

Ruby on Rails (Ruby 1.9.2, Rails 3.1.1) Installation

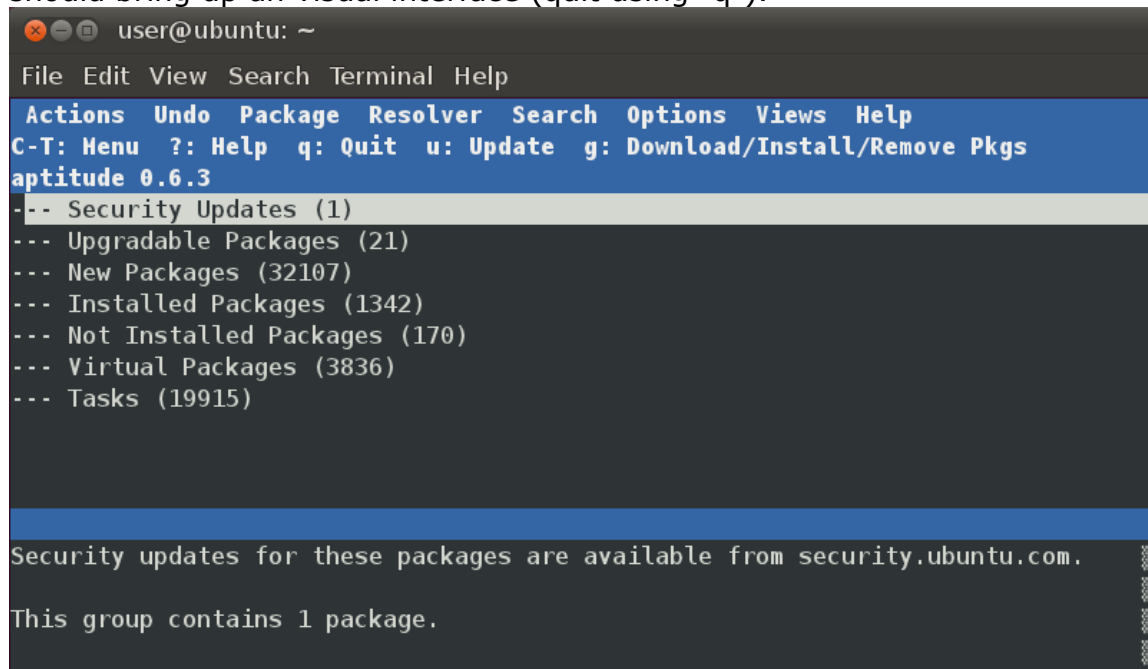
Ubuntu 11.10, 11.04 desktop or server (or on **Linux Mint 11, 12**)

(You are welcomed to share this PDF freely, with no commercial purposes)

First, we will install Ruby on Rails with the default web server WEBrick and database SQLite3.
Second, we will make this installation work with Apache web server and MySQL database.
Third, we will build a very simple application which uses database to test our installation.

Conveniences:

- \$ is the command prompt
- everytime you type a command in the terminal you have to hit Enter to execute it
- when you copy and paste commands from here into the terminal don't copy also \$ sign! It's already there.
- we will use "**aptitude**" as the installer and package manager because it's better than "**apt-get**" at package management (but if you insist on using "apt-get" then you can do so)
- check if you have "**aptitude**" installed by typing "**\$ sudo aptitude**" command in the terminal, it should bring up an visual interface (quit using "q").




```
user@ubuntu: ~  
File Edit View Search Terminal Help  
Actions Undo Package Resolver Search Options Views Help  
C-T: Menu ?: Help q: Quit u: Update g: Download/Install/Remove Pkgs  
aptitude 0.6.3  
-- Security Updates (1)  
--- Upgradable Packages (21)  
--- New Packages (32107)  
--- Installed Packages (1342)  
--- Not Installed Packages (170)  
--- Virtual Packages (3836)  
--- Tasks (19915)  
  
Security updates for these packages are available from security.ubuntu.com.  
  
This group contains 1 package.
```

If you get an error then maybe you need to install it by typing:

```
$ sudo apt-get update  
$ sudo apt-get upgrade  
$ sudo apt-get install aptitude
```

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

We will use a pre-made virtual machine (VMWare or another) with Ubuntu 11.10, 11.04 or Linux Mint 11, 12. If you want to install Ubuntu or Mint from scratch you will have to look for a tutorial for that. In this case a VMWare virtual machine was used. This can be played free using VMWare Player available here: <http://www.vmware.com/products/player/>

After installing VMWare Player you can also download a pre-made virtual machine for VMWare here: <http://www.vmware.com/appliances/directory/cat/508> (look for a Ubuntu 11.10, 11.04 machine, or Linux Mint 11, 12).

If you get a Ubuntu 11.04 virtual machine you can upgrade Ubuntu to the latest version (11.10 in this case) by going to Update Manager and you will see a note there prompting you to upgrade to that latest version. It's up to you if you want to upgrade.


Linux Mint is a fork of Ubuntu so it uses many packages which Ubuntu uses.

Disclaimer

Use this tutorial as is, at your own risk.

I am not responsible of any damage that may occur as a result of using this tutorial.

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

***** **Ruby on Rails working with WEBrick web server and SQLite3 database server** *****
***** (WEBrick and SQLite3 are the default in RoR) *****

1) Update Ubuntu 11.10 or 11.04 to the latest version *(just in case)*

```
$ sudo aptitude update
```

```
$ sudo aptitude upgrade (it will take sometime)
```

(or use Update Manager from Ubuntu)

If you have a server without the desktop interface and you want it then you can install it using:

```
$ sudo aptitude install ubuntu-desktop (it will take sometime)
```

2) Install Git and Curl (for Ruby Version Manager RVM)

```
$ sudo aptitude install build-essential git-core curl
```

```
user@ubuntu:~$ sudo aptitude install build-essential git-core curl
The following NEW packages will be installed:
  git-core
0 packages upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 1,380 B of archives. After unpacking 28.7 kB will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu/ natty/main git-core all 1:1.7.4.1-3
1,380 B]
Fetched 1,380 B in 0s (3,621 B/s)
Selecting previously deselected package git-core.
(Reading database ... 159951 files and directories currently installed.)
Unpacking git-core (from .../git-core_1%3a1.7.4.1-3_all.deb) ...
Setting up git-core (1:1.7.4.1-3) ...

user@ubuntu:~$
```

3) Install Ruby Version Manager RVM (this helps you manage multiple versions of Ruby)

```
$ bash < <(curl -s https://rvm.beginrescueend.com/install/rvm)
```

```
user@ubuntu:~$ bash < <(curl -s https://rvm.beginrescueend.com/install/rvm)
Successfully checked out branch ''
remote: Counting objects: 883, done.
remote: Compressing objects: 100% (281/281), done.
remote: Total 800 (delta 535), reused 764 (delta 500)
Receiving objects: 100% (800/800), 133.66 KiB, done.
Resolving deltas: 100% (535/535), completed with 56 local objects.
From git://github.com/wayneeseguin/rvm
  2flabfb..dec4e82  master    -> origin/master
* [new branch]      shoes     -> origin/shoes
From git://github.com/wayneeseguin/rvm
* [new tag]         1.8.5      -> 1.8.5
* [new tag]         1.8.6      -> 1.8.6
First, rewinding head to replay your work on top of it...
Fast-forwarded master to dec4e82cc651f430e776cedeff8d7bd24654d408.
Successfully pulled (rebased) from origin

RVM:  Shell scripts enabling management of multiple ruby environments.
RTFM: https://rvm.beginrescueend.com/
HELP: http://webchat.freenode.net/?channels=rvm (#rvm on irc.freenode.net)

Installing RVM to /home/user/.rvm/
  Correct permissions for base binaries in /home/user/.rvm/bin...
  Copying manpages into place.
  Recording config files for rubies.
```


4) Add a line to ~/.bashrc (this will load RVM everytime you open the terminal)

```
$ echo '[[ -s "$HOME/.rvm/scripts/rvm" ]] && source "$HOME/.rvm/scripts/rvm"' >> ~/.bashrc
```

```
user@ubuntu:~$ echo '[[ -s "$HOME/.rvm/scripts/rvm" ]] && source "$HOME/.rvm/scripts/rvm"' >> ~/.bashrc
user@ubuntu:~$ █
```

Then close and open again the terminal window to activate it in the terminal (or execute . ~/.bashrc).

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

Check RVM: [\\$ rvm notes](#) (see the results)

```
user@ubuntu:~$ rvm notes

NOTE:
As of 1.8.0 RVM once again loads .rvmrc files, by default, after asking your
permission to trust it of course. If you do not wish to be enabled, simply set:
    export rvm_project_rvmrc=0
Within either your /etc/rvmrc or $HOME/.rvmrc file. This will turn off the
cd/pushd hooks when sourcing the rvm() function into your shell.

Example: echo 'export rvm_project_rvmrc=0' >> $HOME/.rvmrc;
(Then close the current shell and open a new one.)

If you wish to add this for all users, exchange $HOME/.rvmrc with /etc/rvmrc

All individual additional OS requirements/dependencies are now contained in 'rvm requirements'


user@ubuntu:~$
```

5) Install several packages to help Ruby (if some are already installed they won't be installed again)

```
$ sudo aptitude install build-essential openssl libreadline6 libreadline6-dev zlib1g zlib1g-dev zlib
libssl-dev libyaml-dev libsqlite3-0 libsqlite3-dev sqlite3 libxml2-dev libxslt-dev autoconf libc6-dev
ncurses-dev automake libtool bison
```

(this is what RVM tells you later on eventually so better do it now)

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

6) Install Ruby (1.9.2 version in this case)

\$ rvm install 1.9.2

```
user@ubuntu:~$ rvm install 1.9.2
Installing Ruby from source to: /home/user/.rvm/rubies/ruby-1.9.2-p290, this may take a while
depending on your cpu(s)...

ruby-1.9.2-p290 - #fetching
ruby-1.9.2-p290 - #extracted to /home/user/.rvm/src/ruby-1.9.2-p290 (already extracted)
Fetching yaml-0.1.4.tar.gz to /home/user/.rvm/archives
Extracting yaml-0.1.4.tar.gz to /home/user/.rvm/src
Configuring yaml in /home/user/.rvm/src/yaml-0.1.4.
Compiling yaml in /home/user/.rvm/src/yaml-0.1.4.
Installing yaml to /home/user/.rvm/usr
ruby-1.9.2-p290 - #configuring
ruby-1.9.2-p290 - #compiling
ruby-1.9.2-p290 - #installing
Removing old Rubygems files...
Installing rubygems-1.8.10 for ruby-1.9.2-p290 ...
Installation of rubygems completed successfully.
ruby-1.9.2-p290 - adjusting #shebangs for (gem irb erb ri rdoc testrb rake).
ruby-1.9.2-p290 - #importing default gemsets (/home/user/.rvm/gemsets/)
Install of ruby-1.9.2-p290 - #complete
user@ubuntu:~$
```

7) Make RVM use the Ruby version just installed

\$ rvm use 1.9.2

```
user@ubuntu:~$ rvm use 1.9.2
Using /home/user/.rvm/gems/ruby-1.9.2-p290
user@ubuntu:~$
```

Or make it the default:

\$ rvm --default use 1.9.2 (everytime the terminal is started the default Ruby version is used)

```
user@ubuntu:~$ rvm --default use 1.9.2
Using /home/user/.rvm/gems/ruby-1.9.2-p290
user@ubuntu:~$
```

Check: **\$ ruby -v**

```
user@ubuntu:~$ ruby -v
ruby 1.9.2p290 (2011-07-09 revision 32553) [i686-linux]
user@ubuntu:~$
```

8) Install Rails (do not use "sudo" for installing gems!)

