

Notions avancées

En construction. Les liens sont juste donnés. Une introduction et un exemple devrait être proposé pour chaque rubrique, et le nombre de ces rubriques augmenté.

Itérateurs

Itertools, zip,...

- [7 Levels of Using the Zip Function in Python](#)

Générateurs et "yield"

- <http://fr.openclassrooms.com/informatique/cours/pratiques-avancees-et-meconnues-en-python/es-generateurs-2>
- <http://feldboris.alwaysdata.net/blog/python-les-iterateurs-et-les-generateurs-fr.html>
- <https://wiki.python.org/moin/Generators>
- <http://sahandsaba.com/combinatorial-generation-using-coroutines-in-python.html>
- <http://code.activestate.com/recipes/580628-pluggable-python-generators/>
- [Processing Large Data Sets With Yield and Generators](#)

Liste en compréhension

- http://fr.wikipedia.org/wiki/Liste_en_compr%C3%A9hension
- <http://www.pythonforbeginners.com/basics/list-comprehensions-in-python>
- <http://fgallaire.flext.net/comprehension-de-liste-en-python-map-filter/>, remplacement de map() et filter()
- http://www.python-course.eu/list_comprehension.php, suppression de lambda et reduce()
- <https://www.datacamp.com/community/tutorials/python-list-comprehension>
- <https://gist.github.com/bearfrieze/a746c6f12d8bada03589>
- [A Beginner's Guide to Python List Comprehensions](#) (Jonathan Hsu, Medium, 23/04/2020)

map, filter, reduce, lambda, pipe

- [Write Clean Python Code Using Pipes - A Short and Clean Approach to Processing Iterables](#) Khuyen Tran, medium, october 2021
- ...

Le rôle du caractère underscore en Python

- [Role of Underscore\(_\) in Python Tutorial](#) - In this tutorial, you're going to learn about the uses of

[underscore\(_\) in python](#) Hafeezul Kareem Shaik, Medium (DataCmap) October 26th, 2018

- https://www.geeksforgeeks.org/underscore-_-python/

Transformations et manipulations de chaînes (string)

- `translate()` : transformation sur base de correspondances entre des caractères (y compris des caractères spéciaux (\n, \t, \r,...)
 - `string.punctuation` peut être utilisé pour enlever la ponctuation
- `replace()` : remplacement d'une sous-chaîne
- `split()` : découpe en une liste de sous-chaînes. Le caractère utilisé par défaut est l'espace. Le nombre de découpe peut être précisé. `rsplit()` permet de commencer par la droite.
- `partition()` : découpe particulière → `myString.partition("search string")` renvoie un tuple de trois sous-chaînes : (texte précédent, première occurrence de la sous-chaîne cherchée, texte à la suite)
- `strip()`, `lstrip()`, `rstrip()` : élimine des caractères en début et/ou fin, par défaut les espaces
- `zfill(n)` : ajoute des zéros devant pour arriver à une longueur donnée
- `' '.join()` : joint des sous-chaînes en liste pour créer une chaîne en ajoutant ' ' (dans cet exemple) comme sous-chaîne additionnelle
- `upper()` : capitalise
- `lower()`
- `title()` : capitalise les premières lettres des mots
- `swapcase()` : inverse capitales et bas de casse
- `startswith('fizz')`, `endswith('buzz')`, `'fizz buzz' in test_string` : renvoie True ou False

Manipulations de fichiers

- module `os`
 - [8 Must-Know File System Operations In Python - The essential for Python in tasks automation apps](#) Christopher Tao, Medium, 08/02/2021
 - [Python Directory and Files Management](#) (tutorial)
 - official documentation : <https://docs.python.org/3/library/os.html>
 - [10 Python File System Methods You Should Know - Manipulate Files and Folders With os and shutil](#) Jeff Hale, Medium, Feb 15, 2019 (pathlib, OS, shutil)
 - <https://medium.com/better-programming/the-top-10-file-handling-techniques-in-python-cf2330a16e7> : The Top 10 File Handling Techniques in Python - Make working with files easier Yong Cui, Aug 5, 2020
 - <https://medium.com/python-in-plain-english/manipulating-file-paths-with-python-72a76952b832> ??

