Ruby on Rails 2.1.X

tutorialspoint

SIMPLYEASYLEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia

https://twitter.com/tutorialspoint

About the Tutorial

Ruby on Rails is an extremely productive web application framework written in Ruby by David Heinemeier Hansson. This tutorial gives you a complete understanding on Ruby on Rails 2.1.

Audience

This tutorial has been designed for beginners who would like to use the Ruby framework for developing database-backed web applications.

Prerequisites

Before you start Ruby on Rails, please make sure you have a basic understanding on the following subjects:

- Ruby syntax and language constructs such as blocks.
- Relational databases and SQL.
- HTML documents and CSS.

Copyright & Disclaimer

Table of Contents

1. Ruby on Rails 2.1 – Introduction

What is Ruby?

Before we ride on Rails, let us recapitulate a few points of Ruby, which is the base of Rails.

Ruby is the successful combination of:

- Smalltalk's conceptual elegance,
- Python's ease of use and learning, and
- Perl's pragmatism.

Ruby is

- A high-level programming language.
- Interpreted like Perl, Python, Tcl/TK.
- Object-oriented like Smalltalk, Eiffel, Ada, Java.

Why Ruby?

Ruby originated in Japan and now it is gaining popularity in US and Europe as well. The following factors contribute towards its popularity:

- Easy to learn
- Open source (very liberal license)
- Rich libraries
- Very easy to extend
- Truly object-oriented
- Less coding with fewer bugs
- Helpful community

Although we have many reasons to use Ruby, there are a few drawbacks as well that you may have to consider before implementing Ruby:

- Performance Issues - Although it rivals Perl and Python, it is still an interpreted language and we cannot compare it with high-level programming languages like C or C++.
- Threading model – Ruby does not use native threads. Ruby threads are simulated in the VM rather than running as native OS threads.

Sample Ruby Code

Here is a sample Ruby code to print "Hello Ruby".

```
#!/usr/bin/ruby -w

# The Hello Class
class Hello
  # Define constructor for the class
  def initialize( name )
      @name = name.capitalize
  end

  # Define a ruby method
  def salute
      puts "Hello #{@name}!"
  end
end

# Create a new object for Hello class
obj = Hello.new("Ruby")

# Call ruby method
obj.salute
```

This will produce the following result:

```
Hello Ruby
```

For a complete understanding on **Ruby**, please go through our **Ruby** Tutorial

Embedded Ruby

Ruby provides a program called ERb (Embedded Ruby), written by *Seki Masatoshi*. ERb allows you to put Ruby code inside an HTML file. ERb reads along, word for word, and then at a certain point, when it encounters a Ruby code, it starts executing the Ruby code.

You need to know only two things to prepare an ERb document:

- If you want some Ruby code executed, enclose it between **<%** and **%>**.

- If you want the result of the code execution to be printed out, as a part of the output, enclose the code between **<%=** and **%>**.

Here's an example. Save the code in erbdemo.erb file. Note that a Ruby file will have an extension **.rb**, while an Embeded Ruby file will have an extension **.erb**.

```
<% page_title = "Demonstration of ERb" %>
<% salutation = "Dear programmer," %>
<html>
<head>
<title><%= page_title %></title>
</head>
<body>
<p><%= salutation %></p>
<p>This is an example of how ERb fills out a template.</p>
</body>
</html>
```

Now, run the program using the command-line utility erb.

```
c:\ruby\>erb erbdemo.erb
```

This will produce the following result:

```
<html>
<head>
<title>Demonstration of ERb</title>
</head>
<body>
<p>Dear programmer,</p>
<p>This is an example  of how ERb fills out a template.</p>
</body>
</html>
```

What is Rails?

- An extremely productive web-application framework.
- You could develop a web application at least ten times faster with Rails, than you could with a typical Java framework.

- An open source Ruby framework for developing database-backed web applications.
- Your code and database schema are the configuration!
- No compilation phase required.

Full Stack Framework

- Includes everything needed to create a database-driven web application using the Model-View-Controller (MVC) pattern.
- Being a full-stack framework means all the layers are built to work seamlessly with less code.
- Requires fewer lines of code than other frameworks.

Convention over Configuration

- Rails shuns configuration files in favor of conventions, reflection, and dynamic run-time extensions.
- Your application code and your running database already contain everything that Rails needs to know!

