

Ruby On Rails BeijingLUG

by Nathaniel Brown
April 2007

Overview

- Install Ruby
- Install Rails
- (Optional) Install Eclipse
- Talk about Ruby
- Talk about Rails
- Example Application
- Links to Resources

Install Ruby

- `sudo apt-get install ruby irb ri rdoc`
- If you want to install mongrel you will also have to install the dev packages.
- `sudo apt-get install ruby1.8-dev build-essential`

Install Ruby Gems

- `wget` : <http://rubyforge.org/frs/download.php/17190/>
- `tar xzvf rubygems-0.9.2.tgz`
- `cd rubygems-0.9.2`
- `sudo ruby setup.rb`
- `cd ..`
- `rm -rf rubygems-0.9.2`

Install Rails and Mongrel

- `sudo gem install rails --include-dependencies`
- (Optional) `sudo gem install mongrel --include-dependencies`
- (Optional) Select the latest version of each library.

Install MySQL Server

- You could use PostgreSQL or SQLite instead but most examples use MySQL so we will as well.
- `sudo apt-get install mysql-server phpmyadmin libmysql-ruby apache2 libapache2-mod-php5`

(Optional) Install Eclipse

- `sudo apt-get install eclipse sun-java5-jdk sun-java5-fonts`

(Optional) Install RDT, RadRails and Aptana and Subclipse

- Open Eclipse and go to Help ->Software Updates -> Find and Install...
- Select “Search for new features to install” then click on “Next”.
- Add the following remote sites:
RDT : <http://updatesite.rubypeople.org/release>
RadRails : <http://radrails.sourceforge.net/update>
Aptana : <http://update.apтана.com/install/>
Subclipse : http://subclipse.tigris.org/update_1.2.x
- Once they have been added install all of them.

(Optional) Config RDT and RadRails

- Open Eclipse and go to Windows -> Preferences
- Find Ruby and add the Ruby interpreter.
- Find Rails and add the applications. They should all be in `/usr/bin/*app name*`

Ruby is a nice language

- Object Orientated
- Automatic garbage collection
- Sane readable code that makes sense.

Ruby example code 1

- Is readable

```
puts "leon".reverse.capitalize
```

```
staff_list = ["joe", "steve", "bob"]
```

```
staff_list.sort!
```

```
staff_list.each do |first_name|
```

```
  puts first_name.capitalize
```

```
end
```

```
3.times do |zork| puts "#{zork} " end
```

Ruby example code 2

-Tells you if it will change the variable.

```
my_string = "12345"  
new_string = my_string.chop  
puts "String chopped."  
puts "my_string: #{my_string}"  
puts "new_string: #{new_string}"
```

```
my_string = "98765"  
puts "my_string: #{my_string}"  
my_string.chop!  
puts "String chopped."  
puts "my_string: #{my_string}"
```

```
my_number = 999474  
puts my_number.to_s.include?("94")
```

```
my_string = "This is a test."  
puts "#{my_string[3..5]}"  
puts "#{my_string[2]}"  
my_string[10..13] = "Ruby"  
puts my_string
```

Ruby example code 3

-Lots of things are just done really nicely.

```
#create our thread array
threads = []

5.times do |i|
  #create 5 threads into the array
  threads[i] = Thread.new do
    #wait a random amount of time, then print a message
    sleep(rand(10))
    puts "I am thread number #{i}"
  end
end

#let each thread finish before ending the program
threads.each {|t| t.join}
```

Rails is a nice Framework

- Rapid Application Development
- Caching
- Model View Controller
- Nice informative errors

- DRY – Don't Repeat Yourself.
- Convention over Configuration

Rails supports modern features

- AJAX is build into the framework but JavaScript isn't required.
- Pretty URLs
- Active Record which is a type of Object Relation Mapping (ORM)

Rails encourages good development practices

- Validation
- Unit testing
- Use of development, testing and production environments with sane settings for each of them. e.g. Development has caching turned off. Production and Testing have it turned on.
- Version control is encouraged and assumed.

The Bad

- No manual only books.
- Ruby is relatively slow. (See Language Shoot-out)
- Rails is database driven which means it scales with the database.
- You'll often have to do things the “Rails Way”. (But it's often the right way as well)
- Lots of the 3rd party libraries are still unstable. (Improving quickly)

Create Your First Rails Applications

- If you are reading this online then go to the links section find RadRail's home page. He has a fairly good tutorial using RadRails.

Resources Links

- Rails for Fedora - <http://digitalmediaminute.com/howto/fc4rails/>
- RDT - <http://rubyclipse.sourceforge.net/>
- RadRails and Aptana - <http://www.radrails.org/>
- Language Shoot-out - <http://shootout.alioth.debian.org/>

Cheat Sheets Links

- Rails, CSS, Javascript, HTML, etc (A4) - <http://www.ilovejackdaniels.com/>
- Nice Rails summary - <http://www.blainekendall.com/index.php/rubyonrails>
- A couple of Rails Cheat Sheets - <http://www.slash7.com/goodies>

Rails Links

- Ruby - <http://www.ruby-lang.org/en/>
- Ruby on Rails - <http://www.rubyonrails.org/>
- Ruby Forge - <http://rubyforge.org/>
- Free Ruby manual - <http://www.poignantguide.net/ruby/>
- Rails and Caching -
<http://www.railsenvy.com/2007/2/28/rails-caching-tutorial>
- Ruby On Rails to exe -
<http://www.erikveen.dds.nl/distributingrubyapplications/rails.html>
- Rails Tricks -
<http://www.rubyinside.com/19-rails-tricks-most-rails-coders-dont->
- Rails Deployment -
<http://manuals.rubyonrails.com/read/book/17>