Web scraping

Récolter des pages Web dans Python avec Beautiful Soup: les bases

- Beautiful Soup est une bibliothèque Python pour extraire des données à partir de fichiers HTML

- Installation : vous pouvez installer Beautiful Soup 4 à l'aide de pip. Le nom du package est beautifulsoup4 :

```
pip install beautifulsoup4
```

Notion d'analyseur :

Exemple 1 :

```
from bs4 import BeautifulSoup
with open("index.html") as fp:
    soup = BeautifulSoup(fp, 'html.parser')
```

La fonction BeautifulSoup peut accepter deux arguments. Le premier argument est la page HTML ou l'URL, et le second argument est l'analyseur que vous souhaitez utiliser. Les différents analyseurs sont: **html.parser**, **lxml** et **html5lib**. L'analyseur lxml comporte deux versions, un analyseur HTML et un analyseur XML. Le html.parser est un analyseur intégré.

Vous pouvez installer les autres analyseurs à l'aide des commandes suivantes :

```
pip install lxml
pip install html5lib
```

Récupérer des données :

Afficher le titre de la page Web :

```
print(soup.title)           <title>Web scraping</title>
print(soup.title.name)      title
print(soup.title.string)    Web scraping
```

Vous pouvez également racler la page Web pour d'autres informations comme le titre principal ou le premier paragraphe, leurs classes ou l'attribut id.

```
print(soup.a)               <a href="home.html">HOME.HTML</a>
print(soup.a.string)        HOME.HTML
print(soup.a.attrs)         {'href': 'home.html'}
print(soup.a['href'])       home.html
```

```
print(soup.h1)
<h1 class="firstHeading" id="firstHeading" lang="en">Langage Python</h1>

print(soup.h1['lang'])                    en

del soup.h1['id']
print(soup.h1)
<h1 class="firstHeading" lang="en">Langage Python</h1>
```

Afficher le contenu de la balise p :

```
print(soup.p)
<p><b>Texte en gras</b></p>
```

Afficher le contenu de toutes les balises p :

```
for sub_heading in soup.find_all('p'):
    print(sub_heading.text)
```

Navigation dans le DOM

Vous pouvez naviguer dans l'arborescence DOM en utilisant des noms de balises réguliers. Le chaînage de ces noms d'étiquettes peut vous aider à naviguer plus profondément dans l'arbre. Par exemple, vous pouvez obtenir le premier lien dans le premier paragraphe de la page Web en utilisant soup.p.a. Tous les liens du premier paragraphe peuvent être consultés en utilisant soup.p.find_all('a').

Exemple :

Première balise Gras dans p

```
print(soup.p.b)
```

Texte en gras

Toutes les balises b dans p :

```
soup.p.find_all('b')
```

Vous pouvez également accéder à tous les enfants d'une étiquette en tant que liste en utilisant `tag.contents`. Pour obtenir les enfants à un index spécifique, vous pouvez utiliser `tag.contents[index]`. Vous pouvez également itérer sur les enfants d'une étiquette en utilisant l'attribut `.children`.

Les `.children` et les `.contents` ne sont utiles que lorsque vous souhaitez accéder aux descendants directs ou de premier niveau d'une étiquette. Pour obtenir tous les descendants, vous pouvez utiliser l'attribut `.descendants`.

Exemples :

```
<p><b>Texte en gras</b><i>Texte en italique</i><h3>autre</h3></p>
print(soup.p.contents)
[<b>Texte en gras</b>, <i>Texte en italique</i>, <h3>autre</h3>]

print(soup.p.contents[1])
<i>Texte en italique</i>


print(soup.p.contents[1].string)
Texte en italique
```

Obtenir les balises seulement :

```
for child in soup.p.children:
    print(child.name)
```

b i h3

```
print(soup.p.parent.name)
```

body

```
for parent in soup.p.parents:
    print(parent.name)
```

body

html

[document]

Travail à faire :

1. Récupérer les données du tableau HTML(balise table)
2. Convertir ces données en dataframe pandas

Scraper les données en utilisant un lien :

```
import requests

from bs4 import BeautifulSoup


req =
requests.get('https://en.wikipedia.org/wiki/Python_(programming_language)')

soup = BeautifulSoup(req.text, "lxml")
```