INSTITUT
de MATHEMATIQUES
de TOULOUSE

Python intermédiaire pour les scientifiques

Session 5/5 : Perspectives

Laurent Risser

Ingénieur de Recherche à l'Institut de Mathématiques de Toulouse

lrisser@math.univ-toulouse.fr

Résumé

En bref :

- Tour de table.
- Lecture des formats csv, ascii, xml, json.
- Traitement et analyse de données avec Pandas (proche de R).
- Création d'interfaces élémentaires avec Tkinter.
- Retour sur des points à approfondir.

Il exsite de nombreux formats pour sauvegarder des données

Exemple.xml

```
<note>
        <to>Tove</to>
        <from>Jani</from>
        <heading>Reminder</heading>
        <body>Don't forget me this weekend!</body>
        <info>
                <day>27/06/2012</day>
                <hour>10:05</hour>
        </info>
</note>
```

Exemple.csv

```
#mon fichier de donnees
1,2,4
43,0,1
3,0,9
…
```

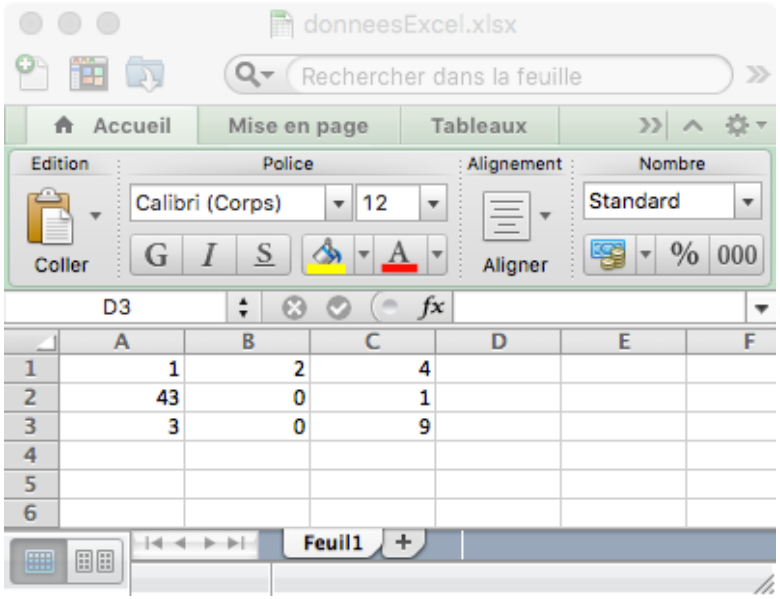
Exemple.json

```
{
        "glossary": {
                "title": "example glossary",
                "GlossDiv": {
                        "title": "S",
                        "GlossList": {
                                "GlossEntry": {
                                        …
                                }
                        }
                }
        }
}
```

Structures variées !

Lecture de fichiers de données

Il exsite de nombreux formats pour sauvegarder des données



Exemple.csv

```
#mon fichier de donnees
1,2,4
43,0,1
3,0,9
…
```

Fichiers ascii / Fichiers binaires

Lecture de fichiers de données

Une des grandes forces de Python est la taille de sa communauté

→ possibilité d'ouvrir de très nombreux formats de données et de les traiter numériquement

Technique de base pour les fichiers ascii : Utilisation de l'objet file

Fichier testFile.csv :

```
1,2,3
4,5,6
7,8,9
```

```
f = open('test.csv', 'r')

toto=f.read()
print toto

→ '1,2,3\n4,5,6\n7,8,9\n'

f.close()
```

Technique de base pour les fichiers ascii : Utilisation de l'objet file

Fichier testFile.csv :

1,2,3
4,5,6
7,8,9

```
f = open('test.csv', 'r')
lines=f.readlines()

for i in range(len(lines)):
        print lines[i]

→ 1,2,3
→ 4,5,6
→ 7,8,9


f.close()
```

Technique de base pour les fichiers ascii : Utilisation de l'objet file

Fichier testFile.csv :

```
1,2,3
4,5,6
7,8,9
```

```
f = open('test.csv', 'r')
lines=f.readlines()


for i in range(len(lines)):
        values=lines[i].split(',')
        print values[0]+' et '+values[1]

→ 1 et 3
→ 4 et 6
→ 7 et 9

f.close()
```

