | 222 |
| 7.2.2 | 异常处理机制               | 225 |
| 7.2.3 | 使用 ensure 回收资源       | 226 |
| 7.2.4 | 使用 retry 再次执行        | 227 |
| 7.3   | rescue 修饰句           | 228 |
| 7.4   | 引发异常                 | 229 |
| 7.4.1 | 引发异常                 | 229 |
| 7.4.2 | 自定义异常类               | 230 |
| 7.4.3 | rescue 和 raise 的同时使用 | 231 |
| 7.5   | 在程序定义中使用异常处理         | 232 |
| 7.5.1 | 在类 / 模块定义中使用异常处理     | 232 |
| 7.5.2 | 在方法定义中使用异常处理         | 235 |
| 7.6   | 本章小结                 | 236 |
| 第 8 章 | 多线程                  | 237 |
| 8.1   | 线程概述                 | 238 |
| 8.1.1 | 线程和进程                | 238 |
| 8.1.2 | 多线程的优势               | 239 |
| 8.2   | 创建多线程                | 240 |
| 8.2.1 | 用 Thread 类创建线程       | 240 |
| 8.2.2 | Ruby 的线程运行方式         | 243 |
| 8.3   | 控制线程                 | 243 |
| 8.3.1 | join 线程              | 244 |
| 8.3.2 | 控制线程的方法              | 245 |
| 8.3.3 | 线程状态                 | 248 |

|        |                           |     |
|--------|---------------------------|-----|
| 8.4    | 深入线程                      | 249 |
| 8.4.1  | 线程和变量                     | 249 |
| 8.4.2  | 线程和异常                     | 252 |
| 8.5    | 多线程的同步                    | 254 |
| 8.5.1  | 线程安全问题                    | 254 |
| 8.5.2  | 使用 Mutex 实现互斥             | 256 |
| 8.5.3  | 使用 ConditionVariable 实现同步 | 258 |
| 8.6    | 本章小结                      | 260 |
| 第 9 章  | 输入 / 输出                   | 261 |
| 9.1    | 基本输入 / 输出                 | 262 |
| 9.1.1  | 基本输出                      | 262 |
| 9.1.2  | 基本输入                      | 265 |
| 9.2    | IO 类简介                    | 267 |
| 9.3    | 文件访问                      | 267 |
| 9.3.1  | 调用系统命令                    | 267 |
| 9.3.2  | 打开/关闭文件                   | 268 |
| 9.3.3  | 读文件                       | 271 |
| 9.3.4  | 使用迭代读文件                   | 274 |
| 9.3.5  | 写文件                       | 275 |
| 9.3.6  | 追加文件                      | 276 |
| 9.4    | 本章小结                      | 276 |
| 第 10 章 | 网络编程                      | 277 |
| 10.1   | 网络编程的基础知识                 | 278 |
| 10.1.1 | 网络基础知识                    | 278 |
| 10.1.2 | IP 地址和端口号                 | 280 |
| 10.2   | Ruby 语言的编程支持              | 281 |
| 10.2.1 | 什么是 Socket                | 281 |
| 10.2.2 | Ruby 的 Socket 支持          | 282 |
| 10.2.3 | 面向应用层的网络编程                | 283 |
| 10.3   | 基于 TCP 协议的网络编程            | 284 |
| 10.3.1 | TCP 协议基础                  | 284 |
| 10.3.2 | 使用 TCPServer 创建 TCP 服务器端  | 285 |
| 10.3.3 | 使用 TCPSocket 创建 TCP 客户端   | 286 |
| 10.3.4 | 加入多线程                     | 287 |
| 10.4   | 基于 UDP 协议的网络编程            | 289 |
| 10.4.1 | UDP 协议基础                  | 289 |

|        |                   |     |
|--------|-------------------|-----|
| 10.4.2 | 使用 UDPSocket 发送数据 | 290 |
| 10.4.3 | 使用 UDPSocket 接收数据 | 291 |
| 10.5   | 面向应用的网络支持         | 292 |
| 10.6   | 本章小结              | 294 |