`$ gem install rails`

```
user@ubuntu:~$ gem install rails
Fetching: multi_json-1.0.3.gem (100%)
Fetching: activesupport-3.1.1.gem (100%)
Fetching: builder-3.0.0.gem (100%)
Fetching: i18n-0.6.0.gem (100%)
Fetching: activemodel-3.1.1.gem (100%)
Fetching: rack-1.3.5.gem (100%)
Fetching: rack-cache-1.1.gem (100%)
Fetching: rack-test-0.6.1.gem (100%)
Fetching: rack-mount-0.8.3.gem (100%)
Fetching: hike-1.2.1.gem (100%)
Fetching: tilt-1.3.3.gem (100%)
Fetching: sprockets-2.0.3.gem (100%)
Fetching: erubis-2.7.0.gem (100%)
Fetching: actionpack-3.1.1.gem (100%)
Fetching: arel-2.2.1.gem (100%)
```

If you get an error here ("**no such file to load -- zlib**") see the Errors section #8.

Check: `$ rails -v`

```
user@ubuntu:~$ rails -v
Rails 3.1.1
user@ubuntu:~$
```

9) Create a Rails project

`$ rails new site` ("*site*" is the name of the new project - it can be any name)

After this, the bundle install command can start automatically ("bundle install" helps you manage application's dependencies).

If it fails you need to look at the error, eventually copy and paste in Google and search for solution among those results.

If you get this error, for example, "**error failed to build gem native extension sqlite3**" then look in the Errors section below at #10.


After solving the error you need to run "bundle install" again from the directory where your project is ("site" in this case).

If the "bundle install" didn't start automatically then you need to start it manually by going in the directory of the newly created application ("site" in this case) and run there.

`$ cd site` (go to the directory "site")

`$ bundle install`

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

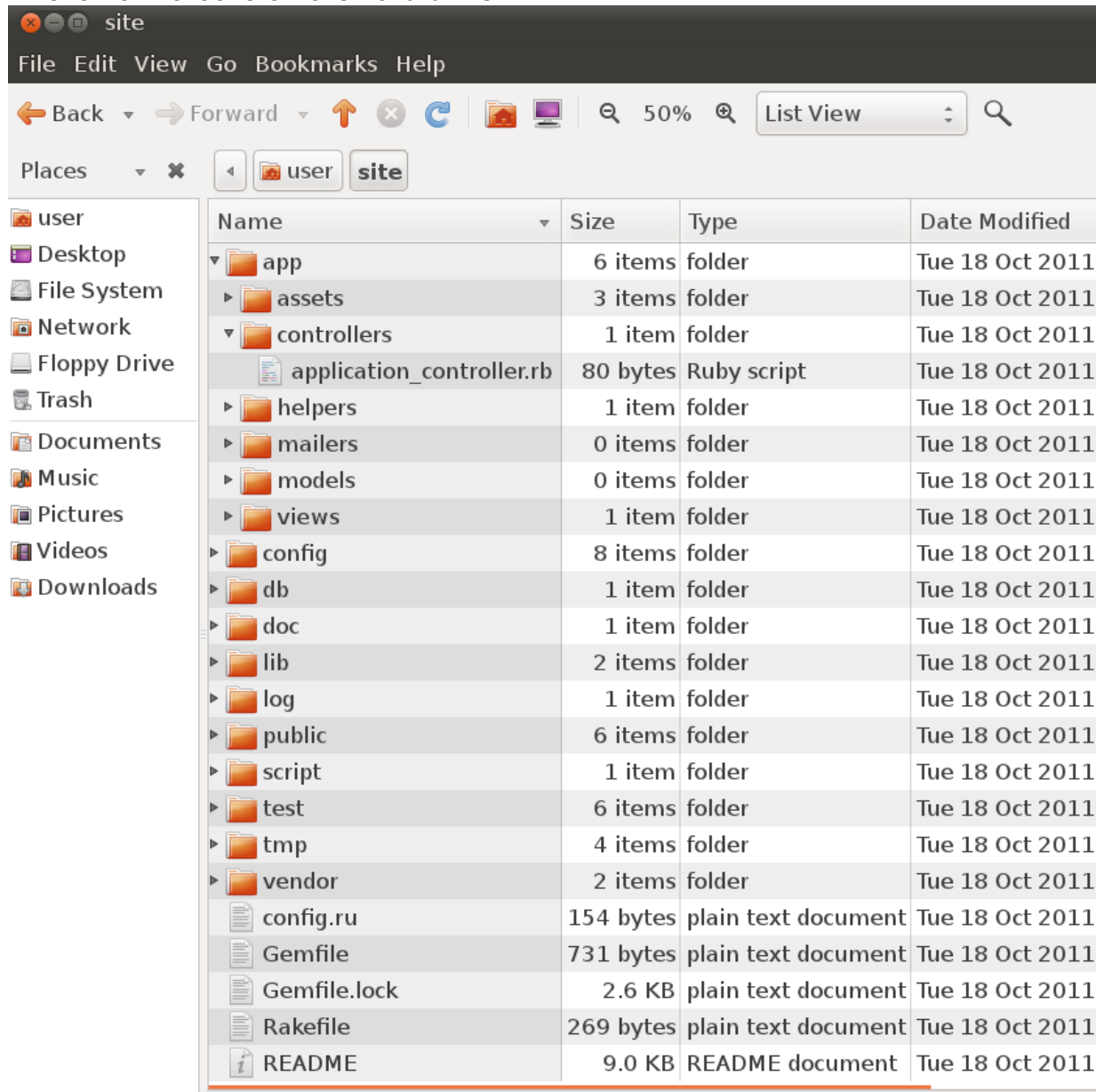
(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

```
user@ubuntu:~$ rails new site
  create
  create  README
  create  Rakefile
  create  config.ru
  create  .gitignore
  create  Gemfile
  create  app
  create  app/assets/images/rails.png
  create  app/assets/javascripts/application.js
  create  app/assets/stylesheets/application.css
  create  app/controllers/application_controller.rb
  create  app/helpers/application_helper.rb
  create  app/mailers
  create  app/models
  create  app/views/layouts/application.html.erb
  create  app/mailers/.gitkeep
  create  app/models/.gitkeep

  create  tmp/cache/assets
  create  vendor/assets/stylesheets
  create  vendor/assets/stylesheets/.gitkeep
  create  vendor/plugins
  create  vendor/plugins/.gitkeep
  run  bundle install
Fetching source index for http://rubygems.org/
Using rake (0.9.2)
Using multi_json (1.0.3)
Using activesupport (3.1.1)
Using builder (3.0.0)
Using i18n (0.6.0)
Using activemodel (3.1.1)
Using erubis (2.7.0)
Using rack (1.3.5)

Installing coffee-rails (3.1.1)
Installing jquery-rails (1.0.16)
Using rails (3.1.1)
Installing sass (3.1.10)
Installing sass-rails (3.1.4)
Installing sqlite3 (1.3.4) with native extensions
Installing turn (0.8.3)
Installing uglifier (1.0.3)
Your bundle is complete! Use `bundle show [gemname]` to see which version
is installed.
user@ubuntu:~$
```



This is how it looks on the hard drive:



10) Start the Rails server (go first in your project directory using `$ cd site` command - "site" in this case)

`$ rails server` (or "`$ rails s`" – shorter version)

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

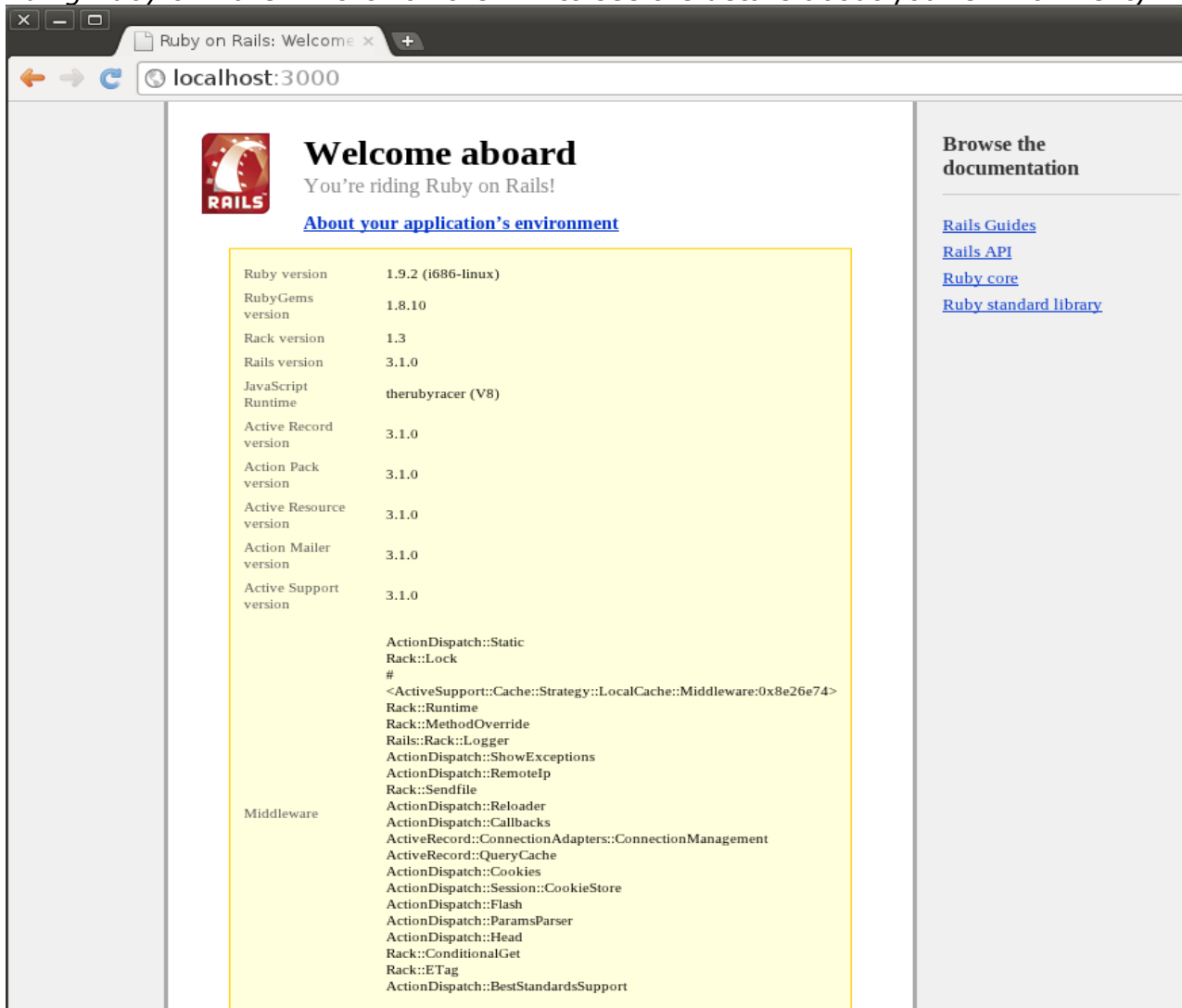
(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

