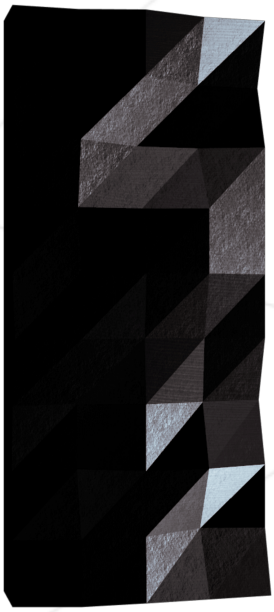

D03 - Django-Python Training

Python - libraries

Damien Cojan dcojan@student.42.fr
42 Sta ff pedago@staff.42.fr

Summary: Today we'll handle some practical bookstores Python.

Contents

Chapter I

Preamble

Geohashing From Wikipedia, the free encyclopedia

Geohashing is an outdoor recreational activity inspired by the webcomic xkcd, participants in qui-have to reach a random rental (Chosen by a computer algorithm), prove Their achievement by Taking a picture of a Global Positioning System (GPS) recei- ver Mobile Reviews another gold device and Then tell the story of Their trip online. Proof is not based electronic navigation est acceptable.

Whereas other outdoor recreational activities like geocaching Have a precise goal, geo- hashing is Mainly fueled by ict pointlessness, Which is deemed funny ict by players. The resulting and geohashing community and culture is extremely THUS tongue-in-cheek, suppor- ting any kind of humorous behavior During the practice of geohashing and resulting and in a parody of traditional outdoor activities. Navigating to a random spot need not be point-less. Some paper geohashers new mapping features They fi nd on the OpenStreetMap project.

Source Wikipedia

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 should assume that the versions of languages and frameworks used are (or later):
  - Python 3
  - HTML5
  - CSS 3
  - Javascript EM6
  - Django 1.9
  - psycopg2 2.6
- Unless otherwise indicated in the subject, the fi les in python each year on
  Python one (d01, d02 and d03) must have their end block if __name__ == '__main__': in order to insert the entry point in the case of a program, or tests in the case of a module.
- Unless otherwise indicated in the subject, each year days on
  Django will have its own application in the project to make for reasons peda- gogiques.
- The exercises are precisely ordered from simple to more complex. In no event shall we bear no 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.
- Unless otherwise specified in the topic you should not include in your rendering:

- The files __ pycache__.
- Any migration.
- The file created by the command collectstatic of manage.py (with the way the value of the variable STATIC_ROOT).
- The fi le Python bytecode (fi les with extension. pyc).
- database fi les (especially with sqlite).
- Any fi le or folder must or can be created by the normal behavior of the rendering work.

It is recommended that you modify your. gitignore in order to avoid accidents.

- When you need to get an accurate output in your programs, it is of course forbidden to a ffi expensive output precalculated instead of performing the exercise cor- rectly.

- 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 are not otherwise specified in the subject ...

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

chapter III

Specific Rules of the day

- No code in the global scope. Make functions!
- Each fi le made must be completed by a function call in the same condition:

```
if __name__ == '__main__':
        your_function (whatever, parameter, is, required)
```

- It is permissible to place a error handling in the same condition.
- No authorized import, except for those explicitly mentioned in the 'Authorized Functions' of the cartridge each year ..
- The interpreter is to use python3.

Chapter IV

Exercise 00

	Exercise: 00 Exercise 00:
Antigravity	
Render Folder: *ex* 00 / Files to make: geohashing.py	
Permitted functions: sys and antigravity modulus	
Remarks: n / A	

This is a small exercise échau ff ly echoing the preamble of the day. Nothing complicated.

Make a small program named geohashing.py that takes as many parameters as needed to calculate a typical Geohash and therefore must obviously calculate this Geohash before a ffi expensive standard output.

On error, the program has a ffi expensive a relevant message you chose before exiting cleanly.

This diagram can, perhaps, help: Geohashing algorithm

Chapter V

Exercise 01

	Activity: 01 year 01:
Pip	
Render Folder: *ex* 01 / files to make: my_script.sh my_program.py	
Permitted functions: path.py Module	
Remarks: n / A	

path.py is a library that implements a Path object around the module os.path
Python, making it very intuitive.

In this exercise, you create a script bash that installs this library, and a program Python who is using it.

The shell script must meet this description:

- Its name must have the extension. sh because it is a script Shell.
- It is a ffi expensive version used pip.
- It must install the development version path.py since its repo GitHub,
  in a file that must be called local_lib, placed in the output folder. If the library has already been installed in this folder, the installation must then crush him.
- He must write the installation log path.py in a file having the extension
  . log
- If the library has been properly installed, it must ultimately carry the small pro- gram that you must also create.

The program Python to create is a composition of your choice, which must nevertheless comply with these restrictions:

- Its extension should be. py because it is a program Python.
- It must import the module path.py from where this library is installed, thanks to the previous script.
- It should create a folder and a file on within that folder, write something in this fi le and then finally read a ffi expensive content.
- It must respect the specific rules of the day c.

Chapter VI

Exercise 02

	Exercise 02 Exercise 02:
querying an API	
Render Folder: *ex* 02 / Files to make: request_wikipedia.py requirement.txt	
Permitted functions: modules requests, JSON, and dewiki sys	
Remarks: n / A	

Wikipedia is a great tool shared you know necessarily. It is dis- ponible on your favorite browser and even as mobile application. I propose now to create a tool that allows you to query the site now become essential, right from your device.

To do this, you must design a program named request_wikipedia.py
that takes a parameter string and e ff ectue a search through the API Wikipedia
before writing the result in a file. You can choose to query the API English or French.