## 第 3 篇 Rails 框架

|        |                                   |     |
|--------|-----------------------------------|-----|
| 第 11 章 | Rails 框架入门                        | 296 |
| 11.1   | Rails 简介                          | 297 |
| 11.1.1 | Rails 的优势和特点                      | 297 |
| 11.1.2 | Rails 应用的目录结构                     | 299 |
| 11.1.3 | Rails 的代码生成器                      | 301 |
| 11.2   | Rails 的配置和约定                      | 302 |
| 11.2.1 | 配置 Rails 的运行环境                    | 302 |
| 11.2.2 | Rails 的命名约定                       | 303 |
| 11.3   | 对象关系映射和活动记录集                      | 304 |
| 11.3.1 | ORM 的概念和优势                        | 304 |
| 11.3.2 | 活动记录集的概念和优势                       | 305 |
| 11.4   | 活动记录集入门                           | 305 |
| 11.4.1 | 活动记录集的基本映射                        | 306 |
| 11.4.2 | 表对应类                              | 307 |
| 11.4.3 | 列对应属性                             | 310 |
| 11.4.4 | 访问属性                              | 315 |
| 11.4.5 | 映射主键                              | 317 |
| 11.4.6 | 映射复合属性                            | 322 |
| 11.5   | 连接数据库                             | 326 |
| 11.5.1 | 通过配置文件指定连接                        | 327 |
| 11.5.2 | 通过代码指定连接                          | 328 |
| 11.5.3 | 调用 establish_connection 方法指定数据库连接 | 330 |
| 11.6   | 本章小结                              | 332 |
| 第 12 章 | Rails 的持久层解决方案                    | 333 |
| 12.1   | 插入记录                              | 334 |
| 12.1.1 | 调用 new()方法和 save()方法增加记录          | 334 |
| 12.1.2 | 调用 create()方法增加记录                 | 338 |
| 12.2   | 查询记录                              | 341 |
| 12.2.1 | 强大的 find()方法                      | 341 |
| 12.2.2 | 根据主键查询                            | 347 |

|        |                                                                          |     |
|--------|--------------------------------------------------------------------------|-----|
| 12.2.3 | 根据普通属性进行查询                                                               | 349 |
| 12.2.4 | 使用 <code>find_by_sql</code> 进行查询                                         | 352 |
| 12.2.5 | 使用 <code>count</code> 或 <code>count_by_sql</code> 统计记录数                  | 355 |
| 12.2.6 | Rails 所支持的运算符                                                            | 357 |
| 12.3   | 更新记录                                                                     | 360 |
| 12.3.1 | 直接更新特定记录                                                                 | 360 |
| 12.3.2 | 使用 <code>update_attribute</code> 或 <code>update_attributes</code> 方法更新记录 | 361 |
| 12.3.3 | 使用 <code>update</code> 方法更新记录                                            | 363 |
| 12.3.4 | 使用 <code>update_all</code> 方法更新记录                                        | 366 |
| 12.4   | 删除记录                                                                     | 367 |
| 12.4.1 | 使用 <code>delete</code> 根据主键删除                                            | 367 |
| 12.4.2 | 使用 <code>delete_all</code> 根据条件子句删除                                      | 369 |
| 12.4.3 | 使用 <code>destroy</code> 根据主键删除                                           | 370 |
| 12.4.4 | 使用 <code>destroy_all</code> 根据条件子句删除                                     | 371 |
| 12.5   | 关联关系                                                                     | 372 |
| 12.5.1 | 数据库中表的关联关系                                                               | 372 |
| 12.5.2 | 数据表中建立关联                                                                 | 374 |
| 12.5.3 | 一对一关联                                                                    | 376 |
| 12.5.4 | 一对多关联                                                                    | 382 |
| 12.5.5 | 多对多关联                                                                    | 387 |
| 12.6   | 深入关联关系                                                                   | 391 |
| 12.6.1 | 自关联                                                                      | 391 |
| 12.6.2 | 多重关联                                                                     | 394 |
| 12.7   | 继承关系                                                                     | 397 |
| 12.7.1 | 单表继承关系                                                                   | 397 |
| 12.7.2 | 多态关联                                                                     | 401 |
| 12.8   | 事务处理                                                                     | 405 |
| 12.9   | 模型校验                                                                     | 407 |
| 12.9.1 | 模型校验的概念                                                                  | 408 |
| 12.9.2 | 常用的模型校验器                                                                 | 410 |
| 12.9.3 | 校验失败后的处理                                                                 | 412 |
| 12.10  | 回调方法                                                                     | 417 |
| 12.11  | 本章小结                                                                     | 419 |

