Installing Ruby on Rails in Linux

General rules

Some general ideas apply throughout the installation.

- Each Linux distribution has its own idiosyncrasies. The steps that work with Red Hat Linux don't necessarily work the same way with SUSE Linux. The same is true for different versions of the same distribution. For example, what works well in Fedora Core 6 may not work exactly the same way in Fedora Core 5.

  Your best bet is to find your distribution's most recent version. (I realize that installing the most recent version isn't always practical. But sometimes, having the most recent version makes the difference between an effortless installation and a pain-in-the-neck installation.)

- Some of the steps described in this document can be executed only with root privileges. The best way to get temporary root privileges is to issue a `sudo` command. For example, to search for all files ending in `.rpm`, and to do so as the big, powerful root user, open a terminal window and type

  ```
  sudo find / -name *.rpm -print
  ```

  After typing a `sudo` command, the system asks you for the root user's password. For newly installed Linux systems, the root password is something that's simple and well-documented. For example, the root password may be `root`, or there may be no root password (in which case, you press Enter when you're prompted for the root password). On established Linux systems (systems that you use at work, for example) you must obtain the root password from your system administrator (after giving the administrator a pound of flesh).

- Don't hesitate to make use of your favorite search engine. Even the most obscure Linux distribution has fans who post advice. If you're running into trouble, chances are good that someone else has already run into the same trouble. And whenever people post problems, other people post solutions.

- On many Linux distributions, you can find out which packages are installed by typing the `rpm` command with the `-q` switch. For example, the command

  ```
  rpm -qa | grep -i ruby
  ```

  queries Linux to display the names of **a**ll installed packages, and displays any names containing the word `ruby` (**i**gnoring upper- versus lower-case).

  Remember that what I call "Ruby" in my book (*Ruby on Rails For Dummies*) may consist of several Linux packages. So even when the `rpm` command displays a `ruby` line, you may not have all the software needed to run the programs in the book.

- For more information on utilities such as `rpm`, `yum`, and `apt`, consult your system's `man` pages. For example, to get an overview of the `rpm` command's options, open a terminal window and type

  ```
  man rpm
  ```

Alternatives within Linux

Linux installations come in several different flavors. In particular, each Linux distribution favors a particular *installation routine* and a particular *format for installation files*.

Installation routines

An *installation routine* is a program that you run (typically by typing a command in the terminal window). An installation routine might have several options – options for listing all installed packages, options for discovering packages that are available for installation, and options for installing and uninstalling packages.

There are two commonly used installation routines; namely, `yum` and `apt`. The `yum` routine comes with Red Hat Fedora Linux and its variants; the `apt` routine comes with Debian Linux and its variants. Knowing which routine your system has may be a simple matter of typing

```
man yum
```

or

```
man apt
```

to see if your system displays manual pages for either. In some cases, a particular Linux system may support both `yum` and `apt`.

Installation file format

There are two commonly used formats for installation packages; namely, `rpm` and `dpkg`. For example, a file named `ruby-1.8.5.i586.rpm` is coded in the `rpm` format.

The `rpm` format works well with Red Hat Fedora Linux and its variants; the `dpkg` format works with Debian Linux and its variants.

Things go smoothly when you figure out which format works best on your system, and look for installation packages in that format. But some Linux systems have an `alien` command. The `alien` command can convert back and forth between `rpm` and `dpkg` formats. This is handy if, for example, your system works best with `.dpkg` files, but your stuck with a file named *`some-package-or-other`*`.rpm.`

For more information on the `alien` command, type

```
man alien
```

Using yum to Install Ruby on Rails

This section describes the use of `yum` to install software. If your system uses `apt` instead of `yum`, then skip this section (and read the next section instead).

Here's how you can install Ruby on Rails using `yum`:

1. Make sure you have an active Internet connection.
2. Open a terminal window.

3. Type the following command:

   ```
   yum install ruby
   ```

   In response, the `yum` installer reaches out along the Internet, and finds repositories containing the appropriate Ruby package (the package that's appropriate for your Linux distribution and version).

