# **Setup Ruby On Rails on**

# **Ubuntu 14.10**

## Prerequisites

a regular, non-root user with sudo privileges configured on server.

OS - Ubuntu 14.10

# **Introduction to MVC(Model View Controller)**

* **Model:** manages data between the rest of the application and the database. You can define how a single entity behaves, this includes data validation, before/after save hooks, etc…
* **Controller:** this is the *glue* between the data managed by the Models and the rendered Views. Http requests that come to your app, are *routed* to a controller which usually will interact with one or more models and finally renders the ouput along with an Http response.git@github.com:swapnild-webonise/cap-deployment.git
* **View:** the final output for a request. It’s usually a piece of HTML, but it might also be JSON or XML. We’ll see this later.

## mvc_schematic

## Figure 1 MVC(Model View Controller)

## 

## Installing Ruby *(Using source code)*

[*https://gorails.com/setup/ubuntu/14.10*](https://gorails.com/setup/ubuntu/14.10)

$sudo apt-get update

$sudo apt-get install git-core curl zlib1g-dev build-essential libssl-dev libreadline-dev libyaml-dev libsqlite3-dev sqlite3 libxml2-dev libxslt1-dev libcurl4-openssl-dev python-software-properties

cd  
wget http://ftp.ruby-lang.org/pub/ruby/2.1/ruby-2.1.3.tar.gz  
tar -xzvf ruby-2.1.3.tar.gz  
cd ruby-2.1.3/  
./configure  
make  
sudo make install  
ruby -v

The last step is to tell Rubygems not to install the documentation for each package locally

echo "gem: --no-ri --no-rdoc" > ~/.gemrc

## Installing Rails

Since Rails ships with so many dependencies these days, we're going to need to install a Javascript runtime like NodeJS. This lets you use Coffeescript and the Asset Pipeline in Rails which combines and minifies your javascript to provide a faster production environment.

To install NodeJS, we're going to add it using a PPA repository:

sudo add-apt-repository ppa:chris-lea/node.js  
sudo apt-get update  
sudo apt-get install nodejs

And now, without further adieu:

gem install rails

If you're using rbenv, you'll need to run the following command to make the rails executable available:

rbenv rehash

Now that you've installed Rails, you can run the rails -v command to make sure you have everything installed correctly:

rails -v  
# Rails 4.1.6

If you get a different result for some reason, it means your environment may not be setup properly.

## Setting Up MySQL

sudo apt-get install mysql-server mysql-client libmysqlclient-dev

## Final Steps

And now for the moment of truth. Let's create your first Rails application:

#### If you want to use SQLite (not recommended)  
rails new myapp  
  
#### If you want to use MySQL  
rails new myapp -d mysql  
  
#### If you want to use Postgres  
# Note that this will expect a postgres user with the same username  
# as your app, you may need to edit config/database.yml to match the  
# user you created earlier  
rails new myapp -d postgresql  
  
# Move into the application directory  
cd myapp  
  
# If you setup MySQL or Postgres with a username/password, modify the  
# config/database.yml file to contain the username/password that you specified  
  
# Create the database  
rake db:create  
  
rails server

You can now visit<http://localhost:3000> to view your new website!

**Hello, world app example**

mkdir projects

cd projects

rails new hello\_app

cd hello\_app

rails srever

Test by <http://localhost:3000> in browser

vim app/controllers/application\_controller.rb

add thhis inside class:

def hello\_app  
 render text: "hello, world!"  
end

**The default (commented-out) root route.**

uncomment

#root 'welcome#index'

and change it to

root 'application#hello\_app'

now visit <http://localhost:3000>

**NOtes**

|  |  |
| --- | --- |
| **File/Directory** | **Purpose** |
| app/ | Core application (app) code, including models, views, controllers, and helpers |
| app/assets | Applications assets such as cascading style sheets (CSS), JavaScript files, and images |
| bin/ | Binary executable files |
| config/ | Application configuration |
| db/ | Database files |
| doc/ | Documentation for the application |
| lib/ | Library modules |
| lib/assets | Library assets such as cascading style sheets (CSS), JavaScript files, and images |
| log/ | Application log files |
| public/ | Data accessible to the public (e.g., via web browsers), such as error pages |
| bin/rails | A program for generating code, opening console sessions, or starting a local server |
| test/ | Application tests |
| tmp/ | Temporary files |
| vendor/ | Third-party code such as plugins and gems |
| vendor/assets | Third-party assets such as cascading style sheets (CSS), JavaScript files, and images |
| README.rdoc | A brief description of the application |
| Rakefile | Utility tasks available via the rake command |
| Gemfile | Gem requirements for this app |
| Gemfile.lock | A list of gems used to ensure that all copies of the app use the same gem versions |
| config.ru | A configuration file for [Rack middleware](http://rack.github.io/) |
| .gitignore | Patterns for files that should be ignored by Git |

**ROR CheetSheet**

PROCESS BASICS

All processes, with params + hierarchy

ps auxww -H

Show all ruby-related PIDs and processes

pgrep -fl ruby

What is a process doing?

strace -f -p $PID

What files does a process have open?

(also detects ruby version of a process)

lsof -p $PID

Flavors of kill

kill xxxx

kill xxxx yyyy zzzz

pkill <name of process>

pkill -f <word in processname>

Keep an eye on a process

watch ‘ps aux | grep ruby’

TIPS N TRICKS

Run Previous command as root

sudo !!

Change to last working dir

cd -

Run something forever

while true;do ruby ghetto.

rb;done

MEMORY

How much mem is free?

free -m

cat /proc/meminfo

Are we swapping?

First line is avg since boot.

vmstat 1

List the top 10 memory hogs

ps aux --sort=-resident|head -11

Detect OOM and other bad things

for i in messages kern.log

syslog; do egrep -i “s[ie]

g|oo(m|ps)” /var/log/$i{,.0};

done

Disable OOM killer for a process

echo -17 > /proc/$PID/oom\_adj

TERMINAL & SCREEN

Start a screen session as the current user

screen -x

Join/re-attach to a screen session

screen -r

Record a terminal session

script filename.out 2> filename.

timing

Playback a recorded terminal session

scriptreplay filename.timing

filename.out

DISK/FILE

S

Check reads/writes per disk

iostat -xnk 5

Files (often logs) marked for deletion but

not yet deleted

lsof | grep delete

Overview of all disks

df -h

Usage of this dir and all subdirs

du -hs

Find files over 100MB

find . -size +100M

Low hanging fruit for free space. Check /var/

log too!

ls -al /tmp

Find files created within the last 7 days

find . -mtime -7

Find files older than 14 days

find . -mtime +14 -type f -name

‘\*.gz’

Delete files older than 14 days

find \*.gz -mtime +14 -type f

-exec rm {} \;

Monitor a log file for an IP or anything else

tail -f file.log | grep

192.168.1.1

NETWORK

TCP sockets in use

lsof -nPi tcp

Get IP/Ethernet info

ip addr

ifconfig

host <=> IP resolution

host 192.168.1.1

host MyRouter

Curl, display headers (I), follow redirects (L)

curl -LI http://google.com

Traceroute with stats over time (top for

traceroute) Requires install

mtr google.com

Traceroute TCP to avoid ICMP blockage

tcptraceroute google.com

List any IP blocks/rules

iptables -L

Drop any network requests from IP

iptables -I INPUT -s

66.75.84.220 -j DROP

Show traffic by port

iftop

Show all ports listening with process PID

netstat -tlnp

D/L speed test (don’t run in prod! :)

wget cachefly.cachefly.

net/100mb.test -O /dev/null