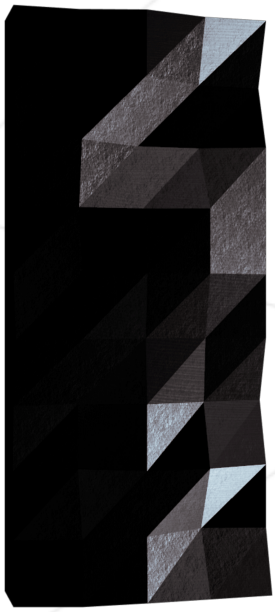


D00 - Pool Ruby On Rails

Web Basics

Stéphane Ballet sballet@student.42.fr
42 Sta ff pedago@staff.42.fr

Summary: This first day will allow you to approach web development bases. On the menu : HTTP, HTML, CSS and integration of existing scripts in JavaScript your pages.

Contents

Chapter I

Preamble

Here is what Wikipedia says the *Balaenoptera musculus:*

The blue whale (*Balaenoptera musculus)* also known as blue whale is a species of cetaceans of the family *Balaenopteridae.* Can exceed 30 meters in length and 170 tons, is the largest animal living in our time and in the present state of our knowledge, the largest that ever lived on Earth.

Long and thin, the body of the blue whale can take various shades of bluish-gray on the back and a little more clear below. There are at least three subspecies dis- tinct: *B. m. musculus* in the North Atlantic and the North Pacific that, *B. m. intermedia* Antarctic Ocean *B. m. brevicauda* discovered in the Indian Ocean and in the south of the Paci fi c Ocean. *B. m. indica* discovered in the Indian Ocean, may be another subspecies. Like other whales, blue whales feed mainly a small crustacean, krill, but also small fish and sometimes squid.

Blue whales were abundant in nearly all oceans until the early twentieth century. For nearly forty years, they were hunted by whalers who brought the species to the brink of extinction before it is protected by the international community in 1966. A 2002 report estimated there were between 5 000 and 12,000 blue whales worldwide, located in at least five groups. More recent studies on the subspecies *B. m. brevicauda* suggest that this may be an underestimate. Before the industrial whaling, the largest population was in the Atlantic, which had approximately 240,000 (in '202 000 and 311

000). The species is considered threatened.

No whales were harmed during the writing of it.

Chapter II

Instructions

- Only this page serve as a reference: Do not fi rm the hallway noise.
- The subject can change up to an hour before rendering.
- If no contrary information is explicitly present, you must assume the following language versions:
  - Ruby 2.3.0
  - for d09 Rails> 5
  - but for all the other days rails 4.2.6
  - HTML5
  - CSS 3
- We prohibit EXPRESSLY using keywords while, for, redo, break, retry and until in Ruby source code that you will make. Any use of these keywords is considered cheating (and / or unfit), giving you the score of -42.
- The exercises are precisely ordered from simple to more complex. In any case, we do not bear attention or take into account a complex exercise if a simpler exercise is not very successful.
- Attention to the rights of your fi les and your directories.
- You must follow the rendering process for all your exercise: only the present work on your GIT repository will be evaluated in defense.
- Your exercises will be evaluated by your pool mates.
- You should leave in your repertoire no other fi le than those explicitly speci fi ed by the forward drills.
- You have a question ? Ask your neighbor to the right. Otherwise, try your left neighbor.
- Your reference manual called Google / man / Internet / ....
- Think discuss on the forum of your pool Intra!
- Read the examples carefully. They might require things that

not otherwise specified in the subject ...

- Please, by Thor and Odin! Re fl échissez dammit!

Chapter III

Exercise 00

	Exercise: 00 00 Exercise: First
shell script	
Render Folder: *ex* 00 / Files to make: myawesomescript.sh	
Permitted functions: curl, grep, cut	
Remarks: n / A	

Yes Twitter has no secrets for you, you are very likely bit.ly :
a shortener handy URLs.

The purpose of this exercise is to write and make a shell script that ffi che the actual address of an address bit.ly **(understand "the address to which the link returns bit.ly ").**

As explicitly written in the cartridge of this exercise, you are not entitled to uili- ser the following three shell **commands to complete this exercise: curl, grep and cut.**
Your best starting point is to read the manual command curl. To do this, type man curl in your device.

An example of the expected behavior of your shell script:

```
$> $ ./Myawesomescript.sh bit.ly/1O72s3U http://42.fr>
```

The above example clearly shows that your script should be executable. interprets the user's / bin / sh.

Meet your script to a folder ex00 to the root of your deposit.

Chapter IV

Exercise 01

	Exercise 01 Exercise 01: Your
CV HTML	
Render Folder: *ex* 01 / files to make: cv.html	
Permitted functions: n / A	
Remarks: n / A	

You must create a CV HTML / CSS and meet the following requirements:

- You must respect the semantics of your HTML tags, and the separation of presentation and content.
- You must produce a file in HTML coherent structure with a minimum imposed content: name, skills and career.
- You need a ffi high to a title with the tag title and a title with the tag h1.
- You must use at least one table with tags table, th, tr and td.
- You must use at least one list with the tag ul and a list with the tag ol. The elements must use the tag li.
- The edges of the tables should be visible (solid). The edges of the tables should be merged (collapse).
- You will need to use a syntactic solution diff erent for each of the previous two rules: first use the tag style in the head your page. For the second, using an attribute style in a tag that is obviously appropriate.
- The lowermost cell to the right of each table must have # 424242 as border color.

No special requirement for the veracity of information. You can make a wacky CV if you feel like it.

Chapter V

Exercise 02

	Exercise: 02
Exercise 02: Sending form of emails	
Render Folder: *ex* 02 / Files to make: form.html	
Permitted functions: n / A	
Remarks: n / A	

Make a form HTML which represents the usual information contact IN ANY. This form is to provide all of the following fields:

- firstname: a text field.
- name: a text field too.
- Age : you must use the specified numeric field as the HTML5.
- Phone: you must use the field as specified in that HTML5.
- E-mail : you must use the specific field email at HTML5.
- Student at 42? : you must use the checkbox field.
- Gender: you must use the radio buttons with values Male, Female and Other.
- A form submit button. The attribute onclick Your button must be: 'DisplayFormContents (this.form).

the tarball d00.tar.gz annexed to this topic contains a subfolder EX02 / which itself contains a file Javascript popup.js written by the son of your boss internship in your company. As it would be unacceptable if you do spend the son of pa- tron for programming incompetant, you can not modify its fi le that should be used as such.

A careful reading and a surface COMPRENSION the provided JavaScript is required to succeed this year.

You must properly integrate this fi le Javascript to your page HTML. If your code HTML is correct, pressing the Form button will bring up an ultramodern popup containing the fields and values of your form. In all other cases, your **code HTML is wrong.**

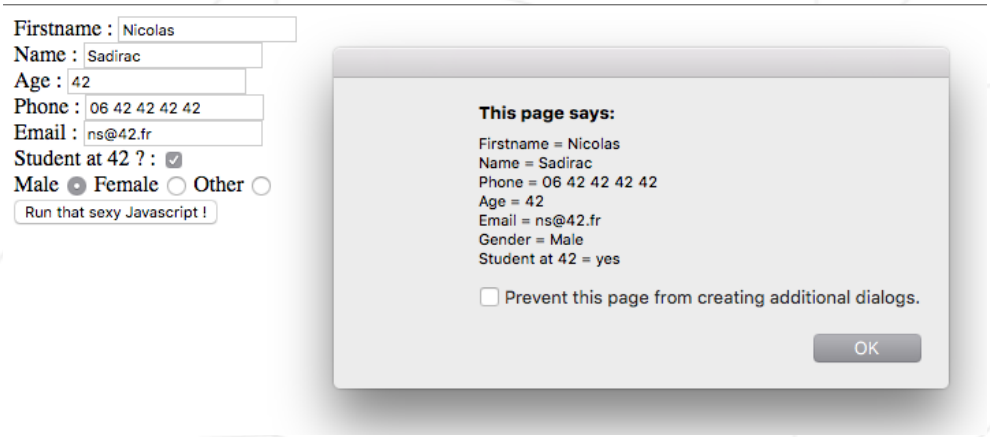


figure V.1 - noncontractual illustration of the expected result.

Chapter VI

Exercise 03

	Exercise: 03
Exercise 03: Reproduction of a web page	
Render Folder: *ex* 03 / Files to make: copy.html	
Permitted functions: n / A	
Remarks: n / A	

A competing company has posted a prettier webpage yours. With an industrial espionage worthy of Hollywood **cinema, your boss provides a screen shot of the page and the fi le css that goes with. These two fi les at your disposition in the Annexes to this in the archive d00.tar.gz and subfolder EX03 /.**

You must reproduce this as faithfully as possible page!

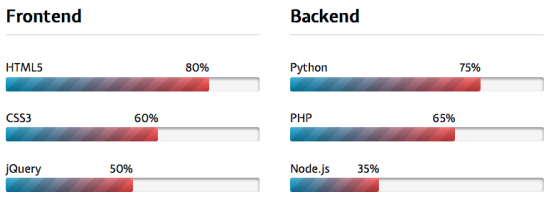


figure VI.1 - The screenshot of the page to duplicate. Scale of noncontractual image.

You will again separate the content and form, respect the semantics of tags used and maintain a logical structure in your document.

You must use the fi le css provided without the modifying. A "fresh" version of this css defense will be used to verify that you have complied with this order.

Chapter VII

Exercise 04

	Exercise: 04
Exercise 04: Integration of JS snippets.	
Render Folder: *ex* 04 / Files to make: snippets.html	
Permitted functions: n / A	
Remarks: n / A	

the tarball d00.tar.gz annexed to this topic contains a subfolder ex04 / which itself contains four fi les: file1.js, file2.js, file3.js and file4.js.

You must create and make a file snippets.html which must import all four scripts so that the pop-up was ffi che **correctly** (so no weird characters).

You can than import scripts in question, you can not modify or add Javascript in your HTML.

Chapter VIII

Exercise 05

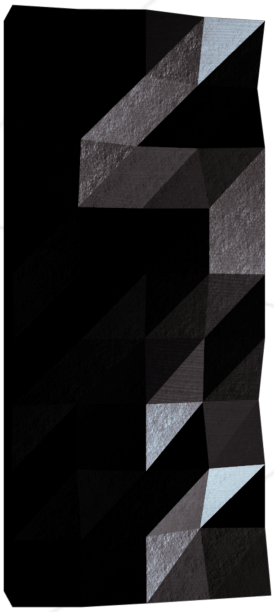
	Exercise 05 Exercise 05:
W3C validation.	
Render Folder: *ex* 05 / Files to make: Your index.html file and the asset	
Authorized Functions: Notes: n	
/ A	

The code is good, the beautiful code is better, and to make good code, it is best to follow a good standard.

The W3C standard is a must in the field, and it is imperative to respect it when you write or you drive HTML.

You will find in the tarball d00.tar.gz annexed to this subfolder
ex05 / which contains the source of a complete webpage. Unfortunately, this page has been written by a developer much worse than you!

Modi fi rm code HTML of the file index. html so that it passes the W3C validation ! This means, therefore, no errors and no *warning*.



D01 - Training Ruby on Rails

syntactic and semantic bases

Stéphane Ballet balletstephane.pro@gmail.com
42 Sta ff pedago@staff.42.fr

Summary: We leave a little dommaine web to focus on the ruby <3, and the syntactic and semantic foundations of this language.

Contents

Chapter I

Preamble

Besides being awesome, Ruby is funny !

- Poignant Guide To Ruby
- TryRuby
- RubyMonk
- Rubyquizz
- RubyWarrior

Chapter II

Instructions

- Only this page serve as a reference: Do not fi rm the hallway noise.
- The subject can change up to an hour before rendering.
- If no contrary information is explicitly present, you must assume the following language versions:
  - Ruby 2.3.0
  - for d09 Rails> 5
  - but for all the other days rails 4.2.6
  - HTML5
  - CSS 3
- We prohibit EXPRESSLY using keywords while, for, redo, break, retry and until in Ruby source code that you will make. Any use of these keywords is considered cheating (and / or unfit), giving you the score of -42.
- The exercises are precisely ordered from simple to more complex. In any case, we do not bear attention or take into account a complex exercise if a simpler exercise is not very successful.
- Attention to the rights of your fi les and your directories.
- You must follow the rendering process for all your exercise: only the present work on your GIT repository will be evaluated in defense.
- Your exercises will be evaluated by your pool mates.
- You should leave in your repertoire no other fi le than those explicitly speci fi ed by the forward drills.
- You have a question ? Ask your neighbor to the right. Otherwise, try your left neighbor.
- Your reference manual called Google / man / Internet / ....
- Think discuss on the forum of your pool Intra!
- Read the examples carefully. They might require things that

not otherwise specified in the subject ...

- Please, by Thor and Odin! Re fl échissez dammit!

chapter III

Specific Rules of the day

- All fi les will be returned with an appropriate shebang and warning fl ag.
- No code in the global scope. Make functions!
- Each fi le made must be completed by a function call.
- No authorized import, except for those explicitly mentioned in the "Permitted functions" of the cartridge each year.

chapter IV

00 Exercise: Class No Class

	Exercise: 00 00 Exercise:
Class No Class	
Render Folder: *ex* 00 / Files to	
make: var.rb	
Permitted functions: n / A	
Remarks: n / A	

Create a named script var.rb in which you must define a function my_var.
In it, declare and initialize variables 4 di ff erent types and print to standard output. You must reproduce exactly the following output:

```
$> Ruby var.rb my
variables:
        Contains a 10's type: Fixnum b Contains: 10 is of type String c
    Contains: ten of type String s Contains: 10.0 is type: Float $>
```

It is understood **not allowed** explicitly write types of your variables in your code prints. Do not forget to call your function at the end of your script as explained in the instructions.

chapter V

01 Exercise: Breakfast

	Exercise 01 Exercise 01:
Breakfast	
Render Folder: *ex* 01 / files to make: croissant.rb	
Permitted functions: n / A	
Remarks: n / A	

For this exercise, you are free to define as many functions as you want and name them as you like.

the tarball d01.tar.gz annexed to this topic contains a subfolder ex01 / wherein is a file numbers.txt containing 1 numbers 100 randomly separated by a comma.

Design a script Ruby appointed croissant.rb whose role is to open the fi le numbers.txt, read the numbers it contains, and the a ffi expensive to standard output, one per line without commas.

Command to extract a tarball: tar -xzf d01.tar.gz

chapter VI

Exercise 02: Hashment well

	Exercise 02 Exercise 02:
Hashment well	
Render Folder: *ex* 02 / Files to	
make: H2o.rb	
Permitted functions: n / A	
Remarks: n / A	

You are free to define as many new features as you want and name them as you like. This set will no longer mentioned, except in cases of explicit contradiction.

