

Introduction à Python

Septembre 2010

Plan

Table des matières

A	Découverte de l'environnement Linux	1
A.1	Le bureau	1
A.2	Arborescence	1
B	Premiers pas	1
B.1	Python en mode console	1
B.2	Python avec Eric4	1
C	Outils de base	2
C.1	Variables et écriture	2
C.2	Boucles et tests	2

B Premiers pas

B.1 Python en mode consol

Lignes de commande

A Découverte de l'environnement Linux

A.1 Le bureau

Ouverture d'une session

- ```
– login : aeinstein
– mot de passe : *****
```

## Le bureau

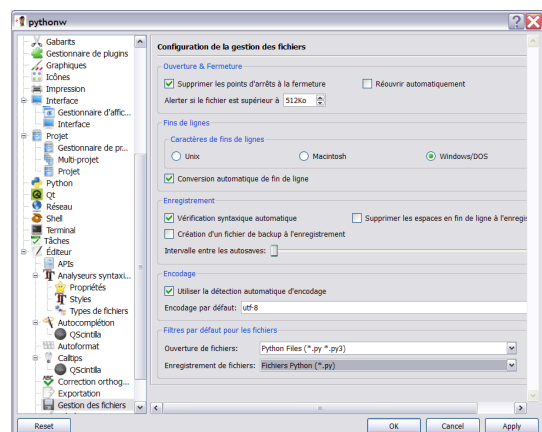
- Les applications
- Les raccourcis
- Le système
- Tableau de bord du bas
  - La barre de tâches
  - Les bureaux virtuels

## Attention à la division !

```
C:\Utilisateurs\Boisgard\Enseignement\
Python 2.6.2 (r262:71605, Apr 14 2009,
) on win32
Type "help", "copyright", "credits" or
>>> 2/3
0
>>> 2./3
0.6666666666666666
>>> █
```

## B.2 Python avec Eric4

## Configuration de l'IDE



## A.2 Arborescence

## Structure des fichiers

## Utilisation du Terminal

- `dir` ou `ls` : affiche le contenu d'un répertoire
- `cd` (pour `change directory`) :
  - `cd nom_du_repertoire_enfant` : descend d'un cran dans l'arborescence
  - `cd ..` : remonte vers le répertoire « père »
- `astuce` : taper les premières lettres puis « `tab` » pour compléter

## Premiers scripts

script1.py

```

1 """
2 Commentaire
3 """
4 g = 9.8
5 v0z = 10
6 t = 10
7
8 z = -0.5*g*t**2+v0z*t
9 print 'z = ', z
10

```

script2.py

```

1 #!/usr/bin/env python
2 #-*- coding: utf-8 -*-
3
4 Calcul de la position verticale d'un mobile lancé
5 avec une vitesse initiale v0z
6 """
7 g = 9.8 # accélération de la pesanteur
8 v0z = 10 # vitesse initiale
9 t = 10 # instant
10
11 z = -0.5*g*t**2+v0z*t
12 print 'z = ', z
13

```

A rajouter automatiquement

- Enregistrer votre script modèle dans « fichiers étiquetés »

## Les bibliothèques (1)

bib1.py

```

1 #!/usr/bin/env python
2 #-*- coding: utf-8 -*-
3
4 racine carrée 1
5 """
6 import math
7
8 a = math.sqrt(2)
9 print a
10

```

bib2.py

```

1 #!/usr/bin/env python
2 #-*- coding: utf-8 -*-
3
4 racine carrée 1
5 """
6 import math as m
7
8 a = m.sqrt(2)
9 print a
10

```

```

1 Python 2.6.2 (r262:71605, Apr 14 20
2 >>> 1.41421356237

```

```

1 Python 2.6.2 (r262:71605, Apr 14 20
2 >>> 1.41421356237

```

## Les bibliothèques (2)

bib1.py

```

1 #!/usr/bin/env python
2 #-*- coding: utf-8 -*-
3
4 racine carrée 1
5 """
6 from math import sqrt
7
8 a = sqrt(2)
9 print a
10

```

bib2.py

```

1 #!/usr/bin/env python
2 #-*- coding: utf-8 -*-
3
4 racine carrée 1
5 """
6 from math import *
7
8 a = sqrt(2)
9 print a
10

