Ruby on Rails Training - 1

Base Rails

Summary: Rails is a powerful open-source framework that combines high productivity and code quality, while benefiting from a large passionate and highly responsive community. With this module04, you will tame its most accessible parts.

Version: 1.1

Contents

I Preamble 2
II General rules 3
III Today's specific instructions 4
IV Exercise 00: CheatSheet 5
V Exercise 01: Moar CheatSheet 7
VI Exercice 02: Quick search 9
VII Exercise 03: Diary 10
VIII Submission and peer-evaluation 11

Chapter I

Preamble

Philosophy

- DRY : Don’t Repeat Yourself
- Convention over configuration
- REST : Representational State Transfer
- CRUD : Create, Read, Update, Delete

Chapter II

General rules

- Your project must be realized in a virtual machine.
- Your virtual machine must have all the necessary software to complete your project. These softwares must be configured and installed.
- You can choose the operating system to use for your virtual machine.
- You must be able to use your virtual machine from a cluster computer.
- You must use a shared folder between your virtual machine and your host machine.
- During your evaluations you will use this folder to share with your repository.
- Your functions should not quit unexpectedly (segmentation fault, bus error, double free, etc) apart from undefined behaviors. If this happens, your project will be considered non functional and will receive a `0` during the evaluation.
- We encourage you to create test programs for your project even though this work **won't have to be submitted and won't be graded**. It will give you a chance to easily test your work and your peers' work. You will find those tests especially useful during your defence. Indeed, during defence, you are free to use your tests and/or the tests of the peer you are evaluating.
- Submit your work to your assigned git repository. Only the work in the git repository will be graded. If Deepthought is assigned to grade your work, it will be done after your peer-evaluations. If an error happens in any section of your work during Deepthought's grading, the evaluation will stop.

Chapter III

Today’s specific instructions

- All submitted files must be Ruby On Rails web applications.
- Every project in this module must include the following line in their Gemfile:

  ```
  gem 'rubycritic', :require => false
  ```

  When executing the "rubycritic" command, the “smells” section of the HTML report page must be EMPTY!
- Unless specified in the subject, NO additional gems are allowed to complete these exercises.
- If no gem is available, you can use Yarn to install what you need.

Chapter IV

Exercise 00: CheatSheet

	Exercise 00
Exercise 00: CheatSheet	
Turn-in directory : *ex*00/	
Files to turn in : `CheatSheet`	
Allowed functions : `bootstrap, rubycritic`	

You are feeling satisfied after completing your first "Hello World" in rails in the previous module, you are motivated to explore further with your new companion.

However, not too far just yet. In this module, we will focus on utilizing Rails in its simplest mechanisms.

If Wikipedia took up most of your time in the previous module, it is advisable to complete Module03 - ex04 before proceeding.

For now, there are, NO DATABASE or CRUD operations.Our focus remainson the views.

You should visit: here and marvel at this wonderful site tful command reminders.

Your task is to replicate that page, including the footer. You are only focusing on the page itself; the navigation system will be addressed later. Remember not to spend too much time on pixel-perfect css, but do aim for some layout and a certain resemblance.

These are the points to be followed:

- The app will be called “CheatSheet”.
- It contains only one controller of you creation `application_controller` which should be left as is.
- The main page will have the title “CheatSheet”.
- The navigation bar (the red thing at the top of the page) does not need to be reproduced at the moment.
- Some layout is required. We don’t expect a production-worthy design, but rather a simple and visually pleasing layout.

Your HTML should be where it’s supposed to be, and the controller should be in its designated folder. No CSS should be present in an HTML file! Make use of Rails as intended.

By the way, don’t be foolish. The CheatSheet lives up to its name and will help you a lot with the Rails application structure and useful commands.

Also, make use the “inspect element” command of your browser and be clever. Remember the crawler in the previous module.

Chapter V

Exercise 01: Moar CheatSheet

	Exercise 01
Exercise 01: Moar CheatSheet	
Turn-in directory : *ex*01/	
Files to turn in : `NewCheatSheet`	
Allowed functions : `bootstrap, rubycritic`	

Well, now that you know how to create a One-Page with Rails, it's time to get serious.

Take your previous application and start implementing the nav-bar (the red thing and all that) and make some changes to the navigation of your CheatSheet.

Specifically, you are separating each section of the CheatSheet into independent pages and assigning a nav-bar button to the corresponding page.

Here's what we expect from you:

- The app is named "NewCheatSheet".
- It still contains only one one controller of your own creation `application_controller` which should be left as is.
- Each page has its dedicated "title" tag.
- Write the code for the "nav-bar" only once!
- The first page is the "Convention" page, which serves as the root page.

- All the following pages must be included:
  - convention
  - console
  - ruby
  - ruby-concepts
  - ruby-numbers
  - ruby-strings
  - ruby-arrays
  - ruby-hashes
  - rails-folder-structure
  - rails-commands
  - rails-erb
  - editor
  - help
- The “Resources” and “About” pages should not be reproduced or included in the nav-bar.
- Some basic styling is required. We don’t expect a production-level design, but rather a simple and visually pleasing layout.

Chapter VI

Exercice 02: Quick search

	Exercise 02
Exercice 02: Quick search	
Turn-in directory : *ex*02/	
Files to turn in : `CheatSheet`	
Allowed functions : `jquery, bootstrap, datatables.net-bs4`	

This CheatSheet is useful, right? But it can still be improved.

You're going to enhance it even more with a `jQuery` plugin and some content re-formatting. Add a tab named "Quick search" that contains a table incorporating the content of ALL the other tables in the CheatSheet.

This table is `properly formatted`, including a header, the correct number of columns, well-placed tags etc.

Specifically you need to:

- Implement a new tab called "Quick search" in the nav-bar.
- Ensure that this tab has its own personalized title tag, just like the rest of the site.
- Make sure that its content is a summary table of ALL the commands from the other pages.

`NB`: If an error popup appears or if there is any Javascript present in the html.erb file, the exercise will be considered incorrect.

Chapter VII

Exercise 03: Diary

	Exercise 03
Exercise 03: Diary	
Turn-in directory : *ex*03/	
Files to turn in : `CheatSheet`	
Allowed functions : `bootstrap, rubycritic, jquery`	

The CheatSheet project is gaining some momentum. But like any good developer, keeping a technical journal would be a valuable asset to incorporate into your project.

Create a new tab named “Log Book” that features a form at the top of the page with a “Write” button and a text field. Once the form is filled out with text and submitted, your app will add the form’s text to a file named “"entry_log.txt"” located at the root of your CheatSheet project folder.

The text should be dynamically formatted by adding the date and time at the beginning of each line, like this:

```
15/07/2016 23:42:04 : The whole crew is asleep, Jim.
15/07/2016 23:41:26 : The whole crew is yawning, Jim.
```

The display of the text is achieved using a loop : each entry should be displayed below the form, each with its own tag starting from the most recent entry to the oldest.

Every time the “Write” button of the form is clicked, you should be redirected to the same page with the new line displayed at the top.

Chapter VIII

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your folders and files to ensure they are correct.

The evaluation process will happen on the computer of the evaluated group.