

Prendre en main Python sous Windows

par Guillaume Durlaud ([Page personnelle](#))

Date de publication : 15/09/2007

Dernière mise à jour : 10/02/2009

Ce tutoriel a pour but de vous permettre de prendre rapidement en main **Python** sous **Windows** à savoir utiliser un interpréteur interactif, écrire un script Python et l'exécuter. Par contre, il n'est pas destiné à vous apprendre le langage **Python**. Pour cela reportez-vous à un tutoriel écrit pour cela que vous pouvez retrouver sur notre page **Cours Python**

I - Introduction.....	3
II - Installation.....	4
II-A - Choix de l'implémentation de Python.....	4
II-B - Installation avec l'installateur de Microsoft Windows.....	4
III - Interpréteur interactif.....	5
III-A - Introduction.....	5
III-B - Shell interactif / console Python.....	5
III-C - IDLE.....	5
IV - Ecriture d'un script Python.....	8
IV-A - sous IDLE.....	8
IV-B - Les autres Editeurs.....	8
V - Exécution d'un script Python.....	9
V-A - Sous IDLE.....	9
V-B - En mode console.....	9
VI - Pour aller plus loin.....	11

I - Introduction

Vu dans la [FAQ](#) **FAQ Python**:

Une présentation du langage Python est faite dans la première partie du cours de Swinnen qui vous montrera les caractéristiques du langage:  **Apprendre à programmer avec Python**. En voici un résumé:

- Python est portable sur de nombreux OS (Unix, Mac, Windows, ...)
- Python est gratuit et utilisable sans restriction dans des projets commerciaux.
- La syntaxe de Python est très simple, l'indentation du code (plutôt que l'usage d'accolades) permet d'avoir des programmes très lisibles et plus compactes
- Python détruit lui-même les objets créés lorsque plus aucune référence ne pointe sur eux
- Il n'y a pas de pointeurs explicites en Python.
- Python est orienté-objet. Il supporte l'héritage multiple et la surcharge des opérateurs.
- Python est dynamiquement typé.
- Python est extensible : On peut l'interfacer avec d'autres langages (C, C++, Delphi, Java, ...)
- La bibliothèque standard de Python est très riches et de nombreuses bibliothèques gratuites peuvent être facilement ajoutées

Nous allons voir ici comment mettre en place l'environnement de travail autour du langage **Python** sur un ordinateur équipé de **Windows**. Vous verrez que c'est vraiment très simple et que vous serez vite opérationnel. Une fois ceci fait, vous aurez alors tout le loisir de commencer l'apprentissage du langage **Python**

II - Installation

II-A - Choix de l'implémentation de Python

Python peut fonctionner sur de multiples systèmes d'exploitation (Windows, Linux, Mac, ...) mais aussi sur différentes plate-forme (Java, .Net, ...). La plus utilisée est l'implémentation **CPython** écrite en C et plus communément appelé **Python**. C'est celle qu'il est impératif d'utiliser pour débiter avec Python, les autres n'étant utiles que si on souhaite travailler également avec Java (**Jython**) ou .net (**IronPython**) par exemple. => [FAQ](#) **Quelle implémentation de Python choisir ?**

Vous trouverez le lien de téléchargement sur le site officiel suivant:  **Téléchargement de CPython**. Il est conseillé de sélectionner la dernière version stable (à ce jour, la 2.6.1). Pour Windows, il existe une version précompilée (selon l'architecture de votre ordinateur) avec un installateur intégré qui nous affranchira d'effectuer cette opération lourde de compilation des sources nécessitant un compilateur C. Cette version précompilée a été elle-même compilée avec une version de Visual Studio.

 *La version 3.0 de CPython est disponible depuis fin 2008. Cette version, qui apporte de nouvelles améliorations à Python, présente cependant certaines incompatibilités avec la version 2 et les ressources existantes (cours, tutoriels, FAQ, sources, ...) n'ayant pas encore évoluées pour se conformer avec la version 3.0, il est conseillé pour le moment d'effectuer son apprentissage de Python avec la version 2.*

II-B - Installation avec l'installateur de Microsoft Windows

Il n'y a rien de plus simple que d'installer Python à partir de la version **Windows installer** (.msi). Vous avez le choix entre plusieurs fichiers d'installation à télécharger dépendant de votre version de windows (32 bits (...msi) ou 64 bits (...64.msi)) et de la version de Python sélectionnée. De préférence, choisissez une version stable (non alpha) qui aura l'avantage de ne plus présenter beaucoup de bogues et la majorité des bibliothèques externes que vous pourrez ensuite installer seront en général compatibles avec cette version.