Remarque : Les values sont des *string* ici. Les convertir en *float* pour les traiter

Technique de base pour les fichiers ascii ou **binaires** :

Fichier testFile.csv :

```
1,2,3
4,5,6
7,8,9
```

```
f = open('test.csv', 'r')
f.seek(2)
print f.read(1)    →    2

f.seek(5)
f.read(1)=='\n'    →    True

f.seek(6)
a=f.read(3)        →    4,5
print a

f.seek(-2,2)
print f.read(1)    →    4,5

f.close()
```

Technique de base pour les fichiers ascii ou **binaires** :

Fichier testFile.csv :

1,2,3
4,5,6
7,8,9

```
f = open('test.csv', 'r')
f.seek(2)
print f.read(1)      →     2

f.seek(5)
f.read(1)=='\n'      →     True

f.seek(6)
a=f.read(3)          →     4,5
print a

f.seek(-2,2)
print f.read(1)      →     9

f.close()
```

Localisation initiale :
0 : Début fichier
1 : localisation courante
2 : Fin fichier

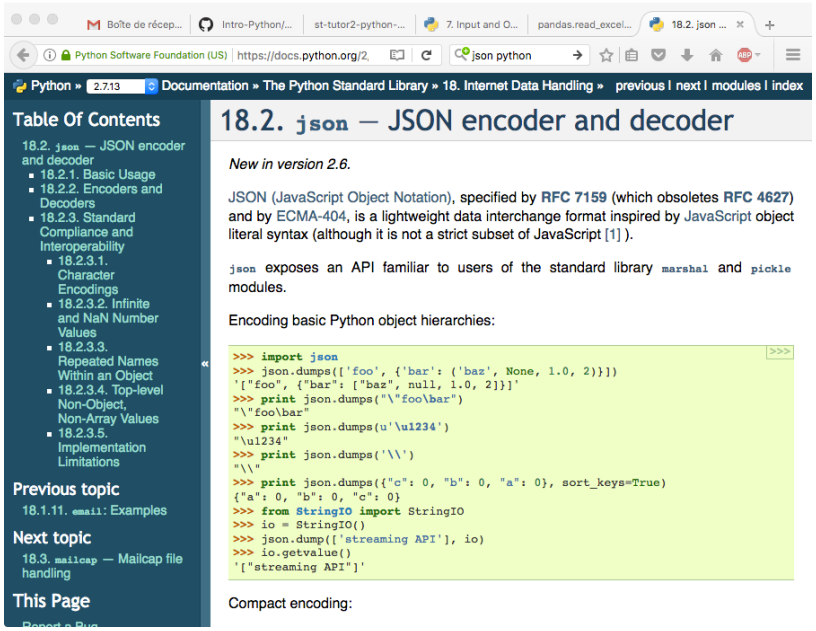
Chemin à parcourir en octets

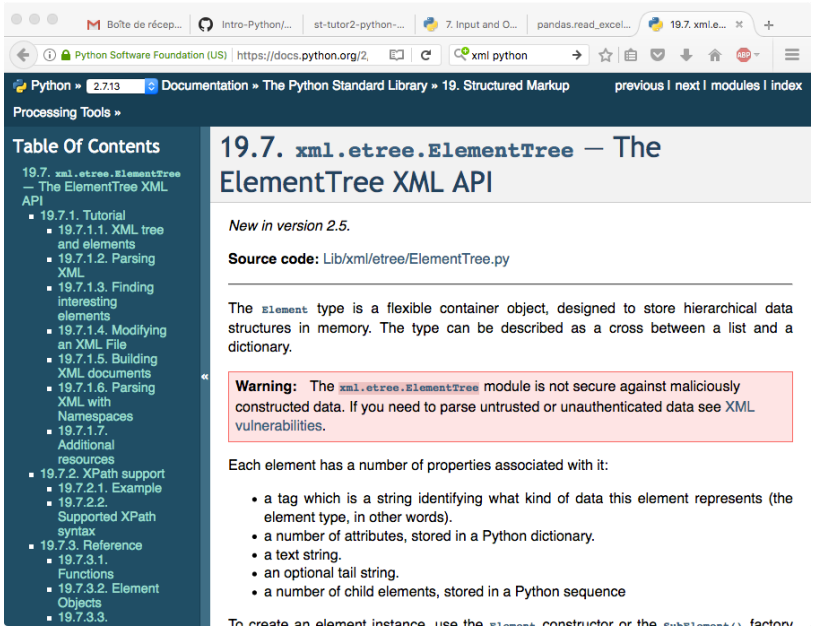
Lecture de fichiers de données

Et pour des fichiers avec une structure ascii plus complexe comme les xml ou json ???

```
import json

...
```

```
import xml.etree.ElementTree as ET

...
```





