Extracted from:

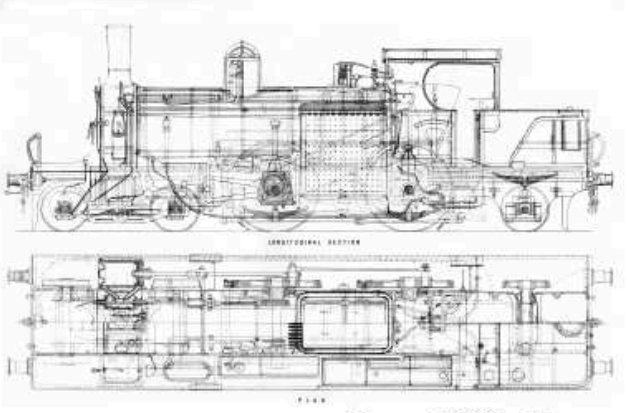
Crafting Rails Applications

Expert Practices for Everyday Rails Development

The Pragmatic Programmers

Crafting Rails Applications

Expert Practices for Everyday Rails Development



José Valim

edited by Brian P. Hogan

The Facets of Ruby Series

Preface

Rails 3 is so much more than the next iteration of an excellent web development framework.

When Rails was first released in 2004, it revolutionized how web development was done embracing concepts like DRY (Don't Repeat Yourself) and "convention over configuration". As Rails gained momentum, the conventions that were making things work so well on the golden path started to get in the way of developers that had the urge to extend how Rails behaves or even replace whole components.

Some developers felt that using Datamapper instead of Active Record was a better fit. Other developers turned to MongoDB and other non-relational databases instead, but still wanted to use their favorite web framework. Then there are those developers that prefer RSpec to Test::Unit. These developers hacked, cobbled, or monkey-patched solutions together to accomplish their goals because previous versions of Rails did not provide a solid API nor the modularity required to make these changes in a clean, maintainable fashion. Rails 3 significantly changes this game by exposing a set of more robust, modular and performant APIs.

This book guides you through these new APIs through practical examples. In each chapter, we will use test-driven development to build a Rails extension or application that covers new Rails 3 features and how these features fit in the Rails 3 architecture. By the time you finish this book, you will understand Rails better and be more productive while writing more modular and faster Rails applications.

Who should read this book?

If you're an intermediate or advanced Rails developer looking to dig deeper and make the Rails framework work for you, this is for you. We'll go beyond the basics of Rails; instead of showing how Rails lets you use its built-in features to render HTML or XML from a controller,

we'll show you how the render method works so you can customize it to accept custom options, like :pdf.

Rails Versions

All projects in *Crafting Rails Applications* were developed and tested against Rails 3.0.3. Future stable releases, like Rails 3.0.4, 3.0.5 and so forth, should be suitable as well. You can check your Rails version with the following command:

```
rails -v
```

And use gem install to get the most appropriate version:

```
gem install rails -v 3.0.3
```

This book also has excerpts from Rails source code. All these excerpts were extracted from Rails 3.0.3.

All of the projects we'll build in this book should be compatible with Rails 3.1. In case we have small compatibility issues and deprecations, they will be posted in the online forum at the book's web site.[1]

Note for Windows developers

Some chapters have dependencies that rely on C extensions. These dependencies install fine in UNIX systems, but Windows developers need the *DevKit*[2], a toolkit that enables you to build many of the native C/C++ extensions available for Ruby.

Download and installation instructions are available at http://rubyinstaller.org/downloads/.

What is in the book?

We'll explore the inner-workings of Rails across seven chapters.

In Chapter 1, *Creating our own renderer*, on page 15, we will introduce *Enginex*[3], a tool used throughout this book to create Rails extensions, and customize render to accept :pdf as an option with a behavior we will define. This chapter starts a series of discussion about Rails' rendering stack.

1. http://www.pragprog.com/titles/jvrails/
2. http://rubyinstaller.org/downloads/
3. http://github.com/josevalim/enginex

In Chapter 2, *Building models with Active Model*, on page 34, we will take a look at Active Model and its modules as we create an extension called *Mail Form* that receives data through a form and sends it to a pre-configured e-mail.

