Sinatra: Up and Running

Take advantage of Sinatra, the Ruby-based web application library and domain-specific language used by GitHub, LinkedIn, Engine Yard, and other prominent organizations. With this concise book, you will quickly gain working knowledge of Sinatra and its minimalist approach to building both standalone and modular web applications.

Sinatra serves as a lightweight wrapper around Rack middleware, with syntax that maps closely to functions exposed by HTTP verbs, making it ideal for web services and APIs. If you have experience building applications with Ruby, you'll quickly learn language fundamentals and under-the-hood techniques as you explore practical examples. Then you'll get hands-on experience with Sinatra by building your own blog engine.

- Learn Sinatra's core concepts, and get started by building a simple application
- Create views, manage sessions, and work with Sinatra route definitions
- Become familiar with the language's internals, and take a closer look at Rack
- Use different subclass methods for building flexible and robust architectures
- Put Sinatra to work: build a blog that takes advantage of service hooks provided by the GitHub API

Purchase the ebook edition of this O'Reilly title at oreilly.com and get free updates for the life of the edition. Our ebooks are optimized for several electronic formats, including PDF, EPUB, Mobi, APK, and DAISY—all DRM-free.
# Table of Contents

**Preface** ................................................................. vii

1. **Taking the Stage** ...................................................... 1  
   - Characteristics of Sinatra ........................................ 2  
   - Is It a Framework? .............................................. 2  
   - Does It Implement MVC? ....................................... 2  
   - Who’s Using It? .................................................. 3  
   - What Does a Production Project Look Like? .................... 3  
   - What’s the Catch? ............................................... 4  
   - Are These Skills Transferrable? ............................... 4  
   - Installation ..................................................... 4  
   - Thin ................................................................. 4  
   - Up and Running .................................................. 5  
     - Breaking Down the Syntax .................................. 6  
     - Testing with Telnet ......................................... 8  
   - Rock, Paper, Scissors or “The Shape of Things to Come” .... 10  
   - Summary ......................................................... 13

2. **Fundamentals** .......................................................... 15  
   - Routing ............................................................ 15  
     - Hypertext Transfer Protocol ............................... 16  
     - Verbs ............................................................ 16  
     - Common Route Definition .................................. 18  
     - Many URLs, Similar Behaviors ............................. 19  
     - Routes with Parameters .................................... 20  
     - Routes with Query String Parameters ..................... 21  
     - Routes with Wildcards ....................................... 21  
     - The First Sufficient Match Wins ............................ 21  
     - Routes with Regular Expressions .......................... 22  
     - Halting a Request ............................................ 23  
     - Passing a Request ........................................... 23
## Table of Contents

**Redirecting a Request**  
Static Files  
Views  
   - Inline Templates  
   - External View Files  
   - Passing Data into Views  
Filters  
Handling Errors  
   - 404 Not Found  
   - 500 Internal Server Error  
Configuration  
HTTP Headers  
   - The headers Method  
   - Exploring the request Object  
Caching  
   - Setting Headers Manually  
   - Settings Headers via expires  
   - ETags  
Sessions  
   - Destroying a Session  
Cookies  
Attachments  
Streaming  
   - Keeping the Connection Open  
   - Finite Streaming  
Summary  

