- Login to Edit

Ubuntu Documentation > Community Documentation > RubyOnRails

RubyOnRails

Contents

Alternative to manual installation

Some users may prefer to skip manual installation by using the unofficial pre-integrated TurnKey Ruby on Rails Appliance based on Ubuntu LTS.

Intro Ubuntu 8.04

These instructions are for Ubuntu 8.04 LTS (hardy). Instead of using `apt` we use gems because of what ReinH put best...

<ReinH> Let gems manage your rails package otherwise gem and apt-get will start fighting like Spock and Captain Kirk in Amok Time. If you haven't seen that episode, trust me: you don't want that.

Intro Ubuntu 9.04 ... Apache2.2 tested only

Slight additions from 8.04 documentation...

- Using Passenger (mod_rails) may be required, it was for me. see http://www.modrails.com/documentation/Users%20guide.html Do NOT try a manual installation, use gems only.
- Do the minimum Ruby installation from Debian packages (apt-get | Aptitude | Synaptic). First install Ruby and Apache and rubygems as described below using Debian packages, then use gems only for the other Ruby-based applications.
- Use an Apache vhost configuration that puts the myapp/public directory at the document

root. see http://httpd.apache.org/docs/2.2/vhosts/name-based.html

- Using an .htaccess file in the myapp/public directory may be required.
- Using fastcgi or fcgid Apache modules is not required, and may not work at all.
- The Rails group is another good place to ask for assistance... http://groups.google.com/group/rubyonrails-talk
- The Rails guide is excellent... http://guides.rubyonrails.org/index.html

Get Ruby

```
sudo apt-get install ruby-full build-essential
```

Note that `ruby-full` will install emacs and a slew of other non-essential packages. You can install all the necessary packages with this command:

```
sudo aptitude install ruby build-essential libopenssl-ruby ruby1.8-dev
```

Server Installation

Now you have ruby, you already got a server included, WEBrick. You may settle with that, also because the installation is much easier, and reboots are faster. But it lacks the configuration options that apache and lighttpd offer. So if you want to have RubyOnRails running on apache or lighttpd, make sure you have installed these first.

Apache

```
sudo apt-get install apache2 apache2-mpm-prefork apache2-prefork-dev
```

WARNING: libapache2-mod-passenger installation uninstalls libapache2-mod-php5 and php5.

Apache + Mongrel

Mongrel Cluster works as a proxy service under Apache (no additional modification is required).

```
sudo a2enmod proxy
sudo a2enmod proxy_http
sudo a2enmod rewrite
sudo /etc/init.d/apache2 restart
sudo apt-get install mongrel
```

Rails Wiki - Apache + Mongrel RubyForge Mongrel - Apache Best Practice Deployment

Lighttpd

Preparing the house:

```
sudo apt-get install lighttpd libfcgi-ruby1.8
```

Installing RubyGems

The link below is to a specific version of RubyGems (version 1.1). You can get a later version from http://rubyforge.org/frs/?group_id=126, but it should not be really necessary because we ask RubyGems to update itself at the end. But to make it more complete, a description of RubyGems 1.2 is also included, below the install instructions of RubyGems 1.2.

RubyGems via apt

```
sudo apt-get install rubygems
```

As of July 27, 2009, RubyGems 1.3.1 is installed via apt.

To make gem work properly you should write gem's path to PATH add to ~/.bashrc:

```
export PATH=/var/lib/gems/1.8/bin:$PATH
```

To export path once just enter previous code to bash.