In Chapter 3, *Retrieving view templates from custom stores*, on page 56, we will revisit the Rails rendering stack and customize it to read templates from a database instead of the filesystem. At the end of the chapter, we will learn how to build faster controllers using Rails 3's modularity.

In Chapter 4, *Sending multipart e-mails with custom template handlers*, on page 78, we will create a new template handler (like ERb and Haml) on top of *Markdown*[4], We'll then create new generators and seamlessly integrate them into Rails.

In Chapter 5, *Subscribing and publishing application events with Rails Engines*, on page 101, we will build a Rails Engine that stores all SQL queries executed by our application in a MongoDB database and expose them for further analysis through a web interface. We will also see how we can use Ruby's Thread and Queue classes in the Ruby Standard Library to do the asynchronous processing;

In Chapter 6, *Writing DRY controllers with Responders and Generators*, on page 124, we will study Rails 3's responders and how we can use them to encapsulate controllers behavior, making our controllers simpler and applications more modular. We will then extend Rails responders to add HTTP Cache and internationalized Flash messages by default. At the end of the chapter, we will learn how to customize Rails' scaffold generators for enhanced productivity.

Finally, in Chapter 7, *Translating applications using Key-Value backends*, on page 150, we will learn about the I18n and customize it to read and store translations in a Redis data store. We will create an application that uses Sinatra as a Rails extension so we can modify these translations from Redis through a web interface. We will protect this translation interface using *Devise*[5] and show *Capybara's*[6] flexibility to write integration tests for different browsers.

4. http://daringfireball.net/projects/markdown
5. http://github.com/plataformatec/devise
6. http://github.com/jnicklas/capybara

How to read this book

We'll build a project from scratch in each chapter. While these projects do not depend on each other, most of the discussions in each chapter depend on what you learned previously. For example, in Chapter 1, *Creating our own renderer*, on the following page we discuss Rails' rendering stack, then we take this discussion further in Chapter 3, *Retrieving view templates from custom stores*, on page 56 and finish it in Chapter 4, *Sending multipart e-mails with custom template handlers*, on page 78. In other words, you can skip around but to get the big picture, you should read the chapters in the order they are presented.

Online Resources

The book's website[7] has links to an interactive discussion forum as well as errata for the book. You'll also find the source code for all the projects we build. Readers of the eBook can click on the gray box above the code excerpts to download that snippet directly

If you find a mistake, please create an entry on the Errata page so we can get it addressed. If you have an electronic copy of this book, there are links in the footer of each page that you can use to easily submit errata to us.

Let's get started by creating a Rails extension that customizes the `render` method so we can learn how Rails' rendering stack works.

José Valim
December, 2010
`jose.valim@plataformatec.com.br`

7. http://www.pragprog.com/titles/jvrails/

The Pragmatic Bookshelf

The Pragmatic Bookshelf features books written by developers for developers. The titles continue the well-known Pragmatic Programmer style and continue to garner awards and rave reviews. As development gets more and more difficult, the Pragmatic Programmers will be there with more titles and products to help you stay on top of your game.

Visit Us Online

Home Page for Crafting Rails Applications
http://pragprog.com//titles/jvrails
Source code from this book, errata, and other resources. Come give us feedback, too!

Register for Updates
http://pragprog.com/updates
Be notified when updates and new books become available.

Join the Community
http://pragprog.com/community
Read our weblogs, join our online discussions, participate in our mailing list, interact with our wiki, and benefit from the experience of other Pragmatic Programmers.

New and Noteworthy
http://pragprog.com/news
Check out the latest pragmatic developments, new titles and other offerings.

Buy the Book

If you liked this eBook, perhaps you'd like to have a paper copy of the book. It's available for purchase at our store: pragprog.com//titles/jvrails.

Contact Us

Online Orders:	www.pragprog.com/catalog
Customer Service:	support@pragprog.com
Non-English Versions:	translations@pragprog.com
Pragmatic Teaching:	academic@pragprog.com
Author Proposals:	proposals@pragprog.com
Contact us:	1-800-699-PROG (+1 919 847 3884)