PROGRAMMATION PYTHON AVANCÉE

Chez le même éditeur

Python 3
2e édition
Bob Cordeau, Laurent Pointal
304 pages
Dunod, 2020

Python pour le data scientist
2e édition
Emmanuel Jakobowicz
320 pages
Dunod, 2021

Python précis et concis
5e édition
Mark Lutz
272 pages
Dunod, 2019

PROGRAMMATION PYTHON AVANCÉE

Guide pour une pratique élégante et efficace

Xavier Olive

Docteur en informatique
Chercheur à l'ONERA Toulouse

DUNOD

Illustration de couverture : © Calin Stan – Adobe Stock

11 rue Paul Bert, 92240 Malakoff
www.dunod.com
ISBN 978-2-10-081598-2

Préface

Lorsque Xavier Olive m'a contacté pour me proposer de relire ce livre que vous tenez actuellement entre les mains, ma première réaction a été « Oh non, encore un livre sur Python ! » Un certain nombre de livres sur Python trônent déjà sur les étagères de mon bureau, même s'il est vrai que la plupart sont en anglais. Mais lors de ce premier échange par mail, Xavier a glissé subrepticement le fait qu'il avait pour but d'*écrire le livre qu'il aurait voulu lire*. Cela a eu pour effet immédiat d'attiser ma curiosité et j'ai donc accepté de relire cet ouvrage sans trop savoir où je mettais les pieds. Après avoir relu les quelque 350 pages qui composent ce livre, je réalise que c'était une très bonne décision de ma part tant le livre est agréable à lire, tant sur la forme que sur le fond.

Sur la forme d'abord, car Xavier a véritablement soigné le style du livre qui tranche avec un certain nombre d'ouvrages que j'ai pu lire par le passé. Ayant une solide expérience quant à la typographie et à la mise en page, je suis admiratif du soin et du souci du détail qui ont été apportés à l'ouvrage. Trop souvent les auteurs négligent cet aspect en se disant que seul le fond importe pour un ouvrage technique alors que la forme peut véritablement jouer un rôle essentiel pour la compréhension de concepts parfois ardus. La relecture et les échanges avec Xavier sur ces aspects ont été d'autant plus agréables qu'il en a une parfaite maîtrise.

Mais la prouesse du livre se trouve bien évidemment sur le fond. Ayant moi-même une assez grande expérience de Python, notamment sur ses versants scientifiques, je reste admiratif de cet ouvrage qui est à la fois bien écrit, bien structuré, bien documenté et surtout extrêmement pédagogique dans son approche. Lors de ma relecture du premier chapitre, je me suis d'abord fait la réflexion qu'il allait un peu vite en besogne en présentant les bases du langage Python, avant de me rappeler qu'il s'agissait d'un ouvrage avancé venant compléter celui de Bob Cordeau et Laurent Pointal qui s'adresse, lui, aux débutants. Or, présenter les bases du langage Python à des utilisateurs avancés est un vrai numéro d'équilibriste. Mais je crois que Xavier a su justement trouver le bon équilibre en choisissant méticuleusement les aspects peut-être moins connus du langage et en les illustrant à l'aide d'exemples pertinents (et pour certains passionnants au niveau théorique, comme les L-systèmes).

Le livre est structuré autour de cinq grandes parties (Les bases du langage Python, L'écosystème Python, Écrire un Python naturel et efficace, Python, couteau suisse du quotidien, Développer un projet en Python) et de trois interludes (Calcul du rayon de la Terre, Reconstruire une carte d'Europe, La démodulation de signaux FM) qui viennent agréablement aérer la technicité de certains chapitres. Étant avant tout un livre pour des utilisateurs avancés, il est évident que Xavier ne pouvait éviter d'être technique sur certains des aspects les plus avancés

de Python. J'avoue ne pas être un grand fan des dernières possibilités offertes par le langage Python car je crois que cela alourdit inutilement le langage mais Xavier a su malgré tout me convaincre de l'utilité de la majorité d'entre elles, notamment lorsque l'on se retrouve à gérer de gros projets collaboratifs.

Je suis donc à la fois très honoré et très heureux d'écrire aujourd'hui cette préface pour un livre qui, je le crois, deviendra un classique. Évidemment, ce n'est que la première édition et, au vu de la rapidité d'évolution du langage Python, je crois aussi que Xavier s'est engagé, sans peut-être le savoir, à écrire une nouvelle édition tous les deux ou trois ans. C'est tout le mal que je lui souhaite. D'ailleurs, à l'heure où j'écris ces lignes (février 2021), le PEP 636 concernant l'identification structurelle de motifs (*Structural Pattern Matching* en anglais) vient d'être accepté, validant ainsi encore un peu plus la 10e règle de Greenspun [1].