RubyGems 1.1.0

```
wget http://rubyforge.org/frs/download.php/34638/rubygems-1.1.0.tgz
tar xzvf rubygems-1.1.0.tgz
cd rubygems-1.1.0
sudo ruby setup.rb
sudo ln -s /usr/bin/gem1.8 /usr/bin/gem
sudo gem update --system
```

RubyGems 1.2.0

```
wget http://rubyforge.org/frs/download.php/38646/rubygems-1.2.0.tgz
tar xzvf rubygems-1.2.0.tgz
cd rubygems-1.2.0
sudo ruby setup.rb
sudo ln -s /usr/bin/gem1.8 /usr/bin/gem
sudo gem update --system
```

The below error is sometimes encountered after running sudo gem update --system.

```
/usr/bin/gem:23: uninitialized constant Gem::GemRunner (NameError)
```

To fix that error follow the below instructions.

```
sudo gedit /usr/bin/gem

In the file find:
require 'rubygems'

Add the following line below the above line:
require 'rubygems/gem_runner'
```

RubyGems 1.3.1 Ubuntu 8.04

```
wget http://rubyforge.org/frs/download.php/45905/rubygems-1.3.1.tgz
tar xzvf rubygems-1.3.1.tgz
cd rubygems-1.3.1
sudo ruby setup.rb
sudo ln -s /usr/bin/gem1.8 /usr/bin/gem
sudo gem update --system
```

RubyGems 1.3.1 Ubuntu 9.04

```
sudo apt-get install rubygems1.8
sudo gem install rubygems-update
sudo update_rubygems     note: this will clean out your gems!
```

Installing rails

We can now use RubyGems to install rails:

```
sudo gem install rails
```

Also we can install rails from apt (not suggested for Ubuntu 9.04)

```
sudo apt-get install rails
```

I think, it would be better to install rails with gem.

If you encounter difficulties installing fcgi (fastcgi) via gem, this might help:

```
sudo aptitude install libfcgi-dev
```

Preparing for your first rails app

Configuration of the webserver also assumes you have by then a rails application set up. Simply enter to get that done (it will create you a first test page):

Default (sqlite3 backend):

```
rails /home/myuser/www/mynewapp
```

For a mysql back-end. Run this one, if you installed rails with gem:

```
rails /home/myuser/www/mynewapp -d mysql
```

Run this one, if you installed rails with apt:

```
rails /home/myuser/www/mynewapp -D mysql
```

You will see a bunch of stuff on the shell... that should be fine (unless you see a error message). That created the rails folder with all the stuff, so now you should have something like have: /home/myuser/www/mynewapp/public

Configure the server

WEBrick (Ruby's server)

No configuring needed, everything is ready to create your first application. You can even try this step before you try launching your application via apache or lighttpd:

```
$cd /home/myuser/www/mynewapp
$ruby script/server
=> Booting WEBrick...
=> Rails 2.1.0 application started on http://0.0.0.0:3000
=> Ctrl-C to shutdown server; call with --help for options
[2008-07-28 04:39:17] INFO  WEBrick 1.3.1
[2008-07-28 04:39:17] INFO  ruby 1.8.6 (2007-09-24)
[2008-07-28 04:39:17] INFO  WEBrick::HTTPServer#start: pid=6860
port=3000
```

Your application is now responding at `http://0.0.0.0:3000`; press Ctrl+C to stop the web server. If you get errors related to the database, **you might need to install a DB connector, or create a database.** Edit this in `$YOUR_RAILS_DIRECTORY/config/database.yml`

Install the sqlite3 connector:

```
sudo apt-get install libsqlite3-dev
sudo gem install sqlite3-ruby
```

Or install the MySQL database and it's connector:

```
sudo apt-get install mysql-server mysql-client
sudo apt-get install libmysql-ruby libmysqlclient-dev
sudo gem install mysql
```

Or install the PostgreSQL database and it's connector:

```
sudo apt-get install postgresql-server postgresql-client
sudo apt-get install libpq-dev
sudo gem install pg
```

Apache

Run the following to install passenger, an apache module that adds support for running rails apps within apache:

```
sudo gem install passenger
sudo apt-get install apache2-dev libapr1-dev libaprutil1-dev
sudo passenger-install-apache2-module
```

WARNING: libapache2-mod-passenger installation uninstalls libapache2-mod-php5 and php5.