## 第 13 章 Rails 的控制器解决方案 420

|        |               |     |
|--------|---------------|-----|
| 13.1   | 控制器入门         | 421 |
| 13.1.1 | MVC 简介        | 421 |
| 13.1.2 | Rails 应用的 MVC | 422 |

|        |                          |     |
|--------|--------------------------|-----|
| 13.1.3 | ApplicationController 简介 | 424 |
| 13.1.4 | 过滤器                      | 426 |
| 13.1.5 | 通过 Filter 设置中文响应         | 430 |
| 13.1.6 | 使用代码生成器创建控制器             | 432 |
| 13.2   | 处理用户请求                   | 432 |
| 13.2.1 | 路由请求                     | 432 |
| 13.2.2 | 指定 URL 规则                | 436 |
| 13.2.3 | 命名路由                     | 437 |
| 13.2.4 | 处理无法识别的 URL              | 438 |
| 13.2.5 | 调用模型方法                   | 438 |
| 13.3   | 生成响应                     | 442 |
| 13.3.1 | 提交数据给视图                  | 442 |
| 13.3.2 | 重定向                      | 451 |
| 13.3.3 | 使用二进制流响应                 | 453 |
| 13.4   | 跟踪用户状态                   | 455 |
| 13.4.1 | Session 和 Cookie         | 455 |
| 13.4.2 | 操作 Session               | 456 |
| 13.4.3 | 操作 Cookie                | 459 |
| 13.5   | 本章小结                     | 462 |
| 第 14 章 | Rails 的表现层解决方案           | 463 |
| 14.1   | 视图层解决方案                  | 464 |
| 14.1.1 | 模板                       | 464 |
| 14.1.2 | RHTML 模板                 | 466 |
| 14.1.3 | RXML 模板                  | 469 |
| 14.1.4 | RJS 模板                   | 471 |
| 14.2   | 常用的工具方法                  | 473 |
| 14.2.1 | 基本工具方法                   | 473 |
| 14.2.2 | 格式化输出方法                  | 476 |
| 14.2.3 | 链接工具方法                   | 479 |
| 14.2.4 | 控制分页方法                   | 482 |
| 14.3   | 表单相关的工具方法                | 485 |
| 14.3.1 | 基本表单工具方法                 | 485 |
| 14.3.2 | 高级表单工具方法                 | 487 |
| 14.3.3 | 时间 / 日期列表框               | 490 |
| 14.3.4 | 文件上传                     | 492 |
| 14.4   | 重构模板                     | 499 |
| 14.4.1 | 使用 Layout 页面装饰           | 499 |