Don't Repeat Yourself (DRY)

DRY is a slogan you will hear frequently associated with Ruby on Rails, which means you need to code the behavior only once and you never have to write similar code in two different places. This is important because you are less likely to make mistakes by modifying your code at one place only.

Rails Strengths

Rails is packed with features that make you more productive, with many of the following features building on one other.

Metaprogramming: Other frameworks use extensive code generation from scratch. Metaprogramming techniques use programs to write programs. Ruby is one of the best languages for metaprogramming, and Rails uses this capability well. Rails also uses code generation but relies much more on metaprogramming for the heavy lifting.

Active Record: Rails introduces the Active Record framework, which saves objects to the database. The Rails version of the Active Record discovers the columns in a database schema and automatically attaches them to your domain objects using metaprogramming.

Convention over configuration: Most web development frameworks for .NET or Java force you to write pages of configuration code. If you follow the suggested naming conventions, Rails doesn't need much configuration.

Scaffolding: You often create temporary code in the early stages of development to help get an application up quickly and see how major components work together. Rails automatically creates much of the scaffolding you'll need.

Ajax at the core: Ajax is the technology that has become a standard to provide interactivity to websites without becoming intrusive. Ruby on Rails has a great support for Ajax technology and it is a part of the core libraries. So, when you install RoR, Ajax support is also made available to you.

Built-in testing: Rails creates simple automated tests you can then extend. Rails also provides supporting code called harnesses and fixtures that make test cases easier to write and run. Ruby can then execute all your automated tests with the rake utility.

Three environments: Rails gives you three default environments: development, testing, and production. Each behaves slightly differently, making your entire software development cycle easier. For example, Rails creates a fresh copy of the Test database for each test run.

What is Rails 2.1.0?

This is the latest version of Ruby on Rails, which has been released by the Rails core team on Saturday May 31, 2008.

This version is a further improvement on RoR 2.0, which was again really a fantastic release, absolutely stuffed with great new features, loads of fixes, and an incredible amount of polish over its previous versions RoR 1.2.x.

This tutorial takes you through all the important features available in the latest RoR version 2.1.0.

After this tutorial, you should be able to build your website using one of the best Web 2.0 technologies called Ruby on Rails v2.1.0.

2. Ruby on Rails 2.1 – Installation

To develop a web application using Ruby on Rails Framework, you would need to install the following software:

- Ruby
- The Rails framework
- A Web Server
- A Database System

We assume that you already have installed a Web Server and Database System on your computer. You can always use the WEBrick Web Server, which comes with standard installation of Ruby. Most sites, however, use Apache or lightTPD in production.

Rails works with many database systems, including MySQL, PostgreSQL, SQLite, Oracle, DB2 and SQL Server. Please refer to a corresponding Database System Setup manual to setup your database.

Let's look at the installation instructions for Rails' Framework on Windows, Mac OS X, and Linux.

Rails Installation on Windows

First, let's check to see if you already have Ruby installed. Bring up a command prompt and type **C:\> ruby -v**. If Ruby responds, and if it shows a version number at or above 1.8.6, then type **C:\> gem --version**. If you don't get an error, skip to step 3. Otherwise, we'll do a fresh installation for Ruby.

1. If Ruby is not installed, then download an installation package from rubyinstaller.rubyforge.org. Follow the **download** link, and run the resulting installer. This is an exe like **ruby186-25.exe** and will be installed in a single click. You may as well install everything. It's a very small package, and you'll get **RubyGems** as well along with this package.

2. With RubyGems loaded, you can install all of Rails and its dependencies through the command line:

```
C:\> gem install rails --include-dependencies
```

The above command may take some time to install all dependencies. Make sure you are connected to the internet while installing gems dependencies.

Congratulations! You are now on Rails over Windows.

NOTE: In case you face any problem with the above installation, there are chances that you may not have the latest versions of Ruby or other Gems. So just issue the following command and you will have everything updated automatically.

```
C:\> gem update
```

Then try the above command with updated gems.

Rails Installation on Mac OS X

1. First, let's check to see if you already have Ruby installed. Bring up a command prompt and type **$ ruby -v**. If Ruby responds, and if it shows a version number at or above 1.8.6 then skip to step 3. Otherwise, we'll do a fresh installation for Ruby. To install a fresh copy of Ruby, the Unix instructions that follow should help.