   The `yum` installer will add this Ruby package to your system. But first, `yum` will check your system to make sure that you have any packages on which the new Ruby package depends. The `yum` installer looks for packages that Ruby needs in order to run, looks for packages that these needed packages require in order to run, and so on. This is a very good thing. (If you've ever had to manually track down package dependencies on a Linux system, you know how painful it can be!)

   When the yum installer text stops racing along in the terminal window, and you see the familiar old command prompt, you have Ruby installed on your system.

Tip

You can download many useful package using the `yum` installer, but the names of these packages change over time. Who knows? Maybe you should type `yum install ruby-core` instead of `yum install ruby`. Maybe the Ruby language comes in several parts, with several different `yum` packages.

You can discover the names of available packages using the `yum` command's `list` option. For example, you want to see the names of all available packages whose names contain the word `ruby`. Then open a terminal window and type the following command:

```
yum list | grep -i ruby
```

Alternatively, you may try either of the following commands:

```
yum provides ruby
yum search ruby
```

Remember

You can check to make sure that the Ruby installation went smoothly. To do so, type

```
rpm -qa | grep -i ruby
```

in the terminal window.

4. In a terminal window type the following command:

   ```
   yum install rubygems
   ```

   RubyGems is the name of Ruby's own installer. RubyGems is like `yum`. What `yum` does for Linux, RubyGems does for Ruby. After installing RubyGems, you'll be able to add extra functionality to your Ruby programming environment.

   Of course, you can't use RubyGems itself to install RubyGems. (That would be like asking a baby to give birth to itself.) So you use `yum` to install RubyGems.

Tip

For more information about RubyGems, visit http://rubygems.org/.

With RubyGems installed, you can use RubyGems to install other Ruby stuff.

5. In a terminal window type the following command:

   ```
   gem install rails --include-dependencies
   ```

   In response, RubyGems installs Rails on your Linux system.

   (In case it's not clear, the command has three dashes: two dashes immediately before the word `include`, and one dash between the words `include` and `dependencies`. There are no blank spaces in the phrase `--include-dependencies`.)

6. Make sure that MySQL is installed on your system. To do so, type the following two commands:

   ```
   yum install mysql
   yum install mysql-server
   yum install mysql-administrator
   ```

   After each command you get a bunch of messages telling you that `yum` is installing the necessary MySQL packages.

7. Visit http://www.radrails.org and download a version of RadRails that's appropriate for your system.

   For many Linux systems, a file with a name like `radrails-0.`*`x`*`.`*`x`*`-linux-gtk.tar.gz` works nicely.

   This `.tar.gz` file is a compressed archive.

8. Examine the contents of the `.tar.gz` file. On most systems you can do this by double-clicking the file's icon.

   The `radrails-0.`*`x`*`.`*`x`*`-linux-gtk.tar.gz` file contains a folder named radrails.

9. Extract the `radrails` folder to a convenient place on your system's directory structure.

   On most systems you can extract stuff from a compressed archive by dragging the stuff from one file manager window to another.

That's it! You're ready to test the installation.

Using apt to Install Ruby on Rails

This section describes the use of `apt` to install software. If your system uses `yum` instead of `apt`, read the previous section.

Here's how you can install Ruby on Rails using `apt`:

1. Make sure you have an active Internet connection.
2. Open a terminal window.

3. Type the following commands:

   ```
   apt-get install ruby ri rdoc mysql-server libmysql-ruby
   apt-get install mysql-admin
   ```

   In response, the `apt` installer reaches out along the Internet, and finds repositories containing the appropriate Ruby package (the package that's appropriate for your Linux distribution and version).

   The `apt` installer will add this Ruby package to your system. But first, `apt` will check your system to make sure that you have any packages on which the new Ruby package depends. The `apt` installer looks for packages that Ruby needs in order to run, looks for packages that these needed packages require in order to run, and so on. This is a very good thing. (If you've ever had to manually track down package dependencies on a Linux system, you know how painful it can be!)

   When the `apt` installer text stops racing along in the terminal window, and you see the familiar old command prompt, you have Ruby installed on your system.