Create a named script H2o.rb in which you must copy the table couples data as is in the next one or another of your duties:

```
data = [[ 'Caleb', 24],
                [ 'Calixte', 84], [ 'Tanager',
                65], [Calvin ', 12], [Cameron',
                54], [ 'Camil', 32], [ 'Camille',
                5], [ ' Can '



                                , 52],
                [ 'Caner', 56], [ 'Cantin', 4],
                [Carl '
                                , 1]
                [ 'Carlito', 23], [ 'Carlo', 19], [
              'Carlos', 26], [ 'Carter', 54], [
              'Casey', 2]]
```

Write code that, when it is executed, and declares the hash transforms with the key to the whole and as value the string and ffi song on the display a message as follows:

```
$> Ruby H2o.rb 24:
Caleb 84: Calixte 65: 12
Tanager Calvin 54:
Cameron 32: 5 Camil
Camille 52: Can 56:
Caner 4: Cantin 1: Carl
23: Carlito 19: Carlo 26:
Carlos 54: Carter 2:
Casey $>
```

chapter VII

Exercise 03: Where am I?

	Exercise: Exercise 03 03:
Where am I?	
Render Folder: *ex* 03 / Files to make: Where.rb	
Permitted functions: n / A	
Remarks: n / A	

Using the following hashes (you recopierez them in a function that either I do the most write):

```
states = {
    "Oregon"                => "OR",
    "Alabama"               => "G"
    "New Jersey" => "NJ", "Colorado"
  => "CO"}


capitals_cities = {
    "OR" => "Salem", "G" =>
   "Montgomery", "NJ" => "Trenton",
   "CO" => "Denver"}
```

Write a program that takes as argument a state (ie Oregon) and a ffi che on output standand its capital (eg Salem). If the argument does not work, your script should a ffi expensive: Unknown state. If there is not, or if too many arguments, your script should do and leave nothing.

```
$> Ruby Where.rb Oregon Salem

$> Ruby foo Where.rb Unknown
state $> ruby Where.rb

$> Ruby Where.rb Oregon Alabama
$> Ruby Where.rb Oregon Alabama Ile-de-France $>
```

chapter VIII

Exercise 04: Backward

	Exercise 04 Exercise 04:
Backward	
Render Folder: *ex* 04 / Files to make: erehW.rb	
Permitted functions: n / A	
Remarks: n / A	

You have again the same two hashes than last year. Create a program that takes a capital argument and a ffi che this time Ci the corresponding state. The rest of the behavior of your program should be identical to those of the previous year.

```
$> Ruby erehW.rb Salem Oregon

$> Ruby foo erehW.rb Unknown
capital city $> $ erehW.rb ruby>
```

chapter IX

Exercise 05: Hal

	Exercise 05 Exercise
05: Hal	
Render Folder: *ex* 05 / Files to make: whereto.rb	
Permitted functions: n / A	
Remarks: n / A	

Always re-hashes of EX03, write a similar program to cedents pre- except that:

- The program should take as argument a string containing many words to search you want, separated by commas.
- For each word of that string, the program should detect if that word is a capital city or a state.
- The program should not be case sensitive, or white spaces.
- If there is no parameter or too many parameters, the program has nothing ffi che.
- When there are two commas in a ffi threaded into the string, the program has nothing ffi che.
- The program needs a ffi expensive results separated by a newline. and using the following specific format:

```
$> Ruby whereto.rb "Salem, Alabama, Toto,                    Montgomery "
Salem is the capital of Oregon (AKR: OR) Montgomery is the capital of
Alabama (AKR: AL) is toto Neither has capital city has state nor Montgomery is
the capital of Alabama (AKR: AL) $>
```

chapter X

Exercise 06: Wait a minute

	Exercise 06 Exercise 06:
Wait a minute	
Render Folder: *ex* 06 / Files to make: CoffeeCroissant.rb	
Permitted functions: n / A	
Remarks: n / A	

Using the following table:

```
data = [[ 'Frank', 33], [
'Stacy', 15], [ 'Juan', 24],
[ 'Dom', 32], [ 'Steve',
24], [ 'Jill', 24] ]
```

Write the code that ffi only che names sorted in order of increasing age and al phabétique when ages are identical, line by line.

```
$> Ruby CoffeeCroissant.rb Stacy Jill Dom
Juan Steve Frank $>
```

The hash of the data will be changed for the defense to verify that the work was done properly.

chapter XI

Exercise 07: elm

	Exercise 07 Exercise
07: elm	
Render Folder: *ex* 07 / Files to	
make: elm.rb	
Permitted functions: n / A	
Remarks: n / A	

the tarball d01.tar.gz annexed to this topic contains a subfolder EX07 / wherein is a file periodic_table.txt which describes the periodic table of ele- ments in a convenient format for IT professionals.

Create a program that uses this fi le to write one page HTML representing the periodic table of elements formatted properly.

- Each element must be a 'case' of a table HTML.
- The name of an item must be in a title tag level 4.
- The attributes of an element should be in list form. Must be listed at least the atomic number, symbol, and atomic mass.
- You must comply with minimum layout of Mendeleev table as we can find it on Google, namely that there must be empty boxes where there must have, as well as linebreaks here where it takes. Your program should create the fi le result periodic_table.html. This fi le HTML

must of course be immediately readable by any browser and must be valid W3C.

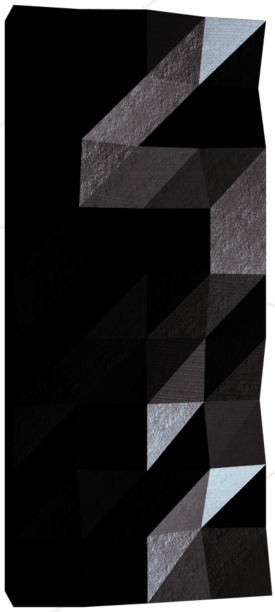
You are free to design your program as you wish. Feel free to split your code into separate and possibly reusable functionality. You can customize the tags with a CSS style "inline" to make your made more attractive (if only that the contour of the table spaces), or even generate a file

periodic_table.css if you prefer.

Here is an excerpt of sample output to give you an idea:

```
[...]
   <Table>
      <Tr>
         <Td style = "border: 1px solid black; padding: 10px">
            <H4> Hydrogen </ h4>
                  <Ul> <li> No 42 </ li>

                        <Li> H </ li> <li> 1.00794 </ li>
                  <li> one electron </ li> <ul> </ td> [...]
```



D02 - Ruby on Rails Training

Classes and Inheritance Exceptions

Stéphane Ballet balletstephane.pro@gmail.com
42 Sta ff pedago@staff.42.fr

Abstract: In this day D02, you will learn to make up classes and get to the heart of OOP, if your exercises seem too wordy is that your code is not

DRY enough!

Contents

Chapter I

Preamble

Besides being awesome, Ruby is beautiful !

This guide (written by Bozhidar Batsov) has emerged as a convention of programmed internal Ruby tion to his business. After a while, it seemed to him that this work could be interesting for community members ruby in general and the world did not really need an internal programming convention more. But the world could certainly bene fi t of a set of practices, idioms and style advice for the Ruby programming.

From the start of this guide, he received a lot of feedback from members of the ex ceptional Ruby community worldwide. Thank you for all the suggestions and support! Together we can create a beneficial resource to all develop- Ruby fears here today.

- The style is what separates the good from the excellent.
- The very widely used Rubocop

Chapter II

Instructions

- Only this page serve as a reference: Do not fi rm the hallway noise.
- The subject can change up to an hour before rendering.
- If no contrary information is explicitly present, you must assume the following language versions:
  - Ruby 2.3.0
  - for d09 Rails> 5
  - but for all the other days rails 4.2.6
  - HTML5
  - CSS 3
- We prohibit EXPRESSLY using keywords while, for, redo, break, retry and until in Ruby source code that you will make. Any use of these keywords is considered cheating (and / or unfit), giving you the score of -42.
- The exercises are precisely ordered from simple to more complex. In any case, we do not bear attention or take into account a complex exercise if a simpler exercise is not very successful.
- Attention to the rights of your fi les and your directories.
- You must follow the rendering process for all your exercise: only the present work on your GIT repository will be evaluated in defense.
- Your exercises will be evaluated by your pool mates.
- You should leave in your repertoire no other fi le than those explicitly speci fi ed by the forward drills.
- You have a question ? Ask your neighbor to the right. Otherwise, try your left neighbor.
- Your reference manual called Google / man / Internet / ....
- Think discuss on the forum of your pool Intra!
- Read the examples carefully. They might require things that

not otherwise specified in the subject ...

- Please, by Thor and Odin! Re fl échissez dammit!

chapter III

Specific Rules of the day

- All fi les will be returned with an appropriate shebang and warning fl ag.
- No code in the global scope. Make functions or classes!
- Each fi le must have made a series of tests to prove its good functioning:

```
if $ PROGRAM_NAME == __FILE__
  ##   your code here end
```

- Each fi le is to be tested in an interactive console (irb or pry choice) like this:

```
require_relative "name \ _of \ _fichier.rb"
>
```

- No authorized import, except for those explicitly mentioned in the "Permitted functions" of the cartridge each year.

chapter IV

00 Exercise: HTML

	Exercise: 00 00
Exercise: HTML	
Render Folder: *ex* 00 / Files to make: ex00.rb	
Permitted functions: n / A	
Remarks: n / A	

Take a class html able to create and populate a file HTML. To do this, you need to Implement:

- A constructor that takes parameters as a fi le (without extension):
  - which a method called head
  - that assigns the title fi le the instance variable @ page_name
- A attr_reader for @page_name
- A method Head which must aposer head of a fi le header html followed by a valid start tag body.
- a method "dump" which takes a string parameter and starts following the "body" tag our string surrounded by "p" tags.
- a method "fi nish" that ends the fi le a body closing tag

All inserts must be in the form of lines.

In a console **ruby,** we must get this:

```
>   require_relative "ex00.rb" => true

>   Html.new a = ( "test")
=> # <Html: @ 0x00000001d71580 page_name = "test">
>   10.times {| x | a.dump ( "titi_number # {x}")}
=> 12
>   a.finish
=> 7
```

And in our shell:

```
cat -e test.html <!
DOCTYPE html> $ <html>
$ <head> $

<Title> Test </ title> $ </ head> $
<body> $

    <P> titi_number0 </ p> $ <p>
   titi_number1 </ p> $ <p>
   titi_number2 </ p> $ <p>
   titi_number3 </ p> $ <p>
   titi_number4 </ p> $ <p>
   titi_number5 < / p> $ <p>
   titi_number6 </ p> $ <p>
   titi_number7 </ p> $ <p>
   titi_number8 </ p> $ <p>
   titi_number9 </ p> $ </ body> $ $>
```

Use as in the example a loop to fulfill its fi le **MUST** function.

chapter V

01 Exercise: Raise HTML

	Exercise 01 Exercise 01:
HTML Raise	
Render Folder: *ex* 01 / files to make: ex01.rb	
Permitted functions: n / A	
Remarks: n / A	

For this exercise, you will reuse the code ex00 and incorporate error handling, throw exceptions when the behavior requires, thus avoiding invalid HTML pages pro- duction and / or wacky. All the following exceptions must **be reproduced accurately, replacing < filename> the fi le name which creates the error:**

- Create a file that already exists (same name) must throw an exception " A file named <filename> already exist ".
- **Write text with dump in a file having no tag body or- vrante should throw an exception " There is no body tag** in <filename> ".
- **Write text with dump after a closing body tag must throw the exception:**
  "Body HAS already been closed in <filename>".
- **Close the body with finish while the closing tag body is already present must throw the exception: "< filename>** has already been closed. "

```
>   require_relative "ex01.rb" => true

>   Html.new a = ( "test")
=> # <Html: @ 0x0000000332b9c0 page_name = 'test'>
>   Html.new a = ( "test")
RuntimeError: test.html already exist! /ex01.rb:15:in from "head"

>   a.dump ( "Lorem_ipsum")
=> Nil
>   a.finish
=> Nil
>   a.finish
RuntimeError: test.html HAS already been closed from / ex01.rb: 39: in "finish"
```

chapter VI

Exercise 02: Rescue HTML

	Exercise 02 Exercise 02:
Rescue HTML	
Render Folder: *ex* 02 / Files to make: ex02.rb	
Permitted functions: n / A	
Remarks: n / A	

Now that you have problem behaviors in their hands, we will pro fi t.

To save process execution correcting shots, with a ffi chage speci fi c errors, and solutions to ensure FUNCTIONING.

Create two classes: Dup_ the fi and Body_closed who will inherit StandardError.
They will both contain implementations of " show_state "," correct " and
"Explain".

- **show_state a ffi che the state before correction**
- **correct corrects the error**
- **explain a ffi che the state after correction**

When an error creating fi le because already now the fi Dup_ the exception is raised. She must :

- a ffi expensive list fi les similares (with full pwd)
- create a new fi le by adding '.new' after the extension, several times if necessary: test.html, test.new.html, test.new.new.html etc etc ... Example:

```
A file named <filename> was already there: /home/desktop/folder_2/<filename>.html .new Appended in order to create requested file:
/home/desktop/folder_2/<filename>.new.html
```

When you try to write AFTER the closing tag of body, the Body_closed exception is thrown and must:

- a ffi expensive line and its number in the fi le
- remove the so-called tag, insert the text and put a closing tag after example:

```
In <filename> body was closed:
   >   In: 25 </ body>: text has-been inserted and tag Moved at end of it
```

chapter VII

Exercise 03: Elem

	Exercise 03 Exercise
03: Elem	
Render Folder: *ex* 03 / Files to	
make: ex03.r	
Permitted functions: n / A	
Remarks: n / A	

Now is the time to change method. Your first engine test
HTML is conclusive and prometeur but it is now question of pushing the paradigm of the subject a little further.

You will now create a class to represent your HTML so that a method to_s at his instance a ffi che code HTML generated.

So, with his method add_content, the class Elem will be able to have content for another instance of Elem.

This architecture permettera use these:

```
html = Elem.new (.....) head =
Elem.new (.....) body = Elem.new (....)

title = Elem.new (Text.new ( "bla bla")) head.add_content (title)

html.add_content ([head, title, body]) puts html
```

To ensure proper implementation, we provide a file in the folder test EX02 / in the tarball d02.tar.gz present with this.

It recapitulates:

- A class Elem with as a construction parameter, a tag type, an array of content, a type of tag (orphaned or not), and a Hash permetant information to implement the in-tag (src, style, data ...).