You should see WEBrick server started ("Booting WEBrick").

```
user@ubuntu:~/site$ rails server
/home/user/.rvm/gems/ruby-1.9.2-p290/gems/rack-1.3.4/lib/rack/backports/uri/common_192.rb:53:
warning: already initialized constant WFKV_
=> Booting WEBrick
=> Rails 3.1.0 application starting in development on http://0.0.0.0:3000
=> Call with -d to detach
=> Ctrl-C to shutdown server
[2011-10-05 01:15:42] INFO WEBrick 1.3.1
[2011-10-05 01:15:42] INFO ruby 1.9.2 (2011-07-09) [i686-linux]
[2011-10-05 01:15:42] INFO WEBrick::HTTPServer#start: pid=26801 port=3000
```

Then open the browser and go to " <http://localhost:3000> " (port 3000 – the default port for Ruby on Rails).

You should see the welcome page of your Ruby on Rails "site" project ("Welcome aboard. You're riding Ruby on Rails!" - click on the link to see the details about your environment).



Welcome aboard
You're riding Ruby on Rails!

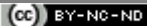
[About your application's environment](#)

Ruby version	1.9.2 (i686-linux)
RubyGems version	1.8.10
Rack version	1.3
Rails version	3.1.0
JavaScript Runtime	therubyracer (V8)
Active Record version	3.1.0
Action Pack version	3.1.0
Active Resource version	3.1.0
Action Mailer version	3.1.0
Active Support version	3.1.0
Middleware	ActionDispatch::Static Rack::Lock # <ActiveSupport::Cache::Strategy::LocalCache::Middleware:0x8e26e74> Rack::Runtime Rack::MethodOverride Rails::Rack::Logger ActionDispatch::ShowExceptions ActionDispatch::RemoteIp Rack::Sendfile ActionDispatch::Reloader ActionDispatch::Callbacks ActiveRecord::ConnectionAdapters::ConnectionManagement ActiveRecord::QueryCache ActionDispatch::Cookies ActionDispatch::Session::CookieStore ActionDispatch::Flash ActionDispatch::ParamsParser ActionDispatch::Head Rack::ConditionalGet Rack::ETag ActionDispatch::BestStandardsSupport

Browse the documentation

- [Rails Guides](#)
- [Rails API](#)
- [Ruby core](#)
- [Ruby standard library](#)

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

If you get an error like "**ExecJS::RuntimeUnavailable**" then look in the Errors section below at #10.

If you encounter an error like "**no such file to load -- openssl**" after starting the Rails server WEBrick then look in the Errors section below at #10.

WEBrick Rails server can be stopped using CTRL+C combination (if that doesn't work try CTRL+Z).

===== Errors =====

8) Error when installing Rails

ERROR: Loading command: install (LoadError)

no such file to load -- zlib

ERROR: While executing gem ... (NameError)

uninitialized constant Gem::Commands::InstallCommand

Solutions: 1) `$ rvm pkg install zlib` (installs the zlib package)

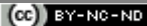
```
user@ubuntu:~$ rvm pkg install zlib
Fetching zlib-1.2.5.tar.gz to /home/user/.rvm/archives
Extracting zlib-1.2.5.tar.gz to /home/user/.rvm/src
Configuring zlib in /home/user/.rvm/src/zlib-1.2.5.
Compiling zlib in /home/user/.rvm/src/zlib-1.2.5.
Installing zlib to /home/user/.rvm/usr
user@ubuntu:~$
```

`$ rvm remove 1.9.2` (removes the Ruby version 1.9.2)

```
user@ubuntu:~$ rvm remove 1.9.2
Removing /home/user/.rvm/src/ruby-1.9.2-p290...
Removing /home/user/.rvm/rubies/ruby-1.9.2-p290...
Removing ruby-1.9.2-p290 aliases...
Removing ruby-1.9.2-p290 wrappers...
Removing ruby-1.9.2-p290 environments...
Removing ruby-1.9.2-p290 binaries...
user@ubuntu:~$
```

`$ rvm install 1.9.2 --with-zlib-dir=$rvm_path/usr` (reinstalls Ruby version 1.9.2 and adds the path to zlib – use the command exactly like that)

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

```
user@ubuntu:~$ rvm install 1.9.2 --with-zlib-dir=$rvm_path/usr
Installing Ruby from source to: /home/user/.rvm/rubies/ruby-1.9.2-p290, this may
take a while depending on your cpu(s)...

ruby-1.9.2-p290 - #fetching
ruby-1.9.2-p290 - #extracting ruby-1.9.2-p290 to /home/user/.rvm/src/ruby-1.9.2-
p290
ruby-1.9.2-p290 - #extracted to /home/user/.rvm/src/ruby-1.9.2-p290
Fetching yaml-0.1.4.tar.gz to /home/user/.rvm/archives
Extracting yaml-0.1.4.tar.gz to /home/user/.rvm/src
Configuring yaml in /home/user/.rvm/src/yaml-0.1.4.
Compiling yaml in /home/user/.rvm/src/yaml-0.1.4.
Installing yaml to /home/user/.rvm/usr
ruby-1.9.2-p290 - #configuring
ruby-1.9.2-p290 - #compiling
ruby-1.9.2-p290 - #installing
Removing old Rubygems files...
Installing rubygems-1.8.10 for ruby-1.9.2-p290 ...
Installation of rubygems completed successfully.
ruby-1.9.2-p290 - adjusting #shebangs for (gem irb erb ri rdoc testrb rake).
ruby-1.9.2-p290 - #importing default gemsets (/home/user/.rvm/gemsets/)
Install of ruby-1.9.2-p290 - #complete
user@ubuntu:~$
```

`$ gem install rails` (try again to install Rails)

2) If the above didn't work try this:

```
$ rvm pkg remove zlib (removes the zlib installed above)
$ rvm remove 1.9.2 (removes the Ruby version 1.9.2)
$ sudo aptitude install zlib1g-dev (installs another version of zlib)
$ rvm install 1.9.2
$ gem install rails (try again to install Rails)
```

9) Error when installing the bundle

error failed to build gem native extension sqlite3

You need to install SQLite package:

```
$ sudo aptitude install libsqlite3-dev
```

Then run the bundle command once again (go first to the directory where your project is using "`$ cd`
`site`" command, "site" in this case):

```
$ bundle install
```

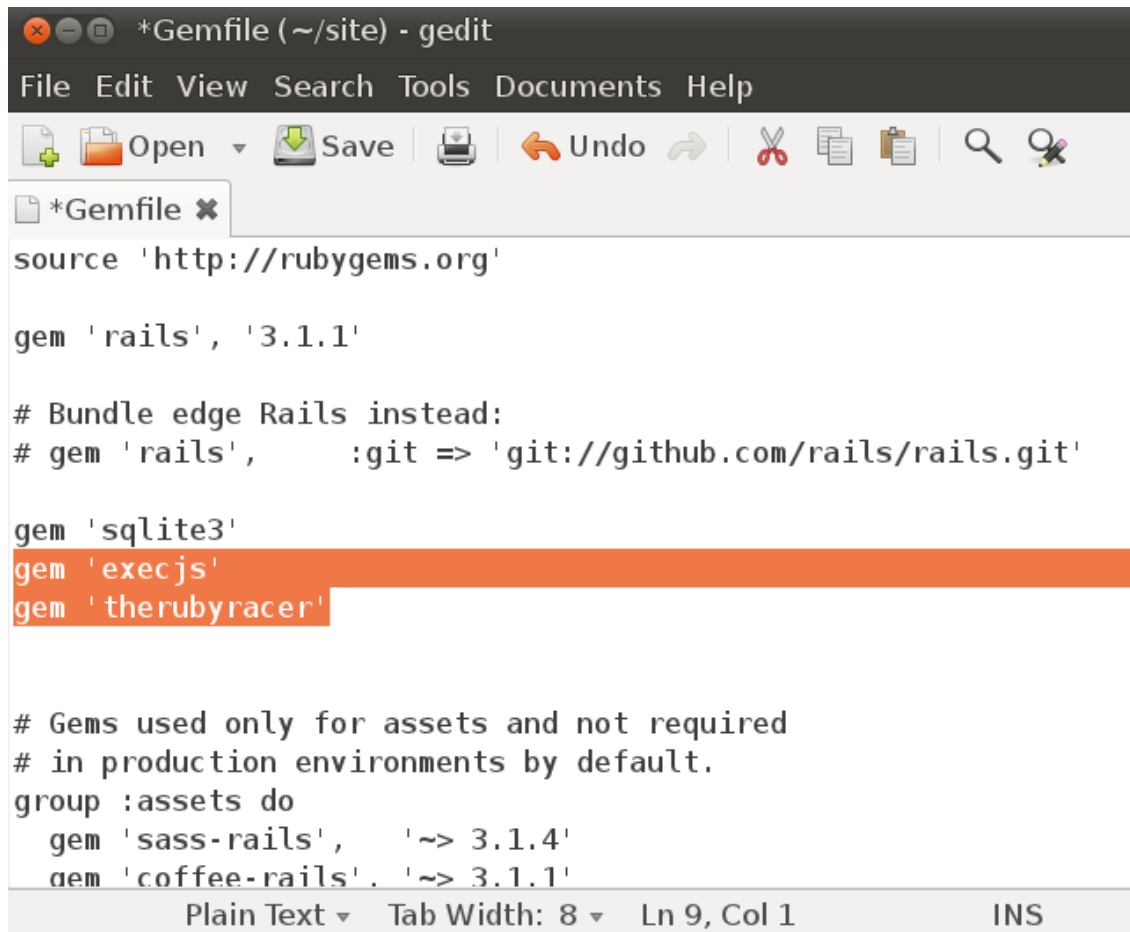
10) Errors when starting WEBrick Rails server

