



# Scraping avec Python

*requests et BeautifulSoup*

Dans ce TD, vous allez vous familiariser avec la notion de crawling d'un site Web. Certains sites comme Twitter, Facebook, Flickr, IMDB offrent des APIs pour explorer et interagir avec leurs contenus, mais dans certains cas de telles APIs n'existent pas ou ne sont pas officielles (comme AlloCine). Un développeur Web est alors obligé de récupérer des informations directement sur des pages Web dont la structure n'est pas forcément très cohérente. Ça peut être le cas pour le parcours complet de sites marchands dont on veut faire l'inventaire par exemple pour établir un comparateur de prix. Des bibliothèques Python comme BeautifulSoup aident grandement à se lancer dans l'aventure du crawling de sites Web. Pour aller plus loin des Frameworks comme Scrapy aident à accélérer et structurer la façon d'explorer un site. L'usage de grosses doses de JavaScript et d'Ajax dans certains sites rend parfois leur exploration assez difficile.

## Exercice 1. requests

```
1 import requests
2 UNI = requests.get("http://www.univ-orleans.fr")
3 print(UNI.text)
```

- Tapez le petit programme ci-dessus
- Lancez le avec Python 3 et observez la réponse
- Quels sont les autres champs de l'objet UNI ?

## Exercice 2. BeautifulSoup

Modifiez le programme comme suit :

```
1 import requests
2 from bs4 import BeautifulSoup
3 UNI = requests.get("http://www.univ-orleans.fr")
4 soup = BeautifulSoup(UNI.text, "lxml")
5 print(soup.h1)
```

- Quel est le résultat ?
- Comment récupérer toutes les divs de la page ?

- Et les images (avec les noms de fichiers et le contenu de leurs attributs alt ?)
- Et les liens (avec leurs attributs href) ?
- Comment récupérer toutes les divs de classe `header-composante` ?

### Exercice 3. Stackoverflow

En visitant <https://stackoverflow.com/questions/tagged/beautifulsoup>, vous obtenez les questions posées sur BeautifulSoup. (Si besoin créez un dictionnaire proxies contenant les proxies http et https)

1. Récupérez et affichez les titres des 10 premières questions
2. Récupérez aussi le nombre de votes et de réponses correspondant à la question
3. En utilisant la librairie Python pandas ou une autre lib, écrire les résultats dans un fichier csv Exemple :

```
1 # data est ici un tuple a 3 elements (triplet)
2 # correspondant aux données d'une question
3 df = pandas.DataFrame(data, columns=["titre", "votes"
4                                , "answers"])
4 df.to_csv("data.csv", index=0)
```

### Exercice 4. Offre de formation Université

Sur la page <http://formation.univ-orleans.fr/fr/formation/rechercher-une-formation.html#nav>, comment récupérer toutes les formations de type Licence Pro offertes par l'Université ? Toutes les formations de niveau Master en Ecologie ? Toutes les formations de l'iUT d'Orléans ?

### Exercice 5. Récupération de définitions de mots

Sur la page <http://services.aonaware.com/DictService/Default.aspx?action=define&dict=wn>, on peut lire les définitions des mots dans différents dictionnaires.

1. Complétez la fonction ci-dessous :

```
1 def get_definition(x):
2     """
3     Get Word definition from aonaware Site
4     """
5     URL = 'http://services.aonaware.com/DictService/
6           Default.aspx?action=define&dict=wn&query={0}'.
7           format(x)
6     html = requests.get(URL).text
7     # ../..
```

Faire de même avec un *payload* de requests

2. On suppose disposer d'un fichier *vocabulary.txt* contenant les mots dont on recherche les définitions (un par ligne) Complétez le code ci-dessous pour obtenir dans *lines* les mots à définir seuls et sans espaces :

```
1 lines = []  
2 with open('vocabulary.txt') as f:  
3     lines = f.readlines()
```

3. Faites en sorte d'écrire les définitions dans un fichier *definitions.txt* correctement indenté.

### Exercice 6. Bonus : Récupération de recettes de cuisine

Comment rechercher sur le site <https://www.marmiton.org> les recettes de cuisine de noix de saint jacques pour 2 personnes prêtes en moins de 30 mn ? Pensez à json-ld par exemple ...