CONSEIL ET EXPERTISE TECHNIQUE

Infrastructure ▪ Cybersecurity ▪ Cloud ▪ Data

Python 01

Matthieu DESTOMBES
Mail : matthieu.destombes@ynov.com

22 Fevrier 2021
Toulouse

C1 - Public

Présentations

- Votre intervenant
  - Qui suis-je ?
  - Ma formation
  - Mon parcours
- Et vous ?
  - Qui êtes-vous ?
  - Votre formation
  - Vos attentes

Objectifs définis

- L'histoire de Python
- Les bases de Python
- La POO avec Python
- Python graphique

L’histoire de **Python**

- Les origines de Python (Histoire)
- Les points forts de python
- L’utilisation

L’histoire de Python
=> Les origines

- Les débuts
- La vertu pédagogique de Python
- Deux générations majeurs

https://fr.wikipedia.org/wiki/Python_(langage)

L’histoire de Python
=> Les points forts

- Facile à lire / comprendre
- Productif
- Puissant
- Portable
- Langage interprété
- Variables facile à créer/utiliser
- Modules multiples
- Libre

L’histoire de **Python**
=> L’utilisation

- Conquête spatiale
- Physique des particules
- Jeux vidéo
- Industrie
- Système d’exploitation
- Domotique

Le fonctionnement de **Python**

- Environnement de travail
- Les commentaires
- L’indentation
- Les erreurs

Le fonctionnement de Python => Environnement de travail

- L’interface de commande
- IDLE
- IDE spécifique gratuit / payant

Le fonctionnement de Python => Les commentaires

- Avec [#] ou ["""]
- Notes explicatives
- Aides mémoire
- Aide visuel
- ...

- A indiquer à l'interpréteur par le caractère « # » en début de

```
# This is a sample Python script.
print("Hello")
```

```
#!/usr/bin/env python3
# coding: utf-8
# ##############################################################################
# Description:  This script contains common function.
#
# Required:     - Run as Standard User.
#               - Python 3.x
#
# Author:       DESTOMBES Matthieu
#
# Date:         2020.11.30
# ##############################################################################

# ==============================================================================
# IMPORTS
# ==============================================================================

import os
import datetime


# ==============================================================================
# SUB FUNCTIONS
# ==============================================================================
def file_format_date(
        input_date=datetime.datetime.now(datetime.timezone.utc)
):
    """
    Return date in string format dedicated for file naming.

    Args:
        input_date:
          (Datetime) Input date in datetime format.
          Default set to current date.

```


Le fonctionnement de Python => L'indentation

- Remplace les délimiteurs comme « { » et « } »
- Définition de bloc
- Utilisation d'espace
- La norme est 4

```
# Create it if not exists
if not os.path.exists(basedir):
    print("- '{0}' created".format(basedir))
    os.makedirs(basedir)
else:
    print("- '{0}' already exist".format(basedir))

# Remove output file if already exist
if os.path.isfile(output_file):
    print("- '{0}' removed before creation".format(output_file))
    os.remove(output_file)
```

Le fonctionnement de Python => Les erreurs

- Erreurs de syntaxe
- Erreurs de logique

- Apprendre de ses erreurs

```
# This is a sample Python script.
print("Hello world)
```

```
  File "C:/Users/MatthieuDestombes/Documents/Ynov/Python/Pycharm/test/version.py", line 2
    print("Hello world)
                       ^
SyntaxError: EOL while scanning string literal
```


Les bases de Python (Part I)

- Les mots réservés
- Les variables et leurs actions associées
- Les opérateurs
- Les structure de contrôle
- Les fonctions simples

Les bases de Python (Part I) => Les mots réservés

- Que veux dire « mots réservés »
- Correctement les utiliser

- and
- as
- assert
- break
- class
- continue
- def
- del
- elif
- else
- except
- exec
- finally
- for
- from
- global
- if
- import
- in
- is
- lambda
- not
- or
- pass
- print
- raise
- return
- try
- while
- with
- yield

Les bases de Python (Part I)
=> Les variables

- Utilité
- Unique
- Affectation
  - [nom_variable] = [valeur_numérique]
  - [nom_variable] = '[chaine_de_caractère]'
  - [nom_variable] = "[chaine_de_caractère]"
- Utilisation
  - [nom_variable]

```
my_string = "Hello"
my_number = 10
```

```
print(my_string)
print(my_number)
```


Les bases de Python (Part I)
=> Les opérateurs

```
my_number_1 = 100
print(my_number_1)
```

```
100
```

```
my_number_2 = my_number_1 + 10
my_number_3 = my_number_1 - 10
my_number_4 = my_number_1 * 10

print(my_number_2)
print(my_number_3)
print(my_number_4)
```

```
110
90
1000
```

- Arithmétiques
  - Addition (+)
  - Soustraction (-)
  - Multiplication (*)
  - Incrémentation (+=)
  - Division (/)
  - Modulo (%)
  - Puissance (**)
  - Décrémentation (-=)

```
my_number_5 = my_number_1 / 10
my_number_6 = my_number_1 % 3
my_number_7 = my_number_1 ** 2

print(my_number_5)
print(my_number_6)
print(my_number_7)
```

```
10.0
1
10000
```

- Conditionnel
  - Affectation (=)
  - Egalité (==)
  - Différence (!=)
  - Supérieur (>)
  - Supérieur ou égal (>=)
  - Inférieur (<)
  - Inférieur ou égal (<=)
  - ET logique (**and**) (&)
  - OU logique (**or**) (|)

Les bases de Python (Part I)
=> Les structures de contrôle

```
print(my_number_1)
print(my_number_2)
print(my_number_3)
print(my_number_4)
```

```
100
110
90
1000
```

- Structure de contrôle ?
- Séquentiel
  - if [condition]
  - elif [condition]
  - else
- Itératif
  - for [element] in [structure]
  - while [condition]

```
if my_number_2 == my_number_1:
    print("Equal")
elif my_number_2 < my_number_1:
    print("Lower")
else:
    print("Upper")
```

```
Upper
```

```
for current_element in my_number_1, my_number_2, my_number_3, my_number_4:
    print(current_element)
```

```
100
110
90
1000
```

```
while my_number_1 >= my_number_3:
    print("Increment")
    my_number_3 += 1
```

```
Increment
Increment
Increment
Increment
Increment
Increment
Increment
Increment
Increment
Increment
Increment
```


Les bases de Python (Part I)
=> Les fonctions simples

- Pourquoi des fonctions ?

- Fonction simple
  - def [nom_de_la_fonction]

```
def my_function():
    print("I'm in the function!")
```

```
for current_position in range(3):
    print("\nPosition: {0}".format(current_position))
    my_function()
```

```
Position: 0
I'm in the function!

Position: 1
I'm in the function!

Position: 2
I'm in the function!
```

```
for current_position in range(10, 13):
    print("\nPosition: {0}".format(current_position))
    my_function()
```

```
Position: 10
I'm in the function!

Position: 11
I'm in the function!

Position: 12
I'm in the function!
```