Now you have to configure apache to point to the passenger module. Simply copy the following suggested code from the `passenger-install-apache2-module` installation to the end of your Apache configuration file, which can be accessed by typing in the command `sudo gedit /etc/apache2/apache2.conf`:

(if you are using RubyGems 1.1):

```
LoadModule passenger_module /usr/lib/ruby/gems/1.8/gems/passenger-
1.0.3/ext/apache2/mod_passenger.so
RailsSpawnServer /usr/lib/ruby/gems/1.8/gems/passenger-
1.0.3/bin/passenger-spawn-server
RailsRuby /usr/bin/ruby1.8
```

(if you are using RubyGems 1.2):

```
LoadModule passenger_module /usr/lib/ruby/gems/1.8/gems/passenger-
2.0.3/ext/apache2/mod_passenger.so
PassengerRoot /usr/lib/ruby/gems/1.8/gems/passenger-2.0.3
PassengerRuby /usr/bin/ruby1.8
```

(if you are using RubyGems 1.3.1):

```
LoadModule passenger_module /usr/lib/ruby/gems/1.8/gems/passenger-
2.0.6/ext/apache2/mod_passenger.so
PassengerRoot /usr/lib/ruby/gems/1.8/gems/passenger-2.0.6
PassengerRuby /usr/bin/ruby1.8
```

(if you are using RubyGems 1.3.6 and Ubuntu 9.10):

```
LoadModule passenger_module
/usr/local/lib/ruby/gems/1.8/gems/passenger-
2.2.11/ext/apache2/mod_passenger.so
PassengerRoot /usr/local/lib/ruby/gems/1.8/gems/passenger-2.2.11
PassengerRuby /usr/local/bin/ruby
```

Now you should point apache to the location of your Ruby app. Create something like this in /etc/apache2/sites-available/ror.myhost.com

```
<VirtualHost *>
    ServerName ror.myhost.com
    DocumentRoot /home/myuser/www/mynewapp/public
    RailsEnv development #It's optional string, but my rails app didn't
work properly without it.
</VirtualHost>
```

Alternatively, you could run rails in a specific sub uri of your current host, e.g. localhost. First, make a symbolic link to your rails public folder from with var/www (or wherever you have the root of your apache www folder), e.g.:

```
ln -s /home/myuser/www/mynewapp/public /var/www/mynewapp
```

In case of localhost (which is the default host with a fresh ubuntu/apache install) edit `default` in `/etc/apache2/sites-available` and add:

```
RailsBaseURI /mynewapp
```

murb: I had to add `RailsEnv development` *as well to get around the 'no route* *found to match "/rails/info/properties" with {:method=>:get}' warning... (can someone elaborate on why?)* apparently this is because /info/properties is buggy and no longer supported: http://www.ruby-forum.com/topic/161924

Save, and enable the rewrite module for apache, and register this site... and finally, restart apache.

```
sudo a2enmod rewrite
sudo a2ensite ror.myhost.com
sudo /etc/init.d/apache2 restart
```

An Alternate Apache Configuration with a named vhost Ubuntu 9.04

If you need to run php-based or other web applications on the same server, and if you can get a DNS change to point another name at the same ip address as your server, this configuration should work for you. Note: for the Rails app you must put its public folder in the Document Root, and that means you'll need two or more vhosts on your server.

Lets say your old PHP apps are available at www.office.example.com, and the Rails app, myrailsapp, will be at www.rails.example.com. Edit your /etc/apache2/sites-available/default file and add this to the end of it:

```
 LoadModule passenger_module /usr/lib/ruby/gems/1.8/gems/passenger-2.2.5/ext/apache2/mod_passenger.so

<VirtualHost *:80>
        ServerAdmin admin@example.com
        ServerName www.rails.example.com
        DocumentRoot /var/www/myrailsapp/public/
        PassengerRoot /usr/lib/ruby/gems/1.8/gems/passenger-2.2.5
        PassengerRuby /usr/bin/ruby1.8

        <Directory "/var/www/myrailsapp/public/">
                Options Indexes +ExecCGI FollowSymLinks
                Order allow,deny
                Allow from all
        </Directory>

</VirtualHost>
```

