

Chapitre 6 : Modules

Yves Guidet pour IPSSI

V1.4.7 March 14, 2017

Les modules ne sont que des fichiers ; ils définissent des espaces de noms, notion que l'on verra avec les classes.

On peut importer tout ou partie d'un module, nous verrons cela avec un certain nombre de modules standard.

On peut les regrouper, à la manière de Java, en « paquetages », même si ce terme n'est pas usité dans le monde Python.

En savoir plus : <http://docs.python.org/2/tutorial/modules.html> (voir le lien dans le « chapeau ») ou la doc ENI, où c'est très bien expliqué.

Un script faisant un peu de math

Considérons le script suivant :

```
yves@bella:Python$ cat zgon.py
#!/usr/bin/env python

from math import sqrt

trinome = (1, 1, -6)

def pSol(a, b, c):
    delta = b*b - 4*a*c
    print 'delta = ', delta
    if delta >= 0:
        x1 = (-b + sqrt(delta))/2*a
        x2 = (-b - sqrt(delta))/2*a
        print 'x1 = ', x1
        print 'x2 = ', x2

pSol(trinome[0], trinome[1], trinome[2])
```

Un module ?

Peut-on importer zgon ?

```
yves@bella:Python$ python
```

```
Python 2.7.4 (default, Apr 19 2013, 18:28:01)
```

```
[GCC 4.7.3] on linux2
```

```
Type "help", "copyright", "credits" or "license" for more in
```

```
>>> import zgon
```

```
delta = 25
```

```
x1 = 2.0
```

```
x2 = -3.0
```

```
>>>
```

Un module ? un .pyc ?

On regarde notre répertoire, et surprise, un nouveau fichier :

```
yves@bella:Python$ ls -rtl
total 15
*rw-r--r-- 1 yves christian 288 mars 18 16:17 zgon.py
*rw-r--r-- 1 yves christian 1343 mars 18 16:24 moduleNonObjet.xhtml
*rw-r--r-- 1 yves christian 553 mars 18 16:38 zgon.pyc
```

Que contient donc ce "*zgon.pyc*" ? Il est créé la première fois qu'on procède à l'import.

```
yves@bella:classesPython$ file zgon.pyc
zgon.pyc: python 2.7 byte-compiled
```

Du *bytecode*, tout simplement.

Import du module dans un script

On tente d'importer *zgon* dans un script :

```
yves@bella:Python$ cat try.py  
#!/usr/bin/env python
```

```
import zgon
```

```
trinome = (1, -2, 1)
```

```
print 'try : ',
```

```
zgon.pSol(trinome[0], trinome[1], trinome[2])
```

Import : ça marche

Et ça marche !

```
yves@bella:Python$ ./try.py
delta = 25
x1 = 2.0
x2 = -3.0
try : delta = 0
x1 = 1.0
x2 = 1.0
```

On constate que le test à la fin du fichier *zgon.py* est exécuté, ce qui est pratique quand on teste le script, mais pas forcément lorsqu'on l'importe.

Import : la variable `__name__`

Que faire pour que ces tests ne soient exécutés que quand on utilise *zgon.py* comme script mais pas quand on importe le module ?

La variable `__name__` est là pour ça ! Elle vaut `__main__` quand on exécute *zgon.py* comme script et vaut en revanche *zgon* quand on importe le module dans un autre script. Ce qui nous permet d'écrire :

```
yves@bella:Python$ cat zgon.py
#!/usr/bin/env python
```

```
from math import sqrt
```

```
trinome = (1, 1, -6)
```

```
def pSol(a, b, c):
```

```
...
```

```
if __name__ == "__main__":
    pSol(trinome[0], trinome[1], trinome[2])
```


Et si on change de répertoire ?

Au niveau du *shell* on peut agir sur la variable `PYTHONPATH` :

```
export PYTHONPATH=/home/yves/.../répertoire_du_module
```

Au niveau de l'interprète *Python* on joue sur la variable `sys.path`.

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

```
from sys import path
```

```
path.append('/home/yves/.../répertoire_du_module')
```

```
import zgon
```

```
...
```

C'est tout bon.