- The program needs to write a result, even if the query is misspelled. Take the original site for example, if it finds you a result for a given request, then your program too.
- The result should be free of any formatting JSON or Wiki Markup before being written in the fi le.
- The name of the file must have the format nom_de_la_recherche.wiki and must not contain any space.
- If no parameters, wrong settings, invalid application of in- formation not found, server problem or any other problem: no expensive fi should be created and an appropriate error message to be a ffi ket on the console.
- Include in your rendering a file requirement.txt that will be used in defense to install the necessary libraries to your program in a virtualenv

or on the system.

The dewiki library is not perfect, we do not seek the cleanest possible result, it is not the purpose of this exercise.

Read the API documentation. Observe the structure of what you will be returned.

Here's an example of what we expect:

```
$> Request_wikipedia.py python3 "chocolatine" $> cat chocolatine.wiki




A chocolatine means:
*   a pastry with chocolate, also called chocolate croissant or chocolate couque;
*   a pastry has patissiere cream and chocolate, also called Swiss;
*   a kind of chocolate candy;
*   a book of Anna Rozen


Despite its ancient usage, the word is not in the dictionary between Webster in 2007 and in the Petit Larousse in 2011.


The use of the term "Chocolatine" is also found in Quebec, whose language has evolved from
    old french french differently from employees in Europe, but this usage proves nor denies any anticipation, dependent chance to use the first trader having introduced in
 Quebec. References




Category: Confectionery Category:
Chocolate
```

Chapter VII

Exercise 03

	Exercise 03 Exercise 03:
Parser html	
Render Folder: *ex* 03 / Files to make: roads_to_philosophy.py, requirement.txt	
Permitted functions: Requests and BeautifulSoup modules	
Remarks: n / A	

Legend says that if you go to any Wikipedia article, you click on the first link in the introduction to this article is neither italicized nor brackets, then you repeat this process loop, you aboutirez invariably on Article Philosophy.

Well aware that this is not a legend! (Make 'Oooooh!' With a bewildered air). Witness this article ... of Wikipedia.

However, as you believe what you see with your own eyes, you **must** create a program that puts this to the test by listing then COMP as all items between your query and the Wikipedia: the **roads to philosophy.**

This program must be named roads_to_philosophy.py and have the following behavior:

- The program must take a string parameter which is a word or set of words corresponding to a single Wikipedia search.
- The program must querying a URL of Wikipedia **English** identical to that of a standard search on a browser. In other words, it is not **not allowed** of UTI Liser API site.

- It must parse the page html through the library BeautifulSoup in order to:
  - Find the possible redirection, and take into account in **roads to philosophy.** Warning, it is not about url redirection.
  - Find the main title of this page and add it to **roads to philosophy.**
  - Find (if it exists) **the first link in the introductory paragraph** condui- health to another Wikipedia article. Rather than ignoring what is in italics and brackets, the program must **ignore** Carefully links that do not point to a new article, such as those leading to the help section Wikipedia.

- The program should start from the **2nd step** obtained starting from the link to the previous step to fall on one of the following cases:
  - The link leads to the page philosophy, in which case it should print on the standard output sections visited and the total consideration of these items in the format < number> roads from <application> to philosophy
  - The page contained no valid links The program has a ffi expensive: It leads to a dead end!.
  - The link leads to a previously visited page, which means that the program is ready to wrap ap- inde fi rect handling. In this case the message ffi ket must be: It leads to an infinite loop!

- At this stage, after a ffi ket on the standard output the necessary messages, the program should exit gracefully.

If at any time in the execution of the program, an error occurs, such as an error connecting to server, setting, request or other: the program should exit gracefully with an appropriate error message.

As in the previous year, vou will need to provide your program with an expensive fi requirement.txt in order to facilitate the installation of libraries.

The output of your program should look like this:

```
$> Python3 roads_to_philosophy.py "42 (number)" 42 (number) number Natural
Mathematics Ancient Greek Greek Language Modern Greek colloquialism Word




Linguistics Science Knowledge
Conscious Awareness
Consciousness Quality
(philosophy) Philosophy



17 roads from 42 (number) to philosophy! $> Python3
roads_to_philosophy.py Accuvio It's a dead end! $>
```

The Wikipedia community regularly puts its products up to date. It is possible that between the time of writing this and that to which you do this training, the example accuvio no longer a dead end.

Chapter VIII

Exercise 04

	Exercise 04 Exercise 04:
virtualenv	
Render Folder: *ex* 04 / Files to make: requirement.txt my_script.sh	
Permitted functions: all	
Remarks: n / A	

Tomorrow is the first day of training on the framework Django. You need to prepare the ground by configuring a little easy installation.

You have to create two elements:

- A file requirement.txt which must contain the latest stable versions of django and of psycopg2.
- A script must have this behavior:
  - Have the extension. sh
  - To create a virtualenv under python3 appointed django_venv.
  - Install the fi le requirement.txt you created in this Virtualenv.
  - Leaving the virtualenv must be **activated.**

Chapter IX

Exercise 05

	Exercise 05 Exercise 05:
Hello World	
Render Folder: *ex* 05 / Files to make: all necessary file	
Permitted functions: all	
Remarks: n / A	

Simply install Django must be frustrating, and that's perfectly understandable.

So you will end this day on the beauty in bookstores by designing your first Bonjour Monde with Django.

In the last year, you must monitor and adapt the tutorial o ffi sky in order to make a web page, from the address http: // localhost: 8000 / helloworld, a ffi che simply text Bonjour Monde ! in your browser.

Your made should be a folder containing the project Django.