Guide du programmeur Python

Ce document est une introduction au langage Python

Il s'appuie sur la version 3.10
dans un environnement Linux

Bonne lecture…

Pierre ROYER Manager | Architecte | Formateur #numérique

I. INDEX

II. PRÉAMBULE

A. Ce document

Informations

Nom du document	DevOps.Python-3.8.docx	Référence	DEVOPS-PYTHON
Version	2020.02.02	Pages	15
Date de création	22/02/2020	Dernière modification :	09/01/2024
Auteur :	Pierre ROYER Tél : (+33) 614 672 909 https://www.linkedin.com/in/pierreau	Contributeur(s) :	
Mode de diffusion	☐ confidentiel ☐ restreint ☒ interne ☐ libre	Liste de diffusion	https://pierreau.fr
Annexes :			

B. Conventions

Les syntaxes utilisées dans ce document :

[root@Rocky ~]# représente un prompt bash en root sur un serveur Rocky
[python@Rocky ~]$ désigne le compte local dédié aux scripts
[root@debian ~]# représente un prompt bash en root sur un serveur Debian
root@FreeBSD:~ # représente un prompt bash en root sur un serveur BSD

Le contenu d'un fichier est encadré, les commandes sont en gras :

```
[python@Rocky ~]# vi /Scripts/python.py
```

```
print ('Il fait beau')
```

Les caractères en italique sont des exemples de paramètres :

```
Ip = 192.168.100.100
HostName = ServeurA
```

Information utile	Attention particulière	Risque important
		

III. INTRODUCTION

A. Présentation

Python est un langage de programmation interprété (non compilé en langage machine), multi-paradigme (objet, fonctionnel, processus) et multiplateformes de haut niveau. Il est distribué sous licence libre.

Les premières versions construites par par Guido van Rossum datent de 1989, et seront diffusées sur Internet, via le réseau (forum Usenet **alt.sources**, environnement Unix) en 1991.

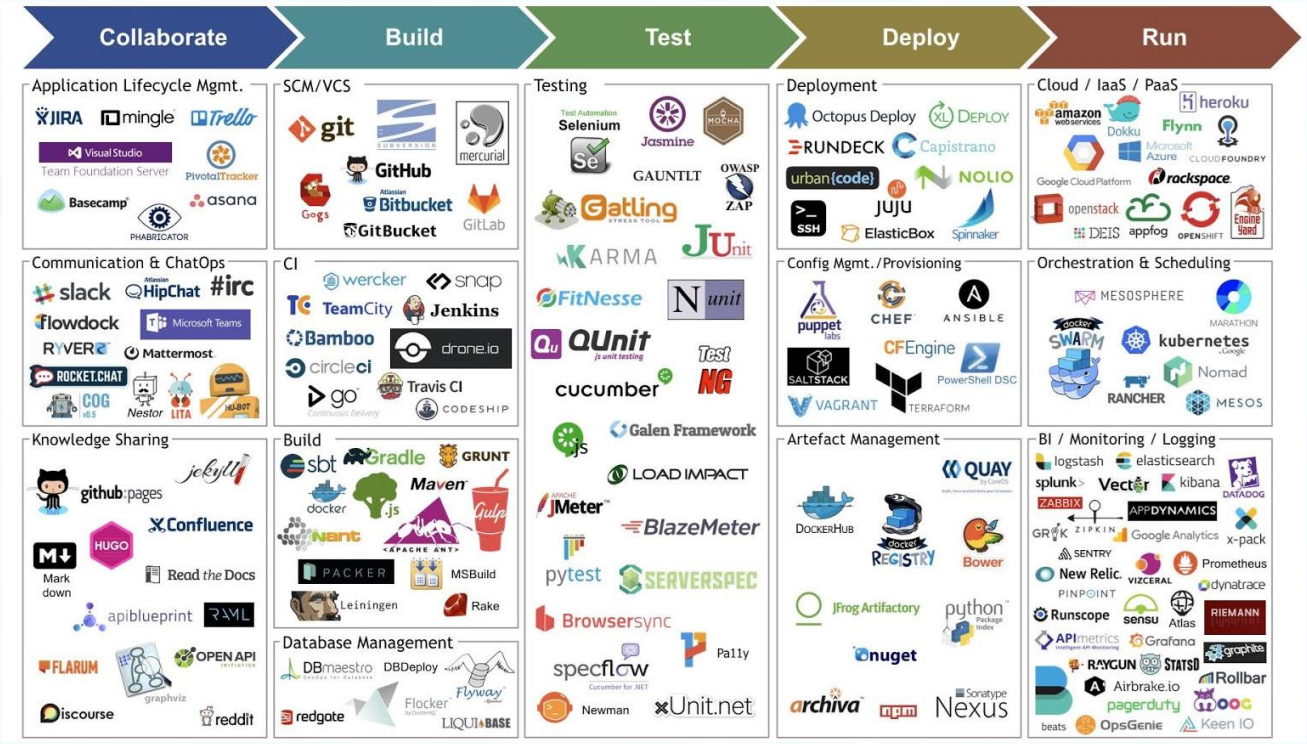
Depuis Python 3.0, il n'y a plus de compatibilité ascendente par défaut pour les anciennes versions.

