About the Tutorial

Ruby is a scripting language designed by Yukihiro Matsumoto, also known as Matz. It runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX.

This tutorial gives a complete understanding on Ruby.

Audience

This tutorial has been prepared for beginners to help them understand the basic to advanced concepts related to Ruby Scripting languages.

Prerequisites

Before you start practicing with various types of examples given in this tutorial, we are making an assumption that you are already aware of computer programs and programming languages in general.

Copyright & Disclaimer

© Copyright 2015 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at contact@tutorialspoint.com
# Table of Contents

About the Tutorial .......................................................................................................................... i

Audience ........................................................................................................................................ i

Prerequisites ..................................................................................................................................... i

Copyright & Disclaimer ................................................................................................................... i

Table of Contents .......................................................................................................................... ii

1. RUBY – OVERVIEW ................................................................................................................... 1

   Features of Ruby .............................................................................................................................. 1

   Tools You Will Need ......................................................................................................................... 2

   What is Next? ................................................................................................................................... 2

2. RUBY – ENVIRONMENT SETUP ................................................................................................. 3

   Try it Option Online ......................................................................................................................... 3

   Local Environment Setup ............................................................................................................... 3

   Ruby Installation on Linux/Unix ........................................................................................................ 3

   Using yum to Install Ruby ............................................................................................................... 4

   Ruby Installation on Windows .......................................................................................................... 4

   Ruby Command Line Options ......................................................................................................... 5

   Ruby Environment Variables ......................................................................................................... 7

   Popular Ruby Editors ...................................................................................................................... 9

   Interactive Ruby (IRb) ..................................................................................................................... 9

   What is Next? ................................................................................................................................ 10

3. RUBY – SYNTAX .......................................................................................................................... 11

   Whitespace in Ruby Program .......................................................................................................... 11

   Line Endings in Ruby Program ....................................................................................................... 11

   Ruby Identifiers .............................................................................................................................. 11
Ruby Ranges

6. RUBY – OPERATORS

Ruby Arithmetic Operators

Ruby Comparison Operators

Ruby Assignment Operators

Ruby Parallel Assignment

Ruby Bitwise Operators

Ruby Logical Operators

Ruby Ternary Operator

Ruby Range Operators

Ruby defined? Operators

Ruby Dot "." and Double Colon ":=" Operators

Ruby Operators Precedence

7. RUBY – COMMENTS

Ruby Multiline Comments

8. RUBY – IF...ELSE, CASE, UNLESS

Ruby if...else Statement

Ruby if modifier

Ruby unless Statement

Ruby unless modifier

Ruby case Statement

9. RUBY – LOOPS

Ruby while Statement

Ruby while modifier

Ruby until Statement
19. RUBY — FILE I/O ................................................................. 123

The puts Statement ...................................................................... 123
The gets Statement ...................................................................... 123
Theputc Statement ...................................................................... 124
The print Statement .................................................................... 124
Opening and Closing Files ............................................................ 125
The File.new Method ................................................................... 125
The File.open Method .................................................................. 125
Reading and Writing Files ............................................................. 126
The syslog Method ...................................................................... 126
The syswrite Method .................................................................. 127
The each_byte Method .................................................................. 127
The IO.readlines Method .............................................................. 128
The IO.foreach Method ............................................................... 128
Renaming and Deleting Files ......................................................... 128
File Modes and Ownership ........................................................... 129
File Inquiries .............................................................................. 130
Directories in Ruby ...................................................................... 132
Navigating Through Directories ................................................... 132
Creating a Directory .................................................................... 132
Deleting a Directory .................................................................... 133
Creating Files & Temporary Directories ....................................... 133
Built-in Functions ...................................................................... 133
File Class and Methods ............................................................... 134
Directory Class and Methods ....................................................... 139