Nicolas P. Rougier

Docteur en informatique et chercheur
à l'Inria en neurosciences computationnelles

Février 2021

1. `https://en.wikipedia.org/wiki/Greenspun's_tenth_rule`

Table des matières

Avant-propos

Python est un langage généraliste et multi-plateforme, développé suivant un modèle open source depuis le début des années 1990 : sa première version vient de fêter ses 30 ans ! C'est un langage interprété, populaire pour sa facilité d'utilisation, pour sa polyvalence, et pour les apports de sa communauté. Python est apprécié parmi les scientifiques et les ingénieurs d'horizons divers.

Dans certaines spécialités, d'autres langages peuvent être plus rapides, plus sûrs ou plus complets, mais Python se démarque par sa polyvalence, par la concision et la lisibilité de sa syntaxe, par la facilité avec laquelle on peut écrire un prototype en quelques heures, construire une chaîne de traitements à partir de briques logicielles écrites par d'autres, parfois dans d'autres langages. Enfin, il reste une solution de choix pour outiller des tâches informatiques simples de notre quotidien.

À qui s'adresse ce livre ?

Ce livre s'adresse à un public qui a déjà une bonne expérience de la programmation, que celle-ci soit avec Python ou non. L'ouvrage propose différentes grilles de lecture, avec un contenu théorique de base et des chapitres complémentaires, adaptés à une deuxième lecture. Ceux-ci seront une opportunité de mettre en pratique les concepts et les outils sur des exemples engageants. L'objectif est de présenter au lecteur un ouvrage qui rappelle les concepts-clés pour une utilisation idiomatique du langage et qui les illustre dans des cadres d'utilisation variés.

Si vous débutez en programmation et souhaitez apprendre Python, ce livre sera difficile à suivre. Les structures de données sont reprises en détail, mais la syntaxe du langage et les fondements de la programmation ne sont pas traités. L'ouvrage *Python 3*, de Bob Cordeau et Laurent Pointal aux éditions Dunod, est plus adapté pour s'initier au langage, apprendre des notions élémentaires (boucles, valeurs, expressions, variables, etc.) et découvrir la syntaxe.

Comment est construit ce livre ?

Le contenu ne se limite pas aux seuls aspects proposés par le langage avec ses apports les plus récents (notamment les versions 3.8 et 3.9) mais expose une approche de l'écosystème

Python dans son ensemble, avec une présentation des principales bibliothèques tierces développées par la communauté, devenues aujourd'hui incontournables. Nous abordons aussi les pratiques recommandées de gestion de projet logiciel en Python.

Ce livre s'appuie sur les bases de l'algorithmique et de la programmation, il présente comment des concepts génériques de programmation non spécifiques au langage sont déclinés en Python. Il aide à appréhender le vocabulaire et les mots-clés propres au langage pour rechercher en ligne de manière autonome et efficace les réponses aux problématiques fréquemment rencontrées lors de l'écriture de code.

Les exemples d'application présentés dans cet ouvrage s'appuient sur des rudiments de culture générale relatifs à des domaines variés tels le calcul numérique, le traitement du signal ou l'intelligence artificielle pour les illustrer et les mettre en évidence de manière naturelle et élégante à l'aide du langage Python.

Ce livre se décompose en cinq grandes parties :

— **Les bases du langage Python.** Cette partie reprend les bases du langage en se concentrant sur les structures de données, avec leurs atouts et leurs limitations. De nombreuses structures avancées sont fournies par le langage ; elles permettent de s'attaquer efficacement à des problèmes difficiles.

— **L'écosystème Python.** Python ne se limite pas à ses fonctionnalités et à ses bibliothèques intégrées déjà bien fournies. C'est également une communauté : certaines bibliothèques écrites par des développeurs indépendants et des laboratoires scientifiques sont devenues incontournables.

— **Écrire un Python naturel et efficace.** Un bon programme Python n'est pas seulement un programme qui fonctionne. C'est un code qui suit les conventions de la communauté et qui utilise le langage comme il a été pensé. Cette partie présente comment exploiter les caractéristiques du langage pour écrire un code qui est clair, concis et facile à maintenir.

— **Python, couteau suisse du quotidien.** Python est un langage adapté pour outiller des tâches du quotidien. Cette partie guide le lecteur pour une utilisation du langage orientée vers la manipulation des fichiers standard (images, CSV, Excel, XML, PDF, JSON et plus) et l'interaction avec des services web ouverts. La production d'outils graphiques et en ligne de commande est également abordée.