- A class text that is built with a simple String parameter.
- A **overload** Method to_s.
- Running the test script must pass **IN-TE-GRA-LEMENT.**

chapter VIII

Exercise 04: Dejavu

	Exercise 04 Exercise
04: Dejavu	
Render Folder: *ex* 04 / Files to make: ex04.rb	
Permitted functions: n / A	
Remarks: n / A	

Congratulations! You are now able to generate any element that HTML
and content. However, it is a bit heavy to generate each item by stating where each attribute to each instantiation. This is an opportunity to use inheritance to other small simplest classes of use.

Perform the following classes in the drifting of Elem:

- Html, Head, Body
- title
- meta
- img
- Table
- Th, Tr, Td
- Ul, ol, Li
- H1, H2
- P
- div
- span
- Hr
- Br

Your code should execute these commands without errors:

```
Html.new puts ([Head.new ([Title.new ( "Hello ground!")])
        Body.new ([H1.new ( "Oh no, not again!")
        Img.new ([], { 'src': 'http: //i.imgur.com/pfp3T.jpg'})])])
```

And ffi expensive:

```
<HTML>
<Head>
<Title> Hello ground! </ Title> </ Head> <Body>


<H1> Oh no, not again! </ H1>
<Img src = 'http: //i.imgur.com/pfp3T.jpg' /> </ body> </ html>
```

If you feel that exercise is daunting is that there may be another solution;)

chapter IX

Exercise 05: Validation

	Exercise 05 Exercise 05:
Validation	
Render Folder: *ex* 05 / Files to make: whereto.rb	
Permitted functions: n / A	
Remarks: n / A	

Despite real progress in your work, you would like that everything is a little cleaner, a little framed, you are like that: you like the constraints and challenges. So why not impose a standard structure for your documents HTML? Commencez by copying the classes of the two previous years in the record of this exercise.

Create a class Page which the manufacturer must take as a parameter an instance of a class inheriting Elem. Your class Page must implement a method isValid ()
which must return true if all the following rules repectées and false if not :

- If during the course of the tree, a node is not type html, head, body, title, meta, img, table, th, tr, td, ul, ol, li, h1, h2, p, div, span, hr br or
  text, the shaft is invalid.
- html must contain exactly Head, then a Body.
- Head should only contain a single title and only this Title.
- Body and div should contain elements of the following types: H1, H2, Div, table, ul, ol, Span, or Text.
- Title, H1, H2, Li, Th, Td must not contain a single text and only this
  Text.
- P should contain only Text.
- span should contain only text or some P.
- IU and ol must contain at least one Li and only Li.

- **Tr must contain at least one Th or td and only Th or some Td. The Th and the td must be mutually** exclusive.
- **Table should contain only Tr and only Tr.**
- **img: must contain a field src and its value is a Text.**

Demonstrate how your class Page by tests of your choice by many su ffi cient to cover all the features. For example, the execution of:

```
if $ PROGRAM_NAME == __FILE__
    foo = Html.new ([Head.new ([Title.new (Text.new ( "Hello ground!"))]) Body.new ([H1.new (Text.new ( "Oh no, not
    again! ")), Img.new ([], { 'src': Text.new ( 'http://i.imgur.com/pfp3T.jpg'})})])]) = Page.new test (foo) test.is_valid?



    tata Html.new = ([Head.new ([Title.new (Text.new ( "Hello ground!"))])
                Body.new ([H1.new (Text.new ( "Oh no, not again!")), Img.new ([], { 'src': Text.new ( 'http://i.imgur.com
      /pfp3T.jpg '})})])]) = test2 Page.new (tata) test2.is_valid? end
```

MUST a ffi expensive:

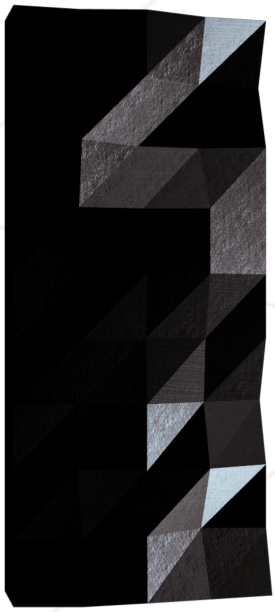
```
Currently Evaluating has Html:
-   root element of the type "html"
-   Html -> Must contains a Head DNA has after-Body Head it is OK

Evaluating a multiple node Currently Evaluating a
Text:
- Text -> Must contains a single text string is OK happy
Evaluating node has multiple Currently Evaluating a Text:


- Text -> Must contains a single string Text happy is OK

Currently Evaluating has Img: Img happy is OK

                            FILE IS OK
```



D03 - Ruby on Rails Training

Gems

Stéphane Ballet balletstephane.pro@gmail.com
42 Sta ff pedago@staff.42.fr

Summary: There is always a Gem for.

Contents

Chapter I

Preamble

The RS7000 is a major groove generation workstation!
It's sort of like Akai's MPC-series (aim for bumps), sampling-combining and sequencing, with an added purpose internal synth engine. The RS7000 is PARTICULARLY suited for dance, techno, hip hop, R & B, and ambient genres.

The sampler section Consists of a 4MB (expandable to 64MB) sampler (5 kHz to 44.1kHz or 32kHz to 48kHz via digital option board). You can use it to sample external sounds, re-sample the RS7000's sounds itself, or load samples from a variety of common formats ! Auto-beat slicing lets you easily sample any loops or sounds and sync them to your sequence tempo ! All the professional sampling and editing features you'd expect are here, and more !

The tone generator offers 62-voice polyphonic AWM2 synthesis, with over 1,000 synth sounds and 63 drum kit sounds (all via ROM). Here you'll find the resonant filters (6 types), advanced LFO modulation, BPM-synchronized LFO waveforms, and more ! Edits made to the internal sounds, as well as to any samples are all stored within your sequence patterns.

The Sequencer is the real meat of the RS7000, where you make music out of the sounds it's got and that you've put into it ! It offers pattern-based recording with 16 tracks each, and a 200,000 note-per-song capacity. Linear sequencer sequencing, like you would do using a software sequencer like Cubase, is also supported by the RS7000. Pattern-based sequences can be converted to the linear format as well. Realtime, grid and step recor- ding methods are also available. Linking patterns into songs can be done in real time and meticulously tweaked.

Total MIDI control, real-time hands on control, 18 assignable knobs and two pads, a Master effect section (with a multi-band compressor, slicer, isolater, other DJ-style master effects), and more make the RS7000 the most professional quality groove/loop/dance machine out there !

Chapitre II

Consignes

- Only this page serve as a reference: Do not fi rm the hallway noise.
- The subject can change up to an hour before rendering.
- If no contrary information is explicitly present, you must assume the following language versions:
  - Ruby 2.3.0
  - for d09 Rails> 5
  - but for all the other days rails 4.2.6
  - HTML5
  - CSS 3
- We prohibit EXPRESSLY using keywords while, for, redo, break, retry and until in Ruby source code that you will make. Any use of these keywords is considered cheating (and / or unfit), giving you the score of -42.
- The exercises are precisely ordered from simple to more complex. In any case, we do not bear attention or take into account a complex exercise if a simpler exercise is not very successful.
- Attention to the rights of your fi les and your directories.
- You must follow the rendering process for all your exercise: only the present work on your GIT repository will be evaluated in defense.
- Your exercises will be evaluated by your pool mates.
- You should leave in your repertoire no other fi le than those explicitly speci fi ed by the forward drills.
- You have a question ? Ask your neighbor to the right. Otherwise, try your left neighbor.
- Your reference manual called Google / man / Internet / ....
- Think discuss on the forum of your pool Intra!
- Read the examples carefully. They might require things that

not otherwise specified in the subject ...

- Please, by Thor and Odin! Re fl échissez dammit!

chapter III

Règles spécifiques de la journée

- Tout les fichiers rendus seront dotés d'un shebang approprié ET du flag de warning.
- Aucun code dans le scope global. Faites des fonctions ou des classes !
- Chaque rendu doit etre une Gem (sauf pour l'exercice 03 !)
- Chaque Gem doit utiliser minitest, la license MIT et ne doivent proposer aucun code de conduite.

- Ces Gems sont à vocation pédagogiques : ne vous occupez pas des repos d'upload !
  Commentez les lignes dans le .gemspec.
- Chaque Gem doit inclure les tests demandés dans l'exercice et ceux-ci doivent être executés par un bundle exec rake dans le dossier racine de la gem
- Aucun import autorisé, à l'exception de ceux explicitement mentionnés dans la section "Fonctions Autorisées" du cartouche de chaque exercice.

Chapitre IV

Exercice 00 : J’aime

	Exercice : 00 Exercice
00 : J’aime	
Dossier de rendu : *ex* 00/ Fichiers à rendre :	
deepthought	
Fonctions Autorisées : colorize	
Remarques : n/a	

Créez une gem, votre première avec nous .

Vous allez ainsi mettre au monde une portion de code reutilisable par la planète entière Procurant de maniere

portable le code suivant :

```
require 'colorize' class

Deepthought
        def initialize end

        def respond(question)
                if question == "The Ultimate Question of Life, the Universe and Everything"
                        puts "42".green return "42"

                elseputs "Mmmm i'm bored".red

                        return "Mmmm i'm bored" end end end
```

Excitant , non ? !

Votre Gem doit répondre aux specificités suivantes :

- Nom de Gem : deepthought
- Des tests ? oui bien sur ! : utilisez minitest
- Un code de conduite ? Non
- Licence : MIT
- Version : '0.0.1'
- La commande grep -Hrn 'TODO' –color=always, exécutée à la racine de votre Gem, ne doit RIEN retourner.
- Vous devez écrire le test qui vérifie que Deepthought.new retourne bien l'objet attendu
- Vous devez écrire le test qui vérifie la valeur de retour des deux cas de la methode 'respond'

Chapitre V

Exercice 01 : ft_wikipedia

	Exercice : 01 Exercice 01 :
ft_wikipedia	
Dossier de rendu : *ex* 01/ Fichiers à rendre : ft_wikipedia	
Fonctions Autorisées : nnokogiri/open-uri	
Remarques : n/a	

Un fait interressant de Wikipedia, est la "route vers la philosophie" Si on clique sur le premier lien en minuscule de la première description de n'importe quelle page, dans 94% des cas, on atteint la page philosophie.

De mon experience, cela n'a jamais depassé 35 liens...

Pour en être sur, vous devez creer une gem "ft_wikipedia" qui s' utilise et s' affiche comme ceci :

```
Ft_wikipedia.search("Kiss")
First search @ :https://en.wikipedia.org/wiki/Kiss https://en.wikipedia.org/wiki/Love
https://en.wikipedia.org/wiki/Affection https://en.wikipedia.org/wiki/Disposition
https://en.wikipedia.org/wiki/Habit_(psychology) https://en.wikipedia.org/wiki/Behavior



https://en.wikipedia.org/wiki/American_and_British_English_spelling_differences https://en.wikipedia.org/wiki/English_orthography
https://en.wikipedia.org/wiki/Orthography https://en.wikipedia.org/wiki/Convention_(norm)
https://en.wikipedia.org/wiki/Norm_(philosophy) https://en.wikipedia.org/wiki/Sentence_(linguistics) https://en.wikipedia.org/wiki/Word
https://en.wikipedia.org/wiki/Linguistics https://en.wikipedia.org/wiki/Science https://en.wikipedia.org/wiki/Knowledge
https://en.wikipedia.org/wiki/Awareness https://en.wikipedia.org/wiki/Conscious https://en.wikipedia.org/wiki/Quality_(philosophy)
https://en.wikipedia.org/wiki/Philosophy => 19
```

Le programme liste toutes les urls visitées, et retourne le nombre de liens qu'il a du traverser avant la page "https ://en.wikipedia.org/wiki/Philosophy".

Sometimes some research as "matter" going in circles! ! you **must** handle this case with an exceptional lifting "StandardError" and a ffi expensive as follows:

```
Ft_wikipedia.search ( "matter")
First search @ https: //en.wikipedia.org/wiki/matter https://en.wikipedia.org/wiki/Atom
https://en.wikipedia.org/wiki/Matter https: //en.wikipedia .org / wiki / Atom


Loop detected there is no way to philosophy here => nil
```

Sometimes some research as "E ff ects_of_blue_lights_technology" are dead ends! ! you **must** manage it with exceptional lifting "StandardError" and AF expensive fi as follows:

```
Ft_wikipedia.search ( "Effects_of_blue_lights_technology")
First search @ https: page atteint //en.wikipedia.org/wiki/Effects_of_blue_lights_technology Dead end => nil
```

You **must** also write all the tests that prove the good operation of your program with the following search "directory", "problem", "einstein", "kiss", "mat- ter".

chapter VI

Exercise 02: TDD

	Exercise 02 Exercise
02: TDD	
Render Folder: *ex* 02 / Files to make: Taillste	
Permitted functions: n / A	
Remarks: n / A	

TDD, or Test Driven Development, is a practice highly prized venue beyond the world of web development

We provide you a Gem empty for you implementer. Tests were already written there. I will not dwell on the features of this Gem, since the purpose of the exercise is that you must guess.

To validate the exercise must ABSOLUTELY than :

- La commande "gem build" à la racine du dossier de votre Gem s'exécute sans erreur (outre les warnings d'aide, de manque de description et de homepage manquante)
- la derniere ligne de l' affichage de la commande "rake" soit :

```
"16 runs, 16 assertions, 0 failures, 0 errors, 0 skips"
```

Chapitre VII

Exercice 03 : Rails

	Exercice : 02
Exercice 03 : Rails	
Dossier de rendu : *ex* 02/ Fichiers à rendre	
: HelloWorld	
Fonctions Autorisées : n/a	
Remarques : n/a	

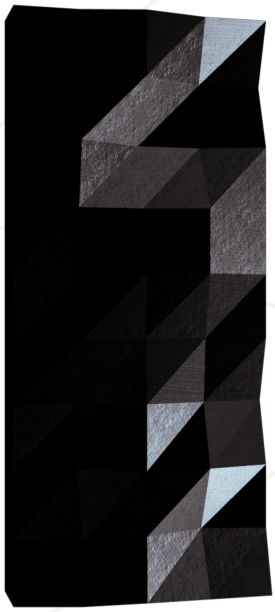
Aaaah ! Hello World ! le célèbre. Vous voici donc maintenant au pied du mur ... Vous devez installer rails sur vos machines, faire une page contenant un titre "Hello World !", et lorsque vous lancez le serveur inclus dans rails, cette page doit être la pre- miere affichée.

Google est remplis de bons conseils, cependant j'en ai un autre pour vous : documen- tez vous avant de lancer des installations ....

Il est primordial que votre installation se déroule correctement. Etant donné que vous aurez a le refaire plusieurs fois, autant prendre un bon départ ! !