→ https://docs.python.org/2/library/json.html
→ https://docs.python.org/2/library/xml.etree.elementtree.html
→ et beaucoup d'autres plus spécialisés … demandez à google ☺

Et pour des fichiers binaires Matlab ou Excel ???

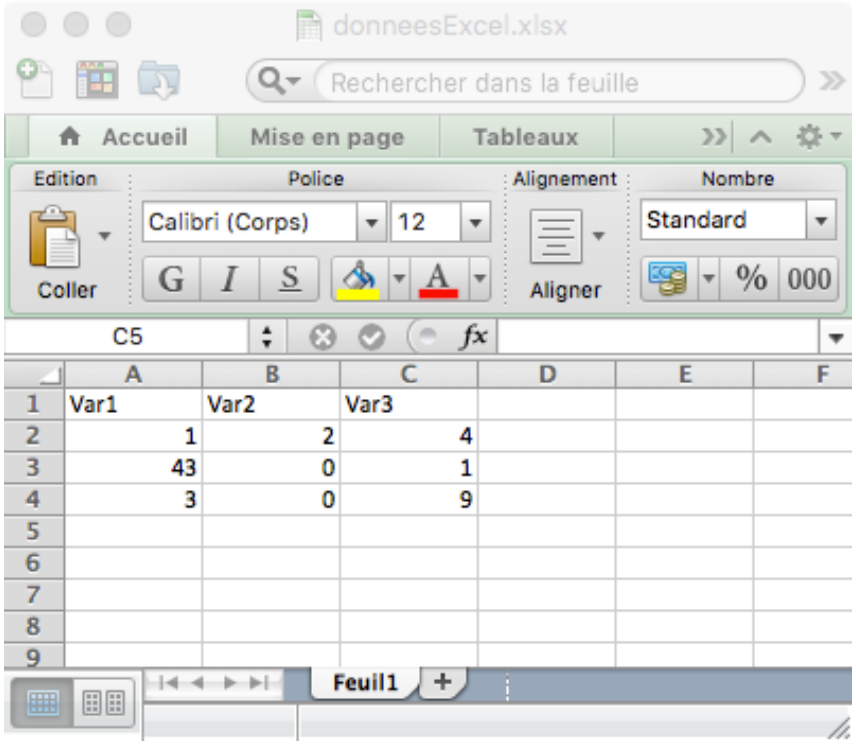
Matlab :

```
import scipy.io as sio
test = sio.loadmat('test.mat')
```

Excel :

```
from pandas import read_excel

myframe=read_excel('ExcelData.xlsx')

print myframe
```

```
→      Var1  Var2  Var3
→  0      1     2     4
→  1     43     0     1
→  2      3     0     9
```



Pandas

Pandas : Librairie Python pour extraire, préparer et éventuellement analyser, des données

- Contient les classes Series et DataFrame (tables de données)
- Lecture des fichiers .csv, xls, hdf5, HTML, XML, JSON, MongoDB, SQL, …
- Selection/Suppression/Ajout de lignes et de colonnes, fusion de DataFrames
- Gestion de données manquantes et abérantes
- Génération de nombres aléatoires
- Tests statistiques élémentaires
- Fonctions graphiques
- Gestion de très grosses données (via HDF5)
- …

```
import pandas as pd
data = {"state": ["Ohio", "Ohio", "Ohio","Nevada"], "year":
[2000, 2001, 2002, 2001], "pop": [1.5, 1.7, 3.6, 2.4]}
frame = pd.DataFrame(data , columns=["year", "state", "pop"])

print frame
→        year    state   pop
→    0   2000    Ohio    1.5
→    1   2001    Ohio    1.7
→    2   2002    Ohio    3.6
→    3   2001    Nevada  2.4

frame2=pd.DataFrame(data, columns=["year", "state", "pop",
"debt"], index=["one", "two", "three", "four"])

print frame2
→            year   state   pop   debt
→    one     2000   Ohio    1.5   NaN
→    two     2001   Ohio    1.7   NaN
→    three   2002   Ohio    3.6   NaN
→    four    2001   Nevada  2.4   NaN

...
```

