



D04 - Ruby on Rails Training

Rails 101

Summary: Rails is a powerful opensource framework combining high productivity and quality code while benefiting the support of a large community of devoted users. With this D04, you will learn to domesticate its most accessible sides.

Contents

I	Preamble	2
II	Ocaml piscine, general rules	3
III	Today's specific instructions	5
IV	Exercise 00: CheatSheet	6
V	Exercise 01: Moar CheatSheet	8
VI	Exercise 02: Quick search	10
VII	Exercise 03: Diary	11

Chapter I

Preamble

Philosophy

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

Library

- [Your new bible](#)
- [Your new toolbox](#)
- [Your new evening news program](#)
- [Your new TV](#)

Chapter II

Ocaml piscine, general rules

- Every output goes to the standard output, and will be ended by a newline, unless specified otherwise.
- The imposed filenames must be followed to the letter, as well as class names, function names and method names, etc.
- Unless otherwise explicitly stated, the keywords `open`, `for` and `while` are forbidden. Their use will be flagged as cheating, no questions asked.
- Turn-in directories are `ex00/`, `ex01/`, ..., `exn/`.
- You must read the examples thoroughly. They can contain requirements that are not obvious in the exercise's description.
- Since you are allowed to use the OCaml syntaxes you learned about since the beginning of the piscine, you are not allowed to use any additional syntaxes, modules and libraries unless explicitly stated otherwise.
- The exercises must be done in order. The graduation will stop at the first failed exercise. Yes, the old school way.
- Read each exercise FULLY before starting it! Really, do it.
- The compiler to use is `ocamlopt`. When you are required to turn in a function, you must also include anything necessary to compile a full executable. That executable should display some tests that prove that you've done the exercise correctly.
- Remember that the special token `;;` is only used to end an expression in the interpreter. Thus, it must never appear in any file you turn in. Regardless, the interpreter is a powerful ally, learn to use it at its best as soon as possible!
- The subject can be modified up to 4 hours before the final turn-in time.
- In case you're wondering, no coding style is enforced during the OCaml piscine. You can use any style you like, no restrictions. But remember that a code your peer-evaluator can't read is a code he or she can't grade. As usual, big functions are a weak style.
- You will NOT be graded by a program, unless explicitly stated in the subject. Therefore, you are given a certain amount of freedom in how you choose to do the

exercises. However, some piscine day might explicitly cancel this rule, and you will have to respect directions and outputs perfectly.

- Only the requested files must be turned in and thus present on the repository during the peer-evaluation.
- Even if the subject of an exercise is short, it's worth spending some time on it to be absolutely sure you understand what's expected of you, and that you did it in the best possible way.
- By Odin, by Thor! Use your brain!!!

Chapter III

Today's specific instructions

- All the turned-in files will be Ruby On Rails web applications.
- Today's projects' Gemfile must include:


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

When running the "rubycritic" command, the html report page's "smells" must be EMPTY!!!

- Except if specified otherwise in the subject, NO additional Gem will be authorized to complete these exercises.

Chapter IV

Exercise 00: CheatSheet

	Exercise 00
Exercise 00: CheatSheet	
Turn-in directory : <i>ex00/</i>	
Files to turn in : CheatSheet	
Allowed functions : bootstrap-sass, rubycritic	

Satisfied with your first "Hello World" in rails, you feel like you've grown wings and choose to push what you've learned from your new companion a little further.

Not too far, though... As a matter of fact, we're gonna spend the day using rails' most simple mechanisms.

If you spent the majority of your time on Wikipedia yesterday, I suggest you take D03-Ex04 before going any further.

NO DATABASE, NO 'CRUD' for now. We will just focus on views.

Go [here](#) and feast your eyes on this wonderful site that will remind you of the useful commands...

You must replicate this page, including footer. It will just be the page this time. The navigation system will be for some time later. By the way, don't waste time making pixel perfect css. However, your result will have to remotely look like this page.