Il suffit d'exécuter le fichier et de suivre les instructions. Il est conseillé de conserver le répertoire par défaut de l'installation, en général **c:\python26**. En gros, cela consiste donc à toujours cliquer sur le bouton **Next** Dans la suite, nous supposerons que l'installation de python a été réalisée dans le répertoire **c:\python26**

III - Interpréteur interactif

III-A - Introduction

L'interpréteur interactif permet d'écrire et d'exécuter du code Python à la volée, de faire des tests rapides, d'obtenir facilement des informations sur une fonction ou un module, ... Il est donc toujours utile d'en avoir un sous la main. La distribution standard de Python en propose 2:

Interpréteur interactif

- Shell interactif: console Python la plus basique
- IDLE: console écrite avec le toolkit Tkinter

III-B - Shell interactif / console Python

Pour lancer le Shell interactif (ou console Python), il suffit d'exécuter le fichier **python.exe** se trouvant dans le répertoire **c:/python26/**

Vous pouvez aussi l'exécuter par le menu en bas à gauche de votre écran **Démarrer -> Tous les programmes -> Python 2.6 -> Python (command line)** (termes dépendant de la version de votre Windows).

```
Python (command line)
Python 2.5.1 <r251:54863, Apr 18 2007, 08:51:08> [MSC v.1310 32 b
win32
Type "help", "copyright", "credits" or "license" for more informat
>>> _
```

le signe **>>>** est le prompt de l'interpréteur Python. Vous pouvez alors taper du code que Python interprétera lorsque vous le validerez par la touche **Entrée**. Si votre instruction n'est pas complète, Python n'exécutera pas le code immédiatement et passera à la ligne suivante en affichant un prompt de signe Une fois l'instruction terminée et validée, vous retrouverez alors le prompt **>>>** qui vous permettra d'écrire une nouvelle instruction.

```
Python (command line)
Python 2.5.1 <r251:54863, Apr 18 2007, 08:51:08> [MSC v.1310 32 b
win32
Type "help", "copyright", "credits" or "license" for more informat
>>> print 5
5
>>> def somme(a, b): return a + b
--
>>> somme(5, 11)
16
>>> _
```