20. RUBY — EXCEPTIONS ............................................................. 142
## Using retry Statement

- Page 143

## Using raise Statement

- Page 144

## Using ensure Statement

- Page 146

## Using else Statement

- Page 147

## Catch and Throw

- Page 148

## Class Exception

- Page 149

### 21. RUBY OBJECT ORIENTED

- Page 151

## Ruby Class Definition

- Page 151

## Define Ruby Objects

- Page 151

## The initialize Method

- Page 151

## The instance Variables

- Page 152

## The accessor & setter Methods

- Page 152

## The instance Methods

- Page 155

## The class Methods and Variables

- Page 155

## The to_s Method

- Page 156

## Access Control

- Page 157

## Class Inheritance

- Page 159

## Methods Overriding

- Page 160

## Operator Overloading

- Page 161

## Freezing Objects

- Page 162

## Class Constants

- Page 164

## Create Object Using Allocate

- Page 165

## Class Information

- Page 166

### 22. RUBY – REGULAR EXPRESSIONS

- Page 167

## Regular-Expression Modifiers

- Page 167

## Regular-Expression Patterns

- Page 168
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular-Expression Examples</td>
<td>171</td>
</tr>
<tr>
<td>Anchors</td>
<td>173</td>
</tr>
<tr>
<td>Special Syntax with Parentheses</td>
<td>174</td>
</tr>
<tr>
<td>Search and Replace</td>
<td>174</td>
</tr>
<tr>
<td>23. RUBY – DBI</td>
<td>177</td>
</tr>
<tr>
<td>Architecture of a DBI Application</td>
<td>177</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>178</td>
</tr>
<tr>
<td>Obtaining and Installing Ruby/DBI</td>
<td>178</td>
</tr>
<tr>
<td>Database Connection</td>
<td>179</td>
</tr>
<tr>
<td>INSERT Operation</td>
<td>180</td>
</tr>
<tr>
<td>Using do Statement</td>
<td>180</td>
</tr>
<tr>
<td>Using prepare and execute</td>
<td>181</td>
</tr>
<tr>
<td>READ Operation</td>
<td>183</td>
</tr>
<tr>
<td>Fetching the Result</td>
<td>184</td>
</tr>
<tr>
<td>Update Operation</td>
<td>190</td>
</tr>
<tr>
<td>DELETE Operation</td>
<td>191</td>
</tr>
<tr>
<td>Performing Transactions</td>
<td>192</td>
</tr>
<tr>
<td>COMMIT Operation</td>
<td>193</td>
</tr>
<tr>
<td>ROLLBACK Operation</td>
<td>193</td>
</tr>
<tr>
<td>Disconnecting Database</td>
<td>193</td>
</tr>
<tr>
<td>Handling Errors</td>
<td>194</td>
</tr>
<tr>
<td>Code Blocks with Methods</td>
<td>195</td>
</tr>
<tr>
<td>Driver-specific Functions and Attributes</td>
<td>196</td>
</tr>
<tr>
<td>24. RUBY – WEB APPLICATIONS</td>
<td>199</td>
</tr>
<tr>
<td>Writing CGI Scripts</td>
<td>199</td>
</tr>
<tr>
<td>Using cgi.rb</td>
<td>199</td>
</tr>
<tr>
<td>Chapter</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>28.</td>
<td>RUBY – WEB SERVICES</td>
</tr>
<tr>
<td></td>
<td>What is SOAP?</td>
</tr>
<tr>
<td></td>
<td>Installing SOAP4R</td>
</tr>
<tr>
<td></td>
<td>Writing SOAP4R Servers</td>
</tr>
<tr>
<td></td>
<td>Writing SOAP4R Clients</td>
</tr>
<tr>
<td>29.</td>
<td>RUBY – TK GUIDE</td>
</tr>
<tr>
<td></td>
<td>Introduction</td>
</tr>
<tr>
<td></td>
<td>Installation</td>
</tr>
<tr>
<td></td>
<td>Simple Tk Application</td>
</tr>
<tr>
<td></td>
<td>Ruby/Tk Widget Classes</td>
</tr>
<tr>
<td></td>
<td>TkFrame</td>
</tr>
<tr>
<td></td>
<td>TkButton</td>
</tr>
<tr>
<td></td>
<td>TkLabel</td>
</tr>
<tr>
<td></td>
<td>TkEntry</td>
</tr>
<tr>
<td></td>
<td>TkCheckButton</td>
</tr>
<tr>
<td></td>
<td>TkRadioButton</td>
</tr>
<tr>
<td></td>
<td>TkListbox</td>
</tr>
<tr>
<td></td>
<td>TkComboBox</td>
</tr>
<tr>
<td></td>
<td>TkMenu</td>
</tr>
<tr>
<td></td>
<td>TkMenubutton</td>
</tr>
<tr>
<td></td>
<td>Tk.messageBox</td>
</tr>
<tr>
<td></td>
<td>TkScrollbar</td>
</tr>
<tr>
<td></td>
<td>TkCanvas</td>
</tr>
<tr>
<td></td>
<td>TkScale</td>
</tr>
</tbody>
</table>
Ruby Profiler ........................................................................................................................................... 401
Additional Ruby Tools .......................................................................................................................... 402
eRuby: Embedded Ruby ........................................................................................................................ 402
ri: Ruby Interactive Reference ............................................................................................................... 403
Ruby is a pure object-oriented programming language. It was created in 1993 by Yukihiro Matsumoto of Japan.