You will have to observe this few instructions:

- The app will be named "CheatSheet"
- It just contains a controller of your own design, `application_controller` which must be left as is
- The main page's title is "CheatSheet"
- You don't have to replicate the nav-bar (the red thing at the beginning of a page) for the moment
- You will have to design your page a little. We don't demand a professional ready-to-produce design, but your page will have to be pleasant to look at.


Your HTML should be where it belongs. The controller should be in its folder. No CSS will be tolerated in an HTML file! Use Rails like you ought to.

Don't be silly. The CheatSheet is aptly named. It will help you in the arborescence of a Rails application as well as in the useful commands.

You should also use the "inspect element" command of your browser and be smart... Remember yesterday's crawler.

Chapter V

Exercise 01: Moar CheatSheet

	Exercise 01
Exercise 01: Moar CheatSheet	
Turn-in directory : <i>ex01/</i>	
Files to turn in : NewCheatSheet	
Allowed functions : bootstrap-sass , rubycritic	

Ok. Now you know how to make a One-Page with Rails, it's time to talk serious business.

Pick your previous application. You will now implement the nav-bar (the red thing...) and change you CheatSheet navigation a little bit.

Practically, you're gonna have to separate each part of this CheatSheet into independent pages and assign a nav-bar button to the matching page.


What's expected:

- The app will be named "NewCheatSheet"
- It stills includes just one controller of your own design, `application_controller` you will leave as is
- Each page will have a dedicated "title" tag
- You must write the "nav-bar" code just once!
- The first page is the "Convention" page. It will be your root page

- All the following pages must be represented:
 - convention
 - console
 - ruby
 - ruby-concepts
 - ruby-numbers
 - ruby-strings
 - ruby-arrays
 - ruby-hashes
 - rails-folder-structure
 - rails-commands
 - rails-erb
 - editor
 - help
- "Resources" and "About" pages won't have to be replicated nor included in the nav-bar.
- You will have to design your page a little. We don't demand a professional ready-to-produce design, but your page will have to be pleasant to look at.

Chapter VI

Exercise 02: Quick search

	Exercise 02
Exercise 02: Quick search	
Turn-in directory : <i>ex02/</i>	
Files to turn in : CheatSheet	
Allowed functions : bootstrap-sass, rubycritic, jquery-datatables-rails	

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

It will take a **jQuery** plugin and a little content reformatting... Add a thumbnail named "Quick search" that will include a table incorporating the content of ALL the other tables of the CheatSheet.

This table will be **properly formatted**, with a header, the proper number of columns, properly located tags etc.

Thus, you will be able to use the **jquery-datatables-rails** Gem that will boost this table.


To be clear, you will have to:

- Implement a new "Quick search" thumbnail in the nav-bar.
- Let this thumbnail have its personalized title tag, like the rest of the site.
- Let its content be a recap table of ALL the other pages' commands.
- Associate the **jQuery-datatables-rails** plugin to this table, and include it to the project via its Gem.

NB: If an error message pops up or Javascript appears in the `html.erb` file, the exercise will be invalid.

Chapter VII

Exercise 03: Diary

	Exercise 03
Exercise 03: Diary	
Turn-in directory : <i>ex03/</i>	
Files to turn in : CheatSheet	
Allowed functions : bootstrap-sass, rubycritic, jquery-datatables-rails	

The CheatSheet project starts looking good. But as any seasoned developer, you're keeping a technical journal, and it would be really neat to include it in your project.

Create a new thumbnail called "Log Book" that features a form with a "Write" button and a field to include text in the page header. Once the form is filled and submitted, your app will add the text in the form to a "entry_log.txt" file, located at the root of your CheatSheet project's folder.

The text will have to be formatted on the fly adding date and time as follows at the beginning of the line:

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

A loop manages the text display: each entry must be displayed after the form, with its own tag from the oldest to the newest entry.

Each time you click on the form's "Write" button, you must be redirected on the same page, with the new line showing on top.