Ruby 2.5 Information and Documentation

OCTOBER, 2018

wlharvey4

Published by:

wlharvey4
Address Line 1
Address Line 2
etc.
Email: wlharvey4@emac.com
URL: http://www.example.com/

This is Edition 0.1a of *Ruby 2.5 Information and Documentation*.

Table of Contents

Preface 1
Intended Audience 1
What Is Covered 1
Typographical Conventions 1
Acknowledgements 1

1 Introduction 2

2 Documentation 3
2.1 Installing Ruby 3
2.1.1 Package Management Systems 3
2.1.1.1 Homebrew (OS X) 3
2.1.2 Installers 3
2.1.2.1 `ruby-build` 4
2.1.2.2 `ruby-install` 4
2.1.3 Managers 4
2.1.3.1 chruby 4
2.1.3.2 `rbenv` 4
2.1.3.3 RVM ("Ruby Version Manager") 4
2.1.3.4 `uru` 4
2.1.4 Building From Source 5
2.2 Getting Started 5
2.3 Manuals 5
2.4 Reference Documentation 5
2.5 Editors and IDEs 5
2.6 Further Reading 5

Appendix A First Appendix Title 6

Appendix B Code Chunk Summaries 7
B.1 Source File Definitions 7
B.2 Code Chunk Definitions 7
B.3 Code Chunk References 7

Bibliography 8

Index 9

Preface

Text here.

Intended Audience

Text here.

What Is Covered

Text and chapter by chapter description here.

Typographical Conventions

This book is written in an enhanced version of Texinfo, the GNU documentation formatting language. A single Texinfo source file is used to produce both the printed and online versions of a program's documentation. Because of this, the typographical conventions are slightly different than in other books you may have read.

Examples you would type at the command-line are preceded by the common shell primary and secondary prompts, '`$`' and '`>`'. Input that you type is shown *`like this`*. Output from the command is preceded by the glyph "⊣". This typically represents the command's standard output. Error messages, and other output on the command's standard error, are preceded by the glyph " error ". For example:

```
$ echo hi on stdout
⊣ hi on stdout
$ echo hello on stderr 1>&2
error  hello on stderr
```

In the text, command names appear in `this font`, while code segments appear in the same font and quoted, '`like this`'. Options look like this: `-f`. Some things are emphasized *like this*, and if a point needs to be made strongly, it is done **like this**. The first occurrence of a new term is usually its *definition* and appears in the same font as the previous occurrence of "definition" in this sentence. Finally, file names are indicated like this: `/path/to/our/file`.

Acknowledgements

1 Introduction

Ruby is . . .

A dynamic, open source programming language with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to write.

2 Documentation

Here you will find pointers to manuals, tutorials and references that will come in handy when you feel like coding in Ruby.

2.1 Installing Ruby

Installation Methods

There are several ways to install Ruby:

- **Package Manager**: When you are on a UNIX-like operating system, using your systems package manager is the easiest way of getting started. However, the packaged Ruby version usually is not the newest one.
- **Installers**: can be used to install a specific or multiple Ruby versions. There is also an installer for Windows.
- **Managers** help you to switch between multiple Ruby installations on your system.
- **Source**: And finally, you can also build Ruby from source.

The following overview lists available installation methods for different needs and platforms.

2.1.1 Package Management Systems

If you cannot compile your own Ruby, and you do not want to use a third-party tool, you can use your systems package manager to install Ruby.

Certain members in the Ruby community feel very strongly that you should never use a package manager to install Ruby and that you should use tools instead. While the full list of pros and cons is outside of the scope of this page, the most basic reason is that most package managers have older versions of Ruby in their official repositories. If you would like to use the newest Ruby, make sure you use the correct package name, or use the tools described further below instead.

2.1.1.1 Homebrew (OS X)

Homebrew

On macOS (High) Sierra and OS X El Capitan, Ruby 2.0 is included.

Many people on OS X use Homebrew as a package manager. It is really easy to get a newer version of Ruby using Homebrew:

```
$ brew install ruby
```

This should install the latest Ruby version.

2.1.2 Installers

If the version of Ruby provided by your system or package manager is out of date, a newer one can be installed using a third-party installer. Some of them also allow you to install multiple versions on the same system; associated managers can help to switch between the different Rubies. If you are planning to use RVM as a version manager you do not need a separate installer, it comes with its own.

2.1.2.1 `ruby-build`

`ruby-build`

`rbenv`

`ruby-build` is a plugin for `rbenv` (see Section 2.1.3.2 "`rbenv`", page 4, that allows you to compile and install different versions of Ruby into arbitrary directories. `ruby-build` can also be used as a standalone program without `rbenv`. It is available for OS X, Linux, and other UNIX-like operating systems.

2.1.2.2 `ruby-install`

`ruby-install` version manager `chruby` version switcher

`ruby-install`

`chruby`

`ruby-install` allows you to compile and install different versions of Ruby into arbitrary directories. There is also a sibling, `chruby` (see Section 2.1.3.1 "chruby", page 4), which handles switching between Ruby versions. It is available for OS X, Linux, and other UNIX-like operating systems.

2.1.3 Managers

Many Rubyists use Ruby managers to manage multiple Rubies. They confer various advantages but are not officially supported. Their respective communities are very helpful, however.

2.1.3.1 chruby

`chruby` allows you to switch between multiple Rubies. `chruby` can manage Rubies installed by `ruby-install` (see Section 2.1.2.2 "`ruby-install`", page 4) or even built from source.

2.1.3.2 `rbenv`

`rbenv`

`ruby-build`

`rbenv` allows you to manage multiple installations of Ruby. It does not support installing Ruby, but there is a popular plugin named `ruby-build` (see Section 2.1.2.1 "`ruby-build`", page 4) to install Ruby. Both tools are available for OS X, Linux, or other UNIX-like operating systems.

2.1.3.3 RVM ("Ruby Version Manager")

RVM

RVM allows you to install and manage multiple installations of Ruby on your system. It can also manage different gemsets. It is available for OS X, Linux, or other UNIX-like operating systems.

2.1.3.4 `uru`

Uru

Uru is a lightweight, multi-platform command line tool that helps you to use multiple Rubies on OS X, Linux, or Windows systems.

2.1.4 Building From Source

Of course, you can install Ruby from source. Download and unpack a tarball, then just do this:

```
$ ./configure
$ make
$ sudo make install
```

By default, this will install Ruby into `/usr/local`. To change, pass the `--prefix=DIR` option to the `./configure` script.

Using the third-party tools or package managers might be a better idea, though, because the installed Ruby wont be managed by any tools.

2.2 Getting Started

2.3 Manuals

2.4 Reference Documentation

2.5 Editors and IDEs

2.6 Further Reading

Appendix A First Appendix Title

Appendix B Code Chunk Summaries

This appendix presents alphabetical lists of all the file definitions, the code chunk definitions, and the code chunk references.

B.1 Source File Definitions

B.2 Code Chunk Definitions

B.3 Code Chunk References

Bibliography

Index

C

`chruby` 4

D

Documentation 3

G

gemsets, manage different using RVM 4

I

installer, third party 3

M

manage Rubies using `chruby` 4
multiple installations, manage using RVM 4
multiple Rubies, command-line tool `uru` 4

R

`rbenv` 4
`rbenv` version manager 4
Rubies, switch between 3
`ruby-build` plugin 4
RVM version manager 3

S

source, building 5

T

Texinfo document formatting language 1

U

`uru` 4

V

version managers 3
versions, multiple installations using `rbenv` 4
versions, switch between using `chruby` 4
versions,multiple 3