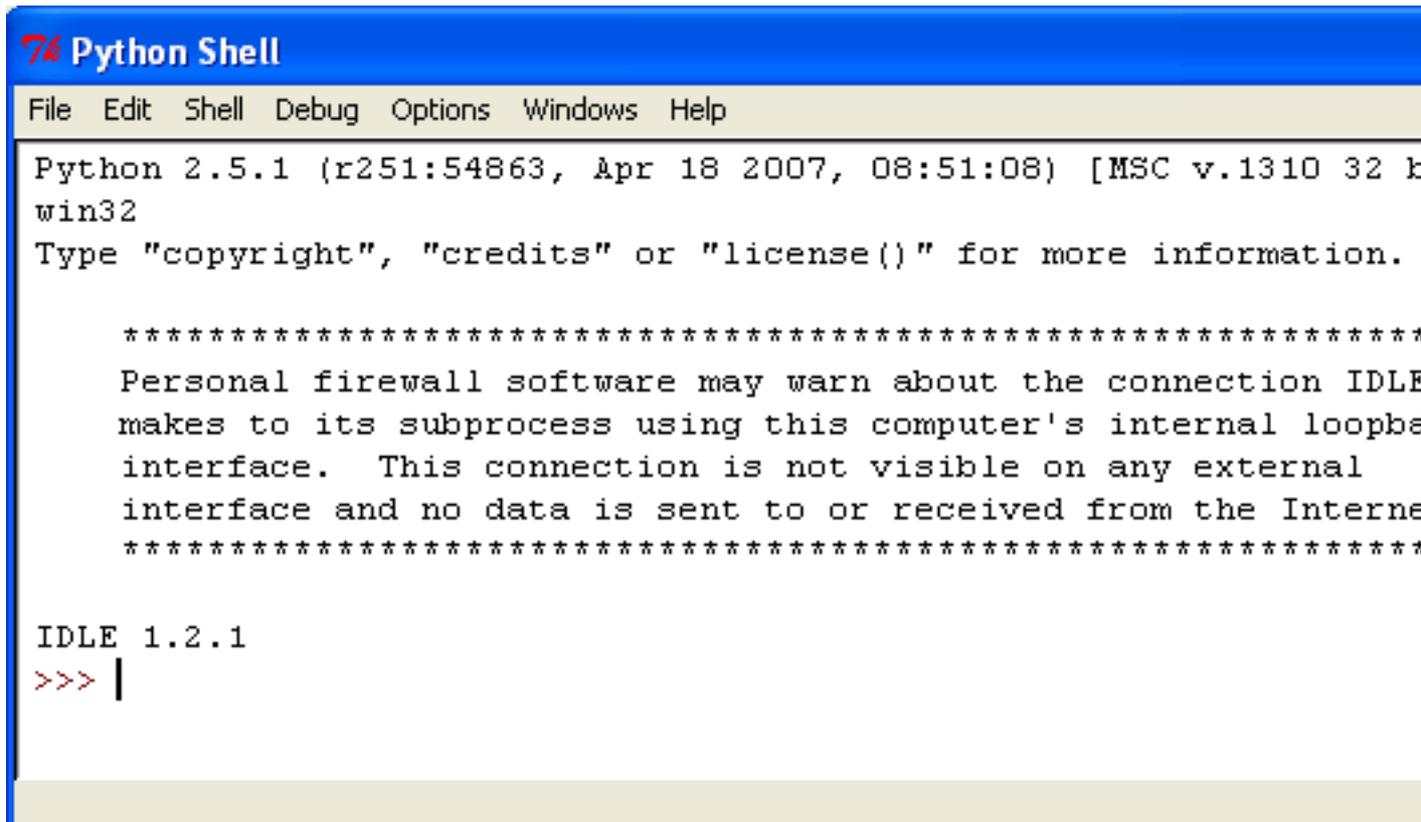
III-C - IDLE

Vous pouvez faire exactement la même chose avec le Shell de **IDLE**.

le Shell de **IDLE** a été écrit en python avec le toolkit graphique Tkinter. Il vous faudra donc exécuter un fichier écrit en Python.

Pour lancer **IDLE**, il vous faudra donc exécuter le fichier **c:/python26/Lib/idlelib/idle.pyw** par l'interpréteur Python. Pour cela, à l'installation de Python, l'installateur a normalement automatiquement associé les fichiers d'extension **.py** à l'interpréteur **c:/python26/python.exe** et les fichiers d'extension **.pyw** à l'interpréteur **c:/python26/pythonw.exe**. Il n'y a aucune différence entre ces 2 interpréteurs excepté que l'interpréteur **pythonw.exe** n'affiche pas la console Python (qui n'est pas utile si on lance une application graphique comme **IDLE**). Un double-clic sur le fichier **c:/python26/Lib/idlelib/idle.pyw** suffira donc pour lancer **IDLE**.

Vous pouvez aussi l'exécuter par le menu en bas à gauche de votre écran **Démarrer -> Tous les programmes -> Python 2.6 -> IDLE (Python GUI)** (termes dépendant de la version de votre Windows).



Vous pouvez écrire et exécuter de la même façon que précédemment des instructions Python. L'avantage est que l'interface est un peu plus élaboré avec quelques fonctionnalités intéressantes comme la coloration syntaxique, les call-tips ou encore l'autocomplétion (avec la touche Tabulation)

76 *Python Shell*

File Edit Shell Debug Options Windows Help

```
Python 2.5.1 (r251:54863, Apr 18 2007, 08:51:08) [MSC v.1310 32 b
win32
```

```
Type "copyright", "credits" or "license()" for more information.
```

```
*****
Personal firewall software may warn about the connection IDLE
makes to its subprocess using this computer's internal loopba
interface. This connection is not visible on any external
interface and no data is sent to or received from the Interne
*****
```

```
IDLE 1.2.1
```

```
>>> a = 5
```

```
>>> if a > 2:
```

```
    print True
```

```
else:
```

```
    print False
```

```
True
```

```
>>> import os
```