You can find the name Yukihiro Matsumoto on the Ruby mailing list at www.ruby-lang.org. Matsumoto is also known as Matz in the Ruby community.

Ruby is "A Programmer's Best Friend".

Ruby has features that are similar to those of Smalltalk, Perl, and Python. Perl, Python, and Smalltalk are scripting languages. Smalltalk is a true object-oriented language. Ruby, like Smalltalk, is a perfect object-oriented language. Using Ruby syntax is much easier than using Smalltalk syntax.

Features of Ruby

- Ruby is an open-source and is freely available on the Web, but it is subject to a license.
- Ruby is a general-purpose, interpreted programming language.
- Ruby is a true object-oriented programming language.
- Ruby is a server-side scripting language similar to Python and PERL.
- Ruby can be used to write Common Gateway Interface (CGI) scripts.
- Ruby can be embedded into Hypertext Markup Language (HTML).
- Ruby has a clean and easy syntax that allows a new developer to learn very quickly and easily.
- Ruby has similar syntax to that of many programming languages such as C++ and Perl.
- Ruby is very much scalable and big programs written in Ruby are easily maintainable.
- Ruby can be used for developing Internet and intranet applications.
- Ruby can be installed in Windows and POSIX environments.
- Ruby support many GUI tools such as Tcl/Tk, GTK, and OpenGL.
- Ruby can easily be connected to DB2, MySQL, Oracle, and Sybase.
- Ruby has a rich set of built-in functions, which can be used directly into Ruby scripts.
Tools You Will Need

For performing the examples discussed in this tutorial, you will need a latest computer like Intel Core i3 or i5 with a minimum of 2GB of RAM (4GB of RAM recommended). You also will need the following software:

- Linux or Windows 95/98/2000/NT or Windows 7 operating system
- Apache 1.3.19-5 Web server
- Internet Explorer 5.0 or above Web browser
- Ruby 1.8.5

This tutorial will provide the necessary skills to create GUI, networking, and Web applications using Ruby. It also will talk about extending and embedding Ruby applications.

What is Next?

The next chapter guides you to where you can obtain Ruby and its documentation. Finally, it instructs you on how to install Ruby and prepare an environment to develop Ruby applications.
Try it Option Online

We already have set up Ruby Programming environment online, so that you can execute almost all the tutorial examples online at the same time when you are doing your theory work. This gives you confidence in what you are reading and to check the result with different options. Feel free to modify any example and execute it online.

Try the following example using the Try it option available on our website at the top right corner of the sample code box given below:

```ruby
#!/usr/bin/ruby -w
puts "Hello, Ruby!";
```

For most of the examples given in this tutorial, you will find a Try it option on our website code sections at the top right corner that will take you to the online compiler. So just make use of it and enjoy your learning.

Local Environment Setup

If you are still willing to set up your environment for Ruby programming language, then let's proceed. This tutorial will teach you all the important topics related to environment setup. We would recommend you to go through the following topics first and then proceed further:

- **Ruby Installation on Linux/Unix**: If you are planning to have your development environment on Linux/Unix Machine, then go through this chapter.
- **Ruby Installation on Windows**: If you are planning to have your development environment on Windows Machine, then go through this chapter.
- **Ruby Command Line Options**: This chapter list out all the command line options, which you can use along with Ruby interpreter.
- **Ruby Environment Variables**: This chapter has a list of all the important environment variables to be set to make Ruby Interpreter works.

Ruby Installation on Linux/Unix

Here are the steps to be followed to install Ruby on a Unix machine:

**NOTE**: Before proceeding, make sure you have root privilege.

- Download a zipped file having latest version of Ruby. Follow Download Link.
After having downloaded the Ruby archive, unpack it and change into the newly created directory:

```
$ tar -xvzf ruby-1.6.7.tgz
$ cd ruby-1.6.7
```

Now, configure and compile the source code as follows:

```
$ ./configure
$ make
```

Finally, install Ruby interpreter as follows:

```
$ su -l root # become a root user
$ make install
$ exit # become the original user again
```

After installation, make sure everything is working fine by issuing the following command on the command-line:

```
$ruby -v
ruby 1.6.7 (2002-06-04) [i386-netbsd]
```

If everything is fine, this should output the version of the installed Ruby interpreter as shown above. You may have installed different version, so it will display a different version.