ExecJS::RuntimeUnavailable

```
user@ubuntu:~/site$ rails server
/home/user/.rvm/gems/ruby-1.9.2-p290/gems/execjs-1.2.9/lib/execjs/runtimes.rb:47:in `autodetect': Could not find a JavaScript runtime. See https://github.com/sstephenson/execjs for a list of available runtimes. (ExecJS::RuntimeUnavailable)
    from /home/user/.rvm/gems/ruby-1.9.2-p290/gems/execjs-1.2.9/lib/execjs.rb:5:in `
```

Add these two lines to your Gemfile text file located in your project directory ("site" in this case). You can add them somewhere in the text file (you can add them after "**gem 'sqlite3'**" for example):

```
gem 'execjs'
gem 'therubyracer'
```



```
*Gemfile (~/.site) - gedit
File Edit View Search Tools Documents Help

source 'http://rubygems.org'

gem 'rails', '3.1.1'

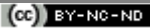
# Bundle edge Rails instead:
# gem 'rails', :git => 'git://github.com/rails/rails.git'

gem 'sqlite3'
gem 'execjs'
gem 'therubyracer'

# Gems used only for assets and not required
# in production environments by default.
group :assets do
  gem 'sass-rails', '~> 3.1.4'
  gem 'coffee-rails', '~> 3.1.1'
end
```

Save the file and then run bundle install command again from the project directory ("site" in this case):

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

```
$ bundle install
```

Start the Rails server again to see if it worked:

```
$ rails server (or "rails s")
```

```
*****
```

```
no such file to load -- openssl
```

```
*****
```

Do this:

- go to the home root using `$ cd ..` command (you are now in your "site" directory)

```
$ rvm pkg install openssl (install openssl)
```

```
$ rvm remove 1.9.2 (uninstall the current version of Ruby)
```

```
$ rvm install 1.9.2 --with-openssl-dir=$HOME/.rvm/usr (run exactly as is, it install again Ruby with openssl)
```

=====

***** **Ruby on Rails working with Apache (mod_rails), Passenger and MySQL** *****

Apache web server #####
#####

1) Install Apache2 web server

\$ sudo aptitude install apache2

```
user@ubuntu:~$ sudo aptitude install apache2
[sudo] password for user:
The following NEW packages will be installed:
  apache2 apache2-mpm-worker{a} apache2-utils{a} apache2.2-bin{a}
  apache2.2-common{a} libapr1{a} libaprutil1{a} libaprutil1-dbd-sqlite3{a}
  libaprutil1-ldap{a}
0 packages upgraded, 9 newly installed, 0 to remove and 2 not upgraded.
Need to get 0 B/3,127 kB of archives. After unpacking 10.5 MB will be used.
Do you want to continue? [Y/n/?] █

Setting up libaprutil1-ldap (1.3.9+dfsg-5ubuntu3) ...
Setting up apache2.2-bin (2.2.17-1ubuntu1.2) ...
Setting up apache2-utils (2.2.17-1ubuntu1.2) ...
Setting up apache2.2-common (2.2.17-1ubuntu1.2) ...
Setting up apache2-mpm-worker (2.2.17-1ubuntu1.2) ...
  * Starting web server apache2                                     [ OK ]
Setting up apache2 (2.2.17-1ubuntu1.2) ...
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place

user@ubuntu:~$ █
```

Check: **\$ apache2 -v**


```
user@ubuntu:~$ apache2 -v
Server version: Apache/2.2.17 (Ubuntu)
Server built:   Sep  1 2011 09:25:29
user@ubuntu:~$ █
```

2) Update Rubygems install (Rubygems manages the Ruby gems)

\$ gem update (if there is nothing to update then it's ok)

```
user@ubuntu:~$ gem update
Updating installed gems
Updating uglifier
Fetching: uglifier-1.0.4.gem (100%)
Successfully installed uglifier-1.0.4
Gems updated: uglifier
Installing ri documentation for uglifier-1.0.4...
Installing RDoc documentation for uglifier-1.0.4...
user@ubuntu:~$ █
```


(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

Check: `$ gem -v`

```
user@ubuntu:~$ gem -v
1.8.10
user@ubuntu:~$
```

3) Install Phusion Passenger (makes RoR work with Apache and Nginx servers)

`$ gem install passenger`

```
user@ubuntu:~$ gem install passenger
Fetching: fastthread-1.0.7.gem (100%)
Building native extensions. This could take a while...
Fetching: daemon_controller-0.2.6.gem (100%)
Fetching: passenger-3.0.9.gem (100%)
Successfully installed fastthread-1.0.7
Successfully installed daemon_controller-0.2.6
Successfully installed passenger-3.0.9
3 gems installed
Installing ri documentation for fastthread-1.0.7...
Installing ri documentation for daemon_controller-0.2.6...
Installing ri documentation for passenger-3.0.9...
Installing RDoc documentation for fastthread-1.0.7...
Installing RDoc documentation for daemon_controller-0.2.6...
Installing RDoc documentation for passenger-3.0.9...
user@ubuntu:~$
```

Check: `$ passenger -v`

```
user@ubuntu:~$ passenger -v
Phusion Passenger version 3.0.9

"Phusion Passenger" is a trademark of Hongli Lai & Ninh Bui.
user@ubuntu:~$
```

4) Install Apache2 module (*mod_rails*)

`$ passenger-install-apache2-module`

```
user@ubuntu:~/site$ passenger-install-apache2-module
Welcome to the Phusion Passenger Apache 2 module installer, v3.0.9.

This installer will guide you through the entire installation process. It
shouldn't take more than 3 minutes in total.

Here's what you can expect from the installation process:

1. The Apache 2 module will be installed for you.
2. You'll learn how to configure Apache.
3. You'll learn how to deploy a Ruby on Rails application.

Don't worry if anything goes wrong. This installer will advise you on how to
solve any problems.

Press Enter to continue, or Ctrl-C to abort.
```

Hit Enter. It will check for software requirements before installing. If something is not found then it will guide you how to install them.

```
Checking for required software...


* GNU C++ compiler... found at /usr/bin/g++
* Curl development headers with SSL support... not found
* OpenSSL development headers... found
* Zlib development headers... found
* Ruby development headers... found
* OpenSSL support for Ruby... found
* RubyGems... found
* Rake... found at /home/user/.rvm/wrappers/ruby-1.9.2-p290/rake
* rack... found
* Apache 2... found at /usr/sbin/apache2
* Apache 2 development headers... not found
* Apache Portable Runtime (APR) development headers... not found
* Apache Portable Runtime Utility (APU) development headers... not found

Some required software is not installed.
But don't worry, this installer will tell you how to install them.

Press Enter to continue, or Ctrl-C to abort.
```

Don't forget to replace "**apt-get**" from the command (like "\$ sudo apt-get install apache2-prefork-dev") with "**aptitude**" (like "\$ sudo aptitude install apache2-prefork-dev"). We are using "aptitude" as the package manager, not "apt-get".

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

Installation instructions for required software

```
* To install Curl development headers with SSL support:  
Please run apt-get install libcurl4-openssl-dev or libcurl4-gnutls-dev, whichever you prefer.  
  
* To install Apache 2 development headers:  
Please run apt-get install apache2-prefork-dev as root.  
  
* To install Apache Portable Runtime (APR) development headers:  
Please run apt-get install libapr1-dev as root.  
  
* To install Apache Portable Runtime Utility (APU) development headers:  
Please run apt-get install libaprutil1-dev as root.
```

If the aforementioned instructions didn't solve your problem, then please take a look at the Users Guide:

```
/home/user/.rvm/gems/ruby-1.9.2-p290/gems/passenger-3.0.9/doc/Users_guide_Apache.html  
user@ubuntu:~/site$
```

After running all those commands to install all missing modules run again "**\$ passenger-install-apache2-module**".

At the end it will tell you how to update your Apache config file (see below).

The Apache 2 module was successfully installed.

Please edit your Apache configuration file, and add these lines:

```
LoadModule passenger_module /home/user/.rvm/gems/ruby-1.9.2-p290/gems/passenger-3.0.9/ext/  
apache2/mod_passenger.so  
PassengerRoot /home/user/.rvm/gems/ruby-1.9.2-p290/gems/passenger-3.0.9  
PassengerRuby /home/user/.rvm/wrappers/ruby-1.9.2-p290/ruby
```

After you restart Apache, you are ready to deploy any number of Ruby on Rails applications on Apache, without any further Ruby on Rails-specific configuration!

Press ENTER to continue.

5) Modify the Apache2 config file

You will be told to do this after Passenger installed mod_rails from above (#4).

You need to open the text file "apache2.conf" from "etc/apache2/" in the root mode.

\$ sudo nano /etc/apache2/apache2.conf (it opens the file in the edit mode)

Then paste what the step above (4) gave you.

Example (copy it from your terminal window, not from here):

```
LoadModule passenger_module /home/user/.rvm/gems/ruby-1.9.2-  
p290/gems/passenger-3.0.9/ext/apache2/mod_passenger.so  
PassengerRoot /home/user/.rvm/gems/ruby-1.9.2-p290/gems/passenger-3.0.9  
PassengerRuby /home/user/.rvm/wrappers/ruby-1.9.2-p290/ruby
```

```
GNU nano 2.2.6      File: /etc/apache2/apache2.conf      Modified

# Include ports listing
Include ports.conf

#
# The following directives define some format nicknames for use with
# a CustomLog directive (see below).
# If you are behind a reverse proxy, you might want to change %h into %{X-Forwarded-For}i
#
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %0 \"%{Referer}i\" \"%{User-Agent}i\" vhost_combined" vhost_combined
LogFormat "%h %l %u %t \"%r\" %>s %0 \"%{Referer}i\" \"%{User-Agent}i\" combined" combined
LogFormat "%h %l %u %t \"%r\" %>s %0" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

# Include of directories ignores editors' and dpkg's backup files,
# see README.Debian for details.

# Include generic snippets of statements
Include conf.d/

# Include the virtual host configurations:
Include sites-enabled/

LoadModule passenger_module /home/user/.rvm/gems/ruby-1.9.2-p290/gems/passenger-3.0.9/ext/ap$
  PassengerRoot /home/user/.rvm/gems/ruby-1.9.2-p290/gems/passenger-3.0.9
  PassengerRuby /home/user/.rvm/wrappers/ruby-1.9.2-p290/ruby

^G Get Help      ^O WriteOut      ^R Read File     ^Y Prev Page     ^K Cut Text       ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is     ^V Next Page     ^U UnCut Text    ^T To Spell
```

Then hit CTRL+X. It will ask you if you want to save the modifications. Hit Y, then Enter. Below you will have also the configuration of the virtual host file.

