

Programmation Python 1 : Prise en main de l'environnement

Prise en main de Python

- Introduction à Python
- Installer l'environnement
- Lancer un programme
- Utiliser l'interpréteur
- Premières instructions

Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

Le langage Python

- Langage multi-plateformes
- Programmation impérative et orientée objet
- Typage dynamique
- Gestion de la mémoire par ramasse-miettes
- License libre (type BSD)
- Très utilisé dans l'industrie et le monde académique
- Site <http://www.python.org>

Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

Installation clé en main

- [Anaconda CE](#) (recommandé)
- [Canopy par Enthought](#). Commercial mais gratuit pour une utilisation académique.

Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

Lancer un programme Python

- Un fichier python termine par ".py":

```
mon_programme.py
```

- Toutes les lignes d'un fichier Python sont exécutées sauf les lignes qui commencent par:

```
# qui sont des commentaires
```

- Pour lancer le programme depuis une ligne de commande ou un terminal:

```
$ python mon_programme.py
```

- Sous UNIX (Linux / Mac OS) il est courant d'ajouter le chemin vers l'interpréteur python sur la première ligne du fichier:

```
#!/usr/bin/env python
```

Cela permet de lancer un programme directement:

```
$ mon_programme.py
```

Prise en main de Python

Introduction à Python
Installer l'environnement
Lancer un programme
Utiliser l'interpréteur
Premières instructions

Exemple

Remarque: Commencer une ligne par ! dans Ipython permet de lancer une commande UNIX.

```
In [1]: !ls mon_programme.py
```

```
mon_programme.py
```

```
In [2]: !cat mon_programme.py
```

```
#!/usr/bin/env python
```

```
print("Bonjour tout le monde!")
```

```
In [3]: !python mon_programme.py
```

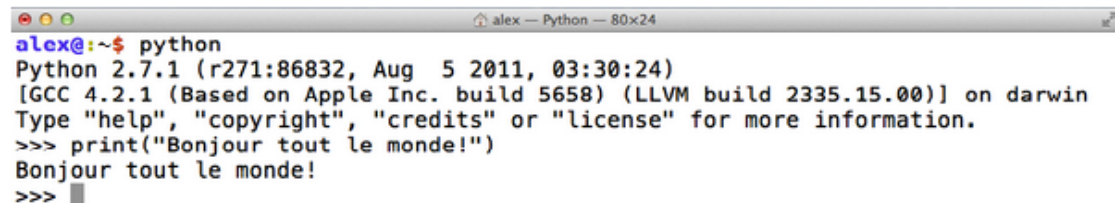
```
Bonjour tout le monde!
```

```
In [4]: !./mon_programme.py # sans "python" devant
```

```
Bonjour tout le monde!
```

L'interpréteur Python (mode interactif)

L'interpréteur Python se lance avec la commande `python`.



```
alex@~$ python
Python 2.7.1 (r271:86832, Aug  5 2011, 03:30:24)
[GCC 4.2.1 (Based on Apple Inc. build 5658) (LLVM build 2335.15.00)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Bonjour tout le monde!")
Bonjour tout le monde!
>>> █
```

Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

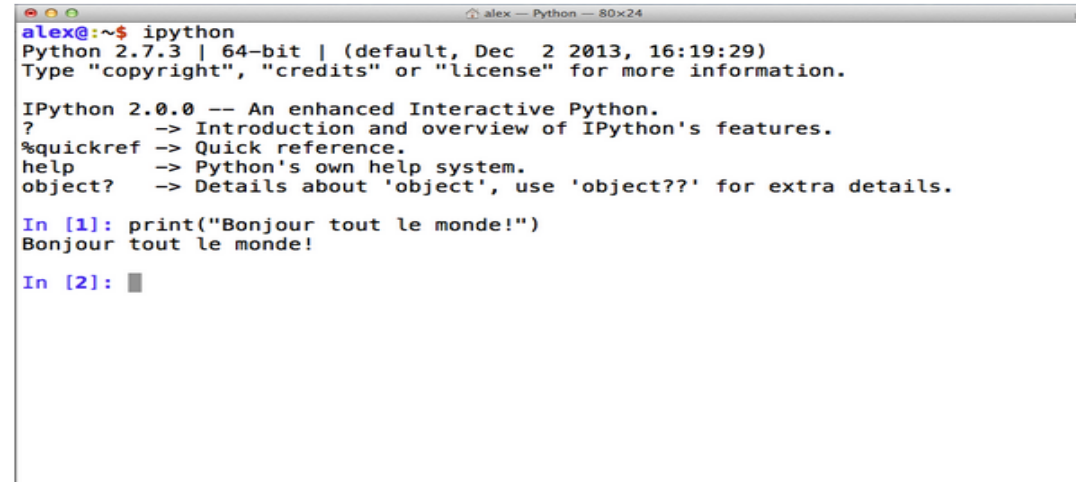
Utiliser l'interpréteur

Premières instructions

Pour sortir taper `exit()` ou Ctrl+D

IPython (<http://ipython.org>)

IPython est un shell interactif beaucoup plus avancé.

A screenshot of a terminal window titled 'alex - Python - 80x24'. The prompt is 'alex@:~\$'. The user has entered 'ipython'. The output shows 'Python 2.7.3 | 64-bit | (default, Dec 2 2013, 16:19:29)' and 'Type "copyright", "credits" or "license" for more information.' Below this is the IPython version '2.0.0' and a list of help options: '?', '%quickref', 'help', and 'object?'. The user then enters 'In [1]: print("Bonjour tout le monde!")' and the output is 'Bonjour tout le monde!'. The next prompt is 'In [2]:' followed by a cursor.

```
alex@:~$ ipython
Python 2.7.3 | 64-bit | (default, Dec 2 2013, 16:19:29)
Type "copyright", "credits" or "license" for more information.

IPython 2.0.0 -- An enhanced Interactive Python.
?      -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help    -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.

In [1]: print("Bonjour tout le monde!")
Bonjour tout le monde!

In [2]: █
```

Il permet notamment de:

- mémoriser les commandes lancées précédemment avec les flèches (haut et bas).
- auto-complétion avec la touche Tab.
- accès simple à la doc
- débogage

Prise en main de Python

Introduction à Python

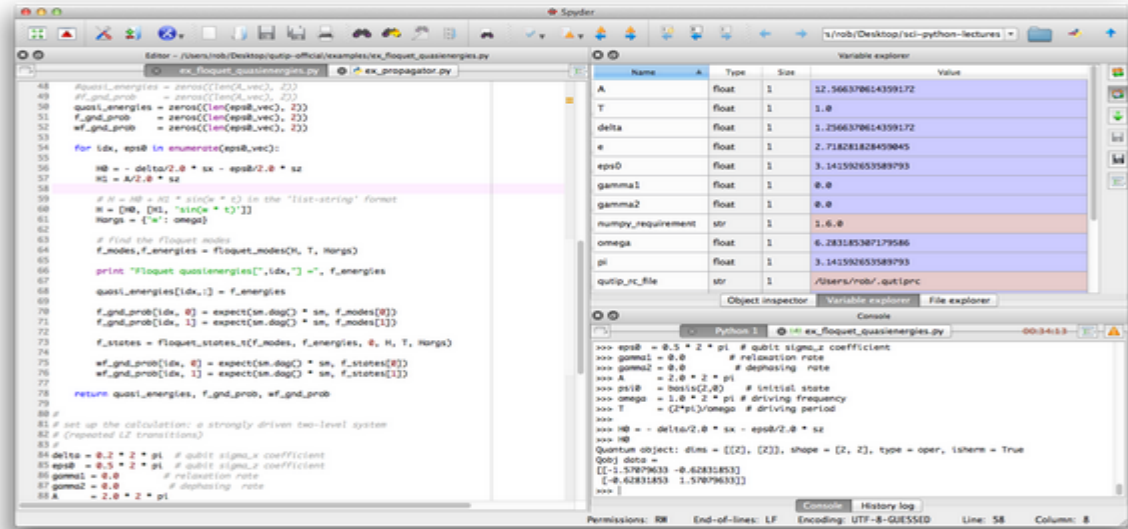
Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

Spyder est un IDE similaire à MATLAB.