### Using yum to Install Ruby

If your computer is connected to the Internet, then the easiest way to install Ruby or any other other RPM is using the `yum` utility. Give the following command at the command prompt and you will find Ruby gets installed on your computer.

```
$ yum install ruby
```

### Ruby Installation on Windows

Here are the steps to install Ruby on a Windows machine.

**NOTE:** You may have different versions available at the time of installation.

- Download a zipped file having latest version of Ruby. Follow [Download Link](#).
- After having downloaded the Ruby archive, unpack it and change into the newly created directory:
- Double-click the Ruby1.6.7.exe file. The Ruby installation wizard starts.
• Click Next to move to the Important Information page of the wizard and keep moving till Ruby installer completes installing Ruby.

You may need to set some environment variables if your installation has not setup them appropriately.

• If you use Windows 9x, add the following lines to your c:\autoexec.bat: set PATH="D:\(ruby install directory)\bin;%PATH%"

• Windows NT/2000 users need to modify their registries.
  o Click Control Panel | System Properties | Environment Variables.
  o Under System Variables, select Path and click EDIT.
  o Add your Ruby directory to the end of the Variable Value list and click OK.
  o Under System Variables, select PATH and click EDIT.
  o Add .RB and .RBW to the Variable Value list and click OK.

• After installation, make sure everything is working fine by issuing the following command on the command-line:

  $ruby -v
  ruby 1.6.7

• If everything is fine, this should output the version of the installed Ruby interpreter as shown above. You may have installed different version, so it will display a different version.

Ruby Command Line Options

Ruby is generally run from the command line in the following way:

$ ruby [ options ] [.] [ programfile ] [ arguments ... ]

The interpreter can be invoked with any of the following options to control the environment and behavior of the interpreter.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>Used with -n or -p to split each line. Check -n and -p options.</td>
</tr>
<tr>
<td>-c</td>
<td>Checks syntax only, without executing program.</td>
</tr>
<tr>
<td>-C dir</td>
<td>Changes directory before executing (equivalent to -X).</td>
</tr>
<tr>
<td>-d</td>
<td>Enables debug mode (equivalent to -debug).</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>-F pat</td>
<td>Specifies pat as the default separator pattern ($;) used by split.</td>
</tr>
<tr>
<td>-e prog</td>
<td>Specifies prog as the program from the command line. Specify multiple -e options for multiline programs.</td>
</tr>
<tr>
<td>-h</td>
<td>Displays an overview of command-line options.</td>
</tr>
<tr>
<td>-i [ ext]</td>
<td>Overwrites the file contents with program output. The original file is saved with the extension ext. If ext isn't specified, the original file is deleted.</td>
</tr>
<tr>
<td>-I dir</td>
<td>Adds dir as the directory for loading libraries.</td>
</tr>
<tr>
<td>-K [ kcode]</td>
<td>Specifies the multibyte character set code (e or E for EUC (extended Unix code); s or S for SJIS (Shift-JIS); u or U for UTF-8; and a, A, n, or N for ASCII).</td>
</tr>
<tr>
<td>-l</td>
<td>Enables automatic line-end processing. Chops a newline from input lines and appends a newline to output lines.</td>
</tr>
<tr>
<td>-n</td>
<td>Places code within an input loop (as in while gets; ... end).</td>
</tr>
<tr>
<td>-0[ octal]</td>
<td>Sets default record separator ($/) as an octal. Defaults to \0 if octal not specified.</td>
</tr>
<tr>
<td>-p</td>
<td>Places code within an input loop. Writes $_ for each iteration.</td>
</tr>
<tr>
<td>-r lib</td>
<td>Uses require to load lib as a library before executing.</td>
</tr>
<tr>
<td>-s</td>
<td>Interprets any arguments between the program name and filename arguments fitting the pattern -xxx as a switch and defines the corresponding variable.</td>
</tr>
<tr>
<td>-T [level]</td>
<td>Sets the level for tainting checks (1 if level not specified).</td>
</tr>
<tr>
<td>-v</td>
<td>Displays version and enables verbose mode</td>
</tr>
<tr>
<td>-w</td>
<td>Enables verbose mode. If program file not specified, reads from STDIN.</td>
</tr>
<tr>
<td>-x [dir]</td>
<td>Strips text before #!ruby line. Changes directory to dir before executing if dir is specified.</td>
</tr>
</tbody>
</table>
-X dir | Changes directory before executing (equivalent to -C).