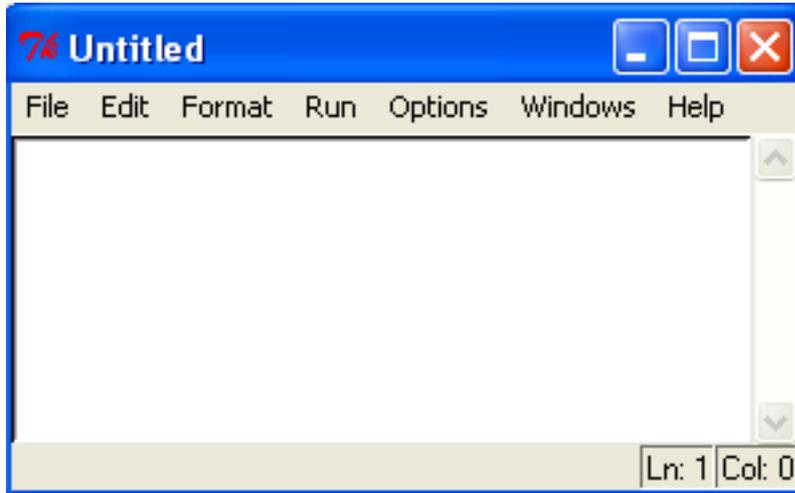
```
>>> os.open(
```

```
open(filename, flag [, mode=0777]) -> fd
```

IV - Ecriture d'un script Python

IV-A - sous IDLE

IDLE en plus d'être un Shell interactif est aussi un éditeur spécialement conçu pour des programmes Python. Pour ouvrir IDLE en mode éditeur, vous pouvez, à partir du mode interactif, aller dans le menu **File - New Window**. S'ouvrira alors l'éditeur de IDLE.



Vous pouvez alors écrire un programme Python et l'enregistrer dans un fichier d'extension **.py**. Une autre possibilité pour accéder à l'éditeur de **IDLE** est tout d'abord de créer à la main un fichier texte (par exemple avec le **bouton droit de la souris -> Nouveau -> Document texte** que vous renommez en un fichier d'extension **.py**. Vous pouvez ensuite ouvrir ce fichier avec **IDLE** en cliquant dessus avec le **bouton droit de la souris -> Edit With IDLE**. En fonction de la configuration de **IDLE**, il se peut que cela ouvre en même temps le shell interactif de **IDLE**.

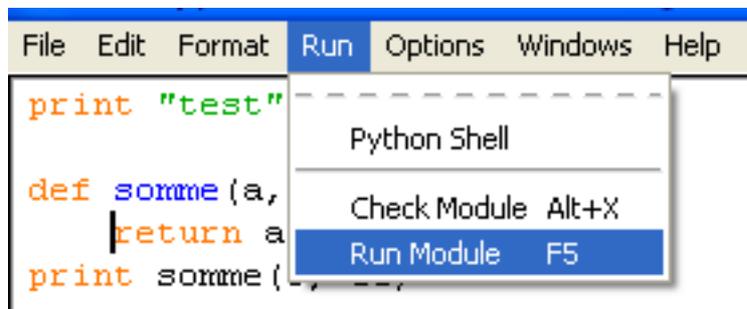
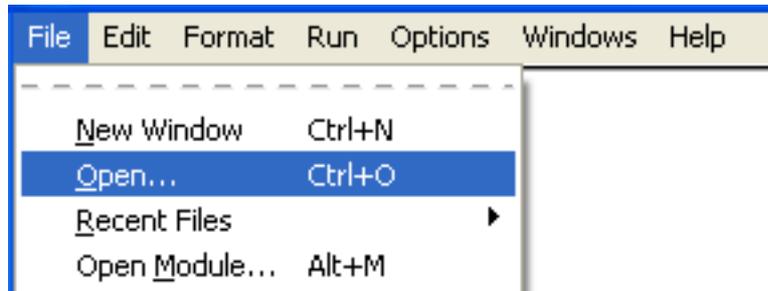
IV-B - Les autres Editeurs

IDLE est l'éditeur standard fourni avec la distribution de Python. Pour débiter, il est largement suffisant et propose les fonctionnalités principales que l'on attend d'un éditeur de programmes. Il existe cependant des éditeurs beaucoup plus évolués que vous pouvez trouver sur notre page [Editeurs pour Python](#). Un comparatif des fonctionnalités des différents éditeurs devraient aussi bientôt voir le jour.