— **Développer un projet en Python.** Le développement d'un projet Python qui prend de l'ampleur se prépare et se sécurise à l'aide d'un certain nombre de pratiques standard : intégration continue, environnements virtuels, suivi de la performance et de la non-régression. Cette partie reprend les différents aspects de la gestion logicielle et présente des outils standards, couramment utilisés dans la plupart des projets logiciels.

Les exemples de code

Tous les chapitres de cet ouvrage contiennent du code source.

Les commandes à entrer dans un terminal (bash, zsh, etc.), un PowerShell Windows ou une invite de commande Anaconda sont préfixées par `$` :

```
$ which python
/home/xo/.conda/envs/pybook/bin/python
```

Les exemples courts démarrent par « >>> » pour refléter l'invite de l'interpréteur Python. Les retours sont alors affichés à la ligne.

```
>>> import math
>>> math.pi
3.141592653589793
```

Les exemples plus longs ne sont pas exécutables tels quels. Tout ou partie d'un fichier Python est imprimé dans le livre et annoté à l'aide de balises numérotées ①, ②, etc. Les fichiers complets sont disponibles sur `https://www.xoolive.org/python/`. La plupart des exemples ont été testés avec Python 3.8 ; ceux qui ne fonctionnent qu'à partir de Python 3.9 sont annotés pour le préciser.

```
import numpy as np  # ①
```

La première partie du livre sur les bases du langage ne nécessite pas de bibliothèques particulières. Dans les parties suivantes, il est recommandé d'installer un environnement Anaconda (☞ p. 69, § 5), de télécharger le fichier `environment.yml` depuis le site du livre, puis de créer un environnement dédié ①.