2. Next, you have to install RubyGems. Go to rubygems.rubyforge.org and follow the download link. OS X will typically unpack the archive file for you, so all you have to do is navigate to the downloaded directory and (in the Terminal application) type the following:

```
tp> tar xzf rubygems-0.8.10.tar.gz

tp> cd rubygems-0.8.10

rubygems-0.8.10> sudo ruby setup.rb
```

3. Now, use RubyGems to install Rails. Issue the following command.

```
tp> sudo gem install rails --include-dependencies
```

The above command may take some time to install all dependencies. Make sure you are connected to the internet while installing gems dependencies.

Congratulations! You are now on Rails over Mac OS X.

```
NOTE: In case you face any problem with above installation, there are chances
that you may not have the latest versions of Ruby or other Gems. So just issue
the following command and you will have everything updated automatically.tp>
sudo gem update
```

Then try the above command with updated gems.

Rails Installation on Linux

1. First, let's check to see if you already have Ruby installed. Bring up a command prompt and type **$ ruby -v**. If Ruby responds, and if it shows a version number at or above 1.8.6, then skip to step 5. Otherwise, we'll do a fresh installation for Ruby.

2. Download ruby-x.y.z.tar.gz from www.ruby-lang.org.

3. Untar the distribution, and enter the top-level directory.

4. Do the usual open-source build as follows:

```
tp> tar -xzf ruby-x.y.z.tar.gz
tp> cd ruby-x.y.z
ruby-x.y.z> ./configure
ruby-x.y.z> make
ruby-x.y.z> make test
ruby-x.y.z> make install
```

5. Install RubyGems. Go to rubygems.rubyforge.org, and follow the **download** link. Once you have the file locally, enter the following at your command prompt:

```
tp> tar -xzf rubygems-x.y.z.tar.gz
tp> cd rubygems-x.y.z
rubygems-x.y.z> ruby setup.rb
```

6. Now use RubyGems to install Rails. Still in the shell, issue the following command.

```
tp> gem install rails --include-dependencies
```

The above command may take some time to install all dependencies. Make sure you are connected to the internet while installing gems dependencies.

Congratulations! You are now on Rails over Linux.

NOTE: In case you face any problem with above installation, there are chances that you may not have the latest versions of Ruby or other Gems. So, just issue the following command and you will have everything updated automatically.tp> sudo gem update

Then try the above command with updated gems.

Keeping Rails Up-to-Date

Assuming you have installed Rails using RubyGems, keeping it up-to-date is relatively easy. Issue the following command:

```
tp> gem update rails
```

This will automatically update your Rails installation. The next time you restart your application, it will pick up this latest version of Rails. While giving this command, make sure you are connected to the internet.

Installation Verification

You can verify if everything is setup according to your requirements or not. Use the following command to create a *demo project* in Rails environment.

```
tp> rails demo
```

This will create a demo rails' project using **SQLite** database. Note that Rails uses **SQLite** as its default database.

We can create an application that will use **MySQL** database. Assuming you have **MySQL** database setup on your machine, issue the following command to create an application that will use MySQL database:

```
tp> rails -d mysql demo
```

We will discuss the database setup part in subsequent chapters. Currently we have to check if our environment is setup properly or not. Use the following commands to run *WEBrick* web server on your machine:

```
tp> cd demo
demo> ruby script/server
=> Rails application started on http://0.0.0.0:3000
=> Ctrl-C to shutdown server; call with --help for options
[2007-02-26 09:16:43] INFO WEBrick 1.3.1
[2007-02-26 09:16:43] INFO ruby 1.8.2 (2004-08-24)...
[2007-02-26 09:16:43] INFO WEBrick::HTTPServer-start:pid=2836...
....
```

Now open your browser and type the following address text box.

```
http://localhost:3000
```

You should receive a message like "Welcome aboard" or "Congratulations".

How to Upgrade?

If you are already running an old version of Rails, then here is the procedure to upgrade it to the latest version 2.1:

1. If you want to move your application to Rails 2.0, you should first move it to Rails 1.2.6.
2. If your application runs fine on 1.2.6 with no deprecation warnings, there's a good chance that it'll run straight up on 2.0.
3. To complete the upgrade, you would have to upgrade your extractions. If you are using *pagination*, you will need to install the *classic_pagination* plugin. If you are using *Oracle*, you will need to install the *activerecord-oracle-adapter* gem.

End of ebook preview

If you liked what you saw…

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