|               |                            |            |
|---------------|----------------------------|------------|
| 14.4.2        | 使用局部模板                     | 504        |
| 14.4.3        | 使用 Helper                  | 506        |
| 14.4.4        | 使用组件                       | 508        |
| 14.4.5        | 使用缓存                       | 511        |
| 14.5          | 本章小结                       | 515        |
| <b>第 15 章</b> | <b>Rails 的邮件抽象层</b>        | <b>516</b> |
| 15.1          | 电子邮件简介                     | 517        |
| 15.1.1        | SMTP 协议简介                  | 517        |
| 15.1.2        | POP3 协议简介                  | 517        |
| 15.1.3        | 电子邮件的用处                    | 518        |
| 15.2          | Ruby 的邮件支持                 | 518        |
| 15.2.1        | 使用 SMTP 发送邮件               | 519        |
| 15.2.2        | 使用 POP3 接收邮件               | 521        |
| 15.3          | Rails 提供的邮件支持              | 523        |
| 15.3.1        | Rails 邮件支持核心               | 523        |
| 15.3.2        | 在 Rails 应用中发送邮件            | 526        |
| 15.3.3        | 发送 HTML 格式邮件               | 531        |
| 15.3.4        | 发送带附件的邮件                   | 532        |
| 15.4          | 注册激活系统                     | 533        |
| 15.4.1        | 基本注册功能                     | 534        |
| 15.4.2        | 发送激活邮件                     | 536        |
| 15.4.3        | 处理激活                       | 537        |
| 15.4.4        | 处理登录                       | 538        |
| 15.5          | 本章小结                       | 539        |
| <b>第 16 章</b> | <b>Rails 和 Ajax</b>        | <b>540</b> |
| 16.1          | Ajax 入门                    | 541        |
| 16.1.1        | Ajax 的含义简介                 | 541        |
| 16.1.2        | 传统的 B/S 结构应用的缺点            | 542        |
| 16.1.3        | Ajax 的革命性进步                | 542        |
| 16.1.4        | Ajax 的关键技术                 | 544        |
| 16.2          | 简单 Ajax 体验                 | 546        |
| 16.2.1        | Ajax 流程                    | 546        |
| 16.2.2        | 创建 XMLHttpRequest 对象       | 547        |
| 16.2.3        | 实现 Ajax 交互                 | 547        |
| 16.3          | Rails 的 Ajax 支持            | 550        |
| 16.4          | Ajax 核心模块: PrototypeHelper | 552        |

|         |                                                |     |
|---------|------------------------------------------------|-----|
| 16.4.1  | 使用 link_to_remote 发送异步请求                       | 552 |
| 16.4.2  | 使用 form_remote_tag 异步提交表单                      | 557 |
| 16.4.3  | 使用 submit_to_remote 异步提交表单                     | 560 |
| 16.4.4  | 使用 periodically_call_remote 周期性发送异步请求          | 561 |
| 16.4.5  | 使用 evaluate_remote_response 执行远程 JavaScript 脚本 | 563 |
| 16.4.6  | 使用 observe_field 监测某个表单域                       | 564 |
| 16.4.7  | 使用 auto_complete_field 实现自动完成                  | 566 |
| 16.4.8  | 使用 text_field_with_auto_complete 生成自动完成文本框     | 569 |
| 16.4.9  | 使用 auto_complete_result 生成自动完成响应               | 570 |
| 16.4.10 | 使用 remote_function 自定义请求的发送时机                  | 572 |
| 16.5    | Ajax 的效果模块: ScriptaculousHelper                | 573 |
| 16.5.1  | 拖动效果                                           | 573 |
| 16.5.2  | 使用 sortable_element 排列列表项                      | 577 |
| 16.5.3  | 视觉效果                                           | 578 |
| 16.6    | 本章小结                                           | 580 |

## 第 4 篇 实例篇

|        |               |     |
|--------|---------------|-----|
| 第 17 章 | 在线考试系统        | 582 |
| 17.1   | 应用背景和需求分析     | 583 |
| 17.1.1 | 系统应用背景        | 583 |
| 17.1.2 | 系统需求分析        | 584 |
| 17.1.3 | 系统 ER 图及数据库设计 | 585 |
| 17.2   | 试题数据管理的实现     | 587 |
| 17.2.1 | 系统的中文化        | 588 |
| 17.2.2 | 页面装饰          | 590 |
| 17.2.3 | 添加考试类型        | 592 |
| 17.2.4 | 浏览全部试题        | 594 |
| 17.2.5 | 添加试题          | 597 |
| 17.3   | 考生数据管理的实现     | 600 |
| 17.3.1 | 浏览全部考生        | 600 |
| 17.3.2 | 删除考生          | 602 |
| 17.3.3 | 添加考生数据        | 603 |
| 17.4   | 考生前台考试的实现     | 606 |
| 17.4.1 | 考生登录          | 606 |
| 17.4.2 | 选择考试类型和进入考试   | 610 |
| 17.4.3 | 提交试题答案        | 616 |