Seule la commande `activate` ② devra être exécutée avant chaque lancement de Python.

```
$ conda create --file environment.yml # ①
$ conda activate pybook # ②
$ python
Python 3.9.1 | packaged by conda-forge | (default, Dec 21 2020, 22:08:58)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Conventions utilisées dans l'ouvrage

Les conventions suivantes sont utilisées au long de l'ouvrage :

— le texte *en italique* retranscrit les termes anglais équivalents au vocabulaire utilisé en français, par exemple « bibliothèque (*library* en anglais) » ;

— le texte à `chasse fixe` retranscrit notamment des noms de variables, de modules et des scripts Python.

Attention !

Cet encart met en valeur les erreurs et pièges courants dans lesquels le programmeur, même averti, peut tomber.

Bonnes pratiques

Cet encart met en valeur les bonnes pratiques, couramment admises, qui améliorent la qualité et la lisibilité du code.

Cas d'application. Cet encart met en valeur un exemple appliqué au milieu d'un chapitre théorique.

En quelques mots...

Cet encart conclut un chapitre en rappelant les éléments essentiels à retenir pour comprendre les chapitres suivants.

Page web du livre

Le code source et les illustrations du livre, qui ont été générées avec Python, sont disponibles sur la page web du livre, avec un erratum qui recense les coquilles relevées après l'impression de l'ouvrage : `https://www.xoolive.org/python/`. Il est possible depuis le site de poser des questions, d'ouvrir des discussions et de proposer des corrections aux éventuelles coquilles qui se seraient glissées dans ces lignes.

Remerciements

Ce projet n'aurait pas vu le jour si Bob Cordeau ne m'avait pas présenté Jean-Luc Blanc des éditions Dunod, juste après la soumission du manuscrit de la deuxième édition de son ouvrage *Python 3. Apprendre à programmer dans l'écosystème Python*. Je tiens à les remercier tous les deux, notamment M. Blanc pour la confiance qu'il m'a accordée pour la définition de ce nouveau projet et pour la souplesse avec laquelle il m'a accompagné.

La rédaction d'un tel ouvrage est un moment intense, un marathon dans lequel sont embarqués malgré eux épouse, enfants, proches et collègues. Tous ont participé d'une manière ou d'une autre à ce travail, en faisant face à une indisponibilité qu'ils n'avaient pas choisie, en apportant leur soutien, leur point de vue critique après relectures, leur réponse pendant les moments de doute, et leurs encouragements. Cet ouvrage leur est dédié.

Je remercie notamment Luis Basora et Thomas Dubot, fidèles soutiens et relecteurs de la première heure, ainsi que Judicaël Bedouet pour sa gentillesse et ses métacommentaires avisés. Junzi Sun et Enrico Spinielli ont également contribué à cet ouvrage sans vraiment le savoir, le fruit de nos riches échanges transparaît dans de nombreux exemples.

Nicolas Rougier, auteur de nombreuses références Python de qualité, a répondu à mon invitation, a accepté de relire l'intégralité de l'ouvrage puis m'a fait l'honneur de le préfacer. Brice Martin, des éditions Dunod, a également été d'une grande aide pour la préparation du manuscript. Leurs commentaires ont contribué à améliorer la lisibilité de l'ensemble.

Il faut enfin saluer tous les étudiants qui, par leurs progrès, leurs doutes, leurs interrogations et reformulations, ont contribué à améliorer les ressources pédagogiques préliminaires sur lesquelles sont construites les fondations de ce livre.

Toulouse, février 2021

Prologue

L'apprentissage d'un langage de programmation est un processus progressif. Comme pour une langue vivante, il est tout à fait possible, et même recommandé, de pratiquer un langage même sans en connaître toutes les subtilités. C'est par la pratique qu'on découvre de nouvelles problématiques, des solutions pour y répondre ; puis vient la réflexion pour généraliser ces solutions à des classes générales de problèmes. Un langage vivant est capable de faire vivre de telles réflexions pour proposer des améliorations de celui-ci.

Ce rôle est joué en Python par les PEP, les *Python Enhancement Proposals*, ces discussions levées par la communauté avec des propositions d'améliorations du langage. Certaines améliorations sont controversées, mais les décisions sont toujours prises après échanges d'arguments. Ces améliorations font évoluer le langage et les pratiques mois après mois. Le Python de 1989 n'est pas le même que le Python de 2003, qui n'est pas le même que le Python de 2020.

Python, un langage fondamentalement orienté objet (☞ p. 201, § **15**), s'est petit à petit positionné par rapport à d'autres paradigmes, notamment celui de la programmation fonctionnelle (☞ p. 155, § **12**), pour finir par en adopter certaines pratiques tout en en délaissant d'autres. Le modèle de développement EAFP (☞ p. 53, § **4.3**) a orienté les interfaces vers le modèle des protocoles (☞ p. 225, § **16**).

Les dernières versions de Python ont chacune apporté leur lot de nouveautés qui infléchissent la pratique du langage. Ces dernières années ont été marquées par :

— les *f-strings* du PEP 498 (Python 3.6, ☞ p. 9, § **1.3**) ;

— les *dataclasses* du PEP 557 (Python 3.7, ☞ p. 50, § **4.2**) ;

— la fonction `__getattr__()` du PEP 562 (Python 3.7, ☞ p. 244, § **17.1**) ;

— le *walrus operator* `:=` du PEP 572 (Python 3.8) au terme duquel le créateur du langage Guido van Rossum a quitté son poste de *Benevolent Dictator For Life* (BDFL) ;

— les types standards génériques du PEP 585 (Python 3.9, ☞ p. 329, § **24**).

Ces pages ont pour vocation d'explorer cet environnement changeant du langage, de son écosystème, construit autour des structures de données efficaces de la bibliothèque NumPy (☞ p. 73, § **6**) et des technologies du web vers lesquelles la communauté se tourne progressivement (☞ p. 109, § **9** ; ☞ p. 135, § **11**), ainsi que des fondements théoriques sur lesquels il s'est construit. Elles guident le lecteur dans le monde des bibliothèques tierces de référence conçues

pour faciliter la réalisation de tâches élémentaires : télécharger des données (☞ p. 293, § **20**), extraire des informations d'un document (☞ p. 283, § **19**), les structurer, les transformer ou les visualiser, le tout dans l'objectif d'y apporter une valeur ajoutée.

Le livre se termine sur la conception d'un projet Python (☞ p. 313, § **22**). Les outils évoluent de jour en jour, il n'a jamais été aussi facile qu'en 2020 de partager du code tout en s'assurant qu'il pourra être exécuté sur une nouvelle plateforme, sur un nouveau système d'exploitation, en embarquant les bibliothèques nécessaires à son bon fonctionnement.

Python reste un outil, pas une fin en soi. Comme le dit A. Maslow, *I suppose it is tempting, if the only tool you have is a hammer, to treat everything as if it were a nail*, « J'imagine qu'il est tentant, si le seul outil dont vous disposez est un marteau, de tout considérer comme un clou. »

Ces lignes visitent parfois des contrées atypiques, elles apporteront sans l'ombre d'un doute des éléments pour clarifier une ligne, une fonction, un module ou un projet. Mais la frontière est parfois ténue entre un code devenu plus clair et un code devenu inutilement complexe.

Keep it simple.

I

Les bases du langage Python