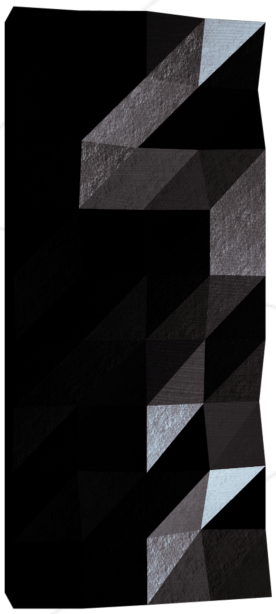


D00 - Piscine Python-Django

Web basics

Summary: This first day will allow you to familiarize with the basics of web development. Here's what's on the menu: `HTTP`, `HTML`,`css` and integration of Javascript scripts existing in your pages.

Contents

Chapter I

Preamble

Here is what Wikipedia has to say about *Balaenoptera musculus* :

The blue whale (Balaenoptera musculus) is a marine mammal belonging to the baleen whale suborder Mysticeti. Reaching a maximum confirmed length of 29.9 meters (98 feet) and weight of 177 tonnes (190 tons), it is the largest animal known to have ever existed.

There are currently five subspecies of blue whale, recognized by the Society for Marine Mammalogy's Committee on Taxonomy: *B. m. musculus* in the North Atlantic and North Pacific, *B. m. intermedia* in the Southern Ocean, *B. m. brevicauda* (the pygmy blue whale) in the Indian Ocean and South Pacific Ocean, *B. m. indica* in the Northern Indian Ocean, and *B. m. unnamed subsp.* in the waters off Chile. The blue whale diet consists almost exclusively of euphausiids (krill).

Blue whales were abundant in nearly all the oceans on Earth until the end of the 19th century. They were hunted almost to extinction by whaling until the International Whaling Commission banned all hunting of blue whales in 1967. The International Whaling Commission catch database estimates that 382,595 blue whales were caught between 1868 and 1978. The global blue whale population abundance is estimated to be 10,000-25,000 blue whales, roughly 3-11smaller concentrations in the Eastern North Pacific (1,647), Central North Pacific (63-133),North Atlantic (1000-2,000), Antarctic (2,280), New Zealand (718), Northern Indian Ocean (270), and Chile (570-760).It is considered an endangered species.

No whale was harmed during the writing of this subject.

Chapter II

Ocaml piscine, general rules

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

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

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

Chapter III

Exercise 00

Exercise 00
Exercice 00: First shell script
Turn-in directory : *ex*00/
Files to turn in : `myawesomescript.sh`
Allowed functions : `curl, grep, cut`

If `Twitter` has no secret to you, you probably know `bit.ly`: a very useful URL shortening service.

The goal of this exercise is to write and turn-in a shell script that displays the real address of a supposedly valid `bit.ly` address (that is, "the address the `bit.ly` link leads towards).

As stated in this exercise header, you can only use the following shell commands: `curl`, `grep` and `cut`. You best bet is to start reading the `curl` manual. To do so, type `man curl` in your terminal.

Here is an example of how your shell script should behave:

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

The example above clearly shows your script must be an executable. You must use `/bin/sh` as an interpreter.

Turn-in your script in the `ex00` folder at the root of your repo.

Chapter IV

Exercise 01

	Exercise 01
Exercice 01: Your resume in HTML	
Turn-in directory : *ex*01/	
Files to turn in : `cv.html`	
Allowed functions : `n/a`	

You will write your resume in `HTML/css` respect the following constraints:

- You must respect the semantics of your HTML tags, as well as the separation between style and content.
- You must create a consistent HTML file with the minimum required content: name, surname, skills and career path.
- You must display at least one title with the `title` tag and a title with the `h1` tag.
- You must use at least one table with the `table`, `th`, `tr` and `td` tags.
- You must use at least a list with the `ul` tag and a list with the `ol` tag. The elements must use a `li` tag.
- The table borders must be visible (`solid`). The table borders must be merged (`collapse`).
- The lowest right cell of a table must have a `#424242` border color.
- You must use a different syntactic solution for each previous instuctions: for the first one, use the `style` tag in the `head` of your page. For the second, use a `style` attribute in the tag you see fit.

`No special instruction about the veracity of informations. You can craft a crazy resume if you like, as long as you follow the instructions above.`

Chapter V

Exercise 02

	Exercise 02
Exercice 02: Email sending form	
Turn-in directory : *ex*02/	
Files to turn in : `form.html`	
Allowed functions : `n/a`	