Pour valider l'exercice, il faudra :

- Installer la Gem rails, dépendances necessaires incluses
- Faire en sorte que le serveur inclus dans rails démarre (la commande est : "rails serveur")
- Faire en sorte que la première page de votre site (accessible via "http ://localhost :3000/") soit une page HTML affichant "Hello World !" dans une balise de titre "<h1>"



D04 - Ruby on Rails Training

rails 101

Stéphane Ballet balletstephane.pro@gmail.com
42 Sta ff pedago@staff.42.fr

Summary: Rails a powerful open source framework aliant high productivity and quality code, while bene fi ting a large passionate community and very responsive.
With the D04, you go to domesticate the most affordable parts.

Contents

Chapter I

Preamble

The philosophy

- DRY: Do not Repeat Yourself
- Convention over configuration
- REST: Representational State Transfer
- CRUD: Create, Read, Update, Delete The

biblioteque

- Your new Bible
- Your new box has tool
- Your new JT
- Your new TV

Chapter II

Instructions

- Only this page serve as a reference: Do not fi rm the hallway noise.
- The subject can change up to an hour before rendering.
- If no contrary information is explicitly present, you must assume the following language versions:
  - Ruby 2.3.0
  - for d09 Rails> 5
  - but for all the other days rails 4.2.6
  - HTML5
  - CSS 3
- We prohibit EXPRESSLY using keywords while, for, redo, break, retry and until in Ruby source code that you will make. Any use of these keywords is considered cheating (and / or unfit), giving you the score of -42.
- The exercises are precisely ordered from simple to more complex. In any case, we do not bear attention or take into account a complex exercise if a simpler exercise is not very successful.
- Attention to the rights of your fi les and your directories.
- You must follow the rendering process for all your exercise: only the present work on your GIT repository will be evaluated in defense.
- Your exercises will be evaluated by your pool mates.
- You should leave in your repertoire no other fi le than those explicitly speci fi ed by the forward drills.
- You have a question ? Ask your neighbor to the right. Otherwise, try your left neighbor.
- Your reference manual called Google / man / Internet / ....
- Think discuss on the forum of your pool Intra!
- Read the examples carefully. They might require things that

not otherwise specified in the subject ...

- Please, by Thor and Odin! Re fl échissez dammit!

chapter III

Specific Rules of the day

- All fi les are made of Ruby On Rails web applications.
- Any project of the day must include in the fi Gem:

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

  During the execution of the command "rubycritic" the "smells" section of the html page report MUST BE EMPTY! !

- Unless stated subject, NO extra Gem is allowed to ac- complir these exercises.

chapter IV

00 Exercise: CheatSheet

	Exercise: 00 Exercise 00:
CheatSheet	
Render Folder: *ex* 00 / Files to make: CheatSeet	
Permitted functions: bootstrap-sass, rubycritic	
Remarks: n / A	

Not content with having made heir your first "Hello World" in rails, you feel Posser wings and decide to go further in the use of your new companion.

Not too far though ... in e ff ect, we'll all day today, exploi- ter rails in its simplest mechanisms.

NO DATABASE OR FOR 'CRUD' for the moment. We will concentrate on the views.

Appointment right here and contemplate what mreveilleux site useful commands reminders .... You must reproduce this site, except for the tabs 'Resources' and 'About' and the footer. Do not waste time doing css pixel, however, some layout and a certain resemblance is appreciable.

Instead you move the page, all tabs buttons should you rediri- ger to a page with the appropriate chapter

These few points are to be respected:

-

- The app will be called "CheatSheet"
- It contains only one controller of your creation, application_controller is to leave as is
- Each page has its tag "title" DEDICATED
- You have to write the code for the "nav bar" once!
- The first page is the "Convention"
- All the following pages should be represented:
  - editor
  - convention
  - ruby
  - console
  - numbers
  - thongs
  - arrays
  - hashes
  - rails_app
  - rails_commands
  - embedded_ruby
- Some layout is required. We do not ask for a design worthy of an app to put into production, but a nice simple layout to watch.

Your HTML should be where it's supposed to be, the controller must be on file. No CSS should be present in a file HTML! Use your rails as it should.

Serve you also "inspect element" and be evil, remember the crawl yesterday.

chapter V

01 Exercise: Quick search

	Exercise 01 Exercise 01:
Quick search	
Render Folder: *ex* 01 / files to make: CheatSheet	
Permitted functions: bootstrap-sass, rubycritic, jquery-rails datatables	
Remarks: n / A	

Rather practice this Cheat Sheet, but it is not yet at the top. You will improve with a plugin jQuery and reformatting the clutter ... Add a tab named "Quick Search" will contain a table that incorporates ALL content of other tables CheatSheet.

This table will be properly trained, will include a header, the many columns, well placed tags etc etc.

Then you can use the gem jquery-datatables rails that will energize you table.

Clearly you must:

- Implementing a new "Quick Search" tab.
- Ensure that tab is the title tag personalized, like the rest of the site.
- To ensure that its contents will be an array of ALL of récaputilation orders from other pages.
- Associating the plugin jQuery-datatables rails in this table, and include it in the project via its Gem.

NB: if an error popup was ffi che or Javascript is present in the fi le html.erb, the exercise will be counted wrong.

chapter VI

Exercise 02: Diary

	Exercise 02 Exercise
02: Diary	
Render Folder: *ex* 02 / Files to make: CheatSheet	
Permitted functions: bootstrap-sass, rubycritic, jquery-rails datatables	
Remarks: n / A	

The project began CheatSheet have some aplomb. But like any good developer, you hold a technical journal, and incorporate into your project would be an important asset.

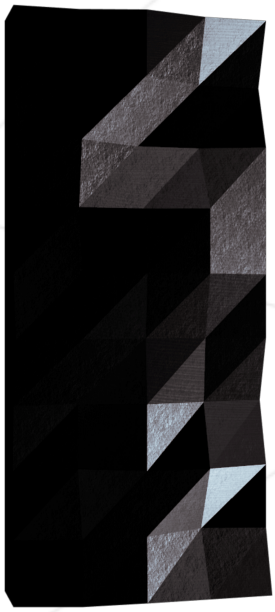
Creez a new tab named "Log Book" which figure in page header, an entry form with a button 'Write'. This button, once the form completed and submitted with the text, add the text of the form to a file "entry_log.txt" on the root of your project file CheatSheet

The text will be formatted on the fly by adding at the beginning of the line, the date and time like this:

```
07.15.2016 11:42:00 p.m.: All the crew is asleep Jim. 07.15.2016 11:41:00 p.m.: All the crew
Jim yawns.
```

The text has ffi drying is done with a loop: Each entry must be a ffi chée following form, with its own tag, and the most recent entry at the entrance most ancienne.door

Every time you click on the "Write" button on the form, you must make sure to be redirected to the same page and the new line is a ffi chée top.



D06 - Training Ruby on Rails

Authentication and permissions

Sta ff 42 bocal@staff.42.fr
Stéphane Ballet balletstephane.pro@gmail.com

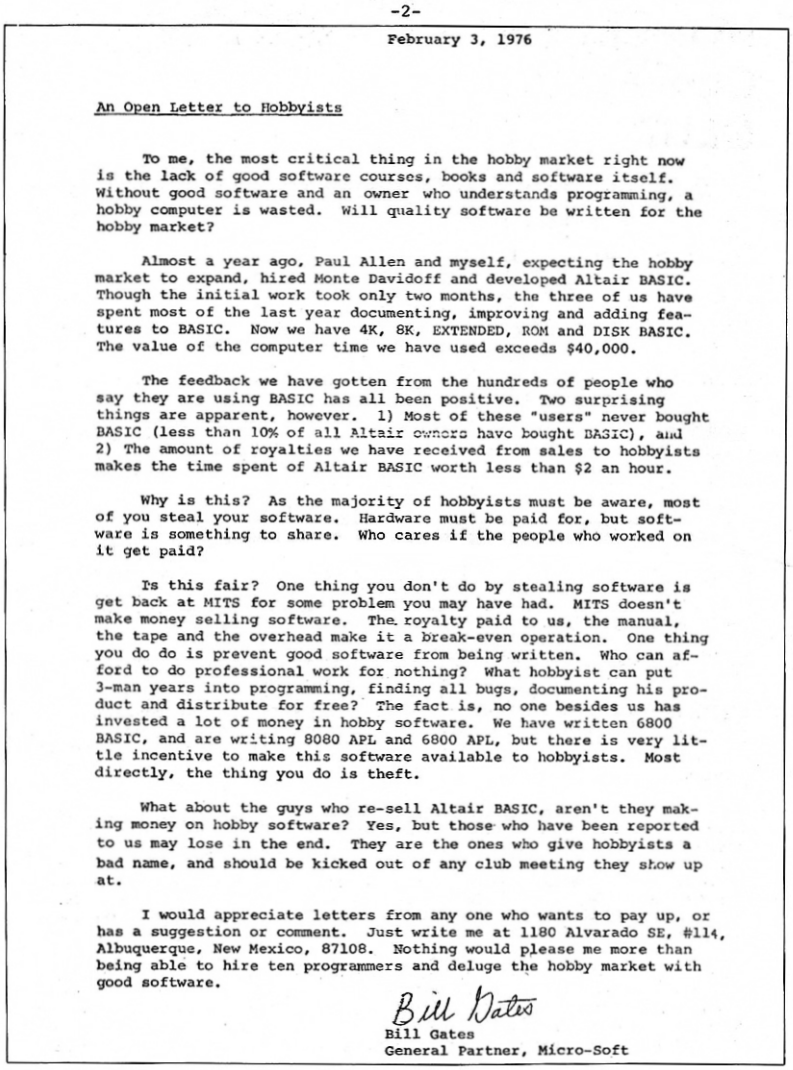
Summary: You will discover the wonderful world of rights, through cookies, fl ags, permissions and privileges.
Incidentally, Rails has a clear error handling, use. At this stage of your learning the 'error-driven development is a good option!

Contents

Chapter I

Preamble

And speaking of permissions and privileges, a small letter Mr Gates dated but that is pretty funny in view of history.



-2-

February 3, 1976

An Open Letter to Hobbyists

To me, the most critical thing in the hobby market right now is the lack of good software courses, books and software itself. Without good software and an owner who understands programming, a hobby computer is wasted. Will quality software be written for the hobby market?

Almost a year ago, Paul Allen and myself, expecting the hobby market to expand, hired Monte Davidoff and developed Altair BASIC. Though the initial work took only two months, the three of us have spent most of the last year documenting, improving and adding features to BASIC. Now we have 4K, 8K, EXTENDED, ROM and DISK BASIC. The value of the computer time we have used exceeds $40,000.

The feedback we have gotten from the hundreds of people who say they are using BASIC has all been positive. Two surprising things are apparent, however. 1) Most of these "users" never bought BASIC (less than 10% of all Altair owners have bought BASIC), and 2) The amount of royalties we have received from sales to hobbyists makes the time spent of Altair BASIC worth less than $2 an hour.

Why is this? As the majority of hobbyists must be aware, most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid?

Is this fair? One thing you don't do by stealing software is get back at MITS for some problem you may have had. MITS doesn't make money selling software. The royalty paid to us, the manual, the tape and the overhead make it a break-even operation. One thing you do do is prevent good software from being written. Who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free? The fact is, no one besides us has invested a lot of money in hobby software. We have written 6800 BASIC, and are writing 8080 APL and 6800 APL, but there is very little incentive to make this software available to hobbyists. Most directly, the thing you do is theft.

What about the guys who re-sell Altair BASIC, aren't they making money on hobby software? Yes, but those who have been reported to us may lose in the end. They are the ones who give hobbyists a bad name, and should be kicked out of any club meeting they show up at.

I would appreciate letters from any one who wants to pay up, or has a suggestion or comment. Just write me at 1180 Alvarado SE, #114, Albuquerque, New Mexico, 87108. Nothing would please me more than being able to hire ten programmers and deluge the hobby market with good software.

Bill Gates

Bill Gates
General Partner, Micro-Soft

Chapter II

Instructions

- Only this page serve as a reference: Do not fi rm the hallway noise.
- The subject can change up to an hour before rendering.
- If no contrary information is explicitly present, you must assume the following language versions:
  - Ruby 2.3.0
  - for d09 Rails> 5
  - but for all the other days rails 4.2.6
  - HTML5
  - CSS 3
- We prohibit EXPRESSLY using keywords while, for, redo, break, retry and until in Ruby source code that you will make. Any use of these keywords is considered cheating (and / or unfit), giving you the score of -42.
- The exercises are precisely ordered from simple to more complex. In any case, we do not bear attention or take into account a complex exercise if a simpler exercise is not very successful.
- Attention to the rights of your fi les and your directories.
- You must follow the rendering process for all your exercise: only the present work on your GIT repository will be evaluated in defense.
- Your exercises will be evaluated by your pool mates.
- You should leave in your repertoire no other fi le than those explicitly speci fi ed by the forward drills.
- You have a question ? Ask your neighbor to the right. Otherwise, try your left neighbor.
- Your reference manual called Google / man / Internet / ....
- Think discuss on the forum of your pool Intra!
- Read the examples carefully. They might require things that

not otherwise specified in the subject ...

- Please, by Thor and Odin! Re fl échissez dammit!

chapter III

Specific Rules of the day

- All the work of the day will be improved versions of the same application rails.
- Any addition to the Gem fi provided in the resources is prohibited.
- Any global variable is prohibited.
- You must provide a seed, you're going to Implement the fi exercises of the day. For correction, no exercise will only be valid if imple- mented fonctionalitées are depicted within the data from the db.
- you must imperatively handle errors. No error page rails will be tolerated for correction.
- You must have a score awarded by rubycritic higher or equal to 90
- You must be NO 'warning' reported by rails_best_practice
- You must ALLOW intact fi le "rails_best_practices.yml" and place it in the root of "app / con fig /" (it contains con fi g softened to d06)

Tips: If you want your corrections are simple, fast and friendly, write tests! It simplifies life for everyone and it is a crucial habit, nay, MANDATORY make for your future.

chapter IV

Exercise 00: It's me

	Exercise: 00 Exercise
00: It's me	
Render Folder: *ex* 00 / Files to make: LifeProTips	
Authorized Functions: Notes: n	
/ A	

You must create an app brand new. She will complexification fi ed all day, so pity (for you), code properly and use git. If the authentication has been encoded with your seat, it will annoy you all day ...