6) Restart Apache

`$ sudo /etc/init.d/apache2 restart` (or "`$ sudo service apache2 restart`")

```
user@ubuntu:~$ sudo service apache2 restart
* Restarting web server apache2
... waiting [ OK ]
user@ubuntu:~$
```

Alternatively, you can stop Apache and start it again ("`$ sudo /etc/init.d/apache2 stop`" or "`$ sudo service apache2 stop`" and "`sudo /etc/init.d/apache2 start`" or or "`$ sudo service apache2 start`").

7) Serving Rails applications with Passenger+Apache

a) Make a directory in your home directory:

`$ mkdir public_html` (you can use also the Ubuntu graphic interface File Explorer)

b) Make "public_html" directory and create a Rails application

Go to that directory and make a Ruby on Rails application ("site" in this case):

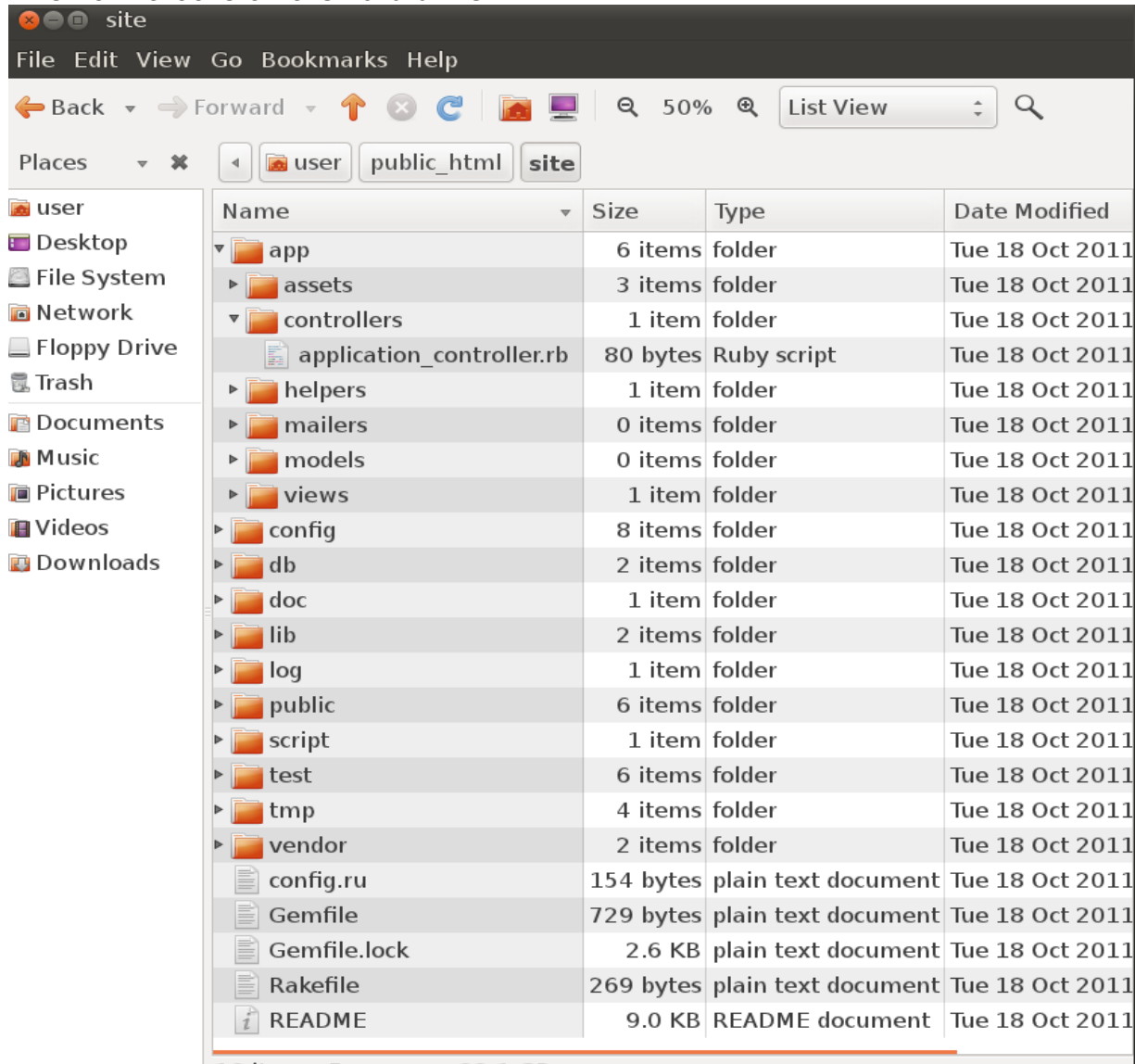
`$ cd public_html` (change directory)

`$ rails new site` (just like you've already done above, in the first part)

If the "bundle install" command doesn't run automatically then you have to run it manually:

```
$ bundle install
```

This how it looks on the hard drive:



Go to the newly created application ("site" in this case) and start the Rails server " `$ rails s` " to see if you get again the error "ExecJS::RuntimeUnavailable".

If you do then repeat the steps from #10 above (Ruby on Rails with WEBrick and SQLite3 installation error).


c) Creating a virtual host file

We have to create a virtual host by creating a file in the "/etc/apache2/sites-available" directory (we will name it "site" - the file won't have any extension but it will be a text file).

```
$ sudo nano /etc/apache2/sites-available/site
```

 (this will create the file named "site" - can be any name - AND open it for editing)

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

Copy and paste this into that file (compare also with what the notes after installing mod_rails tells you):

```
<VirtualHost *:80>
```

```
    ServerName localhost
```

```
    ServerAlias localhost
```

```
    DocumentRoot /home/user/public_html/site/public
```

```
</VirtualHost>
```

Explanations:

VirtualHost *:80 = this is the standard HTTP port, you can change it if you want

ServerName localhost = this if you want to access using "localhost" locally, otherwise put your server name like "domain1.com"

ServerAlias localhost = this if you want to access using "localhost" locally, otherwise put your server name like "www.domain1.com"

DocumentRoot /home/user/public_html/site/public = you can change this if you want to another directory where you have your application

```
GNU nano 2.2.6      File: /etc/apache2/sites-available/site

<VirtualHost *:80>

    ServerName  localhost
    ServerAlias localhost
    DocumentRoot /home/user/public_html/site/public

</VirtualHost>

[ Read 7 lines ]
^G Get Help      ^O WriteOut      ^R Read File     ^Y Prev Page     ^K Cut Text       ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is     ^V Next Page     ^U UnCut Text    ^T To Spell
```

Then hit CTRL+X. It will ask you if you want to save the modifications. Hit Y, then Enter.

d) Enabling the virtual host

Now the new virtual host needs to be enabled:

```
$ sudo a2ensite site
```

e) Enabling mod rewrite in Apache

Rails applications make use of an .htaccess file for various rewrite rules so we need to enable that too in Apache:

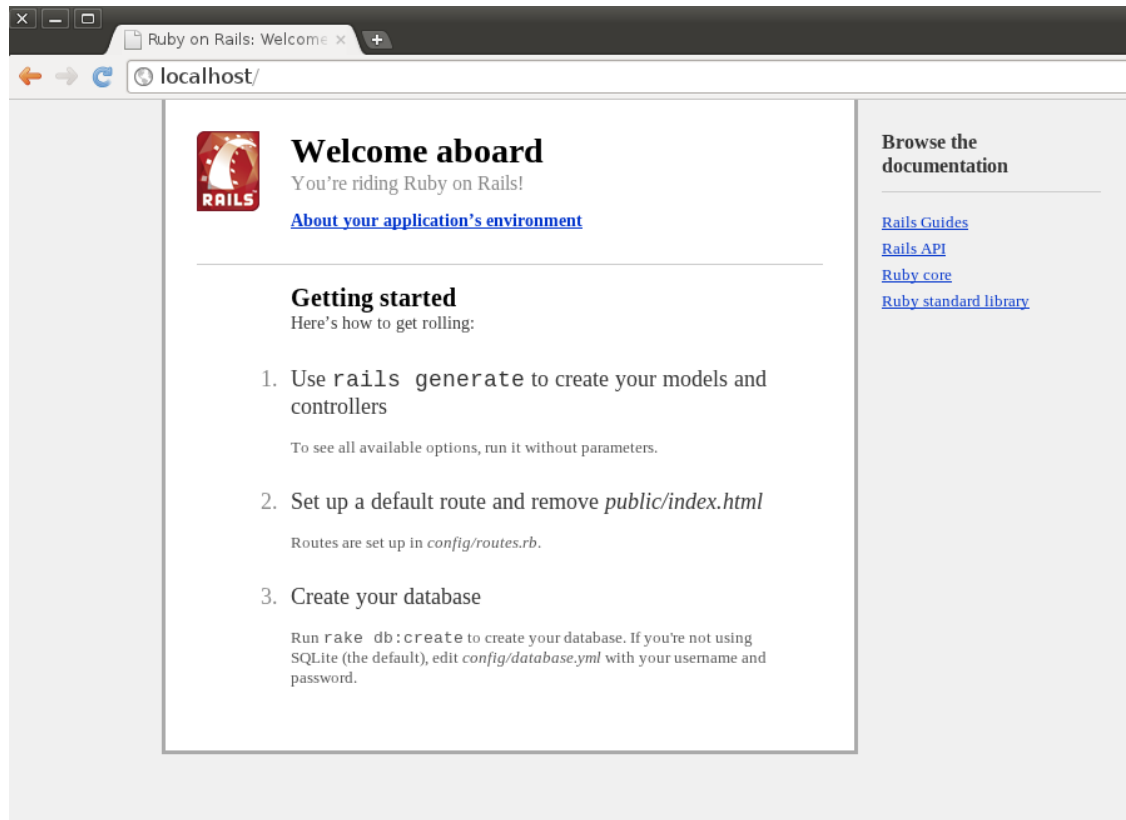
```
$ sudo a2enmod rewrite
```

f) Reload Apache

```
$ sudo /etc/init.d/apache2 reload
```

 (or "\$ sudo service apache2 reload")

Open the browser window and go to "**http://localhost**". You should get the "Welcome aboard You're riding Ruby on Rails!" page. Click on the link. You should see the environment variables.