Expressions rationnelles (régulières)

- ```
import re
```
- <http://howchoo.com/g/zdvmogrlnz/python-regexes-findall-search-and-match>

- <https://docs.python.org/2/howto/regex.html>
- <http://linuxfr.org/news/travailler-avec-des-expressions-rationnelles>
- [https://fr.wikipedia.org/wiki/Expression\\_rationnelle](https://fr.wikipedia.org/wiki/Expression_rationnelle)
- [Beginners Tutorial for Regular Expressions in Python](#)
- [Python Regular Expressions — cheat sheet - Many code examples + useful tips](#) Valeria Aynbinder, Medium, 17/03/2022
- ...

## Décorateurs

- [http://www.python-course.eu/python3\\_memoization.php](http://www.python-course.eu/python3_memoization.php)

## Context managers

## Programmation orienté objet

Page dédiée : [Programmation Python Orientée Objet](#)

- Exemples simples :
  - <http://nbviewer.ipython.org/url/bender.astro.sunysb.edu/classes/python-science/lectures/python-classes.ipynb>
  - <http://jeffknupp.com/blog/2014/06/18/improve-your-python-python-classes-and-object-oriented-programming/>


## Divers

- Définir ses propres types : stack, queue, tree + algorithms : [Data Structures & Algorithms in Python](#) by Papa Moryba Kouate, Aug, 2020, Towards Data Science

## Closures

- <http://stackoverflow.com/questions/36636/what-is-a-closure>
- <http://programmers.stackexchange.com/questions/40454/what-is-a-closure>

## Programmation fonctionnelle

- [Functional Programming in Python](#) ()
- Map, filter, reduce :
  - [How To Replace Your Python For Loops with Map, Filter, and Reduce - Write more semantic code with functional programming](#)

## Débogage, debugging

- [Stop Using Print to Debug in Python. Use Icecream Instead](#)

## Sous le capot (bytecode,...)

- [Efficiently Checking for an Empty List in Python](#) Frank Scholl, Medium, Nov 22, 2019 → comparaison des bytecodes Python/C générés pour 3 solutions proposées

## Performances, temps d'exécution, ...

- [Making Python Programs Blazingly Fast](#), 01/01/2020

## Tests unitaires

## Python Scripting

- [How to Run Python Scripts](#)

## Interface utilisateur graphique (GUI)

- [How to build your first Desktop Application in Python](#) Ampofo Amoh - Gyebi, Medium, 12/12/2020

## emails

smtp lib et pop lib :

- <https://twitter.com/driscollis/status/1508506999268552713>
- <https://twitter.com/driscollis/status/1508570173606928386>

```
import smtplib

HOST = "smtp.mydomain.com"
SUBJECT = "Test email from Python"
TO = "mike@mydomain.com"
FROM = "python@mydomain.com"
text = "blah blah blah"
BODY = "\r\n".join((
 f"From: {FROM}",
```

```
f"To: {T0}",
f"Subject: {SUBJECT}",
"",
text)
)
server = smtplib.SMTP(HOST)
server.sendmail(FROM, [T0], BODY)
server.quit()
```

```
import poplib

mailbox = poplib.POP3('pop3.host.com')
mailbox.user("USERNAME")
mailbox.pass_("PASSWORD")
numMessages = len(mailbox.list()[1])
for i in range(numMessages):
 for j in mailbox.retr(i+1)[1]:
 print(j)
```

## Création et gestion de packages

- [How to create the first python package](#) ChenChih, Medium, Dec 18, 2021

## Références

- [Top 12 most important Python concepts](#) Dacus Augustus, Medium, Feb 23 2021
- [10 Advanced Python Tricks To Write Faster, Cleaner Code - From slotted classes to replacing lists with tuples](#) Erik van Baaren, Medium, 30/06/2021

From:  
<https://dvillers.umons.ac.be/wiki/> - **Didier Villers, UMONS - wiki**

Permanent link:  
[https://dvillers.umons.ac.be/wiki/teaching:progappchim:notions\\_avancees?rev=1651134028](https://dvillers.umons.ac.be/wiki/teaching:progappchim:notions_avancees?rev=1651134028)

Last update: **2022/04/28 10:20**