-y | Enables parser debug mode.

--copyright | Displays copyright notice.

--debug | Enables debug mode (equivalent to -d).

--help | Displays an overview of command-line options (equivalent to -h).

--version | Displays version.

--verbose | Enables verbose mode (equivalent to -v). Sets $VERBOSE to true.

--yydebug | Enables parser debug mode (equivalent to -y).

Single character command-line options can be combined. The following two lines express the same meaning:

```
ruby -ne 'print if /Ruby/' /usr/share/bin
```
```
ruby -n -e 'print if /Ruby/' /usr/share/bin
```

**Ruby Environment Variables**

Ruby interpreter uses the following environment variables to control its behavior. The ENV object contains a list of all the current environment variables set.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLN_LIBRARY_PATH</td>
<td>Search path for dynamically loaded modules.</td>
</tr>
<tr>
<td>HOME</td>
<td>Directory moved to when no argument is passed to Dir::chdir. Also used by File::expand_path to expand &quot;~&quot;.</td>
</tr>
<tr>
<td>LOGDIR</td>
<td>Directory moved to when no arguments are passed to</td>
</tr>
<tr>
<td>Environment Variable</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>PATH</td>
<td>Search path for executing subprocesses and searching for Ruby programs with the <code>-S</code> option. Separate each path with a colon (semicolon in DOS and Windows).</td>
</tr>
<tr>
<td>RUBYLIB</td>
<td>Search path for libraries. Separate each path with a colon (semicolon in DOS and Windows).</td>
</tr>
<tr>
<td>RUBYLIB_PREFIX</td>
<td>Used to modify the RUBYLIB search path by replacing prefix of library path1 with path2 using the format path1;path2 or path1path2.</td>
</tr>
<tr>
<td>RUBYOPT</td>
<td>Command-line options passed to Ruby interpreter. Ignored in taint mode (Where <code>$SAFE</code> is greater than 0).</td>
</tr>
<tr>
<td>RUBYPATH</td>
<td>With <code>-S</code> option, search path for Ruby programs. Takes precedence over PATH. Ignored in taint mode (where <code>$SAFE</code> is greater than 0).</td>
</tr>
<tr>
<td>RUBYSHELL</td>
<td>Specifies shell for spawned processes. If not set, SHELL or COMSPEC are checked.</td>
</tr>
</tbody>
</table>

For Unix, use `env` command to see a list of all the environment variables.

```
HOSTNAME=ip-72-167-112-17.ip.secureserver.net
RUBYPATH=/usr/bin
SHELL=/bin/bash
TERM=xterm
HISTSIZE=1000
SSH_CLIENT=122.169.131.179 1742 22
SSH_TTY=/dev/pts/1
USER=amrood
JRE_HOME=/usr/java/jdk/jre
J2RE_HOME=/usr/java/jdk/jre
PATH=/usr/local/bin:/bin:/usr/bin:/home/guest/bin
MAIL=/var/spool/mail/guest
PWD=/home/amrood
INPUTRC=/etc/inputrc
```
Popular Ruby Editors

To write your Ruby programs, you will need an editor:

- If you are working on Windows machine, then you can use any simple text editor like Notepad or Edit plus.
- **VIM** (Vi IMproved) is a very simple text editor. This is available on almost all Unix machines and now Windows as well. Otherwise, you can use your favorite vi editor to write Ruby programs.
- **RubyWin** is a Ruby Integrated Development Environment (IDE) for Windows.
- Ruby Development Environment (RDE) is also a very good IDE for windows users.

Interactive Ruby (IRb)

Interactive Ruby (IRb) provides a shell for experimentation. Within the IRb shell, you can immediately view expression results, line by line.

This tool comes along with Ruby installation so you have nothing to do extra to have IRb working.

Just type **irb** at your command prompt and an Interactive Ruby Session will start as given below:

```
$irb
irb 0.6.1(99/09/16)
irb(main):001> def hello
irb(main):002:1> out = "Hello World"
irb(main):003:1> puts out
```
Do not worry about what we did here. You will learn all these steps in subsequent chapters.

**What is Next?**

We assume now you have a working Ruby Environment and you are ready to write the first Ruby Program. The next chapter will teach you how to write Ruby programs.
End of ebook preview

If you liked what you saw...

Buy it from our store @ https://store.tutorialspoint.com