```

```

1 Python 2.6.2 (r262:71605, Apr 14 20
2 >>> 1.41421356237

```

```

1 Python 2.6.2 (r262:71605, Apr 14 20
2 >>> 1.41421356237

```

## Bibliothèque (3)

On utilisera essentiellement

- la bibliothèque scipy : `from scipy import *`  
contient toutes les fonctions mathématiques standards + sous-bibliothèques (FFT, solution équations,...)
- la bibliothèque matplotlib : `from pylab import *`  
permet de tracer des plots 2D (et aussi un peu de 3D)

## C Outils de base

### C.1 Variables et écriture

#### Variables

types

- numériques :
  - entier : `x = 3`
  - réel : `x = 3.14156`
  - complexe : `x = 5 + 4j`
- texte : `T = "toto"`
- ensembles :
  - liste : `L = [4,6,8]` → modifiable
  - tuplet : `t = (4,6,8)` → non modifiable
  - tableau : `T = array([1,2,3])` → type non standard (bibliothèque Numpy)

## Affectations et opérations (1)

op\_num.py

```

3 """
4 opérations
5 """
6 x = 2
7 y = 4
8 z = x + y
9 print z
10

```

op\_texte.py

```

3 """
4 opérations
5 """
6 T1 = 'toto'
7 T2 = 'titi'
8 T = T1 + ' ' + T2
9 print T
10

```

```

1 Python 2.6.2 (r262:
2 >>> 6

```

```

1 Python 2.6.2 (r262:71605,
2 >>> toto titi

```

## Affectations et opérations (2)

liste1.py

```

3 """
4 opérations sur les listes
5 """
6 L1 = [1, 2, 3]
7 L2 = [11, 12, 13]
8 L = L1 + L2
9 print L
10 print L[0], L[3]

```

```

1 Python 2.6.2 (r262:71605, A
2 >>> [1, 2, 3, 11, 12, 13]
3 1 11

```

array1.py

```

4 """
5 opérations sur les listes
6 """
7 from scipy import *
8 A1 = array([1, 2, 3])
9 A2 = array([11, 12, 13])
10 A = A1 + A2
11 print A

```

```

1 Python 2.6.2 (r262:71605, A
2 >>> [12 14 16]

```

## Listes et tableaux sur mesure

liste2.py

```

4 """
5 opérations sur les listes
6 """
7 L1 = range(10)
8 print L1
9 L2 = range(3, 12, 2)
10 print L2
11 L3 = range(10, 2, -1)
12 print L3

```

```

1 Python 2.6.2 (r262:71605, Apr 14
2 >>> [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
3 [3, 5, 7, 9, 11]
4 [10, 9, 8, 7, 6, 5, 4, 3]

```

array2.py

```

3 """
4 opérations sur les listes
5 """
6 from scipy import *
7 A1 = arange(2, 5, 0.3)
8 print A1
9 A2 = linspace(0, 10, 7)
10 print A2

```

```

1 Python 2.6.2 (r262:71605, Apr 14 2009, 22:40:02) [MSC v.15
2 >>> [2. 2.3 2.6 2.9 3.2 3.5 3.8 4.1 4.4 4.7]
3 [0. 1.66666667 3.33333333 5. 6.66666667
4 8.33333333 10.]

```

## C.2 Boucles et tests

### Boucle for

```
boucle_for.py
2 #-*- coding: utf-8 -*-
3 """
4 opérations sur les listes
5 """
6 L = []
7 for n in range(10):
8 L = L + [2*n]
9 print L

Python 2.6.2 (r262:71605, Apr 14 2009, 22:
2 >>> [1, 2, 4, 8, 16, 32, 64, 128, 256, 512]
3
```

## If then else

```
test_if.py
4 tests
5 """
6 res1 = 2==3
7 print res1
8 res2 = 20==20
9 print res2
10 for n in range(5):
11 if n%2==0:
12 print n, ' est pair'
13 else:
14 print n, ' est impair'

2 >>> False
3 True
4 0 est pair
5 1 est impair
6 2 est pair
7 3 est impair
8 4 est pair
```