

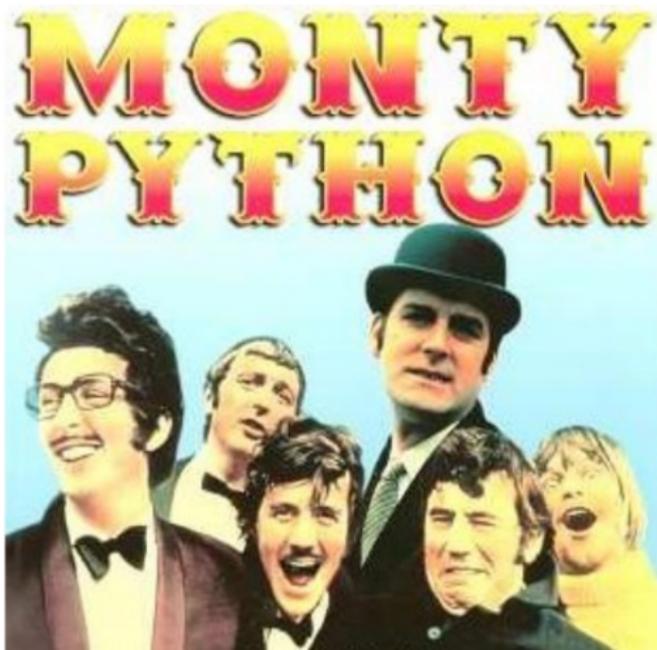
Introduction à Python

Eric BERTHOMIER

`eric.berthomier@free.fr`

1^{er} janvier 2016





Préférer le beau au laid,
l'explicite à l'implicite,
le simple au complexe,

le complexe au compliqué,
le déroulé à l'imbriqué,
l'aéré au compact.



Préférer le beau au laid,
l'explicite à l'implicite,
le simple au complexe,

le complexe au compliqué,
le déroulé à l'imbriqué,
l'aéré au compact.

La lisibilité compte.



Préférer le beau au laid,
l'explicite à l'implicite,
le simple au complexe,

le complexe au compliqué,
le déroulé à l'imbriqué,
l'aéré au compact.

La lisibilité compte.

<http://legacy.python.org/dev/peps/pep-0020/>



- Les cas particuliers ne le sont jamais assez pour violer les règles, même s'il faut privilégier l'aspect pratique à la pureté.



- Les cas particuliers ne le sont jamais assez pour violer les règles, même s'il faut privilégier l'aspect pratique à la pureté.
- Ne jamais passer les erreurs sous silence, ou les faire taire explicitement.



- Les cas particuliers ne le sont jamais assez pour violer les règles, même s'il faut privilégier l'aspect pratique à la pureté.
- Ne jamais passer les erreurs sous silence, ou les faire taire explicitement.
- Face à l'ambiguïté, ne pas se laisser tenter à deviner.



- Les cas particuliers ne le sont jamais assez pour violer les règles, même s'il faut privilégier l'aspect pratique à la pureté.
- Ne jamais passer les erreurs sous silence, ou les faire taire explicitement.
- Face à l'ambiguïté, ne pas se laisser tenter à deviner.
- Si l'implémentation s'explique difficilement, c'est une mauvaise idée.



- Les cas particuliers ne le sont jamais assez pour violer les règles, même s'il faut privilégier l'aspect pratique à la pureté.
- Ne jamais passer les erreurs sous silence, ou les faire taire explicitement.
- Face à l'ambiguïté, ne pas se laisser tenter à deviner.
- Si l'implémentation s'explique difficilement, c'est une mauvaise idée.
- Si l'implémentation s'explique facilement, c'est peut-être une bonne idée.



- Les cas particuliers ne le sont jamais assez pour violer les règles, même s'il faut privilégier l'aspect pratique à la pureté.
- Ne jamais passer les erreurs sous silence, ou les faire taire explicitement.
- Face à l'ambiguïté, ne pas se laisser tenter à deviner.
- Si l'implémentation s'explique difficilement, c'est une mauvaise idée.
- Si l'implémentation s'explique facilement, c'est peut-être une bonne idée.

<http://legacy.python.org/dev/peps/pep-0020/>



Duck Typing

*When I see a bird that walks like a duck
and swims like a duck
and quacks like a duck,*



Duck Typing

*When I see a bird that walks like a duck
and swims like a duck
and quacks like a duck,
I call that bird a duck.*

https://en.wikipedia.org/wiki/Duck_typing



Documentation en console

```
Python 3.4.1 Shell
File Edit Shell Debug Options Windows Help
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> help (print)
Help on built-in function print in module builtins:

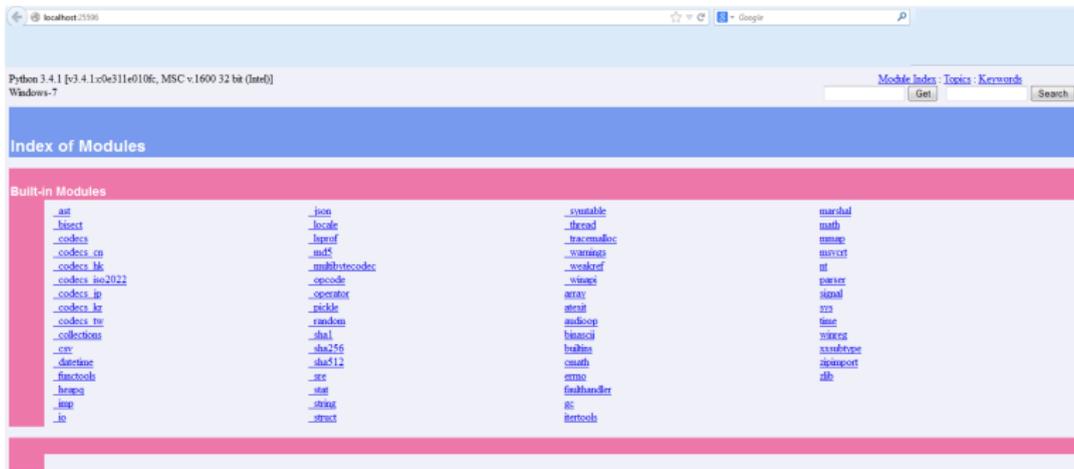
print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file: a file-like object (stream); defaults to the current sys.stdout.
    sep:  string inserted between values, default a space.
    end:  string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.

>>>
```



