

---

**j11**

os

# OS

- disponible sur presque tout système
- `os.uname` pas sur win

```
import os

os.uname() #OS details

os.mkdir("folder")
os.makedirs("folder/subfolder")

os.listdir() #list the content

os.chdir("folder") #change dir
os.getcwd() #current working dir

os.rmdir("folder") #remove dir

os.system("a_command") #
```



# datetime

- Date

```
from datetime import date, datetime
```

```
d = date.today() # a date object
```

```
dt = now()
```

```
d.day, d.month, d.year # int
```

```
christmas = date(2021,12,25)
```

```
dt1 = datetime(2021,11,21,9,30)
```

```
t1 = dt1.time() # 9:30:00:000000
```

# datetime

- Le format de date ISO est
  - YYYY-MM-DD
- Methode replace
- Récupérer le jour de la semaine
- Formater une date
- Formater une heure
- Convertir une chaine en date

```
valentine = \
    date.fromisoformat('2022-02-14')
```

```
bastille= valentine.replace(\
    month=7)
```

```
d.weekday() # Monday is 0
d.isoweekday() # Monday is 1
```

```
d.strftime('%Y/%m/%d')
t.strftime("%H:%M:%S")
```

```
datetime.strptime\
    ("2019/11/04 14:53:00",
    "%Y/%m/%d %H:%M:%S"))
```

<https://docs.python.org/3/library/datetime.html#strftime-and-strptime-format-codes>





# time

- time est un float
- Il donne le nombre de secondes depuis le 1<sup>er</sup> janvier, 1970, 00:00:00
- *timestamp*, timbre à date, horodatage

```
import time
t = time.time() # a timestamp
time.gmtime(t1) #universal
time.localtime(t1) #local
```

- Convertir un time stamp en date
- Convertir un timestamp en string
- Attendre

```
d = date.fromtimestamp(t)
```

```
time.ctime(t) # "Mon Nov 22 09:30:..."
```

```
time.sleep(3) # 3 seconds
```



# timedelta

- une différence de date
- jours, heures

```
from datetime import timedelta  
delta = dt2 - dt1  
delta = timedelta(weeks=3, days=2)  
# 23 days, 0:00:00
```



# calendar

- Calendrier d'une année, d'un mois
- définir le 1<sup>er</sup> jour de la semaine
- définir l'en-tête des semaines
- quel jour de la semaine tombe une date donnée ?
- y est-elle une année bissextile ?

```
import calendar
calendar.calendar(year)
calendar.month(year, month)

calendar.setfirstweekday(
    calendar.MONDAY)

calendar.weekheader(2)

calendar.weekday(2022, 5, 8)

calendar.isleap(y)
```



# pickle

- Mettre en conserve les objets d'un programme Python
  - sérialisation, *serialization*
  - désérialisation, *deserialization*
  - persistance, *persistence*
- Dans un fichier binaire
- Dans une suite d'octets, *byte stream*

## Attention :

- Pas les fonctions, ni les méthodes
- Fournir la définition des classes
- Versions de Python et de la bibliothèque
- Accès séquentiel
- Risque de sécurité en cas de lecture de données malignes

```
import pickle
```

```
x = ...  
with open('data.pckl','wb') as fo:  
    pickle.dump(x,fo)  
...  
with open('data.pckl','rb') as fi:  
    x1 = pickle.load(fo)
```

```
x = ...  
b = pickle.dumps(x)  
...  
x1 = pickle.loads(b)
```







# shelve

- Base de données organisée comme un dictionnaire
  - Accès direct par la clé
  - La clé doit être une str
  - Toutes les méthodes des dict
- Paramètre flag pour l'ouverture
  - 'r' read, existing db
  - 'w' read and write, existing db
  - 'c' read, write and create if db not exists
  - 'n' create new db for read and write
- Utilise pickle

```
import shelve
```

```
shlv = open("data.shlv", "c")  
shlv["foo"] = 12  
shlv.close()
```

```
with open("data.shlv", "r") as shlv:  
    print(shlv["foo"]) #12
```





# json

## *JavaScript Object Notation*

- Standard ECMA et IETF (RFC 7159)
- Douglas Crockford
- Convertir les objets en string et réciproquement - Texte
- Syntaxe ([www.json.org](http://www.json.org))
  - Lire les diagrammes de Conway
  - tentative de synthèse en EBNF ci-contre
  - Les espaces en dehors des guillemets ne sont pas significatifs
- Vérifier la syntaxe  
<https://jsonlint.com/>



```
json ::= val | object | array  
  
val ::= literal | object | array  
  
array ::= [ val, ... ] | []  
  
object ::= string:val, ... } | {}  
  
literal ::= int | float | string |  
           boolean | `null`  
  
boolean ::= `true` | `false`  
  
string ::= double " et pas ', \ pour échapper  
           \u ou Uxxx pour Unicode  
  
int ::= décimal, signe - optionnel, pas de 0  
        en tête, ni de +  
  
float ::= point décimal, notation scientifique
```

<https://www.ecma-international.org/publications-and-standards/standards/ecma-404/>  
<https://datatracker.ietf.org/doc/html/rfc7159.html>



# json

- Bibliothèque JSON
- Convertir un objet en chaîne JSON
- Et réciproquement
- Fonctionne pour les types natifs
  - pas pour les complexes
- Pour un classe utilisateur, on obtient
  - pour dumps, TypeError
  - Pour loads, un dictionnaire et non un objet de la classe attendue
- Il faut fournir les fonctions encode, decode spécifiques ou customiser les classes JSONEncoder/Decoder fournies par JSON

```
import json
```

```
s = json.dumps(x)
```

```
x = json.loads(s)
```

```
object is not JSON serializable
```

Classeur Jupyter ecjson





# type

- Le type d'un objet est sa classe
- Une classe est un objet.  
Voici l'anatomie d'une classe :
  - `__name__` pour une classe
  - `__class__` d'un objet ou d'une classe
  - `__bases__` pour une classe (tuple)
  - `__dict__` pour un objet ou une classe
- Créer une classe – instancier un type avec 3 arguments

```
>>> type(int)
<class 'type'>
```

```
>>> type(type)
<class 'type'>
```

```
Cls = type(name, bases, dic)
```



# metaclass

- La métaclasse est la classe d'une classe et hérite de la classe 'type'
- En surchargeant la méthode `__new__` on introduit les membres communs à toutes les classes de ce type
- Personnaliser le contrôle de l'instanciation d'une classe

```
class MC(type):  
    def __new__(mcs, name, bases, dic):  
        o = super().__new__(mcs, name, base, dic)  
        o.prop = "Nouvel attribut ajouté"  
        return o
```

```
class C(metaclass=MC):  
    pass
```

# Merci !

- Restons en contact :
  - Georges Georgoulis – [ggeorgoulis@alteractifs.org](mailto:ggeorgoulis@alteractifs.org) – 06 12 68 40 06



Coopérative d'activité et d'entrepreneurs [www.alteractifs.org](http://www.alteractifs.org)