V - Exécution d'un script Python

V-A - Sous IDLE

Sous **IDLE**, l'exécution d'un script Python est très simple. Il vous faudra tout d'abord charger dans l'éditeur votre script s'il n'y est pas déjà présent par le menu **File -> Open**. Ensuite pour l'exécuter, il suffit d'appuyer sur la touche **F5** (ou bien aller dans le menu **Run -> Run Module**). L'exécution se fera dans la fenêtre interactif de **IDLE**.



IDLE peut être utilisé pour exécuter vos programmes mais n'en abusez pas. Je le déconseille si vous devez lancer des programmes assez lourd ou contenant une interface graphique, **IDLE** étant programmé en **Tkinter**, il peut arriver qu'il y est des interférences entre votre programme et **Tkinter**. Il y a ainsi parfois des bogues qui surviennent à cause de l'exécution d'un script sous **IDLE**.

En ce qui me concerne, je ne me sers de **IDLE** uniquement pour écrire mes programmes et du mode interactif pour faire des tests ou obtenir de l'aide sur des fonction/modules/classes/... . Dès qu'il s'agit d'exécuter mes programmes, je passe directement par le mode console.

V-B - En mode console

Comme nous l'avons entreaperçu au paragraphe **III-C**, pour exécuter un programme Python, il faut passer par l'interpréteur `c:/python26/python.exe` ou `c:/python26/pythonw.exe`.

Si votre fichier `.py` est associé à `python.exe`, un double-clic sur le fichier lancera le programme. En revanche, la console Python se refermera automatiquement à la fin du programme si aucune instruction ne bloque l'exécution ou si une erreur survient à un moment de l'exécution du code, ce qui est tout de même assez gênant notamment lorsqu'on débogue.

Vous avez donc plusieurs possibilités. Si votre programme ne présente pas de bogue, vous pouvez ajouter à la fin de votre programme l'instruction `raw_input("Appuyer sur Entrée pour quitter")`, l'instruction `raw_input([prompt])` bloquant le programme jusqu'au moment où l'utilisateur appuie sur Entrée.

Vous pouvez également (et c'est ce que je fais toujours) créer un fichier d'extension `.bat` (il suffit de créer un fichier texte que vous renommez avec une extension `.bat`). dans le même répertoire que votre programme. Dans ce fichier `.bat`, vous écrivez simplement si votre programme s'appelle `monprogramme.py` :

```
monprogramme.py
pause
```

ou

```
c:/python26/python.exe monprogramme.py  
pause
```

La première ligne va permettre d'exécuter votre programme et l'instruction pause attend que vous appuyez sur une touche pour fermer la fenêtre. Un double-clic sur ce fichier **.bat** lancera alors votre programme.

Vérifiez-bien aussi que dans vos options des dossiers, l'extension des fichiers connus ne soit pas masquée; sinon, votre fichier **.bat** restera un fichier **.txt** et le double-clic ouvrira alors votre fichier en mode texte plutôt que de l'exécuter (sous **XP**, **Outils -> Options des dossiers -> Affichage -> Masquer les extension des fichiers dont le type est connu**. sous **Vista**, il faut passer par le **Panneau de Configuration -> Options des dossiers -> ...**)

VI - Pour aller plus loin

Vous êtes maintenant à même de commencer votre apprentissage de Python.

Vous trouverez sur le site <http://python.developpez.com/> toutes les ressources nécessaires pour maîtriser le langage. Commencez notamment par un cours pour apprendre le langage : **Apprentissage de Python**

N'oubliez pas de consulter les **FAQ Python** qui vous permettra d'obtenir une précieuse aide sur les erreurs habituellement rencontrées et les **Source Sources Python** qui vous donneront des exemples de programmes Python (qui peuvent être aussi bien être très simples que très complets).

Il est important aussi d'avoir toujours un oeil sur la  **Python Library Reference**

Et n'hésitez pas à poser toutes vos question sur le **forum Python** et ses sous-forums.