Documentation en local, interface web



Python 3.4.1 [v3.4.1.rc6311e010f6; MSC v.1600 32 bit (Intel)]
Windows-7

Module Index - Topics - Keywords

Get Search

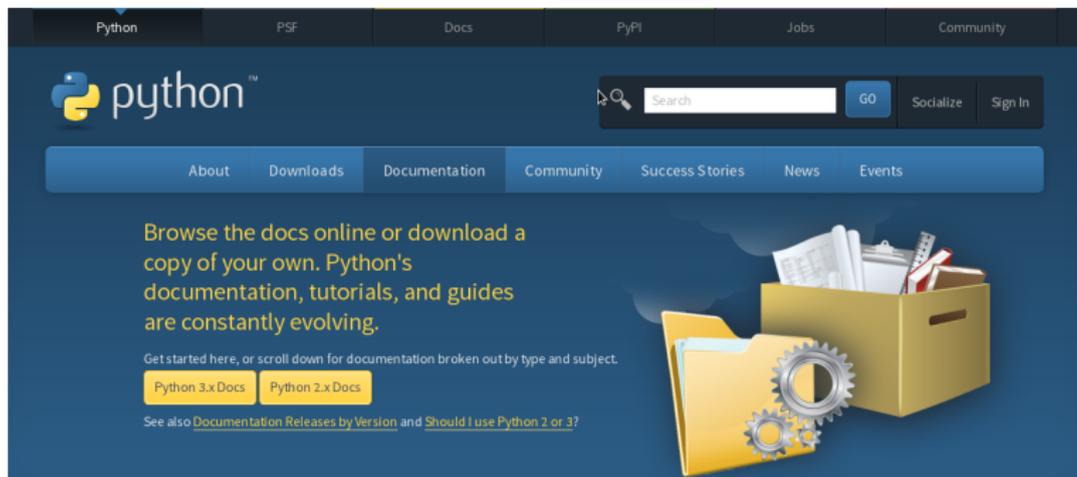
Index of Modules

Built-in Modules

| | | | |
|---------------------------------|---------------------------------|------------------------------|--------------------------|
| _ast | _json | _syntable | marshal |
| _bisect | _locale | _thread | math |
| _codecs | _lxml | _tracemalloc | mmap |
| _codecs_cn | _md5 | _warnings | mover |
| _codecs_iso2022 | _multibytecodec | _weakref | msilib |
| _codecs_jp | _opcode | _winreg | mutex |
| _codecs_kr | _operator | array | signal |
| _codecs_tw | _pickle | array | sys |
| _collections | _random | audioclip | time |
| _cmr | _sha1 | binascii | unittest |
| _datetime | _sha256 | binhex | urllib |
| _functools | _sha3 | cmath | urllib2 |
| _heapq | _sre | errno | urlparse |
| _imp | _stat | fcntl | zipfile |
| _io | _string | gc | |
| | _struct | _socket | |



Documentation sur la toile



The screenshot shows the Python.org website with a dark blue header and navigation bar. The Python logo is on the left, and a search bar with a 'GO' button is on the right. Below the navigation bar, there's a main content area with the text: "Browse the docs online or download a copy of your own. Python's documentation, tutorials, and guides are constantly evolving." Below this text are two buttons: "Python 3.x Docs" and "Python 2.x Docs". To the right of the text is an illustration of a yellow folder and a box containing papers and gears. At the bottom of the screenshot, there are navigation icons for back, forward, and search.

Python PSF Docs PyPI Jobs Community

python™

Search GO Socialize Sign In

About Downloads Documentation Community Success Stories News Events

Browse the docs online or download a copy of your own. Python's documentation, tutorials, and guides are constantly evolving.

Get started here, or scroll down for documentation broken out by type and subject.

Python 3.x Docs Python 2.x Docs

See also [Documentation Releases by Version](#) and [Should I use Python 2 or 3?](#)

<https://www.python.org/doc/>



- Using the `__future__` module
- The `print` function
- Integer division
- Unicode
- `xrange`
- Raising exceptions
- Handling exceptions
- The `next()` function and `.next()` method
- For-loop variables and the global namespace leak
- Comparing unorderable types
- Parsing user inputs via `input()`
- Returning iterable objects instead of lists

<https://docs.python.org/2/library/2to3.html>

http://nbviewer.ipython.org/github/rasbt/python_reference/blob/master/tutorials/key_differences_between_python_2_and_3.ipynb



Windows

windows.py

```
#!c:\python34\python  
# -*- coding: UTF-8 -*-
```

Linux

linux.py

```
#!/usr/bin/python3  
# -*- coding: UTF-8 -*-
```



EnteteProjet.py

```
__author__ = "Rob Knight, Gavin Huttley, and Peter Maxwell"  
__copyright__ = "Copyright 2007, The Cogent Project"  
__credits__ = ["Rob Knight", "Peter Maxwell", "Gavin Huttley",  
              "Matthew Wakefield"]  
__license__ = "GPL"  
__version__ = "1.0.1"  
__maintainer__ = "Rob Knight"  
__email__ = "rob@spot.colorado.edu"  
__status__ = "Production"
```

