Extracted from:

The dRuby Book

Distributed and Parallel Computing with Ruby

The Pragmatic Bookshelf

Dallas, Texas • Raleigh, North Carolina

The Pragmatic Programmers

The dRuby Book

Distributed and Parallel Computing with Ruby

Masatoshi Seki

Translated by Makoto Inoue

Foreword by Yukihiro "Matz" Matsumoto

Edited by Susannah Davidson Pfalzer

The Facets of Ruby Series

The team that produced this book includes:

Susannah Pfalzer (editor)
Potomac Indexing, LLC (indexer)
Kim Wimpsett (copyeditor)
David J Kelly (typesetter)
Janet Furlow (producer)
Juliet Benda (rights)
Ellie Callahan (support)

Original Japanese edition:
“dRuby niyoru Bunsan Web Programming” by Masatoshi Seki
Copyright © 2005. Published by Ohmsha, Ltd

Printed in the United States of America.
ISBN-13: 978-1-934356-93-7
Encoded using the finest acid-free high-entropy binary digits.
Book version: P1.0—March 2012

Preface

Stateful web servers are a core concept of dRuby. dRuby lets you pass normal Ruby objects and call their methods across processes and networks seamlessly. With dRuby, you'll experience the world of distributed computing as a natural extension of Ruby.

The most widely used distributed system in the world is probably the Web. It's one of the most successful ways to distribute documents around the world —and dRuby's history is related to the Web. Back when Ruby was still in version 1.1, a web server called shttpsrv was available. shttpsrv was similar to WEBrick, but WEBrick was so innovative that Shinichiro Hara—one of the core committers of Ruby and the author of shttpsrv—decided to ditch the new version of shttpsrv in favor of WEBrick (which now comes as part of Ruby's standard libraries). But I really liked the small and cool web server called shttpsrv, so I wrote a servlet extension for it. With this extension, shttpsrv transformed from an ordinary web server to a special TCP server with state. And that is how dRuby started.

This is the third edition of *The dRuby Book* (the previous two editions were in Japanese). For this edition, I've rewritten the book to cover the latest dRuby information and new libraries. If you are looking for theoretical definitions of distributed objects or detailed comparisons of various systems, look elsewhere! This book is full of hands-on exercises and interesting code examples. I hope you put this book to use by writing code as you read and discovering new things along the way.

Ruby changes your thinking process, and so does dRuby. dRuby is not just a tool to extend a method invocation. You'll discover new techniques, programming styles, and much more as you learn how dRuby works.

dRuby will show you a side of Ruby you've never seen before. Let's explore together!

Who This Book Is For

You'll gain a lot from this book if you are...

- Interested in finding out about the benefits of writing apps using dRuby
- Excited by the concept of "distributed systems" such as NoSQL but think most of the existing systems are too complicated
- Interested in client-server network programming and web programming but are interested in a more lightweight alternative to Ruby on Rails or Sinatra
- Interested in adding concurrent programming, such as multithreading, messaging, and the Actor model, to your applications

You don't need to know much about distributed systems as a prerequisite for reading this book, but you should know the basic Ruby syntax, know the standard Ruby classes, and be able to write some simple code.

More important, you don't need big infrastructure to apply what you will learn in this book. I created most of the libraries to solve problems I was having. Because many personal computers come with multicore processors these days, everyone can benefit from multiprocessing libraries such as dRuby. dRuby and my other libraries will give you some basic constructs to build tools that will make your personal computing environment flexible and powerful. After reading this book, you'll be ready to start making your own distributed tools.

Environment

All the sample programs have been tested on OS X with Ruby 1.9.2. Some of the code runs differently depending on your operating system (especially on Windows machines). I'll mention the differences as we go along.

Throughout this book, we'll do lots of experiments using the interactive Ruby shell (irb). When invoking irb, we pass the `--prompt simple` option to switch the command prompt to a simpler version (>>). Also, we have omitted some of the output prompts (=>) for a more concise display. Finally, you may want to specify `--noreadline` if you are an OS X user and experience problems using dRuby from irb (for more details, see *OS X and readline*, on page ?).

What's in This Book

This book covers a wide range of topics related to distributed computing and more. The main focus is on dRuby, but you'll also find out about other libraries I created, such as ERB, Rinda, and Drip, and how to integrate them with dRuby. You'll learn about some advanced Ruby techniques, such as multi-threading, security, and garbage collection. dRuby exposes some unique problems that you might not often encounter, so you'll find out how to deal with those situations too.

Chapter 1, *Hello, dRuby*, on page ?
: The fun part starts here. We'll launch multiple terminals and access dRuby via irb. You'll learn how to use dRuby and write some simple programs to explore the power of dRuby.