Where was I ... Oh yes! The authentication. You must create a decent, with encryption, redirect and everything goes well. Serve you has_secure_password.
Creez for now a 'root page' and make a vacuum line user page header (you are not designers, I know, but a layout -peu- please '):

"Welcome Stéphane! Log_out"

If visteur is unknown, his name will consist of a random animal name followed by "_visiteur":

"! Welcome pony_visitor sign_in log_in" The animal name at

random is like this:

```
url = "https://www.randomlists.com/random-animals" doc = HTML :: Nokogiri (open
(url)) doc.css name = ( 'li'). first.css ( 'span'). text
```

So you need to do that:

- An unregistered user is acceuilir with "Welcome <animal_name> _visitor" followed by a registration link and login.
- The anonymous user name stored in a cookie valid for one minute (easier for corrections). Thus, after a minute, his name will be changed to the next loading.
- After registering, the new user is automatically logged.
- The users must have a name and an email UNIQUE.
- The registration form will not let the apparent passwords.
- Passwords do not appear bright in the log server either.
- The registration form asks:
  - A name
  - An email
  - A password (min 8 char)
  - A password_con fi rmation
- The form has errors ffi che with a message.
- You must create a seed with some users.
- You should have tests.
- The utility 'rails_best_practice' must not return any warning:

```
%> Rails_best_practices
Source Code: | ============================================== ================ | Please go to
http://rails-bestpractices.com to see more Rails Best Practices. No warning found. Cool!
```

chapter V

Exercise 01: Add me in

	Exercise 01 Exercise 01:
Add me in	
Render Folder: *ex* 01 / files to make: LifeProTips	
Authorized Functions: Notes: n	
/ A	

Ok great! We have a DB and users of authentication. Sum- mayor is, but it's a start.

You will add a section admin in order to administer users. For safety it will be based on a design pattern that uses namespacing.

In short: the URL for the actions of requirent admins rights com- menceront by "/ admin / ...". This allows you to Implement a controller For users without conflict with the former.

The new controller will have prototype like this:

```
:: class Admin UsersController <ApplicationController

    # ================
    #   code here
    # ================ end
```

Thus, the URL http: // localhost: 3000 / admin / users is available and leads to the created page.

I hope for you that you used good agreements and that you therefore current_user is the user logged or if the cookie.

Directors must be diff érenciés by admin true field or false (default false) in the users table.

Only they can access the list of all users and edit values (name and email), and remove them.

Each user can, however, change his name or email.

When a user is admin its connection, the message is followed by a link that leads to more page admin / users (the index of our new controller):

"Welcome Stéphane! Administrate_users Log_out" In addition to 'callback', you will need to

properly con fi gure your roads .

chapter VI

Exercise 02: Need an account

	Exercise 02 Exercise 02:
Need an account	
Render Folder: *ex* 02 / Files to make: LifeProTips	
Authorized Functions: Notes: n	
/ A	

The basic back-o ffi ce is almost fi ned, we'll add some meaning to our app. In e ff ect, the LifeProTips application is intended to receive and share "life pro tips" but it remains somewhat advantageous tricks we will restrict access to members, but not too ...

You must create using a sca ff old Post an object that contains:

- a 'user_id' (required)
- a 'title' (single and longueure min 3 char)
- a 'content' (which must contain a lot of text)

Obviously, the author does not put its id itself to the establishment is in control- them that the thing is done automatically with current_user

The index must list the titles and authors by name and sorted all time-high (youngest first).

Only the index is visible to visitors, the title should be a link to the action 'show post' and if a visitor clicks on it, it must be redirected to the root page 'with instructions like: "You need an account pour voir cette "

The new root page is index posts

A user must be able to edit all posts (for now).

Repeat the same operation as in FY01: an admin page for posts or administrator has all the rights.

The line in the header is (if one is admin):

"Welcome Stéphane! Administarte_users Administrate_posts Log_out" The link "Administrate_users" leads to a page listing all created posts. This one allows access to deletion and modification.

chapter VII

Exercise 03: Peer edit

	Exercise 03 Exercise 03:
Peer edit	
Render Folder: *ex* 03 / Files to make: LifeProTips	
Authorized Functions: Notes: n	
/ A	

Although badly reseigné one who believes to know everything.

The Life Pro Tips are editable, you have a ffi expensive on the show page for each post by whom and when the post was modified last time:

Original author: bob2
Title: How to tell if the water is too hot?
Content: Dip your baby in it and check its temperature with the inside of your wrist Edited by: Stéphane

Date of Modification: 2016-07-19 1:55:51 p.m. UTC

You need to Implement the functionality that is represented by the last two lines of the example

You are free to choose your solution. There are many but the good are rare ..

Warning no global variables or cookie-based solutions

chapter VIII

Exercise 04: UvDv

	Exercise 04 Exercise
04: UvDv	
Render Folder: *ex* 04 / Files to make: LifeProTips	
Authorized Functions: Notes: n	
/ A	

Add on the page "post show", the possibility of 'Upvote' and 'downvote' and an a ffi chage of the total votes (possibly negative).

Vote on the same page redirects

Nb: no global variable or another of this style .The votes have to be an object that inherits from ActiveRecord.

Again seen must have an admin interface as for exerecie 01 (with the same design).

This interface will have a view that lists the votes grouped by posts indicating the title of the post in the header and the names of associated voting.

The administrator can remove votes.

chapter IX

Exercise 05: Can you?

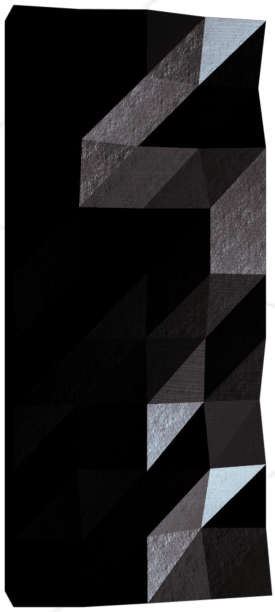
	Exercise 05 Exercise 05:
Can you?	
Render Folder: *ex* 05 / Files to make: LifeProTips	
Authorized Functions: Notes: n	
/ A	

You need for this last exercise of the day, set up privileges based on the total votes.

- 0 to 3: no speci fi c law
- 3 to 6: right to upvote
- 6 to 9: right to downvote
- above 9 right to edit posts

Now a user can edit his posts, unless it is above 9 upvote.

The detail of the user page (user show) a ffi che what right has the user in question, and these rights also fi gurent on page user administration.



D07 - Ruby on Rails Training

Getting advanced situation and practices

Sta ff 42 bocal@staff.42.fr
Stéphane Ballet balletstephane.pro@gmail.com

Summary: A day on applied practices of web dev. You are put in a position via 'stories' that will lead to the creation of a basic e-commerce site.

Contents

Chapter I

Preamble

Write a user story (scenario, story)

Une histoire utilisateur est une suite d'actions effectuées par un utilisateur de notre application suivies d'une ou plusieurs assertions (test) validant que notre histoire s'est bien déroulée.

Pour expliquer en langage courant, une histoire ressemble à un problème de mathé- matiques dans lequel on a :

- des hypothèses
- un élément perturbateur
- quelque chose à démontrer

Exemple de problème mathématique (version collège) :

- Soit 2 personnes (Pierre et Jean)
- Pierre possède 3 bananes
- Jean possède 2 fois plus de bananes que Pierre quand Jean mange une banane
- alors combien de bananes possède Jean ?

Transformons cet exemple de manière mathématique (version prépa, merci M. Bool) :

- Soit 2 personnes (Pierre et Jean)
- Pierre possède 3 bananes
- Jean possède 2 fois plus de bananes que Pierre quand Jean mange une banane
- alors prouvons que Jean possède 5 bananes.

Si maintenant nous voulons écrire une histoire utilisateur validée par notre système, nous écrirons :

- Soit 2 personnes (Pierre et Jean)
- Pierre possède 3 bananes
- Jean possède 2 fois plus de bananes que Pierre quand Jean mange une banane
- **alors Jean devrait posséder 5 bananes. C'est ce qu'on appelle un test : Jean devrait posséder 5 bananes.**

Si nous écrivons des tests, c'est parce que notre système doit se comporter comme tel. Cela nous permet de valider que notre application réagit bien TOUJOURS de la même manière, même après de nombreux mois / années (donc après de nombreuses modifica- tions).

Docs, sources et references :

- Rspec, cucumber book
- Le wiki du github de cucumber
- Le site de cucumber
- Motivational

Chapitre II

Consignes

- Seule cette page servira de référence : ne vous fiez pas aux bruits de couloir.
- Le sujet peut changer jusqu'à une heure avant le rendu.
- Si aucune information contraire n'est explicitement présente, vous devez assumer les versions de langages suivantes :
  - Ruby 2.3.0
  - pour d09 Rails > 5
  - mais pour tout les autres jours Rails 4.2.6
  - HTML5
  - CSS 3
- Nous vous interdisons FORMELLEMENT d'utiliser les mots clés while, for, redo, break, retry et until dans les codes sources Ruby que vous rendrez. Tout utilisation de ces mots clés est considérée comme triche (et/ou impropre), vous donnant la note de -42.
- Les exercices sont très précisément ordonnés du plus simple au plus complexe. En aucun cas, nous ne porterons attention ni ne prendrons en compte un exercice complexe si un exercice plus simple n'est pas parfaitement réussi.
- Attention aux droits de vos fichiers et de vos répertoires.
- Vous devez suivre la procédure de rendu pour tous vos exercices : seul le travail présent sur votre dépot GIT sera évalué en soutenance.
- Vos exercices seront évalués par vos camarades de piscine.
- Vous ne devez laisser dans votre répertoire aucun autre fichier que ceux explicite- ment specifiés par les énoncés des exercices.
- Vous avez une question ? Demandez à votre voisin de droite. Sinon, essayez avec votre voisin de gauche.
- Votre manuel de référence s'appelle Google / man / Internet / ....
- Pensez à discuter sur le forum Piscine de votre Intra !
- Lisez attentivement les exemples. Ils pourraient bien requérir des choses qui ne

sont pas autrement précisées dans le sujet...

- Par pitié, par Thor et par Odin ! Réfléchissez nom d'une pipe !

Chapitre III

Règles spécifiques de la journée

- All the work of the day will be improved versions of the same application rails.

- Any addition to the supplied Gem fi is prohibited.
- Any global variable is prohibited.
- You must make a seed in accordance with the presented functionalities. When your defense your concealer should be able to view your work.
- rubycritic award you must score at least 89/100
- you must imperatively manage any errors rails error page will be tolerated for correction.

Tips: If you want your corrections are simple, fast and friendly, write tests! It simplifies life for everyone and it is a crucial habit, nay, MANDATORY make for your future.

chapter IV

00 Exercise: MySQL is a bad habit

	Exercise: 00
00 Exercise: MySQL is a bad habit	
Render Folder: *ex* 00 / Files to	
make: acme	
Permitted functions: functions_authorized	
Remarques : n/a	

Commençons par un peu d’admin sys, installez postgresql sur la machine et créez un template et un user. De ce fait vous pourrez creer une application rails nomée ’acme’ et utilisant le gemfile que vous trouverez dans la tarball d07.tar.gz

La commande "rake db :create" doit passer sans erreurs.

Story:

- L’application s’appele "acme". Je veux pouvoir mettre en ligne l’application sur Heroku

Chapitre V

Exercice 01 : You Sir ?

	Exercice : 01 Exercice
01 : You Sir ?	
Dossier de rendu : *ex* 01/ Fichiers	
à rendre : acme	
Fonctions Autorisées : functions_authorized	
Remarques : n/a	

Vous avez une db, dans une app toute fraiche. Hier vous avez crée une authentification à la main, mais dans la vraie vie on ne fait pas ca.
En effet, meme pour un bloginet (c'est un petit blog), un serveur est present, et doit
etre protegé.

Pour ce faire, on utilise la très bonne librairie devise, outre le petit dèj (à ce stade, le café est déjà trop loin), elle gère les authentifications.

Si vous inspectez le gemfile, vous verrez qu'elle est deja présente. Il vous suffit main- tenant de l'installer et de la faire gérer un model "User" de sorte que votre seed puisse executer les commandes suivantes :

```
User.create!(bio: FFaker::HipsterIpsum.paragraph, name: 'admin',

email:'admin@gmail.com',
password:'password',
password_confirmation: 'password' )
```

Stories:

- Un utilisateur peux créer un compte avec un mot de passe (nécessaire), un nom (nécessaire), un email(nécessaire) et une biographie (optionelle).
- Il peut re-editer tout ces champs apres creation du compte via l' application (confi- gurez vos "strong parameters")

Chapitre VI

Exercice 02 : Give me something to sell

	Exercice : 02
Exercice 02 : Give me something to sell	
Dossier de rendu : *ex* 02/ Fichiers	
à rendre : acme	
Fonctions Autorisées : functions_authorized	
Remarques : n/a	

Créez des produits à vendre et des marques, et comme tout produit outre une des- cription tres avantageuse et vantant des mérites que la physique réfute, il faut une image, et c'est là notre problème.

En effet, pour permettre correctement l'upload de fichier d'image, on va utiliser la gem carrierwave (the classier solution), rien de trop compliqué jusque là. Le hic c'est qu'un hé- bérgement gratuit supporte trèèès mal le stock de données qui peuvent être volumineuses.

C'est à cet effet que figure dans le Gemfile, une gem appelée coudinary, vous devez creer un compte gratuit chez cloudinary, et vous octroyer par la un espace de stockage distant spécialisé dans les images et autres medias.

A ce stade votre seed peut executer :

```
50.times do |tm|
    mk = Brand.create!(name: FFaker::Product.brand,
                       avatar: open(FFaker::Avatar.image))

    50.times do |tw|
        Product.create!(name: FFaker::Product.product,
                        pict: open(FFaker::Avatar.image), description:
                        FFaker::HipsterIpsum.paragraph, brand_id: mk.id, price:price.sample)

    end
end
```

Stories:

- En se connectant sur le site de l’application, on voit un catalogue listant des produits.
- Chaque produit est contitué d’un nom, d’une image, d’une description d’une marque et d’un prix.
- Une marque a un nom et une image.
- Peut importe la taille de l’image uploadée la page des produit doit afficher les 2500 images,dans une version thumbnail afin de charger rapidement.
- On peut créer et/ou editer touts les champs en ligne des marques et des produits.
- A terme des roles seront attribués, déterminant qui peux editer quoi.

Chapitre VII

Exercice 03 : Panier

	Exercice : 03 Exercice
03 : Panier	
Dossier de rendu : *ex* 03/ Fichiers	
à rendre : acme	
Fonctions Autorisées : functions_authorized	
Remarques : n/a	

Nous avons besoin de faire un panier, un Cart qui va se voir associer des copies des produits sous forme de CartItem, copiés en OrderItem et rassemblés dans un Order lors de la validation du panier.

Ces objets particuliers ne nécéssitant pas de CRUD à proprement parler, seul un mo- del leur est utile. Cependant, des fonctionalités devront etre placés dans un "Concern" afin de pouvoir inclure des méthodes relatives au panier dans d'autres controleurs comme celui des "Products". Fabriquez vous une méthode current_cart qui se base sur la ses- sion_id.

Remember to destroy objects and unnecessary records, for example, when the cancellation of a basket, the 'CartItems' associates have to be destroyed.