Tip

You can download many useful package using the `apt-get` installer, but the names of these packages change over time. Who knows? Maybe you should type `apt-get install ruby-core` instead of `apt-get install ruby`. Maybe the Ruby language comes in several parts, with several different packages.

You can discover the names of available packages using the `apt-cache` command's `search` option. For example, you want to see the names of all available packages whose names contain the word `ruby`. Then open a terminal window and type the following command:

```
apt-cache search ruby --names-only
```

4. Visit http://rubyforge.org and download a package named RubyGems.

   Most likely, you'll be downloading a file named `rubygems-`*`something-or-other`*`.tgz`. The `.tgz` extension indicates that this file is a compressed file in tar/gzip format. You can uncompress the file by double-clicking the file's icon with your system's file browser. If that doesn't work, try typing

   ```
   tar -xvzf rubygems-file-name.tgz
   ```

5. Find the directory containing the uncompressed contents of the rubygems file. In that directory (or in one of its subdirectories), look for a file named `setup.rb`. This `setup.rb` file is a Ruby program that installs the RubyGems package. So, to install RubyGems, issue the following command:

   ```
   ruby setup.rb
   ```

RubyGems is the name of Ruby's own installer. RubyGems is like `apt`. What `apt` does for Linux, RubyGems does for Ruby. After installing RubyGems, you'll be able to add extra functionality to your Ruby programming environment. Of course, you can't use RubyGems itself to install RubyGems. (That would be like asking a baby to give birth to itself.) So you use Ruby to install RubyGems.

Tip

For more information about RubyGems, visit http://rubygems.org/.

With RubyGems installed, you can use RubyGems to install other Ruby stuff.

6. In a terminal window type the following command:

   ```
   gem install rails --include-dependencies
   ```

   In response, RubyGems installs Rails on your Linux system.

   (In case it's not clear, the command has three dashes: two dashes immediately before the word `include`, and one dash between the words `include` and `dependencies`. There are no blank spaces in the phrase `--include-dependencies`.)

7. Visit http://www.radrails.org and download a version of RadRails that's appropriate for your system.

   For many Linux systems, a file with a name like `radrails-0.`*`x`*`.`*`x`*`-linux-gtk.tar.gz` works nicely.

   This `.tar.gz` file is a compressed archive.

8. Examine the contents of the `.tar.gz` file. On most systems you can do this by double-clicking the file's icon.

   The `radrails-0.`*`x`*`.`*`x`*`-linux-gtk.tar.gz` file contains a folder named radrails.

9. Extract the `radrails` folder to a convenient place on your system's directory structure.

   On most systems you can extract stuff from a compressed archive by dragging the stuff from one file manager window to another.

In the next section, you test the Ruby on Rails installation.

Running Ruby on Rails

Here's how you test your Ruby on Rails installation:

1. To run the MySQL server, open a terminal window and type the following command:

   ```
   /etc/rc.d/init.d/mysqld start
   ```

   If that doesn't work, poke around for things named `mysqld` on your system. If you're logged on as root, you can find `mysqld` by issuing the following command:

   ```
   find / -name mysqld -print
   ```

Technical Stuff

The name `mysqld` stands for “mysql daemon.” On a Linux system, a program that runs constantly in the background (a database server, for example) is called a “daemon.”

2. In a file manager window, navigate to the `radrails` directory (the directory that you created in Step 9 of the “Using yum” section).

That radrails directory contains an executable file named `RadRails`.

3. Double-click the `RadRails` file's icon.

   The RadRails IDE opens in all ofits glory.

From that point on, you follow the steps in my *Ruby on Rails For Dummies* book: the steps for creating a new project, running some Ruby code, creating and testing a Rails application, and so on.

Tip

When I first tried to run db:migrate I got an error message telling me that Rake couldn't find the `mysql.sock` file. So I looked for my system's `mysql.sock` file by typing the following command:

```
find / -name mysql.sock -print
```

Then I went back to RadRails and edited my project's `config/database.yml` file. In the file's `development` section, I added a line pointing to my system's `mysql.sock` file:

```
development:
  adapter: mysql
  socket: /var/lib/mysql/mysql.sock
```

That worked like a charm.