BES Langages avancés

Romain Alis, Pierre Elyakime

22 septembre 2024

Le langage Python

Introduction

https ://www.python.org

C'est un langage initié par Guido van Rossum dans les années 1990 et qui est maintenant géré par the Python Software Foundation.

Caractéristiques

- Multi-plateformes
- Langage interprété
- Programmation orienté objet
- Open source
- Grande communauté d'utilisateurs
- Bibliothèque de modules
- Évolution rapide

Exemples d'applications

- Traitement de données.
- Simulation numérique.
- Interfaces graphique.
- Machine Learning/IA.
- Bases de données.
- Imagerie médicale.
- Contrôle/acquisition de capteurs.

Python permet de tout faire **SI** on sait où chercher.

Le langage Python

Quelques points essentiels

Quelle version de Python ?

La nomenclature est faite avec 3 numéros : Python X.Y.Z

- X –> Numéro majeur de la version
- Y –> Numéro mineur de la version
- Z –> Numéro de révision de la version

Actuellement, la dernière version de Python mise en ligne est la 3.12.6.

Attention !

Python 2.x.x n'est plus maintenue, il ne faut plus l'utiliser.

Python 2.x.x et Python 3.x.x

La compatibilité n'est pas assurée entre Python 2.x.x et Python 3.x.x :

- Python 2.x.x –> print a
- Python 3.x.x –> print(a)

Le langage Python

L'utilisation de Python

Comment fonctionne Python ?

Un interpréteur lit les instructions et demande à la machine de les exécuter.

Méthode 1

- Lancer l'interpréteur dans un terminal.
- Taper les commandes au fur et à mesure.

Méthode 2

- Écrire les commandes dans un fichier (.py).
- Lancer l'interpréteur avec le nom du fichier en argument.

Méthode 3

- Les IDEs, Integrated Development Environment
- Spyder, Anaconda, Pycharm, Jupyter, ...

Le langage Python

Deux approches (paradigmes) différentes

L'approche "scripting"

- On écrit la liste les opérations à éffectuer dans un fichier.
- On exécute le fichier.

```
#!/usr/bin/python
# -*- coding:Utf-8 -*-

a = 4
b = 10

print("a+b=", a + b)

a = 15
b = -10

print("a+b=", a + b)
```

L'approche orienté objet

- On définit des objets : quelles données et comment les manipuler.
- On utilise ces objets.

```
#!/usr/bin/python
# -*- coding:Utf-8 -*-

class Compute:
    def __init__(self, a=0, b=0):
        self.a = a
        self.b  = b

    def show_sum(self):
        print("a+b=", self.a+self.b)

if __name__ == "__main__":
    obj_comp = Compute()
    obj_comp.show_sum(a=4, b=10)
    obj_comp.show_sum(a=15, b=-10)
```

C'est à l'utilisateur de choisir en fonction de ses compétences et des ses objectifs.

L'approche orienté objet en Python

Les concepts essentiels

- **Classes** :
  Définition d'un type d'objet.
- **Attributs** :
  Variables définies dans une **classe**.
- **Méthodes** :
  Fonctions agissant sur les **attributs** d'une **classe**.
- **Instances** :
  Objets crées à partir d'une **classe**.

```python
#!/usr/bin/python
# -*- coding:Utf-8 -*-

class Compute:
    def __init__(self, a=0, b=0):
        self.a = a
        self.b  = b

    def show_sum(self):
        print("a+b=", self.a+self.b)

if __name__ == "__main__":
    obj_comp = Compute()
    obj_comp.show_sum(a=4, b=10)
    obj_comp.show_sum(a=15, b=-10)
```

- **L'héritage** :
  S'appuyer une classe existante pour définir une nouvelle classe.

L'approche orienté objet en Python

L'importance des modules

- **Un module** :
  Un fichier définissant des variables, des fonctions et des classes.
- **Un paquet/package** :
  Un dossier qui contient des fichiers **modules** (ou même d'autres **packages**).
- **Un gestionnaire de paquets/packages** :
  Un exécutable qui permet de manipuler les **paquets/packages**.

La réutilisation de code existant est un pilier essentiel pour programmer.

Python Package Index (PyPI), `https://pypi.org/`

Site regroupant l'ensemble des packages validés par la Python Software Foundation.

- Permet d'avoir accès à 575k packages.
- Facilite l'utilisation et le partage d'outils Python qui ont déjà été développés.
- Packages développés par la communauté Python suivant la norme PEP.

L'approche orienté objet en Python

Utiliser des modules et des paquets/packages

Importation sélective

```
>>> from fibo import fib, fib2
>>> fib(50)
0 1 1 2 3 5 8 13 21 34
>>> fib2(50)
0 1 1 4 9 25 64 169 441 1156
```

Importation totale

```
>>> import fibo
>>> fibo.fib(50)
0 1 1 2 3 5 8 13 21 34
```

Importation totale à bannir

```
>>> from fibo import *
>>> fibo.fib(50)
0 1 1 2 3 5 8 13 21 34
```

Importation sélective en renommant

```
>>> from fibo import fib as fibonacci
>>> from fibo import fib2 as fibonacci_square
>>> fibonacci(50)
0 1 1 2 3 5 8 13 21 34
>>> fibonacci_square(50)
0 1 1 4 9 25 64 169 441 1156
```

Importation totale en renommant

```
>>> import fibo as fib
>>> fib.fib(50)
0 1 1 2 3 5 8 13 21 34
```

Bonus, dur de savoir ce qu'on manipule...

```
>>> import sound.effects.echo
>>> from sound.effects import echo
>>> from sound.effects.echo import echofilter
```

Installer un paquet/package Python disponible sur PyPi

```
python -m pip install SomePackage          # latest version
python -m pip install SomePackage==1.0.4
```

Une interface graphique Tkinter

Objectifs

- En utilisant l'approche orienté objet, construire une interface graphique Tkinter permettant d'écrire un fichier de paramètres.
- Savoir naviguer dans l'univers de Python.

Deux sujets au choix

- **Sujet 1 :** faire une interface graphique pour rentrer les paramètres d'un code résolvant l'équation de la chaleur 2D.
- **Sujet 2 :** faire une interface graphique pour rentrer les paramètres de CPIV, un code de PIV.

Notation

- Interface fonctionnelle.
- Code structuré et orienté objet.
- Code propre, commenté et lisible.

Ressources

Tutoriels et documentations

- https://www.python.org
- https://pypi.org/
- https://pythonforge.com/ et https://pythonforge.com/classes-python/
- https://www.pythontutorial.net/tkinter/
- https://python.doctor/page-tkinter-interface-graphique-python-tutoriel
- https://sebsauvage.net/python/gui/index_fr.html
- https://www.cours-gratuit.com/tutoriel-python/tutoriel-python-programmer-une-interface-graphique-avec-tkinter-partie-1

Deux conseils pour démarrer

- Savoir associer une variable et un bouton.
- Savoir écrire une variable dans un fichier.

Exemples à analyser (3ème séance)

- Une interface graphique implémentant un jeu.
- Une interface graphique écrivant un fichier.