Web Scraping - BeautifulSoup4

January 31, 2018

```
In [2]: from bs4 import BeautifulSoup
        import requests
        # import psycopg2
        import re
        import pandas
```

0.1 Jerarquía de elementos HTML

0.2 DOM (Document Object Mode)

0.2.1 Obteniendo nuestro primer documento web

```
In [3]: URL = 'http://nostarch.com'
        URL = BeautifulSoup(requests.get(URL).text, "lxml")
```

BeautifulSoup nos otorga una estructura de datos para manipular el objeto:

```
In [4]: URL.title, URL.title.name, URL.title.string
```

```
Out[4]: (<title>No Starch Press | No Starch Press</title>,
         'title',
         'No Starch Press | No Starch Press')
```

```
In [ ]: URL.title.parent.name
```

```
In [ ]: # HTML: <p>
        URL.p
```

```
In [ ]: URL.body.b, URL.a
```

0.3 Ejercicio:

Crear una función que reciba una URL y regrese el título (title)

```
In [ ]: # Ejercicio
```

```
        # Ejercicio
```

```
In [ ]: # Buscar todas las etiquetas

        URL.find_all('p')
```

```
In [ ]: # Buscar todas las etiquetas

        URL.find_all('a')
```

```
In [5]: # Seleccion por clase

        URL.find_all('div', class_='product-body')

        lista = URL.find_all('div', class_='product-body')
```

Comprobar los resultados:

```
In [6]: lista[0].a.get_text(), lista[0].a["href"]


        ---------------------------------------------------------------------------

        IndexError                                Traceback (most recent call last)

        <ipython-input-6-a50bb97424e3> in <module>()
    ----> 1 lista[0].a.get_text(), lista[0].a["href"]


        IndexError: list index out of range
```

Una vez que sabemos como se compone la lista de resultados, podemos exportarla a un DataFrame:

```
In [7]: # Creamos un nuevo diccionario
        resultado_dic = {}

        # Copiamos los resultados
        for element in lista:
            resultado_dic[str(element.a.get_text())] = element.a["href"]
```

```
In [8]: # Creamos un nuevo DataFrame
        resutado_dataframe = pandas.DataFrame.from_dict(resultado_dic, orient='index')
        resutado_dataframe.head()
```

```
Out[8]: Empty DataFrame
        Columns: []
        Index: []
```

```
In [9]: # Renombramos las columnas
        resutado_dataframe.rename(columns={0: 'URL'}, inplace=True)

        resutado_dataframe.head()
```

```
Out[9]: Empty DataFrame
        Columns: []
        Index: []

In [ ]: # Exportar a CSV

        resutado_dataframe.to_csv('data/nostarch_lista.csv')
```

0.4 Selección por herencia DOM

```
In [10]: URL = 'https://news.ycombinator.com/news'
         URL = BeautifulSoup(requests.get(URL).text, "lxml")
```

0.4.1 Herencia DOM: tr > td > a

0.4.2 Recuerdan REGEX?

```
In [11]: a_list = URL.select('tr > td  > a[href*="."]')

In [12]: a_list

Out[12]: [<a href="http://www.ycombinator.com"><img height="18" src="y18.gif" style="border:1p
          <a class="storylink" href="https://www.sec.gov/litigation/litreleases/2017/lr23870.h
          <a class="storylink" href="https://www.eff.org/alice">Saved by Alice</a>,
          <a class="storylink" href="https://drikerf.com/building-pixels-a-daily-source-of-insp
          <a class="storylink" href="https://blog.2ndquadrant.com/what-is-select-skip-locked-f
          <a class="storylink" href="https://open.nytimes.com/react-relay-and-graphql-under-th
          <a class="storylink" href="http://ruslanledesma.com/2016/06/17/why-does-heap-work.ht
          <a class="storylink" href="http://biosrhythm.com/?page_id=1453">WiFi232  An Internet
          <a class="storylink" href="http://deako.com/careers" rel="nofollow">Deako (YC W16) I
          <a class="storylink" href="https://www.discretization.de/media/filer_public/2014/09/
          <a class="storylink" href="https://github.com/mitnk/cicada">Cicada  Unix shell writt
          <a class="storylink" href="http://www.gemini.edu/node/12679">Striking Gemini Images
          <a class="storylink" href="https://www.mikeash.com/pyblog/friday-qa-2017-06-30-disse
          <a class="storylink" href="http://www.nybooks.com/daily/2017/06/29/myth-maker-of-the
          <a class="storylink" href="https://www.microsoft.com/en-us/research/publication/what
          <a class="storylink" href="https://www.buzzfeed.com/williamalden/theres-a-fight-brew
          <a class="storylink" href="https://www.independent.co.uk/news/world-0/nevada-cannabi
          <a class="storylink" href="https://www.theatlantic.com/business/archive/2017/06/chin
          <a class="storylink" href="https://dolphin-emu.org/blog/2017/07/01/dolphin-progress-
          <a class="storylink" href="https://www.theatlantic.com/technology/archive/2017/06/so
          <a class="storylink" href="https://www.ibiblio.org/harris/500milemail.html">The case
          <a class="storylink" href="http://blogs.bl.uk/digitisedmanuscripts/2017/06/making-a-g
          <a class="storylink" href="http://www.slate.com/articles/technology/future_tense/201
          <a class="storylink" href="https://github.com/Microsoft/ELL">Microsofts Embedded Lea
          <a class="storylink" href="https://www.nytimes.com/2017/06/29/well/live/what-to-blam
          <a class="storylink" href="https://www.nytimes.com/2017/06/30/technology/women-entre
          <a class="storylink" href="https://vlad.d2dx.com/the-great-assistant-skills-compari
          <a class="storylink" href="https://motherboard.vice.com/en_us/article/8xa4ka/iphone-
```

```
<a class="storylink" href="http://www.npr.org/sections/krulwich/2010/07/01/128170775/
<a class="storylink" href="https://redditblog.com/2017/06/30/why-we-chose-typescript/
```

0.5 Ejercicios

- Image Site Downloader
- Link Verification

1 Pattern

Web mining module for Python, with tools for scraping, natural language processing, machine learning, network analysis and visualization. http://www.clips.ua.ac.be/pages/pattern