

Conventions de style en python

M2BI Python avancé

Jonathan Barnoud

15 septembre 2011



Introduction

Pourquoi ?

Améliorer la lisibilité

Convention ?

Cohérence

module > projet > équipe > langage

Document de référence

PEP 8

Python Enhancement Proposal

Plan

Introduction

Organisation du code

Commentaires

Noms de variables

Références

Organisation du code

Indentation

Espaces ou tabulations

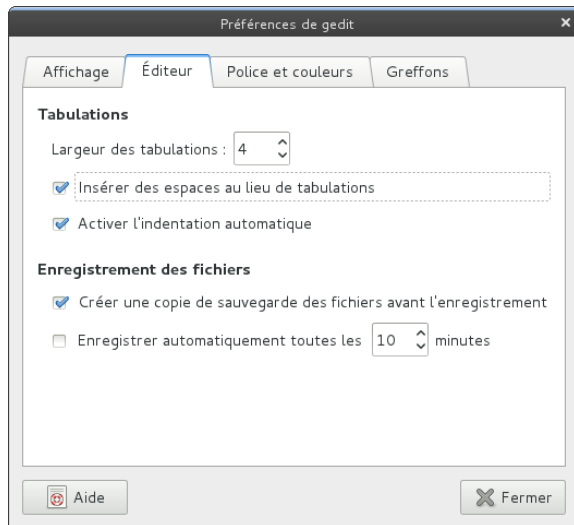
Pas de mélange

Indentation

4 spaces

Indentation

Configurer l'éditeur



Taille d'une ligne

Pas de lignes trop longues

Taille d'une ligne

79 caractères

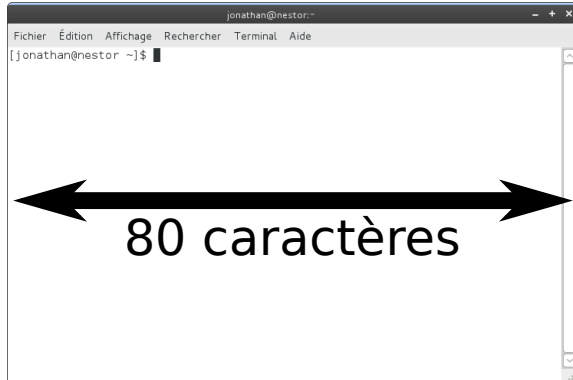
Taille d'une ligne



Crédits : Mutatis mutandis, Creative commons by,

http://commons.wikimedia.org/wiki/File:Punched_card.jpg

Taille d'une ligne



Taille d'une ligne

```
morpion = [ ["X", "X", "O"],  
             ["O", "X", "X"],  
             ["X", "O", "O"] ]
```

```
atome = { "type" : " CA ",  
          "aa" : "Lys",  
          "residu" : 43,  
          "coordonnees" : (43.5, 67.8, 12.9),  
        }
```

Taille d'une ligne

```
if ( -87 < phi < -27  and  
    -77 < psi < -17 ) :  
    print "helix"
```

```
if -87 < phi < -27 and \  
    -77 < psi < -17 :  
    print "helix"
```


Sauts de lignes

- ▶ Constitue des blocs visuels dans le code
- ▶ **1** saut de ligne après une définition de fonction
- ▶ **2** sauts de ligne après une définition de classe

Commentaires

Commentaires

```
# Un commentaire sur une ligne
```

```
# Un commentaire un peu plus long  
# sur deux lignes
```

Docstrings

```
>>> def hello_world(name) :  
...     """  
...     Souhaite le bonjour à quelqu'un.  
...     """  
...     print "Hello %s" % name  
...  
>>> help(hello_world)  
Help on function hello_world in module __main__:  
  
hello_world(name)  
    Souhaite le bonjour à quelqu'un.
```

Docstrings

- Pour les classes

```
class MaClasse :  
    """  
    Une classe qui fait pleins de trucs.  
    """
```

- Pour les modules

```
#!/usr/bin/env python  
#-*- coding:utf-8 -*-  
  
"""  
Un super module !  
"""
```