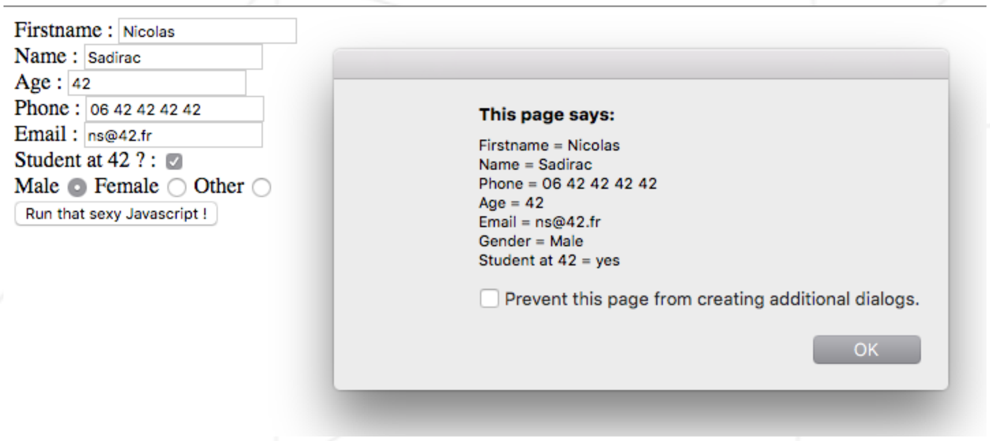
Create a `HTML` form that represents the usual informations of any contact. This form will show the following fields:

- `Firstname`: a text field.
- `Name`: a text field, also.
- `Age`: you must use the specific numeric field specific to the `HTML5`.
- `Phone`: you must use the tel field specific to the `HTML5`.
- `Email` : you must use the email field specific to the `HTML5`.
- `Student at 42?`: you must use the checkbox field.
- `Gender`: you must use radio buttons with the values `Male`, `Female` and `Other`.
- A form submission button. The `onclick` attribute of your button must be: `'displayFormContent`

The tarball `d00.tar.gz` in this subject appendix contains a `ex02/` sub-folder that contains a `Javascript` `popup.js` file written by your boss's son, who's an intern in your company. And since you would not like to have your boss's son feel like an incompetent slob as far as programming goes, you cannot modify his file, which must be used as is.

`A thorough reading and a superficial understanding of the provided Javascript code are required to complete this exercise.`

You must correctly integrate this `Javascript` file in your `HTML` page. If your `HTML` code is correct, pushing the form button will make a super modern popup appear. It will contain the fields and values of your form. If it doesn't, your `HTML` code is flawed.



Firstname : Nicolas
Name : Sadirac
Age : 42
Phone : 06 42 42 42 42
Email : ns@42.fr
Student at 42 ? : ✓
Male ● Female ○ Other ○

Run that sexy Javascript !

This page says:

```
Firstname = Nicolas
Name = Sadirac
Phone = 06 42 42 42 42
Age = 42
Email = ns@42.fr
Gender = Male
Student at 42 = yes
```

Prevent this page from creating additional dialogs.
OK

Figure V.1: Non contractual illustration of the expected result.

Chapter VI

Exercise 03

	Exercise 03
Exercice 03: Web page replicating	
Turn-in directory : *ex*03/	
Files to turn in : `copy.html`	
Allowed functions : `n/a`	

A competing business has uploaded a website that's nicer than yours. Thanks to a serious mission of industrial espionage, your boss gets a screenshot of a page and its `css` file. You can access both those files in the appendix of this subject in the `d00.tar.gz` archive and its `ex03/` sub-folder.

You must replicate this page as faithfully as possible!

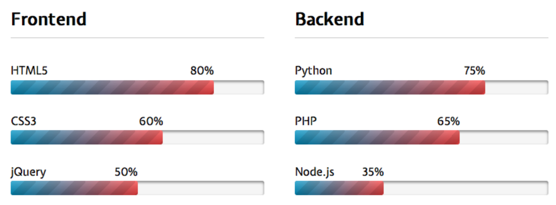


Figure VI.1: The screenshot of the page you must replicate. Image scale is non-contractual.

Once again, you will have to separate style and content, respect the tags' semantics you will use and keep the structure logical in your document.

You must use the provided `css` file without modifying it. A "fresh" version of the `css` will be used during the evaluation in order to check whether you have followed this instruction.

Chapter VII

Exercise 04

	Exercise 04
Exercice 04: Snippets JS integration.	
Turn-in directory : *ex*04/	
Files to turn in : `snippets.html`	
Allowed functions : `n/a`	

The `d00.tar.gz` tarball in this subject appendix contains a `ex04/` sub-folder that contains the same 4 files: `file1.js`, `file2.js`, `file3.js` and `file4.js`.

You must create and turn-in a `snippets.html` file that must import the 4 scripts so that the pop-up appears **correctly** (meaning no strange characters should show).

```
You cannot import the specified scripts.  You cannot modify them.
You cannot add Javascript in your HTML code.
```

Chapter VIII

Exercise 05

	Exercise 05
Exercice 05: W3C validation.	
Turn-in directory : *ex*05/	
Files to turn in : `Your edited index.html`	
Allowed functions :	

Code is nice. Nice code is better. And write nice code, you should follow a nice norm.

The `norme W3C` is a staple, and you have to respect its form when writing or generate `HTML`.

In the `d00.tar.gz` tarball located in this subject appendix you will find the `ex05/` subfolder. It contains the sources of a complete web page. Unfortunately, it was written by a developer far less skilled than yourself!

Edit the `HTML` code of the `html` index file so it can pass the W3C validation! This means neither error nor *warning*.

Your must *edit* the file, not truncate it. This means the file's content you will edit *must* be included in it totality in your repo.