Stories:

- In the catalog page, insert a 'basket' is present.
- items can be added by clicking the "Add to cart" present on each item.
- The user can start an order, fill your basket and close the browser. Upon returning to the site's shopping cart is reloaded.
- The user can increase and decrease the number of each item in the basket.
- The basket of lines ffi Chent one type of item quantity and a button

button less, and the price on the formula: quantity * price.

- The user can cancel a basket and make void with a button.
- Un bouton "Checkout" affiche un recapitulatif de la commande avec le prix total.

Chapitre VIII

Exercice 04 : One panel to rule them all

	Exercice : 04
Exercice 04 : One panel to rule them all	
Dossier de rendu : *ex* 04/ Fichiers	
à rendre : acme	
Fonctions Autorisées : functions_authorized	
Remarques : n/a	

Vous devez maintenant créer un paneau d'administration digne de ce nom. Dans le Gemfile, une gem rails_admin est présente, allez voir la doc et initialisez la.

On peut y accéder (pour l'instant) des qu'on possede un compte. Creez sur la page catalogue un link qui mène au dashboard fraichement disponible.

Stories:

- Un utilisateur enregistré peux se connecter et aller sur le panneau d'administration du site.

- De là, on peut éditer et visualiser l'ensemble des données du site.

Chapitre IX

Exercice 05 : One account to rule them all

	Exercice : 05
Exercice 05 : One account to rule them all	
Dossier de rendu : *ex* 05/ Fichiers	
à rendre : acme	
Fonctions Autorisées : functions_authorized	
Remarques : n/a	

Il est, vous me l'accorderez, plutot inutile de disposer d'un panneau d'administration, si tout le monde a la possiblité d'y faire sa loi, pour peu qu'il ait pris la peine de s'y inscrire.

Pour cette occasion, j'ai inclus la gem cancancan qui facilite la gestion des droits, ainsin que la gem rolify qui elle crée un modele de groupe et y attribue les id des users

La "rolification" doit etre présente aussi dans la seed.

Ces deux outils se combinent plutot bien, mais qu' en est il de l'association avec rails admin ?

Stories:

- Un administrateur crée en meme temps que la db peut attribuer des roles.
- Deux rôles sont disponibles : "admin" et "mod".
- Un administrateur peux TOUT editer.
- Un moderateur peux lui SEULEMENT editer les marques et les produits : creation, modification et suppression
- Un utilisateur simple peux s'identifier mais n'aura aucun de ces privilèges,a moins qu'un admin ne lui attribue un role.

- Aucun "url re-writing" ne permet a quiquonque d’outrepasser les permissions dues à son role

Chapitre X

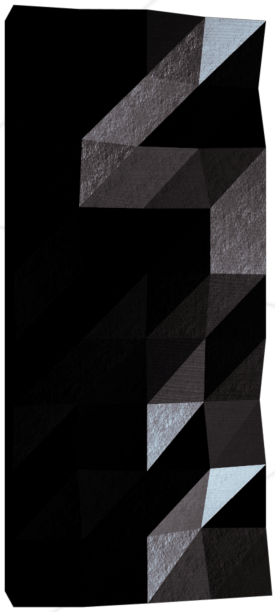
Exercice 06 : Show me what you got

	Exercice : 06
Exercice 06 : Show me what you got	
Dossier de rendu : *ex* 06/ Fichiers	
à rendre : acme	
Fonctions Autorisées : functions_authorized	
Remarques : n/a	

Creez un compte sur Heroku <3 (oups, ca m'a echappé). Créez un compte disais-je et mettez en ligne votre application.

Stories:

- Notre communauté de beta testeurs sont prets l'application est disponnible sur le web.
- Elle est disponible @ : "https ://votrelogin-acme.herokuapp.com"
- Un administrateur peux TOUT editer.
- An application stand script can fill the application with 2,500 products, 50 brands, 20 users with an "admin" and 5 "models
- All fonctioanlitées put in evidence through the stories of the days are Functional ern online, upload pictures, authentication, roles basket etc etc ....



D08 - Ruby on Rails Training

servers

Sta ff 42 bocal@staff.42.fr
Stéphane Ballet balletstephane.pro@gmail.com

Summary: A few notions of system administration and shell-scripting will be necessary to the aparition your application on the web. You will learn how to con fi gure nginx, cougar, and capistrano.// a rails application and a server con fi gurent proudly diff year depending on the environment .You will learn all about

-Setting prod- famous.

Contents

Chapter I

Preamble

Why Are So Many Websites Hosted On Linux?

tl; dr: Because of Windows windowsy's things, and Linux's unixy things

1. Stability

Linux systems are Well Known for Their Ability to run for years without failure; In fact, Many Linux users-have never seen a crash. That's great for users of every kind, PARTICULARLY goal it's valuable for small and medium-sized businesses, for qui downtime can-have Disastrous consequences.

Aussi Linux handles a wide number of processes running at once much better than Windows does-that's something, in fact, That tend to degrade Windows' stability who- CKLY.

Then there's the need for rebooting. Whereas in Windows configuration exchange ty- Pically require a reboot-causing inevitable downtime-there's no need to restart Generally Linux. Almost all Linux configuration exchange can be done while the system is running and without a ff ecting unrelated services.

Similarly, whereas Windows servers must be Often defragmented frequently, that's all aim eliminated on Linux. Let your concurrents endure the plentiful downtime inevitable Tably That goes hand-in-hand with Windows; trusty Linux will keep you up and running and serving your customers around the clock.

2. Security

Linux est innately more secure than Windows is, whether on the server, the desk- top or in an embedded environment. That's due to the fact That Largely Linux, qui est person based on Unix, Was designed from the start to be a multiuser operating system. Only the administrator, or root user, administrative privileges HAS, and Fewer users and applica- tions-have permission to access the kernel gold Each Other. That keeps everything modular and protected.

Of course, Linux gets attacked This aussi less frequently by viruses and malware, and vul- nerabilities tend be found and fi xed by more Quickly icts legions of developers and users. Even the six-year-old kernel bug That Was recently fi xed, for instance-an extremely unusual

instance in the Linux world-had never been exploited.

Internally, Meanwhile, users of a Windows system can hide Sometimes the order from the system administrator. On Linux, HOWEVER, the sys admin always: has a clear view of the fi the system and is always in control.

3. Hardware

Whereas Windows Typically requires frequent hardware upgrades to Accommodate icts ever-Increasing resource demands, Linux is slim, t rim, flexible and scalable, and it is just about Performs admirably Any computer, Regardless of processor machines gold ar- chitecture.

Linux can be aussi Easily recon fi gured to include only the services needed for your business's Purposes, THUS further Top Reducing memory requirements, Improving performance and keeping things Even simpler.

4. TCO

There's no beating Linux's total cost of ownership, since the software is free Generally. Even an enterprise version of you purchased with corporate backing Will Be Cheaper overall than Windows or other proprietary software, qui Generally Involve user-based licensing and a host of expensive add-ons, Especially For security.

Same goes for MOST of the tools and applications might be used That was Linux server. The overall TCO simply can not be beat.

5. Freedom

With Linux, there is no commercial vendor try trying to lock you into some products or protocols. Instead, you're free to mix and match and choose what works best for your business.

In short, with all the advantages Many Provides Linux in the server realm, it's no wonder gouvernements, organisms and major companies around the world, Including Amazon and Google Rely on the open source operating system In Their Own Production systems.

If you're looking for a Linux distribution to run your business it's servers, you'd do well to Consider CentOS (or RHEL, the paid version from Red Hat CentOS That is based on), Slackware, Debian and Gentoo.

Chapter II

Instructions

- Only this page serve as a reference: Do not fi rm the hallway noise.
- The subject can change up to an hour before rendering.
- If no contrary information is explicitly present, you must assume the following language versions:
  - Ruby 2.3.0
  - for d09 Rails> 5
  - but for all the other days rails 4.2.6
  - HTML5
  - CSS 3
- We prohibit EXPRESSLY using keywords while, for, redo, break, retry and until in Ruby source code that you will make. Any use of these keywords is considered cheating (and / or unfit), giving you the score of -42.
- The exercises are precisely ordered from simple to more complex. In any case, we do not bear attention or take into account a complex exercise if a simpler exercise is not very successful.
- Attention to the rights of your fi les and your directories.
- You must follow the rendering process for all your exercise: only the present work on your GIT repository will be evaluated in defense.
- Your exercises will be evaluated by your pool mates.
- You should leave in your repertoire no other fi le than those explicitly speci fi ed by the forward drills.
- You have a question ? Ask your neighbor to the right. Otherwise, try your left neighbor.
- Your reference manual called Google / man / Internet / ....
- Think discuss on the forum of your pool Intra!
- Read the examples carefully. They might require things that

not otherwise specified in the subject ...

- Please, by Thor and Odin! Re fl échissez dammit!

chapter III

Specific Rules of the day

You do not have the right to install the "gem" nginx ".

No script will be uploaded and / or fetché in rendering script, only the app "foubarre" is uploaded to bitbucket and is synchronized during the deployment by
capistrano in ex01

Chapter IV

Exercise 00

	Exercise: 00 Exercise 00:
Shell_ fl avored	
Render Folder: *ex* 00 / Files to make: create_server.sh	
Authorized Functions: Notes: n	
/ A	

In this day early on soft and servers into production must go to the ex00, a script executed in a fresh whole vm'll install and commission a server (puma) in applying " foubarre ".

To do this it is imperative that it operates:

- the git installation
- installing curl
- installation of vim or emacs
- installing rvm
- install ruby
- rail installation
- installing all the necessary dependencies
- the creation of the control application "foubarre"
- compiling assets
- The configuration of the 'SECRET_KEY_BASE'
- and the server start-up in production environments The script make á must integrate the following code to the creation of the moin te- application.

```
$> Mkdir / home / vagrant / Site $> cd / home / vagrant / Site
$> New foubarre rails -d postgresql $> cd foubarre


$> G scaffold rails component great_data
$> Echo "Component.create (great_data 'foo_bar_name')" >> db / seeds.rb $> bundle install

$> Sed -i -e "s / username: foubarre / username: vagrant / g" config / database.yml $> = RAILS_ENV output rake db: create $> =
RAILS_ENV output rake db: migrate $> = RAILS_ENV output rake db : seed


$> Sed -i "2iroot to 'components # index" config / routes.rb $> echo "<h1> <% =% Rails.env> </ h1>"> app / views /
components / index.html .erb
```

First, please verify that you have virtualbox and of vagrant. Then create a folder named "ex00". Start at the root command:

```
hashicorp vagrant init / jessie64
```

Replace y fi le "Vagrant end the" original by:

```
#   - * -    fashion: ruby - * -
#   vi: set ft = ruby:
VAGRANTFILE_API_VERSION = "2"

Vagrant.configure (VAGRANTFILE_API_VERSION) do | config |
    #   Use Ubuntu 14.04 Trusty Tahr 64-bit operating system as our config.vm.box = "debian / jessie64"


    #   Configurate the virtual machine to use 2GB of RAM config.vm.provider: virtualbox
    do | vb |
        vb.customize [ "modifyvm" id, "--memory", "1024"] end


    ##   Un-comment this line in order to copy your script into the VM
    ##   with the "vagrant provision" command:
    # config.vm.provision "file" source "create_server.sh" destination "~ / create_server.sh"


    #   Forward the Rails server default harbor to the host config.vm.network: forwarded_port, guest:
    3000, host: 3031 config.vm.network: forwarded_port, guest: 80, host: end 8090
```

This will give you a vm representing your server. Do not update the distribution!

You can now mount the vm using the command:

```
vagrant up
```

And log in using:

```
vagrant ssh
```

You must assign a password for the root user vagrant, it is strongly advised to use 'vagrant'.

You must change the line of the file "/ etc / hosts"

```
127.0.0.1 localhost
```

in :

```
0.0.0.1 localhost
```

You should make the script that will install the necessary elements and con fi gure so that an application named "foubarre" located at the root of "/ home / vagrant / site" (in the guest system) or accessible via your browser (in the host system) has the address "http: //0.0.0.0: 3031 /"

Uncomment the "con fi g.vm.provision" fi the "..." in the fi Vagrant to copy your script in the vm with the command:

```
vagrant provision
```

If all is well, you should see your application served by puma á "http: //0.0.0.0: 3031 /" in your browser.

Chapter V

Exercise 01

	Activity: 01 year 01: Ruby_ fl
avored	
Render Folder: *ex* 01 / files to make: create_server_2.sh, create_app.sh	
Authorized Functions: Notes: n	
/ A	

In this exercise we will use Capistrano, an automation library deployment on remote server.

This means that since our environment development in local you sign in ssh on your server to synchronize your data from an SVN of your choice, keeping 5 and previous versions of symlinks fi le configuration com- Muns the sake of disk usage optimization.

So deploy on a machine remote (here a vm) an application in an environment of production, and if everything works, with a single command.

You must also use nginx in "Reverse Proxy" which will allow us to have a ready base á receive configurations of "load balancing", in bene fi ting from increased security

In this exercise, you must return two scripts.

The FIRST "create_app.sh":

This script serves á 'recreate the application and so insŕer the configuration of Nginx when correcting. This script should begin as follows:

```
$> Cat create_app.sh
new rails foubarre2 -d postgresql cd foubarre

rails g scaffold component great_data
echo "Component.create (great_data 'foo_bar_name')" >> db / seeds.rb sed -i "2iroot to 'components # index"
config / routes.rb echo "<h1> <% =% Rails.env > </ h1> "> app / views / components / index.html.erb echo"
group: development do ">> Gemfile echo" gem 'capistrano'

                                                    require: false "Gemfile >>
echo "gem 'Capistrano-rvm'                          require: false "Gemfile >>
echo "gem 'rails capistrano'                        require: false "Gemfile >>
echo "gem 'Capistrano-bundler' require: false" >> Gemfile echo "gem 'capistrano3-puma'
                                                    require: false "Gemfile >>
echo "end" >> Gemfile bundle install
install cap

echo "_votre_configuration_ICI_"> config / deploy.rb
#   Your nginx configuration file MUST be in your
#   config file in your app touch config / nginx.conf

echo "_votre_configuration_ICI _"> config / nginx.conf [...]
```

the creation script must also initialize git á the root of your project if you do not have an account bitbucket create one now. Made a new private repo on the web platform and link it to your local project:

```
    git init git add.

    git commit -m "first commit"
    git remote add origin git@bitbucket.org: user_name / repo_name -u git push origin master
```

Now your folder is synchronized with the online repository. For sake of accuracy and fairness, you have to destroy the depot and start every correction.