When you reloaded Apache, if you encountered the error "**apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1 for ServerName**" then do this:

- Open the "httpd.conf" file from "/etc/apache2/httpd.conf" directory (it will be blank usually)

```
$ sudo nano /etc/apache2/httpd.conf
```

- Add the following line there: `ServerName localhost`

Hit CTRL+X. It will ask you if you want to save the modifications. Hit Y, then Enter.

- Restart Apache again

```
$ sudo /etc/init.d/apache2 restart (or " $ sudo service apache2 restart ")
```


Reload again the webpage above. If it breaks, then modify back the "httpd.conf" file by taking out that line you put there (then save). Restart again the server. All should be working again now.

g) Open the browser and go to `http://localhost`.

You should be able to see the Ruby on Rails start page: "Welcome aboard You're riding Ruby on Rails!"

Click on the link "About your application's environment". You should be able to see the environment variables.

(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

If you get this error:

No route matches [GET] "/rails/info/properties"

Then it means you are running the Rails application in the production mode (in the production mode that link doesn't work).

Open again the config file "site" for virtual host:

```
$ sudo nano /etc/apache2/sites-available/site
```

You need to run Rails application in the development mode so add this line to the Virtual Host config file from "/etc/apache2/sites-available/" ("site" in this case):

Add this like to what you have there already, like in the example below:

RailsEnv "development"

So, in the end the Virtual Host config file "site" will look like this:

<VirtualHost *:80>

ServerName localhost

ServerAlias localhost

DocumentRoot /home/user/public_html/site/public

RailsEnv "development"

</VirtualHost>

Explanations:

VirtualHost *:80 = this is the standard HTTP port, you can change it if you want

ServerName localhost = this if you want to access using "localhost" locally, otherwise put your server name like "domain1.com"

ServerAlias localhost = this if you want to access using "localhost" locally, otherwise put your server name like "www.domain1.com"

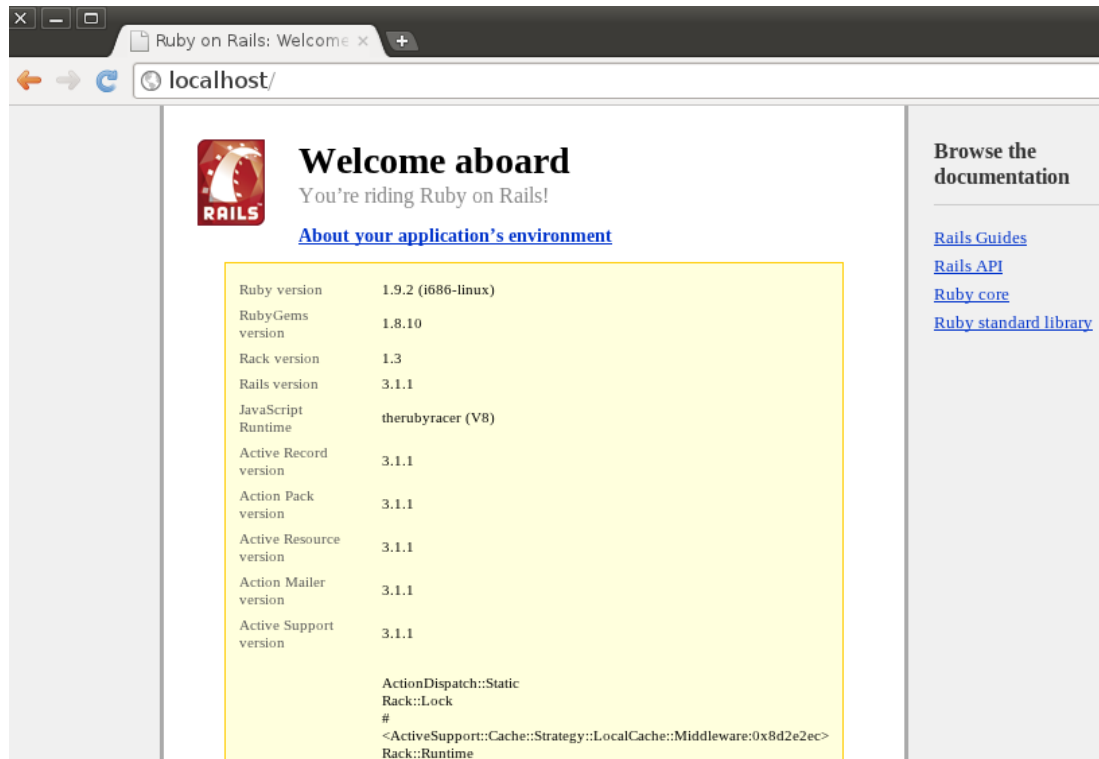
DocumentRoot /home/user/public_html/site/public = you can change this if you want to another directory where you have your application

RailsEnv "development" = this will get rid of the RoR routing error which you will get if the Rails environment would be production, not development

Reload Apache again:

```
$ sudo /etc/init.d/apache2 reload (or " $ sudo service apache2 reload ")
```

Now you can open your browser and go to "**http://localhost**" and all should be working.



```
##### MySQL #####  
#####
```

By default we used SQLite3 database with Ruby on Rails.
Now, below, we will use MySQL as the database of choice.

1) Install MySQL server, client, admin and a library

```
$ sudo aptitude install mysql-server mysql-client mysql-admin libmysqlclient-dev
```

```
user@ubuntu:~$ sudo aptitude install mysql-server mysql-client mysql-admin libmysqlclient-dev  
[sudo] password for user:  
The following NEW packages will be installed:  
  mysql-admin mysql-client mysql-gui-tools-common{a} mysql-query-browser{a}  
0 packages upgraded, 4 newly installed, 0 to remove and 2 not upgraded.  
Need to get 3,729 kB of archives. After unpacking 11.9 MB will be used.  
Do you want to continue? [Y/n/?]   
  
Setting up mysql-gui-tools-common (5.0r14+openSUSE-2.2ubuntu1) ...  
Setting up mysql-admin (5.0r14+openSUSE-2.2ubuntu1) ...  
Setting up mysql-client (5.1.54-1ubuntu4) ...  
Setting up mysql-query-browser (5.0r14+openSUSE-2.2ubuntu1) ...  
  
user@ubuntu:~$
```

If it prompts you to set up a password for the root MySQL user then do it.
If it doesn't prompt you for a password then it can use the Linux password you have setup for your system. You will need this password below.

Check if the server was installed and running (after installation it should run automatically):

```
$ sudo service mysql stop  
$ sudo service mysql start
```

Enter the admin mode for MySQL to create databases, use queries, etc:

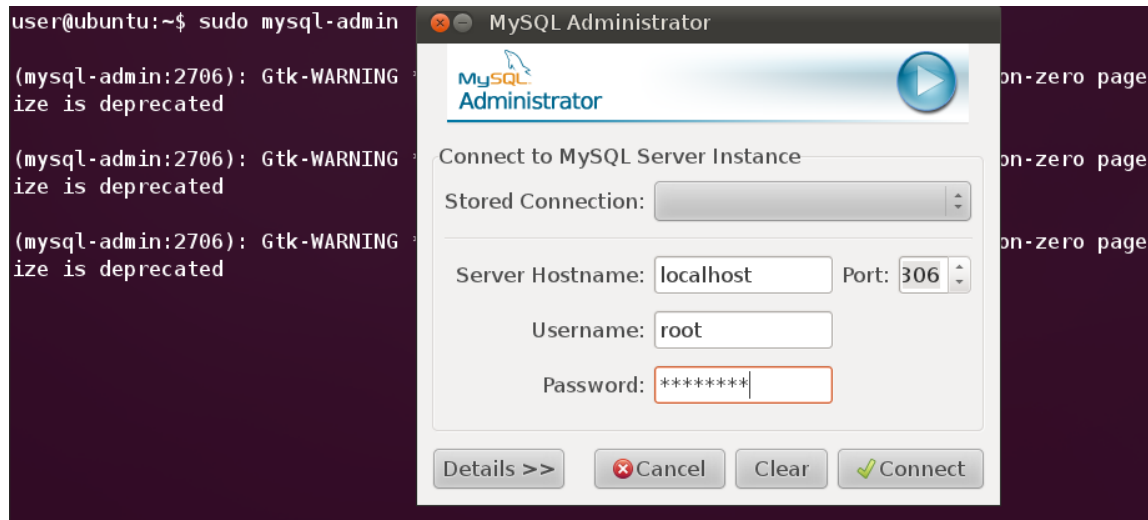
```
$ sudo mysql -u root -p (it will prompt you for a password if you created one)
```

You will see the MySQL command prompt: mysql>

You can exit from the MySQL admin mode by typing "exit", then Enter.

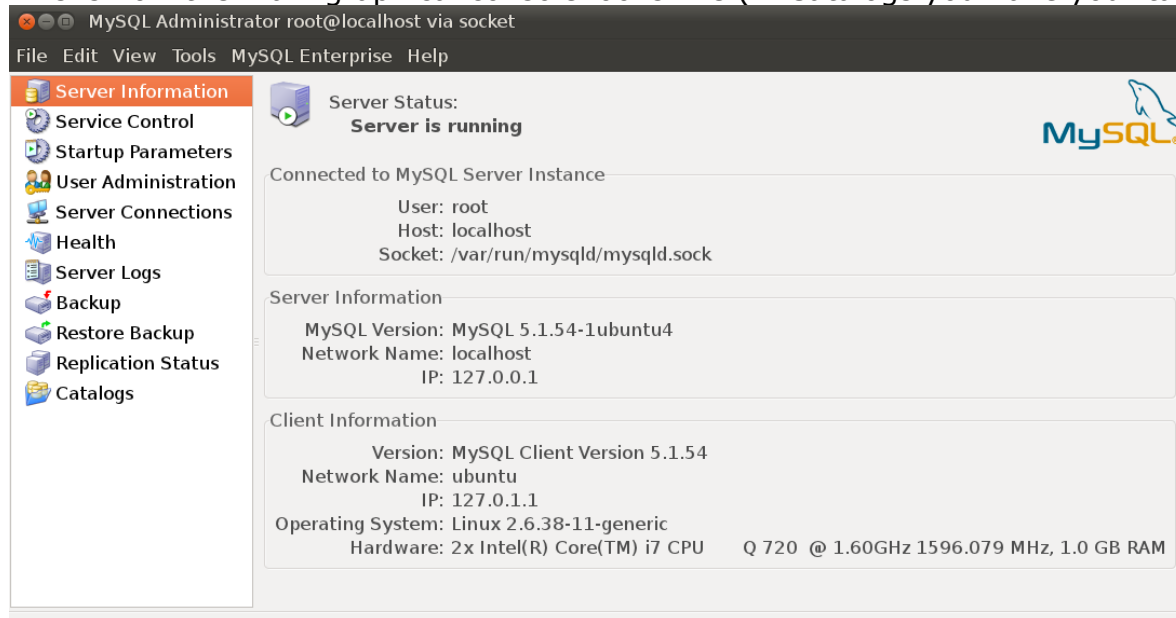
You can also enter in the graphical mode of administering MySQL by typing:

```
$ sudo mysql-admin
```




In the graphical admin interface you have to fill out the form to connect to the database server:
Server Hostname: localhost
Username: root (you can log as another user too)
Password: your password

This is how the main graphical console looks like (in Catalogs you have your tables):



(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

2) Install the MySQL Ruby library.