### 3. A Peek Behind the Curtain

- Application and Delegation  
  - The Inner Self  
  - Where Does get Come From?  
  - Exploring the Implementation  
Helpers and Extensions  
  - Creating Sinatra Extensions  
  - Helpers  
  - Combining Helpers and Extensions  
Request and Response  
  - Rack  
  - Sinatra Without Sinatra  
Rack It Up  
  - Middleware  
  - Sinatra and Middleware  
Dispatching
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Modular Applications</td>
<td>67</td>
</tr>
<tr>
<td>Subclassing Sinatra</td>
<td>68</td>
</tr>
<tr>
<td>Running Modular Applications</td>
<td>68</td>
</tr>
<tr>
<td>About Settings</td>
<td>71</td>
</tr>
<tr>
<td>Subclassing Subclasses</td>
<td>73</td>
</tr>
<tr>
<td>Dynamic Subclass Generation</td>
<td>76</td>
</tr>
<tr>
<td>Better Rack Citizenship</td>
<td>78</td>
</tr>
<tr>
<td>Chaining Classes</td>
<td>78</td>
</tr>
<tr>
<td>On Return Values</td>
<td>82</td>
</tr>
<tr>
<td>Using Sinatra as Router</td>
<td>83</td>
</tr>
<tr>
<td>Extensions and Modular Applications</td>
<td>85</td>
</tr>
<tr>
<td>Helpers</td>
<td>85</td>
</tr>
<tr>
<td>Extensions</td>
<td>86</td>
</tr>
<tr>
<td>Summary</td>
<td>87</td>
</tr>
<tr>
<td>5. Hands On: Your Own Blog Engine</td>
<td>89</td>
</tr>
<tr>
<td>Workflow Concept</td>
<td>89</td>
</tr>
<tr>
<td>File-Based Posts</td>
<td>89</td>
</tr>
<tr>
<td>Git for Distribution</td>
<td>90</td>
</tr>
<tr>
<td>Semistatic Pages</td>
<td>91</td>
</tr>
<tr>
<td>The Implementation</td>
<td>92</td>
</tr>
<tr>
<td>Displaying Blog Posts</td>
<td>92</td>
</tr>
<tr>
<td>Git Integration</td>
<td>96</td>
</tr>
<tr>
<td>Glueing Everything Together</td>
<td>99</td>
</tr>
<tr>
<td>Summary</td>
<td>103</td>
</tr>
</tbody>
</table>
When people speak of Ruby web development, it has historically been in reference to the opinionated juggernaut that is Rails. This is certainly not an unfounded association; Hulu, Yellow Pages, Twitter, and countless others have relied on Rails to power their (often massive) web presences, and Rails facilitates that process with zeal.

Why, then, are people so interested in Sinatra, the tiny little domain-specific language that could?

Rails was a breath of fresh air to many developers exhausted by the “old ways”; Sinatra enters the arena with a similar game-changer: a beautifully minimalistic, “I’ll get out of your way” approach. No generators, no complex folder hierarchies, and a brief yet expressive syntax that maps closely to the functionality exposed by the Hypertext Transfer Protocol verbs.

In short, Sinatra is for classy web development.

Our goal is to provide the core concepts and accompanying examples to help you feel comfortable using Sinatra as quickly as possible. By the end, you should have a working knowledge of Sinatra and how it fits into the larger Ruby web development ecosystem. You should know when Sinatra will get the job done quickly and when it might be better to lean on Rails, Padrino, or similar frameworks. You should also have a better sense of the internals of Sinatra, as well as the Rack specification and accompanying gem.

No worries, we won’t short-change you on the reference aspects; you can certainly return to this book to recall how to perform daily tasks without excessive searching.

With that said, let’s get you up and running.
Who This Book Is For

*Sinatra: Up and Running* is for developers with some Ruby experience under their belt, and ideally some web development experience as well. Some concepts that are core to web development (the HTTP specification, HTML, CSS, etc.) are critical to understanding how to be productive with Sinatra; we recommend that you have at least a passing familiarity with these concepts to make the experience a little easier.

If you’ve written some web applications before but not specifically in Ruby, that’s no problem. Our discussion of other tools is primarily limited to comparing and contrasting with how Sinatra does things.

Our plan is to try to address the needs of several distinct camps of readers: those with a Ruby web development background in Rails but no experience with Sinatra, as well those who are familiar with Sinatra but would like a tour of its internals and philosophy. Where possible, we’d also like to help bring developers without direct web experience into the fold. Pretty tricky if you think about it, but we’ll do our best to speak to all the seats in the house by the conclusion.

Given these stated goals, we’ve divided the materials into three sections. The beginning of the book focuses on the bare minimum you need to know to work with Sinatra. Here you’ll find the fundamentals, such as how to craft routes, manage sessions, create views, and so on. Immediately afterward, we will lift the veil and examine some of the techniques behind the scenes, which will open up a world of possibilities for implementation and integration. Finally, we will wrap up the discussion with some practical applications, including developing a GitHub-powered blog.