Chapter 2, *Architectures of Distributed Systems*, on page ?
: You'll learn about distributed object systems in general and how dRuby is different from others.

Chapter 3, *Integrating dRuby with eRuby*, on page ?
: eRuby is a templating system often used to render HTML. ERB is an implementation of eRuby that I wrote, and it's also part of the Ruby standard libraries. In this chapter, I'll explain how easily you can integrate ERB with dRuby.

Chapter 4, *Pass by Reference, Pass by Value*, on page ?
: Even though dRuby is a seamless extension of Ruby, there are a few differences. In this chapter, you'll learn two ways of exchanging objects over processes: by reference and by value.

Chapter 5, *Multithreading*, on page ?
: You need to know about multithreading to have a better understanding of how dRuby works. When using dRuby, multiple processes work in coordination with multithreading. In this chapter, you'll learn about threading in Ruby and how you can synchronize threads, which is important for avoiding unexpected bugs.

Chapter 6, *Coordinating Processes Using Rinda*, on page ?
: Linda is a system for multiple processes to coordinate with one another. In this chapter, you'll learn how to coordinate processes via `TupleSpace` using Rinda, the Ruby implementation of Linda.

Chapter 7, *Extending Rinda*, on page ?
: Rinda started as a port of Linda, but I added a few extra functionalities I thought necessary while developing applications with Rinda. You'll also learn about a service registration service called Ring, which comes with Rinda.

Chapter 8, *Parallel Computing and Persistence with Rinda*, on page ?
: After releasing Rinda, I created an extension library called *more_rinda* that adds parallel computing capability and a persistence layer to Rinda. They are not part of Ruby standard libraries but have interesting extensions—with some drawbacks. I'll explain why. If you're interested in parallel computing or NoSQL, this is a chapter you shouldn't miss.

Chapter 9, *Drip: A Stream-Based Storage System*, on page ?
: If more_rinda is the trial and error of all my attempts at the art of distributed programming, Drip is my solution. Drip is a stream-based storage system, with fault tolerance and a messaging system built in. I will explain the basic usage of Drip by comparing `Queue` and `Hash` and also talk about the design policy behind Drip.

Chapter 10, *Building a Simple Search System with Drip*, on page ?
: We'll create a simple desktop search system using Drip. You will experience how you can use Drip as both a storage system and a process coordination tool. We will also talk about the `RBTree` data structure we used in the search system, which Drip uses internally.

Chapter 11, *Handling Garbage Collection*, on page ?
: You may not need to worry about garbage collection when you use Ruby daily, but there are a few things you have to know when you use dRuby. Ruby has a garbage collection system that cleans up unused objects, but this doesn't consider how dRuby passes references across processes. In this chapter, you'll see how to protect dRuby referenced objects from garbage collection and what you have to know about garbage collection when you are building applications.

Chapter 12, *Security in dRuby*, on page ?
: dRuby lets you communicate with other processes seamlessly, but this also means you have to be more careful about security to prevent unintended access. You'll learn what dRuby does and doesn't do when it comes to security and what you have to do at the application level. I'll also explain how to use dRuby over networks using SSH port forwarding.

Everyone should read Chapter 1, *Hello, dRuby*, on page ? and Chapter 6, *Coordinating Processes Using Rinda*, on page ? to get a basic understanding of dRuby and Rinda. If you already use dRuby and are seeking some practical tips, then you'll find the following chapters packed with detailed explanations: Chapter 4, *Pass by Reference, Pass by Value*, on page ?; Chapter 5, *Multi-threading*, on page ?; Chapter 11, *Handling Garbage Collection*, on page ?; and Chapter 12, *Security in dRuby*, on page ?. If you're new to dRuby, you might find the level of detail in these chapters overwhelming. Feel free to read only the first section of these chapters and jump to the following chapters. You can always refer to these chapters as a reference when you encounter problems using dRuby.

Newly added for this English edition or greatly modified are the following chapters: Chapter 3, *Integrating dRuby with eRuby*, on page ?; Chapter 8, *Parallel Computing and Persistence with Rinda*, on page ?; and Chapter 9, *Drip: A Stream-Based Storage System*, on page ?. They're packed with unique ways to use each library and also contain many new concepts.

Conventions Used in This Book

Ruby method names follow the convention of the Ruby manual. For example, `String.new` represents a class method, and `String#chomp` represents an instance method. The arguments are just examples, and you should add your own arguments when working on the code.

The book's website[1] has a place to submit errata for the book and to participate in its discussion forum. You'll also find the source code for all the projects we build. You can click the box before the code excerpts to download that snippet directly.

Let's get started!

1. http:/pragprog.com/titles/sidruby