Les **PEPs** (Python Enhancement Proposal) sont des propositions textuelles d'amélioration de Python : orientation pour le développement (process PEP), proposition technique (Standard Track PEP) ou une simple recommandation (Informational PEP).

➔ https://www.python.org/dev/peps/

B. Contexte CI / CD

Pyhon fait pleinement partie de la chaîne d'indégration continue DevOps, entre autres dans les phases de déploiement. Ansible est d'ailleurs une solution 100% écrite en Python.



IV. Installation

A. Rocky

[root@Rocky ~]# représente … un système Rocky Linux

B. Debian

Mise à jour des dépôts :

```
[root@debian ~]# apt-get update && apt upgrade -y && apt install ansible -y
```

C. FreeBSD

```
root@FreeBSD:~ # pkg install python
```

D. Environnements de développement

- https://thonny.org/
-

V. Prise en main

A. Interactions avec prompt

1. Affichage

```
>>> print ('Il fait beau')
Il faut beau
```

2. Saisies au clavier

```
>>> firstname = input('Quel est ton prénom ? ')
Quel est ton prénom ? Pierre
>>> print('Bonjour', firstname !')
Bonjour Pierre !
```

3. Calculs

```
>>> (4+2*5)/2
7.0
```

B. Sortie de l'interpréteur

Ctrl-Z et Entrée

VI. LES VARIABLES

Python est un langage de typage dynamique. Il n'est pas nécessaire de déclarer le nom et le type des variables, pour ensuite leur assigner un contenu.

A. Types de données

1. Booléen

```
b = True
print(b)
True
print(type(b))
<class 'bool'>
```

Opérateurs de comparaison :

Description	Notation
Égal	==
Différent	!=
Strictement plus petit	<
Plus petit ou égal	<=
Strictement plus grand	>
Plus grand ou égal	>=

```
>>> a=1
>>> a==1
True
>>> a==0
False
>>> a!=1
False
>>> a!=0
True

>>> 1 + 2 <= 3 + 4
True
```

Est équivalent à :

```
>>> (1 + 2) <= (3 + 4)
True
```

Opérateurs logiques :

Description	Notation
« non » logique	not
« et » logique	and
« ou » logique	or

```
>>> a==1 and b==1
```

```
True
>>> a==True and b==True
True
```

Priorités sur les opérateurs :

Priorités	
1	les quatre comparaisons : **<**, **<=**, **>** et **>=**
2	les égalités : **==** et **!=**
3	le « non » logique : **not**
4	le « et » logique : **and**
5	le « ou » logique : **or**

Enchainement des opérateurs

```
>>> 0 <= a and a <= 1
True
```

Est equivalent à :

```
>>> 0 <= a <= 1
True
```

B. Chaines de caractères

1. Principe

2. Répétition

3. Découpages

Il est possible d'extraire une sous-chaîne (**slice**) d'une variable :

4. Concaténation

5. Formatage

6. Séparateurs et fin de chaine

C. Nombres entiers et flottants

D. Nombres complexes

VII. Formatages & Placeholder

A. Placeholder

B. Formats d'affichage

1. Numériques
2. Alignements

C. Les dates

VIII. Premier script

IX. Les conditions

A. if

1. Condition simple
2. Condition alternative
3. Imbrication de conditions

X. Données structurées

A. La liste

https://python.sdv.univ-paris-diderot.fr/04_listes/

1. **Indexation**
2. **Indices négatifs**
3. **Consultations et extractions**
4. **Slices**
5. **List & Range**
6. **Liste de liste**

A. Le tuple

1. **Principes**
2. **Le tuple nommé**

B. Les stacks

1. **LIFO**
2. **FIFO**

C. Le dictionnaire

XI. LES BOUCLES

A. for

B. while

1. **Interruption d'une boucle**

. . .

XII. Les fonctions

A. Fonction sans paramètre

B. Fonction avec un paramètre

C. Fonction avec plusieurs paramètres

D. Paramètres par défaut

E. Packing et unpacking

F. Valeur de retour

G. Portée des variables

XIII. Les modules

A. Module OS

Ce module est propre à la gestion du système d'explotation

1. **Environnement**
2. **Fichiers**
3. **Random**
4. **Reste du module**

B. Modules personnalisés

XIV. Docstring

A. Format Google

B. ReStructuredText

XV. La Programmation objet

La programmation orientée objet (POO) permet de créer des entités (objets) que l'on peut manipuler, et qui peuvent interagir entre eux. C'est à mon sens une évolution par rapport à la programmation « procédurale ».

A. Les classes

B. Les objets

1. Attributs

2. Héritages multiples et surcharge

3. Méthodes

On appelle les fonctions d'une classe des « méthodes ».

Les méthodes de classe

Les méthodes statiques

4. Encapsulation

XVI. Services Web

A. Serveur

B. Client

XVII. Gestion des erreurs

XVIII. Approche avec GIT

XIX. Travaux pratiques

I. Ressources externes

Documentation officielle : https://docs.python.org/fr/3/contents.html

Wiki : https://wiki.python.org

Ememples : https://www.w3schools.com/python/

Site en français :

https://python.doctor/
https://www.tresfacile.net/
https://python.developpez.com/cours/apprendre-python3

Cours scientifiques : https://courspython.com/