Les avantages de Spyder:

- Bon éditeur (couleurs, intégré avec le debugger).
- Explorateur de variables, intégration de IPython
- Documentation intégrée.

Prise en main de Python

Introduction à Python

Installer l'environnement

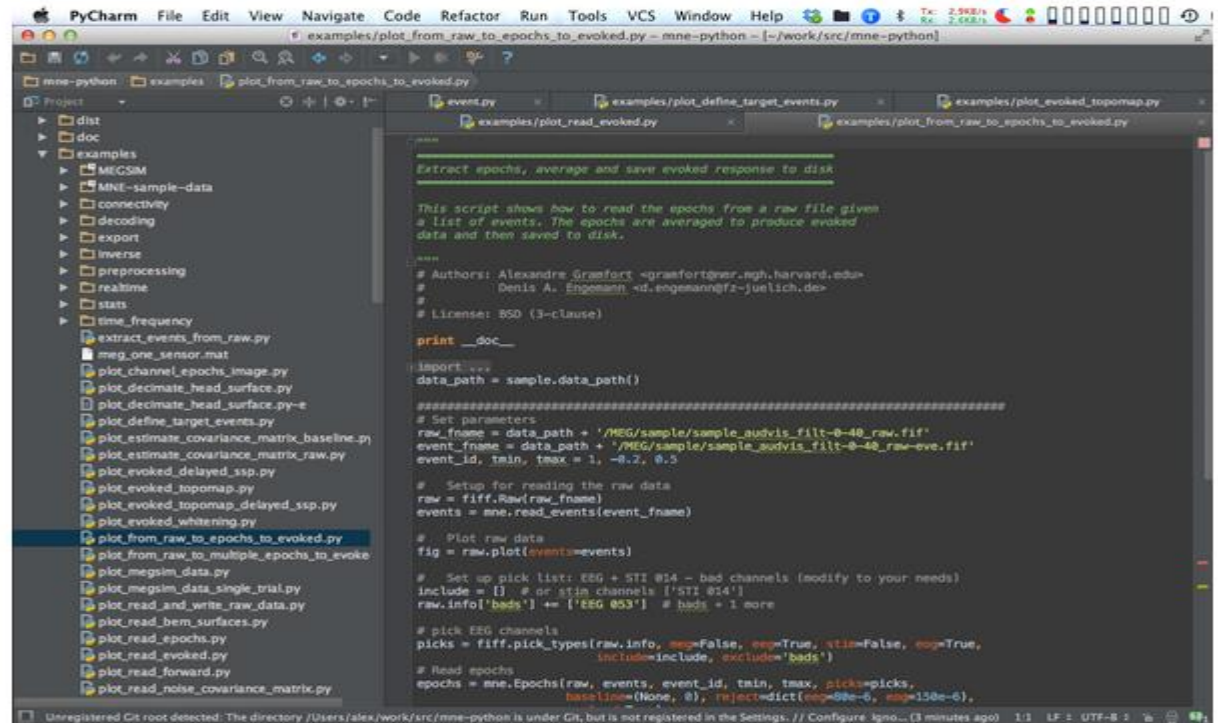
Lancer un programme

Utiliser l'interpréteur

Premières instructions

PyCharm

PyCharm est un IDE complet.



La version community edition est gratuite.