We’ve also tried to inject as much related information as possible for the various topics covered within, ranging from gotchas to other resources where one could explore subtopics in greater depth.

One aside: if you encounter a section explaining information you’re already well-versed in, please bear with us as other readers may benefit from the discussion. We strive to keep the pace brisk, but we’d prefer not to leave any folks out.

How This Book Is Organized

*Sinatra: Up and Running* is organized as follows.

The Basics

Chapter 1, *Taking the Stage*, serves as a high-level introduction to some of the core concepts in Sinatra. It also discusses how to install the Sinatra gem, and walks through the creation of a simple application.
Chapter 2, *Fundamentals*, covers the different features of Sinatra, such as route definitions, creating views, managing sessions, and so on. It also serves as something of a reference chapter, with each topic discussed in granular fashion.

If you’ve already built some Sinatra applications of your own and are fairly comfortable doing so, you can likely just skim through Chapters 1 and 2, although the newest release of Sinatra (version 1.3.1) contains a number of changes that are worth exploring (including support for the HTTP PATCH verb, streaming, etc.).

**Digging Deeper**

In Chapter 3, *A Peek Behind the Curtain*, we discuss the internals of Sinatra and its implementation; this includes coverage of Rack, building middleware, and other topics that clarify what really happens under the hood.

Chapter 4, *Modular Applications*, covers the various approaches available for subclassing Sinatra, allowing you to create significantly more flexible and robust architectures.

If you’ve built some Sinatra applications but have never really explored the source code (or written Rack applications directly), this section will help to flesh out your knowledge. Understanding the modular application approach is critical to taking full advantage of what Sinatra offers.

**Hands On**

In Chapter 5, *Your Own Blog Engine*, we put the theory into application and create a Markdown-powered blog that takes advantage of the service hooks provided by the GitHub API.

**Conventions Used in This Book**

The following typographical conventions are used in this book:

*Italic*
- Indicates new terms, URLs, email addresses, filenames, and file extensions.

*Constant width*
- Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords.

*Constant width bold*
- Shows commands or other text that should be typed literally by the user.
Constant width italic

Shows text that should be replaced with user-supplied values or by values determined by context.

This icon signifies a tip, suggestion, or general note.

This icon indicates a warning or caution.

Using Code Examples

This book is here to help you get your job done. In general, you may use the code in this book in your programs and documentation. You do not need to contact us for permission unless you’re reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing a CD-ROM of examples from O’Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product’s documentation does require permission.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: “Sinatra: Up and Running by Alan Harris and Konstantin Haase (O’Reilly). Copyright 2012 O’Reilly Media, Inc., 978-1-449-30423-2.”

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at permissions@oreilly.com.

Safari® Books Online

Safari Books Online is an on-demand digital library that lets you easily search over 7,500 technology and creative reference books and videos to find the answers you need quickly.

With a subscription, you can read any page and watch any video from our library online. Read books on your cell phone and mobile devices. Access new titles before they are available for print, and get exclusive access to manuscripts in development and post feedback for the authors. Copy and paste code samples, organize your favorites, download chapters, bookmark key sections, create notes, print out pages, and benefit from tons of other time-saving features.
About the Authors

Alan Harris is a software engineer with a decade of professional experience, and author of several books on software development spanning multiple platforms and languages. He has delivered numerous scalable, elegant solutions for companies ranging from non-profits to military subcontractors; he has also been a featured contributor in the developerWorks community. He currently works and resides in the DC area.

Konstantin Haase, as a current maintainer of Sinatra, is an Open Source developer by heart. Ruby has become his language of choice since 2005. He actively participates in the Ruby community and regularly contributes to different widespread projects, like Rubinius and Rack. In 2010, he successfully took part in the Ruby Summer Of Code, working on Rails internals. Haase is currently studying IT Systems Engineering at the Hasso Plattner Institute in Potsdam, Germany, and works part time as a software engineer at finnlabs in Berlin.