`$ sudo aptitude install libmysql-ruby`

```
user@ubuntu:~$ sudo aptitude install libmysql-ruby
The following NEW packages will be installed:
  libmysql-ruby libmysql-ruby1.8{a} libreadline5{a} libruby1.8{a}
0 packages upgraded, 4 newly installed, 0 to remove and 2 not upgraded.
Need to get 1,904 kB of archives. After unpacking 7,942 kB will be used.
Do you want to continue? [Y/n/?] █

Setting up libreadline5 (5.2-7build1) ...
Setting up libruby1.8 (1.8.7.302-2) ...
Setting up libmysql-ruby1.8 (2.8.2-1) ...
Setting up libmysql-ruby (2.8.2-1) ...
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place

user@ubuntu:~$ █
```

3) Install the gem "mysql2"

`$ gem install mysql2` (don't use "sudo")

```
user@ubuntu:~$ gem install mysql2
Fetching: mysql2-0.3.7.gem (100%)
Building native extensions. This could take a while...
Successfully installed mysql2-0.3.7
1 gem installed
Installing ri documentation for mysql2-0.3.7...
Enclosing class/module 'mMysql2' for class Result not known
Enclosing class/module 'cMysql2Result' for alias size count not known
Enclosing class/module 'mMysql2' for class Client not known
Installing RDoc documentation for mysql2-0.3.7...
Enclosing class/module 'mMysql2' for class Result not known
Enclosing class/module 'cMysql2Result' for alias size count not known
Enclosing class/module 'mMysql2' for class Client not known
user@ubuntu:~$ █
```

Check:

```
$ irb (hit Enter - this will switch to Ruby command prompt)
ruby-1.9.2-p290 :003 > require 'mysql2' (hit Enter - this should show "true")
=> true (exit the Ruby command prompt by typing "exit", then Enter)
```

4) Re-create your Rails application

Delete your Rails application created with the name "site" from "public_html" directory (or you can create another one with another name - but then you have to change things (the paths) in Apache

etc/apache2/sites-available/site file (the virtual host file)...from "site" to whatever you want for your new application).

Go to the "public_html" directory then run again the Rails command to create a new application in there:

```
$ rails new site -d mysql
```

 ("d mysql" will make sure the file "database.yml" from /public_html/site/config/ directory will be modified by adding MySQL as the database of use, not SQLite3)

If the "bundle install" command didn't run automatically then you need to start it manually. Go to your application directory (in this case "public_html/site/" and run:

```
$ bundle install
```

If you open up the Gemfile from your application directory ("site" in this case) you will see that now it uses "gem mysql2" as the database and not SQLite3.

Type "rails s", then Enter to start the WEBrick web server just to you see if you encounter the errors you encountered when installing Ruby on Rails with WEBrick and SQLite3 (see #9 and #10 above) follow the steps described there to solve them.

6) Open a browser window and type "http://localhost"

You should see the welcome page of your Ruby on Rails "site" project ("Welcome aboard. You're riding Ruby on Rails!" - click on the link to see details about your environment).

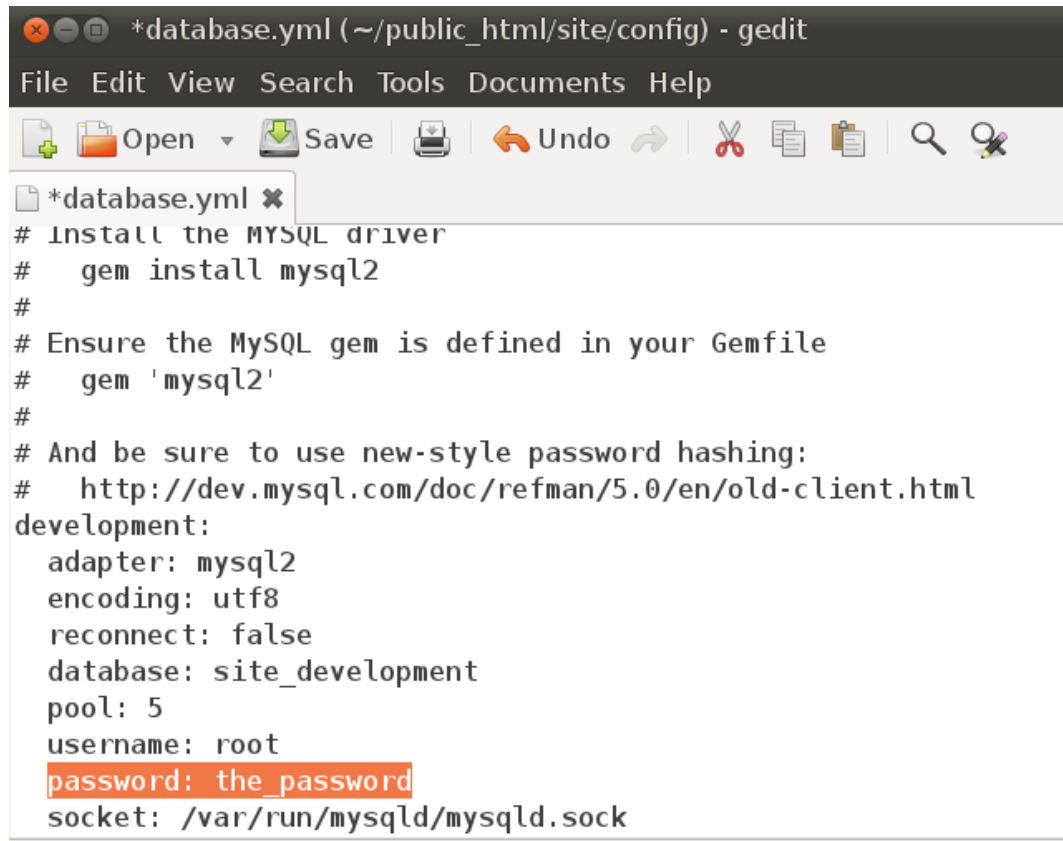
If you click on the link and you get this error:

Mysql2::Error (Access denied for user 'root'@'localhost' (using password: NO)):

Then open "database.yml" file /public_html/site/config/ directory and look for the lines "password". Type there your root MySQL password (if you didn't setup a password when you installed MySQL then you should get this error). You can also try to use your Linux system password you have.

```
$ sudo gedit /public_html/site/config/database.yml
```

 (if this "database.yml" file looks blank, then close it without saving and open it using your File Explorer and a text editor and edit it there)



```
*database.yml (~/.public_html/site/config) - gedit
File Edit View Search Tools Documents Help

# Install the MySQL driver
# gem install mysql2
#
# Ensure the MySQL gem is defined in your Gemfile
# gem 'mysql2'
#
# And be sure to use new-style password hashing:
# http://dev.mysql.com/doc/refman/5.0/en/old-client.html
development:
  adapter: mysql2
  encoding: utf8
  reconnect: false
  database: site_development
  pool: 5
  username: root
  password: the_password
  socket: /var/run/mysqld/mysqld.sock
```

You may have to stop/restart Apache.
Reload the webpage.

If now you get this error:

mysql2::Error (Unknown database 'site_development'):

Then you need to login to MySQL and create a blank database called "site_development" (it's the same name you have in "database.yml" file, look for it). See below how to do it.

a) Login to MySQL server as root

\$ mysql -u root -p ("-u" means user, "-p" will ask you the password)

You will now have the MySQL prompt "mysql>".

b) Create a blank database named "site_development"

mysql> create database site_development; (don't forget ; - then hit Enter)

You will get "Query OK, 0 rows affected (0.06 sec)" which means the database was created.

c) Reload now the localhost webpage and click again on that link

You should get the environment variables.

If still doesn't work you could try to restart Apache web server "**\$ sudo service apache2 restart**". Try again the link.

Now you have Ruby on Rails working together with Apache web server (via Phusion Passenger) and MySQL database.

You can start develop your own applications.

Anytime you encounter an error, then copy and paste it into Google and search/sort through the answers. You may have to try several solutions until you find the right one for your setup, just like I did for this tutorial.

See below a sample application created using Ruby on Rails scaffolding.

Creating an example to test the setup with Apache and MySQL #####
#####

We will create an example of a small web app (named "products") which interacts with the Apache web server and MySQL database.

We will use scaffolding to create this web app.

Scaffolding creates a model, controller and views for creating, retrieving, updating and deleting the resource we specify. It is also creating the database tables and columns for our data using migrations.

Scaffolding is use for demonstrations on development environment but usually professional Ruby on Rails developers **don't use** scaffolding in production.

1) Creating a new RoR application

We have already created one above named "site" so we don't need to create another one.

If you want to create a new one then, as usual, run the "\$ rails new your_app" command in the "public_html" directory.