The SECOND "create_server_2.sh":

Create a new vm to the application root:

```
hashicorp vagrant init / jessie64
```

Replace the fi le "Vagrant end the" original by:

```
#   - * -   fashion: ruby - * -
#   vi: set ft = ruby:
VAGRANTFILE_API_VERSION = "2"

Vagrant.configure (VAGRANTFILE_API_VERSION) do | config |
   #   Use Ubuntu 14.04 Trusty Tahr 64-bit operating system as our config.vm.box = "debian / jessie64"


   #   Configurate the virtual machine to use 2GB of RAM config.vm.provider: virtualbox
   do | vb |
      vb.customize [ "modifyvm" id, "--memory", "1024"] end


config.vm.provision: shell path: "create_server_2.sh"

   #   Forward the Rails server default harbor to the host config.vm.network: forwarded_port,
   guest: 80, host: 8090
   config.vm.network: forwarded_port, guest: 22, host: 2222 id: "ssh" disabled: true config.vm.network: forwarded_port, guest: 22, host: 64673,
  auto_correct: true end
```

Thus, with the command " up vagrant " your vm will provisionéee with the script "create_server_2.sh" pretty simple right? .

You must adapt this script by using the "create_server.sh" so that it operates:

- creating the user 'deploy' with password 'deploy_password'
- the git installation
- installing curl
- installation of vim or emacs
- installing rvm
- install ruby
- rail installation
- install nginx
- installing all the necessary dependencies
- create a script called "post_deploy_symlink.sh"
- the crushing of the file "/ etc / hosts" by:

```
127.0.0.1                   foubarre.com
127.0.1.1                   jessie.raw                  jessie

:: 1             localhost localhost ip6-ip6-loopback ff02 :: 1 ip6-allnodes ff02
:: 2-ip6 allrouters
```

CHOOSE how many user needs to install anything. Knowing that the user deploy should not be included in the sudoers and Capistrano must log in and make its operations as deploy, which must have its added ssh key has bitbucket

The fi expensive hosts must contain a jump address foubarre.com IP on the vm that is loopback. This will declare in the configuration nginx:

```
server_name foubarre.com;
```

Nginx needs as "/ etc / nginx / sites-available / foobarre" is a symbolic link to "/ home / deploy / apps / foobarre / **current / con fi g / nginx.conf". This is the fi le that will contain your configuration for nginx works by** reverse proxy **Puma and serve. It is to this e ff ect the script post_deploy_symlink.sh must be created and executed after the first** deployment.

If all goes well you can make your first deployment:

```
bundle exec producing cap deploy: initial
```

For the exercise to be valid on the host system:

- We execute the script "create_app.sh"
- It initializes the vm via Vagrant fi on the subject
- It is checked that the application is accessible via "foubarre.com" on the browser after déploiment and execution in the script vm "post_deploy_symlink.sh"
- can alter the local app and re-deploy:

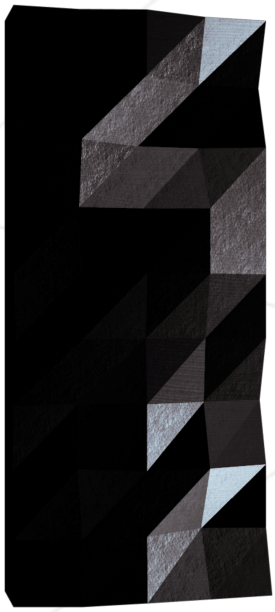
```
modified_file git add git commit -m "message" git push
origin master course bundle exec producing deploy
```

And you must verify that within the VM:

- the app is in "/ home / deploy / apps"
- **the app's file is "own" the user deploy**

- the user deploy neither root nor in the sudoers

Do not go head down, and read your logs: it is the key.



Ruby on Rails Training - Rush 00

Moviemon

Stéphane Ballet balletstephane.pro@gmail.com

Summary: This is the first complex project (everything is relative) you have to do.

Contents

Chapter I

Instructions

- Only this page serve as a reference: Do not fi rm the hallway noise.
- The subject can change up to an hour before rendering.
- If no contrary information is explicitly present, you must assume the following language versions:
  - Ruby 2.3.0
  - for d09 Rails> 5
  - but for all the other days rails 4.2.6
  - HTML5
  - CSS 3
- We prohibit EXPRESSLY using keywords while, for, redo, break, retry and until in Ruby source code that you will make. Any use of these keywords is considered cheating (and / or unfit), giving you the score of -42.
- The exercises are precisely ordered from simple to more complex. In any case, we do not bear attention or take into account a complex exercise if a simpler exercise is not very successful.
- Attention to the rights of your fi les and your directories.
- You must follow the rendering process for all your exercise: only the present work on your GIT repository will be evaluated in defense.
- Your exercises will be evaluated by your pool mates.
- You should leave in your repertoire no other fi le than those explicitly speci fi ed by the forward drills.
- You have a question ? Ask your neighbor to the right. Otherwise, try your left neighbor.
- Your reference manual called Google / man / Internet / ....
- Think discuss on the forum of your pool Intra!
- Read the examples carefully. They might require things that

not otherwise specified in the subject ...

- Please, by Thor and Odin! Re fl échissez dammit!

chapter II

Specific Rules of the day

- You must make an application stateless
- You should not use database
- You should not use ActiveRecord
- This means:
  - Your class in / model (s) will inherit from nothing
  - Your / your controller (s) inherit (as by default) of ApplicationControl- l
  - You are entitled (the app is not intended to be multi-user) to 4 global variables: $ view, $ selected, $ $ and game player
- The backups will be stored valid JSON.

Chapter III

Preamble

Intro scene from the movie La_Classe_Américaine

V12 - *V12 called Captain George Abitbol, V12 called Captain George Abitbol. Someone asks you on the bridge.*

George - *Who ?*
V12 - *A man named Jose.*
George - *OK, I'm coming, V12.*
José - *Ah, that finally the king of the class!*
José - *The man too well-dressed, Abitbol!*
José - *So, you've been elected the most world class man! Let me laugh !*
José - *Style big Playboy Seabed kind that thrills household. Except me I fuck me, housewives, right? It's not true ?*

George - *Listen to me, my little Jose. You fuck housewives, well, you should have your ass shining. But that's not what we call the class. I'm telling you this as a more world class man.*

José - *Well, I'll stop now. The class is to be smart in the way he ha-biller.*

José - *Nothing better than going to Azzedine Alaia. . . . . or even to buy pullovers at Yohji Yamamoto!*

George - *Sorry to tell you this, my poor Jose, but you confuse a little while.*
George - *You make an amalgam between coquetry and class. You are crazy. You spend all your money on clothes and. . . Fashion Accessories. . . but you're ridiculous. Finally if you like it. . . It is you who the doors. But me, if you want my opinion, it's a little. . . has been.*

José - *The cow ! Me, I look has-been? I have for over a clothes bar on me. So fuck you put it!*

George - *You're really not very nice. But the train of your insults rolling on the rail of my indi ff erence. I prefer to leave rather than to hear that rather than being deaf.*
José - *Well ! Considers that one is no longer friends, Abitbol!*

chapter IV

mandatory part

IV.1 Introduction - FAQ

This rush to aim to make you encode a small single-player game with a web interface.

What is the purpose of this game?
This game is called Poke .. Heuu wait no, this game is called MovieMon and its goal is to capture all ...
-Movie-Mons- hiding on a game grid by using
-Movie-balls-.

What is Moviemon?
A Moviemon is a film available on IMDB. Ideally a film Monster.

How do you get a Moviemon?
By lowering its hit points to 0 before you. The IMDB rating of Moviemon
will be its strength. A high rating makes Moviemon more di ffi cult to catch a low rating. The strength of the player, which is the number of Moviemons in his possession, increases the chances of catching.

How is a typical new part?
When a player starts a new game, the game application on IMDB all films required before sending the 'Worldmap', the main page of the game. On this page, the player moves freely and cheerfully from box to box on a grid of fixed size. Random boxes on which he walks, he uncovers a Moviemon. Two choices S'o ff then rent to him if it feels that its chances, the player tries to capture it. Otherwise, he fled cowardly.

If its energy falls below 0 before moviemon, this one escapes and leaves the game. If, instead, the moviemon is 0 before the player, then the moviemon is captured! The player can then consult its fi proudly Moviedex which lists all Moviemons

captured, before returning to hunt for all catch them! !

IV.2 instructions

IV.2.1 Rulez

Your game must obey some rules:

- the design of the front must be that of a gray laptop video game station has green screen (see screenshots).
- buttons should be clickable areas of the main screen layout map image-
- The size of the game grid must be a minimum of ten squares tall and ten squares wide.
- the game must start with a title screen
- the game is set on the grid screen (map) which should fi gure a map image and a player image (defining its position).
- on the game screen, the button 'start'va the "Movie_dex" and returns to the game screen via the' start 'button.
- if functionality via the buttons are present, a legend should indicate (eg "Press [A] to skip").
- click another button as defined by the legend does nothing (not even an error).
- the "movies_mons" must be loaded at the beginning of the game.
- the "movie_dex" is a list of "movie_mons" captured, the right and left buttons will to cycle through the pages of this circular fashion list: go left from the first item in the list should take you to the end of that list.

The API IMDB is rather obscure, you can use OMDB Which is an unofficial API, or even: Random movie .

IV.2.2 game data

You will need to retain data between di ff erent pages at a scéance game. A typical website would use cookies, or a system of server side ses- sions. But here there is no question of typical website.

You will need to store data in global variables. You have four at your disposal 'view', 'game', 'player' and menus 'selected'

So you must also create in your project, the logic needed to update and use of this fi le that should contain the following information:

- The player's position on the map.
- The names (or identifiers) of all Moviemons of Moviedex.
- Full details of all Moviemons of the game, as obtained on IMDB.

IV.2.3 Data Management

You must also create one (or more) class (es) whose mission is to manage these data set This (These) class (es) doi (s) may contain at minimum the following methods.:

- 'Initialize'
- 'Save': Writes the data set to a valid JSON.
- 'Load': Plays the game data from a file and uses it to assign these data to the variables of the game.
- 'Get_movie': Returns an array or hash containing all the details of name Moviemon Spent parameters necessary to page Detail.

You can add to vo (be) s class (es) all methods and attributes as necessary.

On the title screen, the button 'select' a ffi che celèbres the 3 'save slots' and load a backup. On the game screen (the map), the button 'select' as a ffi che three 'save slots' and it can save the current game. On both screens to 'save slots', navigating with the fl up and down arrows.

IV.2.4 Aesthetics game

The a ffi chage the game will naturally on your browser via the HTML and CSS.
You do not have permission to use Javascript.

The a ffi chage of the game is split into two parts to be visually distinguishable perfectly:

- Screen: A ffi che what happens in the game. No interaction is possible at this point that should never contain any link or any form.

- Controls : Located below or on either side of the screen, they can interract with the game and are contextual, ie they change compor- ment according to the state of the game. This also means that they are not necessarily all ne- assets systematically. However, even inactive, they must be visible and keep the same place permanently.

  There must be new 'buttons':

  - Four directions, such as one might find on a directional pad Joystick: Up, Right, Down, Left

  - A button select.
  - A button start.
  - A button Power (small red LED)
  - A button AT
  - A button B

You do not have permission to add / remove 'buttons' nor a ffi expensive infor- mation, apart from the name of 'buttons' in this area.
Beyond this minimum distinction, aesthetics does not matter in the mandatory part of the subject.

The behavior of buttons for each view is described in the next section.

IV.2.5 Pages

You must create pages / views / behaviors listed if after. A 'button' men- tioned not a page is a 'button' inactive. Moreover, if the destination is not specified for a check is that it sent back the same page, potentially amended.

TitleScreen

- Description: Welcome Screen
- Display: Must a ffi expensive the name of the game and 'Start - New Game' and 'Select - Load'.
- controls:
  - Start access Thu
  - Select access Save slots
  - Power " reboot "the game, reset global variables and returns to Title screen. worldmap

- Description: card game, where the character moves, and flushes of Moviemons.
- Screen: A grid whose size is that defined in the settings. On the box corre- sponding to the current position of the player must find a representation (image, character, etc ...) clearly identi fi able character.

- controls:
  - directions: Each department must move the character one square in the same direction. The player must not get out of the map. Each movement has a chance to flush a Moviemon
  - start: access Moviedex.
  - select: access Save slots.

Refresh this page does not change the position of the character on the card.

Battle

- Screen: A ffi che post the name Moviemon, the director's name, and your energy

  If captured, you must also expensive ffi a sentence like "You caught it" to mark the event.

  If a leak, you must also expensive ffi a sentence like "You coward!" .

- controls:
  - AT : hit moviemon
    Here the player launches a hit and subtracts its hit_point to the energy of moviemon, who also hit the turn with his hit_point to him who his' imdbRating. If successful (the energy of moviemon reaches 0 before you), the Moviemon
    is captured and stored in the MovieDex. You ffi Cherez has a right message, and return the energy of the player to the max, but also increase their hit points (and yes, experience it pays).

    If unsuccessful, a ffi you Cherez an adequate message and you delete the mo- viemon the list available and you would redirect the player on the map.

  - B: flight to worldmap
    In this case the moviemon is released, player's energy is restored and back on the map.

Moviedex

- Screen: A Moviemon captured with information: Year, Genre, Director,
  imdbRating, Synopsis and Poster
- controls:
  - Right and left : Directions to select a film diff erent. You must use at least two directions: left and right.

  - select: Back to page worldmap

Default arriving on the page, the first film in the list is selected.

Save slot

- Description: Allows you to make backups of the game in the fi le 'save.json' or from the title screen to 'load' backup.
- Screen: A ffi che three slots to save / load
- controls:
  - Up and down : Selects the slot to handle
  - Select: return to the screen precedant
  - the rest : AD LIB

Chapter V Part

Bonus

Once your required part perfectly realized, you can implement additional functionality in order to earn bonus points.

For your checker sees these features as successful, you will have to convince their good achievement.

An unhandled error invalidate the functionality in question.

Only under the bonus, the use of Javascript is tolerated as long as the latter does not affect the operation of the compulsory part (which must work without JavaScript). The AJAX or Websockets are not allowed.

Here are some bonus ideas you could implement:

- Match the controls to the keyboard in order to avoid having to use the mouse to play.
- In order to vary the game, made that each new part load a selection di ff E- monsters pension.
- Add variety to the worldmap with impassable items.

chapter VI

Rendering and peer-assessment

You must make a project tracks perfectly configured.

Apart from what is imposed on you in the subject, you are free to organize the project as you wish.

No server error will be tolerated. Test out the behavior of your views. You must provide a file requirement.txt containing all the necessary libraries to run your project.