Prise en main de Python

Introduction à Python

Installer l'environnement

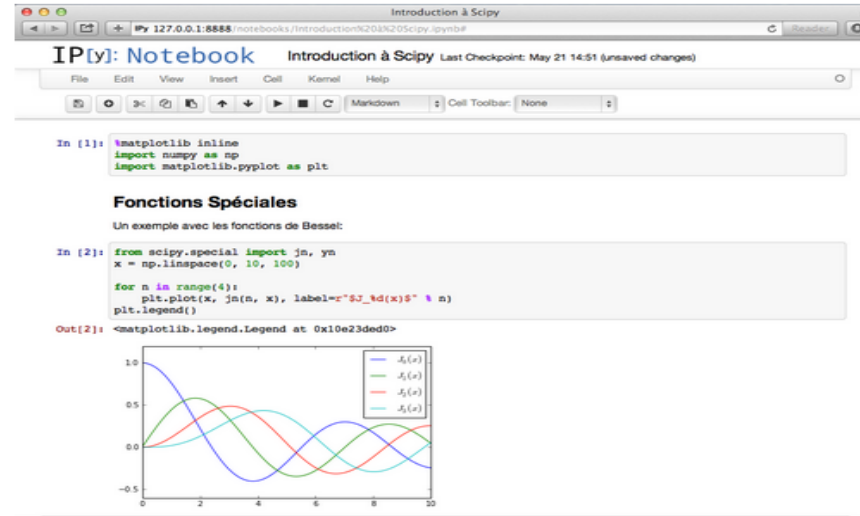
Lancer un programme

Utiliser l'interpréteur

Premières instructions

IPython notebook

IPython notebook comme Mathematica ou Maple dans une page web.



Pour lancer ipython-notebook:

```
$ ipython notebook
```

depuis un dossier où seront stockés les notebooks (fichiers *.ipynb)

Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

Les nombres

```
In [5]: 2 + 1 # ceci est un commentaire
```

```
Out[5]: 3
```

```
In [6]: 3 * 4
```

```
Out[6]: 12
```

```
In [7]: 2 ** 3 # 2 à la puissance 3
```

```
Out[7]: 8
```

Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

Assignement de variable et type

```
In [8]: a = 4  
print(a)  
print(type(a))
```

```
4  
<type 'int'>
```

"a" est du type entier **int** (integer en Anglais)

Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

Les mots clés du langage

Les noms de variable peuvent contenir a–z, A–Z, 0–9 et quelques caractères spéciaux tels que `_` mais commencent toujours par une lettre.

Par convention les noms de variables sont en minuscule.

Quelques noms de variable ne sont pas autorisés car déjà utilisés par le langage:

```
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
```

```
In [9]: int a = 1;  # en C
```

```
File "<ipython-input-9-850b11336ead>", line 1
```

```
    int a = 1;  # en C
        ^
```

```
SyntaxError: invalid syntax
```

Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

Les nombres (suite)

```
In [10]: c = 2.1  # nombre flottant  
print type(c)
```

```
<type 'float'>
```

```
In [11]: a = 1.5 + 1j  # nombre complexe  
print(a.real)  
print(a.imag)  
print(type(a))
```

```
1.5
```

```
1.0
```

```
<type 'complex'>
```

Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

```
In [12]: 3 < 4  # booléens True et False (attention à la majuscule)
```

```
Out[12]: True
```

```
In [13]: 3 < 2
```

```
Out[13]: False
```

```
In [14]: test = (3 > 4)
          print(test)
          print(type(test))
```

```
False
<type 'bool'>
```

Prise en main de Python

Introduction à Python
Installer l'environnement
Lancer un programme
Utiliser l'interpréteur
Premières instructions

Les nombres et changement de type

```
In [15]: print(7 * 3.)  # int x float -> float
          print(type(7 * 3.))

          21.0
          <type 'float'>
```

Attention !

```
In [16]: 3 / 2  # int x int -> int (vrai en Python 2)
```

```
Out[16]: 1
```

```
In [17]: 3 / 2.  # OK
```

```
Out[17]: 1.5
```

```
In [18]: a = 2
          3 / float(a)  # OK
```

```
Out[18]: 1.5
```

```
In [19]: 3 // 2  # division entière
```

```
Out[19]: 1
```


Prise en main de Python

Introduction à Python

Installer l'environnement

Lancer un programme

Utiliser l'interpréteur

Premières instructions

Mais

In [20]:

```
cos(2)
```

```
-----  
-----  
NameError
```

Traceback (most recent call last)

Traceback (most recent call last)

<ipython-input-20-43abd96808db> in <module>()

----> 1 cos(2)

NameError: name 'cos' is not defined