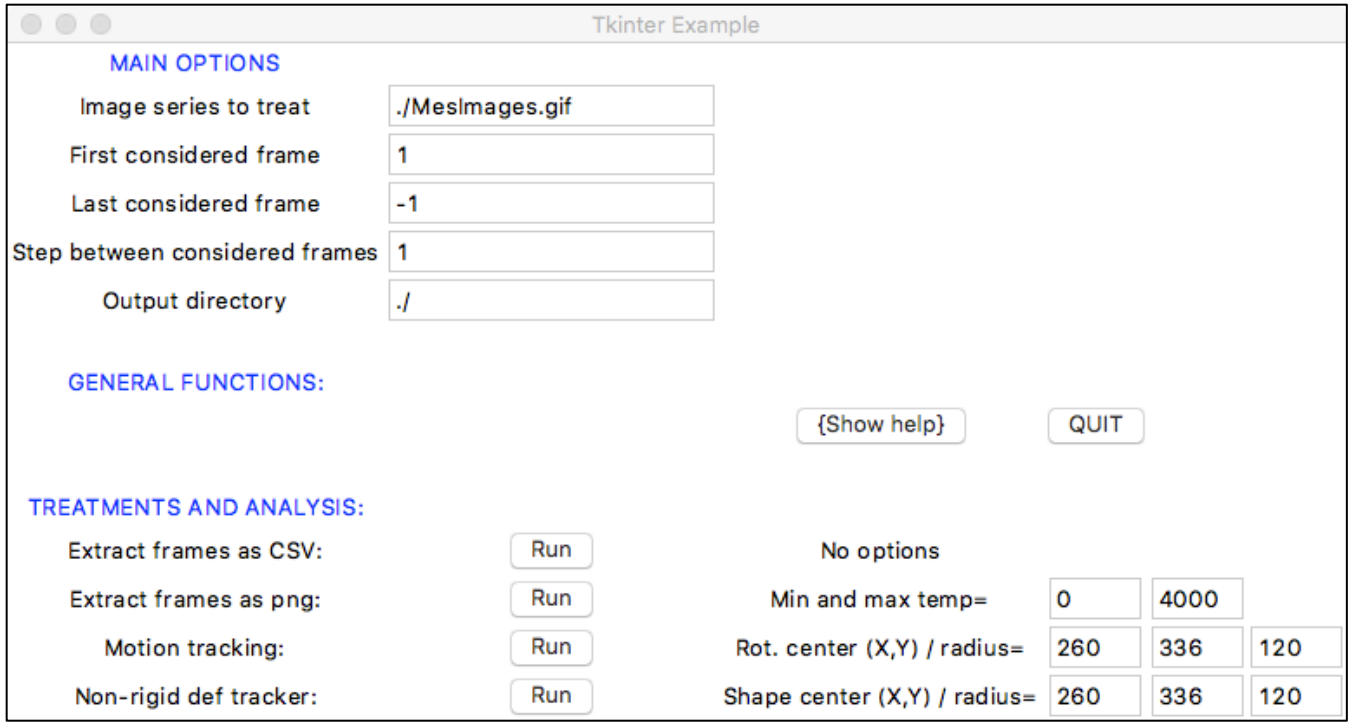
```
...

frame["state"]

→    0      Ohio
→    1      Ohio
→    2      Ohio
→    3    Nevada
→    4    Nevada

frame2["debt"] = 16.5
frame2.set_value('four', 'debt', 10)
print frame2

→            year   state   pop   debt
→    one     2000   Ohio    1.5   16.5
→    two     2001   Ohio    1.7   16.5
→    three   2002   Ohio    3.6   16.5
→    four    2001   Nevada  2.4   10.0
```

Pour aller plus loin avec Pandas:

- https://github.com/wikistat/Intro-Python/blob/master/Cal2-PythonPandas.ipynb

- http://pandas.pydata.org/pandas-docs/stable/

Tkinter (ou tkinter en Python 3): outil standard de Python pour écrire des interfaces graphiques



- Outil relativement simple et assez portable
- https://docs.python.org/2/library/tkinter.html

MERCI A VOUS ☺