Your models should not inherit anything, you may have migration generated by rails but nothing in the db

Veri fi cation with the following easy commnade executed in the console:

```
ActiveRecord :: Base.subclasses.map {| forbid | forbid.name}
```

chapter VII

Example Rendering

figure VII.1 - Your titlescreen Could look like that

figure VII.2 - This game Moviemon Appears to happen in A familiar map

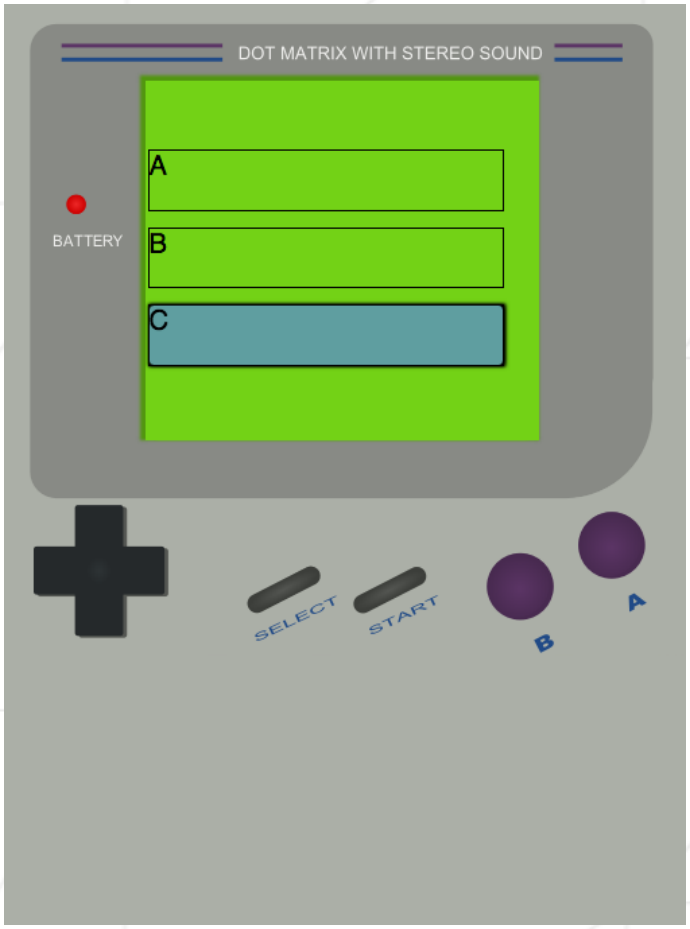


figure VII.3 - Save slot with 'c'selected

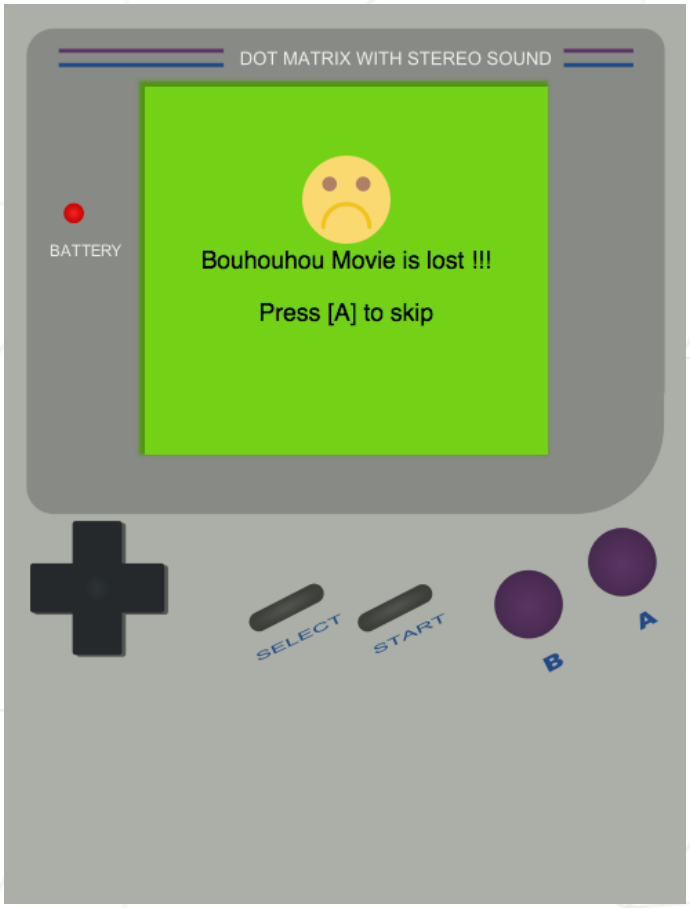
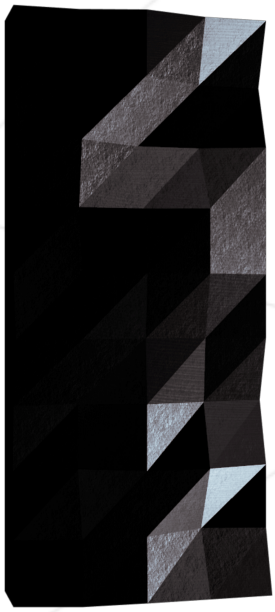


figure VII.4 - A battle lost messages

figure VII.5 - A fi ght screen

figure VII.6 - Your moviedex Could look like that



Ruby on Rails training - Rush 01

Every companies want some

Stéphane Ballet balletstephane.pro@gmail.com

Summary: This is the second complex project (everything is relative) you have to do.

Contents

Chapter I

Instructions

- Only this page serve as a reference: Do not fi rm the hallway noise.
- The subject can change up to an hour before rendering.
- If no contrary information is explicitly present, you must assume the following language versions:
  - Ruby 2.3.0
  - for d09 Rails> 5
  - but for all the other days rails 4.2.6
  - HTML5
  - CSS 3
- We prohibit EXPRESSLY using keywords while, for, redo, break, retry and until in Ruby source code that you will make. Any use of these keywords is considered cheating (and / or unfit), giving you the score of -42.
- The exercises are precisely ordered from simple to more complex. In any case, we do not bear attention or take into account a complex exercise if a simpler exercise is not very successful.
- Attention to the rights of your fi les and your directories.
- You must follow the rendering process for all your exercise: only the present work on your GIT repository will be evaluated in defense.
- Your exercises will be evaluated by your pool mates.
- You should leave in your repertoire no other fi le than those explicitly speci fi ed by the forward drills.
- You have a question ? Ask your neighbor to the right. Otherwise, try your left neighbor.
- Your reference manual called Google / man / Internet / ....
- Think discuss on the forum of your pool Intra!
- Read the examples carefully. They might require things that

not otherwise specified in the subject ...

- Please, by Thor and Odin! Re fl échissez dammit!

chapter II

Specific Rules of the day

-
- Outside the rails gems you can have only these Gems (including bonus):
  - Pdf generation:
    - Prawn
    - Wicked pdf
    - PDFKit
    - Wkhtmltopdf
  - Registration:
    - Motto
    - OmniAuth
    - Authlogic
  - Styling:
    - Bootstrap-sass
    - Rails-bootstrap-forms
  - Ajax:
    - Best in Place
  - Rich text:
    - wysiwyg-rails
  - CSV:
    - roo
  - Charts:
    - LazyHighCharts
  - Date range:
    - jquery datepicker

Chapter III

Preamble

usefull Facts

Google runs on 5000 times more code than the original space shuttle The microcontroller inside a Macbook load is about as Powerful as the original Macintosh computer.

"The quick brown fox jumps over the lazy dog" is an English-language pangram, a sentence That contains all of the letters of the alphabet.

If you eat a teaspoon of sugar after-eating something spicy, it will completely Call neutralized lize the heat.

A day on Venus is along than a year on Venus.

The term "nerd" originated from Dr. Seuss' 1950's book, If I Ran The Zoo. "Scotland's national

animal is the unicorn

In ancient Rome, When a Man testi fi ed in court Would he swear On His testicles. A donkey will sink in

quicksand goal mule will not.

In the 40's, the Bich pen Was changed to Bic for fear That Americans Would Pronounce it'Bitch. '

A woodchuck breathes only 10 times in hibernation. A cat's jaw can

not move sideways. An ostrich's eye is bigger than icts brain.

If you put a drop of liquor was scorpion, it will go mad and sting Instantly Itself to death.

The average person falls asleep in seven minutes.

The Declaration of Independence Was written on hemp (marijuana) paper. A group of kangaroos

s'intitule a mob.

IBM's motto is' Think. Apple later made Their motto 'Think diff erent. Albert Einstein and Charles

Darwin Both married Their fi rst cousins. Charlie Chaplin once won third prize in a Charlie Chaplin

look-alike contest. Most dust particles in your house are made from dead skin!

In the "May I have a wide container of co ff ea" award, the number of letters in words Each Makes the (begining of) the number pi.

It Would take Approximately 31.7 years to count o ff 1 trillion seconds. Bikinis and buffers

Were invented by men. Kissing is more hygienic and Healthier than shaking hands. It is not

possible for a human to sneeze with your eyes open.

chapter IV

mandatory part

You have arrived at this second rush that will close two weeks of swimming. This is an opportunity for you to demonstrate your skills with Rails to prove your extraordinary e ffi ciency of statup-o-freelanc-o-wak-o-Developer.

The 'views' are not imposed, it is for you to de fi ne if everything is on one page, or if there is one word per page. However, the layout is required and a minimum of structure:

- incitement: leads the user to e ff ectuer of specific actions.
- distinction: Group functions similar facilitating access, mémorisa- tion and learning.
- readability: clarify and limit the number of elements di ff erent present on the in- terface
- brevity: a ffi expensive content to facilitate the use of data

These are not the rules of the rush, but an incentive to facilitate common sense, too, your correction.

You do not have to do tests. I mean: it will not be assessed, but it is never a safe from the e ff ects of the development of a feature.

You must by cons have seed that su ffi t demonstrate the good operation of each functionality.

It is also advised to read the whole topic before starting;).

IV.1 Iteration 1: Functionality Not Trade

You must first perform a basic project with several parts, chacunes gathered in a sprint:

IV.1.1 Sprint1: User accounts

You must set up a sytem of authentication, with the compulsory feature:

- No password in clear (in logs or DB)
- Form // sign-in sign-out
- A ffi indication:
  - Log in
  - noti fi cation system of flash events
  - Opportunities for modifying their information users er And a management

privileges for the types of accounts:

- admin: has its back-o ffi ce own with total control all the CRUD.
- operator: Basic functionality (access to personal data), and must belong to one of the following groups (it can belong has one):
  - Production
  - International
  - Commercial
- manager: Basic functionality (access to personal data), access to back-o ffi ces of EVERY sprints (except admin), read access to the personal data (all), and assigning users to groups.
- unregistered: access only to public areas.

Remember: You do not have permission to use the gems that will taff in your place, like 'rolify', 'can-can' ...

IV.1.2 Sprint 2: Admin back-o ffi ce

As stated above, there must be all CRUD any application re- presented here, all you'll develop throughout the years realize now must be added here and open access for purposes of administration.

Only able to him:

- assign / modify privileges to users.
- create / modify a 'user account' and delete.

The admin is -all-puissant- on the data, but does not necessarily creating errors: 'Stored nil gains', 'no single IDS' or 'unvalid data'

Remember: You do not have permission to use gems like 'admin-track', 'Active-admin ...

IV.1.3 Sprint 3: storefront

A home page to make a presentation of your company sham. This part is public

This page is your responsibility and must have must:

- links to: register, login.
- pictures, one (or more) video (s) Back-o ffi

ce:

- No

IV.1.4 Sprint 4: In email

Do I explain the concept of 'in email? Well, when in doubt, I list at least the mandatory features:

- in box: for all users registered with all the in-mail to destinatiion the user connected, and mention of status' or read again.
- out box: for all users registered with all the in-mails emvoyés by the user connected. With mention 'read' if the recipient opened.
- send: anyone can send 'messages in' individuelement.
- Contact list: list of all registered users with links to "in email that person" (not I ca not explain), and mention the group to which it belongs
- pdf: the in-mail Receipts / sent have a button that generates a PDF with the content of the email (through the browser's print function will not be admitted, going ... you guessed it.). Back-o ffi ce:

- Sending mail in-groups.
- Sending mail to all.
- activity history, list of items with text 'read' if opened by the recipient, in the manner of a log [date / time] [title] [sender] [recipient] [link to content]:
  - For everyone
  - For users
  - For groups
- Similarly, a manager can read through the history in all-mails (not meuu is not fl ippant, Mr Tuttle)

IV.2 Iteration 2: Commercial Functionalities

After having set up a basic infrastructure, you now have to oc- cuper the profession of the heart of your sham company.

IV.2.1 Sprint 1: Customer Database

As its name suggests, this is a list of customers and will prototype as follows: name, first name, email, phone,

company, function

It should be accessible by all registered users and readable order tick attending literacy and company. It should also be able to import new via CSV through the gem roo . Back-o ffi ce:

- Export to CSV
- History of activity: by company and by customer, we must see actions concerning these entities. (In the current state of the code is irrelevant, but with the features of the next sprint, ca will make sense). There must fi gure Entre others:
  - Imports, when and by whom
  - Quotes, when and by whom
  - Orders, when and by whom
  - Invoices, when and by whom
- deleting entries
- Customer assignment to operators (all groups combined)

IV.2.2 Sprint 2: Project Suite

One project:

- a name, a client and one or more quotes (quotation proposal to a client by an operator)
- an order (command creates an operator at the customer's request)
- one or more in-going invoices (FUNCTIONING fee invoice, freight)
- an out-going invoice (final customer billing, consistent with good order) Each of these items has:

- A header which figure:
  - a customer
  - his company
  - its operator
  - a creation date
- items that have:
  - a price
  - a description
- an intro 'rich text'
- total euro '€'

Operators can create invoices, quotes and orders that have to belong to a project.

you are exempt from VAT concept not thank me

Back-o ffi ce:

You will always calculate on a granularity of 1 week (X-axis) of MON tives (y-axis), and on time ranges selectable via the plugin 'pick-a-day' in the page header. There are three pages: by customer, by business and for all, where fi gurent the following two graphs:

- Total revenue (out-going invoices) minus expenses (in-going invoices)
- Sum of orders

Only the manager can create a project.

IV.2.3 Sprint 3: Survey

Each operator can create a poll, polls are public and you must save the mails (syntactically valid) non-registered respondents.

The surveys include:

- A title
- An intro en'rich-text '
- questions to answer yes / no
- a thank you message en'rich- text '. Back-o ffi ce:

- creating multiple surveys
- validation (setting public / visible) surveys
- Import CSV of new polls
- Export CSV surveys of results
- Listing and export CSV of collected mail

IV.3 Bonus

You probably imagine that the bonuses will be only evaluated than if all else fonctione perfectly.

Many 'smart-features' have been omitted in the subject are consistent and useful things to add to the few they use any additional gem.

A Bonus is an improvement of A sprint and the layout, it's not serious bonus.