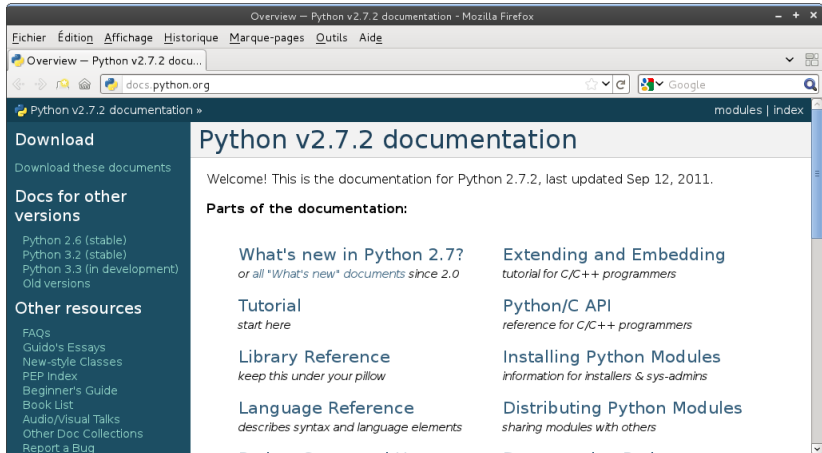
Docstrings

- ▶ Qu'est-ce que ça fait ?
- ▶ Pourquoi ça le fait ?
- ▶ D'où ça vient ?
- ▶ **Quels arguments ?**
- ▶ **Quels retours ?**
- ▶ Quels exceptions ?
- ▶ Comment ça s'utilise ?

Doctests

```
def somme(a, b) :  
    """  
    Calcul la somme de a et b.  
  
    >>> somme(2, 3)  
    5  
    """  
    return a + b  
  
import doctest  
doctest.testmod()
```

Sphinx



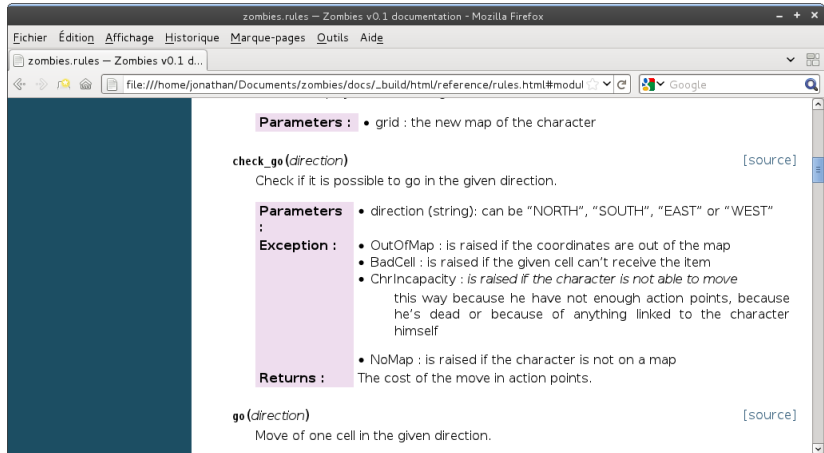

```
def check_go(self, direction) :
    """
    Check if it is possible to go in the given direction.

    :Parameters:
        - direction (string): can be "NORTH", "SOUTH", "EAST" or "WEST"

    :Exception:
        - OutOfMap : is raised if the coordinates are out of the map
        - BadCell : is raised if the given cell can't receive the item
        - ChrIncapacity : is raised if the character is not able to move
            this way because he have not enough action points, because
            he's dead or because of anything linked to the character himself
        - NoMap : is raised if the character is not on a map

    :Returns:
        The cost of the move in action points.
    """
```

Sphinx



Sphinx

`http://sphinx.pocoo.org/`

Noms de variables

Choix d'un nom de variable

Noms explicites

Conventions de nommage

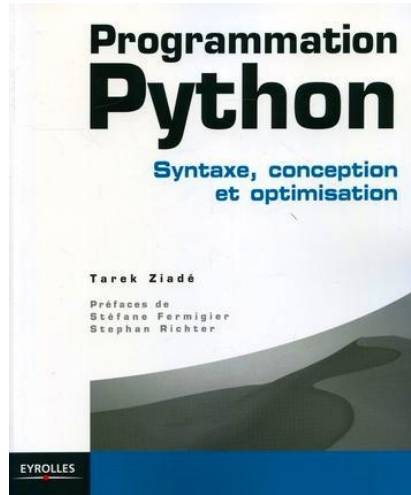
- ▶ variables et fonctions : `ma_variable`
- ▶ constantes : `MA_CONSTANTE`
- ▶ classes : `MaClasse`
- ▶ exeptions : `MyError`

Références

Pylint

```
pylint mon_script.py
```


Références



Références

PEP 8 <http://www.python.org/dev/peps/pep-0008/>

Bonnes pratiques <http://www.biologeeek.com/bonnes-pratiques,conferences,django,python,traduction/bonnes-pratiques-et-astuces-python/>

Sphinx <http://sphinx.pocoo.org/>

pylint <http://www.logilab.org/857>

Conclusion

`import this`



Ce document est mis à disposition selon les termes de la licence Creative Commons Attribution 3.0 non transposée

<http://creativecommons.org/licences/by/3.0/>