- The loadmodule passenger line is required and will be given to you by the passenger setup script. Use whatever path it provides.
- You MUST change your local DNS to point www.rails.example.com to the same ip address as www.office.example.com, see your administrator or your ISP.
- The PassengerRoot and PassengerRuby are also required.
- As usual with Apache there are many other ways to set up vhosts, see http://httpd.apache.org/docs/2.2/vhosts/

The .htaccess file was needed by my application in the myapp/public directory (note that it assumes a dispatch.cgi file):

```
Options +FollowSymLinks +ExecCGI

RewriteRule ^(.*)$ dispatch.cgi [QSA,L]
RewriteEngine On

RewriteRule ^$ index.html [QSA]
RewriteRule ^([^.]+)$ $1.html [QSA]
RewriteCond %{REQUEST_FILENAME} !-f
```

Apache + Mongrel

You have to start Mongrel Cluster (under www-data ?).

```
sudo mongrel_rails start -d -p 8000 -e production -P
/full/path/to/log/mongrel-1.pid
```

And configure Apache's vhost:

```
<VirtualHost *:80>
        ServerName otherserver.me

        ProxyPass / http://otherserver.me:8000/
        ProxyPassReverse / http://otherserver.me:8000
        ProxyPreserveHost on

  <Proxy *>
    Order allow,deny
    Allow from all
  </Proxy>

</VirtualHost>
```

RubyForge Mongrel - Apache Best Practice Deployment

Lighttpd

Edit lighttpd.conf, e.g.:

```
sudo kate /etc/lighttpd/lighttpd.conf
```

and here starts the tricky part, paste at the end of the file:

```
server.modules   += ( "mod_fastcgi", "mod_rewrite" )


$HTTP["host"] == "localhost" {
        server.document-root =
"/home/santiago/proyectos/beholder/public/"

        server.dir-listing          = "disable"
        server.error-handler-404 = "/dispatch.fcgi"

        fastcgi.server      = ( ".fcgi" => ( "localhost" => (
                "min-procs"       => 1,
                "max-procs"       => 1,
                "socket" => "/tmp/ruby-beholder.socket",
                "bin-path" =>
"/home/santiago/proyectos/beholder/public/dispatch.fcgi",
                "bin-environment" => ( "RAILS_ENV" => "development" )
        ) ) )

}
```

now run

```
sudo /etc/init.d/lighttpd restart
```

and you're set, go to http://localhost and you should see the rails index... click on the link "About your rails environment" and if you don't see a 404 or 500 error... then you're set (you should see either a rails error, or a rails message or something like that).

Known Problems

just in case you get a message like this:

```
kmin@metakmin:~/ruby_test$ gem -v
1.2.0
kmin@metakmin:~/ruby_test$ rails -v
Rails 2.2.2
kmin@metakmin:~/ruby_test$ gem update --system
Updating RubyGems
Nothing to update
kmin@metakmin:~/ruby_test$ script/server
Rails requires RubyGems >= 1.3.1 (you have 1.2.0). Please `gem update
--system` and try again.
kmin@metakmin:~/ruby_test$
```

dont panic, just run:

```
sudo gem install rubygems-update
sudo update_rubygems
```

If you get a message like this when starting WEBrick on 9.10

```
/usr/lib/ruby/gems/1.8/gems/rails-2.3.5/lib/initializer.rb:271:in
`require_frameworks': no such file to load -- net/https (RuntimeError)
```

You may need to install these:

```
sudo apt-get install build-essential libopenssl-ruby
sudo apt-get install ruby rubygems ruby1.8-dev libsqlite3-dev
```

* thanks to Tomasz Przybysz who mailed me with the problem and then with the solution

for any comments on the lighttpd install please feel free to mail me to santiago-ve@ubuntu.com

CategoryDevelopment

RubyOnRails (last edited 2010-04-29 08:01:21 by https://login.launchpad.net/+id/W7RnhRW @ 93-172-96-108.bb.netvision.net.il[93.172.96.108]:bricel)

- Page History