Go to "site" directory from "public_html" directory:

```
$ cd public_html/site
```

2) Generate the scaffold

```
$ rails generate scaffold products title:string description:text price:decimal quantity:integer  
starts_at:datetime ends_at:datetime
```

```
user@ubuntu:~/public_html/site$ rails generate scaffold products title:string description:text
price:decimal quantity:integer starts_at:datetime ends_at:datetime
/home/user/.rvm/gems/ruby-1.9.2-p290/gems/rack-1.3.4/lib/rack/backports/uri/common_192.rb:53:
warning: already initialized constant WFKV_
Plural version of the model detected, using singularized version. Override with --force-plura
l.
  invoke  active_record
  create  db/migrate/20111007060153_create_products.rb
  create  app/models/product.rb
  invoke  test_unit
  create  test/unit/product_test.rb
  create  test/fixtures/products.yml
  route  resources :products
  invoke  scaffold_controller
  create  app/controllers/products_controller.rb
  invoke  erb
  create  app/views/products
  create  app/views/products/index.html.erb
  create  app/views/products/edit.html.erb
  create  app/views/products/show.html.erb
  create  app/views/products/new.html.erb
  create  app/views/products/_form.html.erb
  invoke  test_unit
  create  test/functional/products_controller_test.rb
  invoke  helper
  create  app/helpers/products_helper.rb
  invoke  test_unit
  create  test/unit/helpers/products_helper_test.rb
  invoke  assets
  invoke  coffee
  create  app/assets/javascripts/products.js.coffee
  invoke  scss
  create  app/assets/stylesheets/products.css.scss
  invoke  scss
  identical app/assets/stylesheets/scaffolds.css.scss
user@ubuntu:~/public_html/site$
```

Scaffolding will create new files and directories in your app directory (usually they start with the scaffold name, "products" in this case):

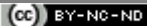
- db/migrate/20111007060153_create_products.rb
- app/models/product.rb
- test_unit/test/unit/product_test.rb
- test_unit/test/fixtures/products.yml
- scaffold_controller/app/controllers/products_controller.rb
- scaffold_controller/erb/app/views/products
- scaffold_controller/test/functional/products_controller_test.rb
- scaffold_controller/helper/app/helpers/products_helper.rb
- scaffold_controller/helper/test_unit/test/unit/helpers/products_helper_test.rb
- assets/coffee/app/assets/javascripts/products.js.coffee
- scss/app/assets/stylesheets/products.css.scss

3) Setting up the table "products" in MySQL

We use "rake" command (you will see the database Ruby script generated above in /site/db/migrate/. That script will be executed to populate the database with the command below.)

```
$ rake db:migrate
```

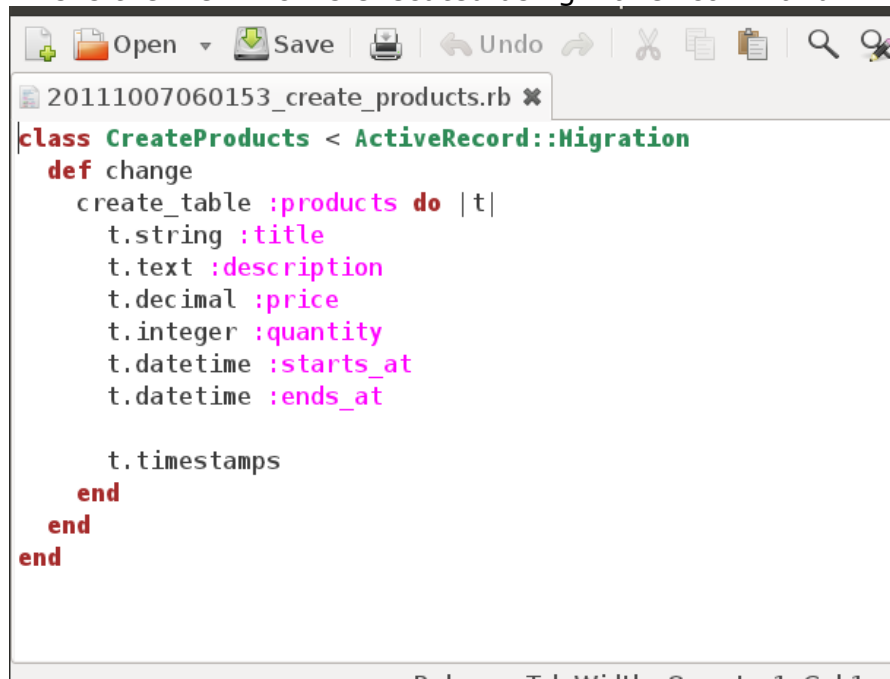
(with Apache and MySQL for Ubuntu 11.10, 11.04, Linux Mint 11,12)

(under Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License  – contact me for more details)

```
user@ubuntu:~/public_html/site$ rake db:migrate
== CreateProducts: migrating =====
-- create_table(:products)
-> 0.0525s
== CreateProducts: migrated (0.0526s) =====

user@ubuntu:~/public_html/site$
```

This is the file which is executed using "rake" command:



```
class CreateProducts < ActiveRecord::Migration
  def change
    create_table :products do |t|
      t.string :title
      t.text :description
      t.decimal :price
      t.integer :quantity
      t.datetime :starts_at
      t.datetime :ends_at

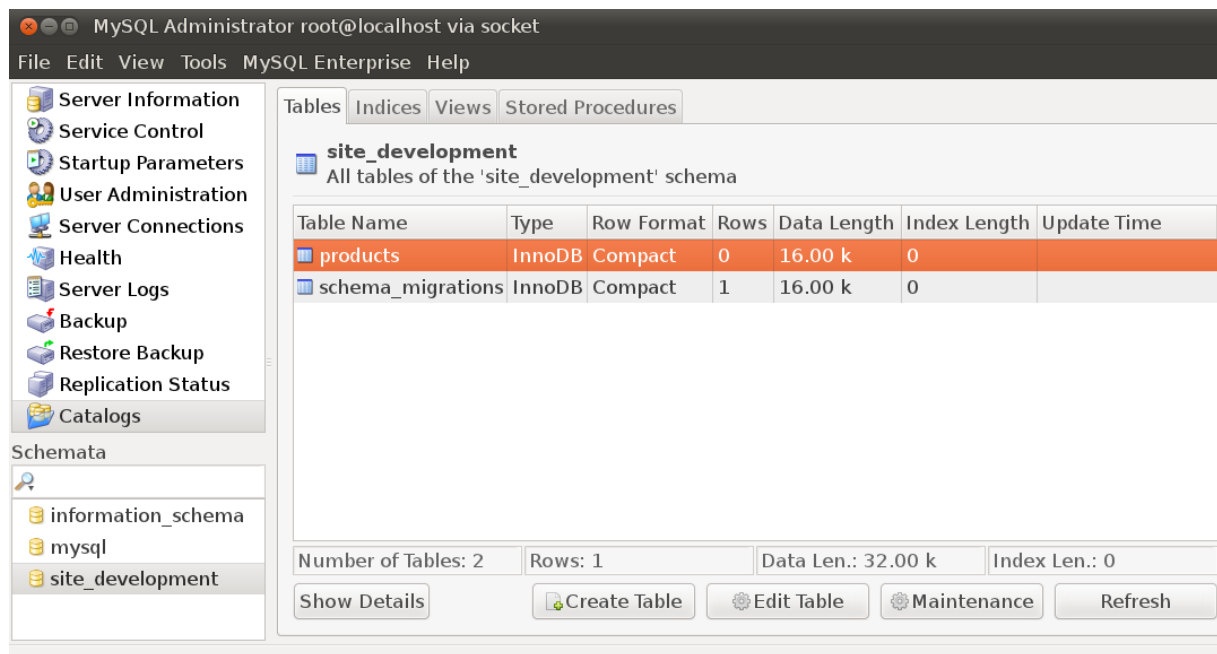
      t.timestamps
    end
  end
end
```

4) See the table created in MySQL

Open the MySQL admin console so you can see there the table "products" created by the "rake" command

\$ mysql-admin (then enter the hostname, username and password)

Open Catalogs, then "site_development" and see the table "products" there (we use "site_development" database because this is what we created by the RoR installation above).



Alternatively, you can use MySQL command prompt to see the table:

```
$ mysql -u root -p (enter the password and hit Enter)
```

See what databases you have there:

```
mysql> show databases;
```

Now switch to the database "site_development":

```
mysql> use site_development;
```

See what tables you have there:

```
mysql> show tables;
```

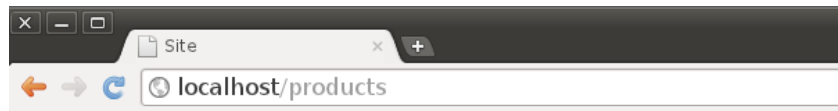
You should see now the table "products" (and another table "schema_migrations" produced by "rake" command - don't worry about this).

You can try to select something from the table "products" using SQL but it will be empty because we haven't added anything yet:

```
mysql> select * from products;
```

5) See the Products page in the browser

Open the browser and go to "**http://localhost/products**" to see your "products" page.
The "products" page is empty and we can add new records by clicking on "New Product" link.



Listing products

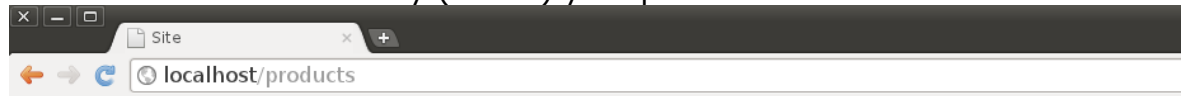
Title Description Price Quantity Starts at Ends at

[New Product](#)

Add some products in your app by clicking New Product.

A screenshot of a web browser window showing the 'New product' form. The form has fields for Title, Description, Price, Quantity, Starts at, and Ends at. The Starts at and Ends at fields are date pickers. There is a 'Create Product' button at the bottom and a 'Back' link.

You can also Edit or Destroy (delete) your products.



Listing products

Title	Description	Price	Quantity	Starts at	Ends at	
Product 1	Test product on Ruby on Rails 3 with Apache and MySQL.	25	5	2011-10-19 04:26:00 UTC	2011-10-19 04:26:00 UTC	Show Edit Destroy
Product 2	This is another test product for Ruby on Rails 3.	10	10	2011-10-19 04:27:00 UTC	2011-10-19 04:27:00 UTC	Show Edit Destroy
Product 3	This is the third test product for Ruby on Rails 3.	50	3	2011-10-19 04:28:00 UTC	2011-10-19 04:28:00 UTC	Show Edit Destroy

[New Product](#)

If you try now to select your records at the MySQL prompt you will see your products:

```
mysql> select * from products;
```

We have tested now that our Ruby on Rails installation with Apache web server and MySQL database works and we can start building some more serious applications using this framework.

I highly recommend starting using this tutorial for building a simple Twitter application:

<http://ruby.railstutorial.org/ruby-on-rails-tutorial-book>

It will give you the basis of Ruby on Rails web development.

Resources:

www.rubyonrails.org (the official Ruby on Rails website)

www.railsinstaller.org (Windows Ruby on Rails installer)

www.ruby-lang.org (the official Ruby programming language site)

<http://httpd.apache.org> (the official Apache web server site)

www.mysql.com (the official MySQL database site)

www.vmware.com (the official site of the company which builds virtual machine software VMware)

www.ubuntu.com (the official Ubuntu Linux site)

www.linuxmint.com (the official Linux Mint site)

<http://ruby.railstutorial.org/ruby-on-rails-tutorial-book> (the site of Michael Hartl's Ruby on Rails Tutorial which teaches you how to build a Twitter-like application)

www.railscasts.com (video tutorials by Ryan Bates - some are free)

www.stackoverflow.com (very useful site for developers of all kind)

<http://groups.google.com/group/rubyonrails-talk> (Google group for Ruby on Rails developers)

<http://www.youtube.com/watch?v=LADHwoN2LMM> (Berkeley Ruby on Rails class – five parts)

http://www.youtube.com/results?search_query=ruby+on+rails+tutorials (tutorials on Youtube)