

Python

Caractéristiques générales

Nicolas Delestre

Historique

- Langage créé en 1990 par Guido van Rossum
 - Branche 2.0 à partir de 2000
 - Branche 3.0 à partir de 2008
- S'inspire de plusieurs langages : ABC, Modula, C, Smalltalk, langages de scripts
- Langage open source (GPL), plusieurs implantations (**CPython**, Pypy, Jython, etc.)
- Versions utilisées dans ce cours **3.9**

Un langage « interprété »

- Deux utilisations possibles de Python :

- ① Depuis un environnement interactif : chaque commande est interprétée avec l'affichage possible d'un résultat
 - python ou python3 : interpréteur de base
 - ipython ou ipython3 : interpréteur étendu (complétion, erreurs plus lisibles, « commandes magiques », etc.)
 - jupyter : interpréteur en ligne (web) très utilisé en science des données
- ② En lançant un script python depuis la ligne de commande
 - Python vérifie l'existence d'un .pyc
 - s'il n'existe ou est plus ancien que le .py, Python compile le code python dans un bytecode
 - Python interprète alors le bytecode

Attention

- Le terme Python désigne quelque fois le langage, quelque fois l'interpréteur

Paradigmes de programmation : introductions ou rappels

Paradigme de programmation

- Un paradigme de programmation est une façon d'aborder informatiquement la résolution d'un problème
- Il propose des concepts et des « outils »
- Un sous paradigme de programmation précise ou ajoute des concepts ou des outils

Paradigme de programmation...

- | | | |
|--|--|-----------------------------|
| • Imperatif <ul style="list-style-type: none">• Structuré | • Orienté Objet <ul style="list-style-type: none">• Orienté Classe• Prototype | • Concurrente |
| • Déclaratif <ul style="list-style-type: none">• Descriptive• Fonctionnel• Logique | • Événementiel | • Orienté métaprogrammation |
| | | • Orienté pile |
| | | • ... |

Un langage peut appartenir à plusieurs paradigmes



Paradigmes de programmation orienté objet

Paradigmes de programmation orienté objet

- Le paradigme de la programmation orienté objet repose sur « la définition et l'interaction de brique logiciels appelées objet [qui] représente un concept, une idée ou toute entité du monde physique » (Wikipédia)
- Un objet a un état qui est défini par les valeurs de ces attributs
- Un objet est dit muable si son état peut changer, immuable sinon
- Les méthodes permettent d'interroger l'état d'un objet, de le modifier (lorsque c'est possible) ou d'obtenir, de créer ou de référencer d'autres objets

Paradigmes de programmation orienté classe

- Le paradigme de la programmation orienté classe est un sous paradigme de programmation orienté objet qui indique qu'un objet est créé à partir d'une classe par instanciation
- L'instanciation crée un objet dans un état initial

Python un langage multi-paradigmes

Avant tout structuré et orienté classe

- Un seul type de sous programme : les fonctions avec passage de paramètre par référence
- Tout est objet : types, fonctions, classes, modules, packages, etc.
- Un objet est l'instance d'une classe qui détermine son type
- Un objet peut être muable ou immuable

Mais aussi du paradigme de la programmation

- Programmation orientée prototype : on peut ajouter, retirer des attributs à un objet
- Fonctionnelle : expression *lambda*, fonction map, filter, etc.
- Concurrente : coroutine
- Métaprogrammation : les classes sont des objets instances de métaclasses

Affectation et typage

Affectation

- Une variable référence un objet grâce à l'opérateur/instruction d'affectation (=) : la valeur retournée est l'objet référencé
- L'instruction `i = 1` signifie que la variable `i` référence l'objet de type `int` représentant l'entier `1` (et non pas `i` vaut `1`)

Typage

- Typage fort : il n'y a aucun transtypage implicite
- Typage dynamique : le type d'une variable est celui de l'objet dernièrement référencé
- *Duck typing* : le type d'un objet est défini par les messages qu'on peut lui envoyer, par ce qu'il « sait faire », et non pas par le nom de son type
« Si je vois un oiseau qui vole comme un canard, cancanne comme un canard, et nage comme un canard, alors j'appelle cet oiseau un canard » (citation attribuée à James Whitcomb Riley, Wikipédia)
Le concepteur d'une fonction doit indiquer dans la documentation des paramètres formels leurs types attendus en termes d'opérations/fonctions applicables

- Fichier obligatoirement en UTF-8 (python 3)
 - on peut utiliser des caractères étendus « accentués, arabes, asiatiques, cyrilliques, grecs, etc. » comme identifiant
- L'indentation marque les blocs
- Sensible à la casse
- Commentaires (commençant par un #) et documentations (chaîne de caractères, ou *docstring*) sont représentés différemment

Bonnes pratiques (<https://www.python.org/dev/peps/pep-0008/>)

- Indentation avec 4 espaces
- Ligne d'au maximum 72 caractères (utilisation de \ pour indiquer la coupure de ligne)
- « Formes » des identifiants varient : en *snake_case* minuscule (variables, fonctions, méthodes, modules, packages), *SNAKE_CASE* en majuscule (constantes) ou en *CamelCase* (classes)
- Une ligne vide entre chaque fonction
- Espaces entre opérateurs (sauf dans les paramètres formels des fonctions) et après les virgules

Conclusion

Nous avons vu dans ce cours

- Python est un langage interprété
- Python est un langage mutiparadigmes, reposant principalement sur les paradigmes de la programmation structuré et orienté classe
- Tout est objet en Python
- L'affectation en Python référence des objets
- Python est un langage à typage fort, dynamique et qui applique le *duck typing*