S12-T02

April 3, 2022

1 S12 T02 - Web scraping

1.1 Descripción

Aprende a realizar web scraping. _____ Objetivos - Web scraping - Documentar datos recogidos con web scraping

[1]:
```
import pandas as pd

import requests
from bs4 import BeautifulSoup

from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver import Chrome
from webdriver_manager.chrome import ChromeDriverManager

from docx import Document
from docx.shared import Inches
from docx.enum.text import WD_LINE_SPACING
from docx import text
from docx.shared import Pt

import warnings
warnings.filterwarnings('ignore')

import scrapy
from scrapy import signals
from scrapy.exporters import CsvItemExporter
from scrapy.http import TextResponse
from scrapy import Selector
from scrapy.pipelines.images import ImagesPipeline
from scrapy.exceptions import DropItem
from scrapy import Request
from scrapy.spiders import CrawlSpider, Rule, Spider
from scrapy.linkextractors import LinkExtractor
```

```
from scrapy.exceptions import CloseSpider
from scrapy.crawler import CrawlerRunner, CrawlerProcess
from fake_useragent import UserAgent
from scrapy.http import TextResponse

import csv
import json
import logging
```

1.2 Nivel 1

1.2.1 - Ejercicio 1

Realiza web scraping de una página de la bolsa de Madrid (https://www.bolsamadrid.es) utilizando BeautifulSoup y Selenium.

> **web scraping** es el proceso de recopilar informacion de Internet.
>
> *Challenges* : Variedad en la estructura de la informacion y Durabilidad del sitio web (webistes cambian constantemente)

Primero se hace scrap del html de la pagina:

[20]:
```
#get url
URL = "https://www.bolsamadrid.es"
page = requests.get(URL)

page
```

[20]:
```
<Response [200]>
```

1.2.2 Beautifiul Soup

Primero utilizaremos la libreria Beautiful soup para hacer "parsing" de los datos. Esta libreria permite interactuar con HTML de manera similar que se interactua con la pagina web utilizando los developer tools.

[3]:
```
#create Beautiful soup object
soup = BeautifulSoup(page.content, "html.parser")
#page.content avoids parsing problems, html parser makes the appropriate parser
```

[4]:
```
#get the title of the webpage
title = soup.find("title").get_text().replace("\r\n","").replace("\t","")
print(title)
```

```
Bolsa de Madrid
```

Ahora, se va a buscar la url que contiene la informacion referente a las acciones

[6]:
```
#search in the shares tag
shares_tag = soup.find('a', string = 'Acciones')
```

```
print(shares_tag)
```

```
<a href="/esp/aspx/Mercados/Precios.aspx?indice=ESI100000000"
target="_self">Acciones</a>
```

[8]:
```
#get url by the tag
link_s = shares_tag.get(key='href')
print(link_s)
```

```
/esp/aspx/Mercados/Precios.aspx?indice=ESI100000000
```

Y ahora se combina la URL de la pagina principal con la ubicacion de las acciones y creamos otro beautiful soup object

[21]:
```
#set the url of the data
url_shares = URL+link_s.replace('esp','ing')
print(url_shares)
html_shares = requests.get(url_shares)
```

```
https://www.bolsamadrid.es/ing/aspx/Mercados/Precios.aspx?indice=ESI100000000
```

[22]:
```
#create beautifulsoup element for new url
soup_s = BeautifulSoup(html_shares.content, 'html.parser')
```

Ahora, se busca el elemento que incluye la tabla, se itera para extraer los datos y, finalmente, creamos un dataframe con ellos.

[23]:
```
#find the data by tag
table_tag = soup_s.find(id='ctl00_Contenido_tblAcciones')
```

Una de las opciones para obtener la informacion requerida es iterar en la jerarquia de los elementos de la web, ahora identificamos que la tabla es un child element, por lo que se va a iterar en él para obtener la informacion relativa a las acciones y generar una la tabla:

[25]:
```
#iteration for gather data
rows = []
for child in table_tag.children: #iterate child element
    element = []
    if child != "\n":
        for i in child:
            if i != "\n":
                element.append(i.text)
        rows.append(element)
```

[28]:
```
#save to dataframe
df_acciones = pd.DataFrame(rows[1:], columns=rows[0])
df_acciones
```

[28]:

	Name	Last	% Dif.	High	Low	Volume	\
0	ACCIONA	169.6000	-0.64	173.1000	168.4000	182,238	

1	ACERINOX	10.2700	1.73	10.3000	10.1000	1,350,393
2	ACS	24.9300	0.93	25.1000	24.7500	879,936
3	AENA	150.2000	2.18	150.6000	147.5500	113,071
4	ALMIRALL	11.7500	-5.62	12.5300	11.7100	666,050
5	AMADEUS	60.7000	7.28	61.0400	57.0200	1,359,484
6	ARCELORMIT.	29.9750	-1.38	30.7800	29.2150	798,387
7	B.SANTANDER	3.2485	5.47	3.2630	3.1140	71,880,085
8	BA.SABADELL	0.8016	7.02	0.8040	0.7512	55,519,643
9	BANKINTER	5.4500	6.78	5.4600	5.2060	4,206,500
10	BBVA	5.5010	6.71	5.5530	5.2080	35,106,942
11	CAIXABANK	3.1630	4.56	3.1640	3.0350	16,581,598
12	CELLNEX	44.0300	0.99	44.3100	43.5400	2,057,550
13	CIE AUTOMOT.	21.4200	7.10	21.4200	20.1200	393,235
14	ENAGAS	19.7700	-2.51	20.4000	19.6550	1,205,706
15	ENDESA	19.1600	1.19	19.2300	18.9150	1,103,494
16	FERROVIAL	24.5900	1.70	24.8100	24.1500	936,605
17	FLUIDRA	27.8000	4.91	28.0000	26.7500	302,745
18	GRIFOLS CL.A	15.6250	1.99	15.6750	15.3950	847,625
19	IAG	1.7660	5.65	1.7790	1.6870	33,210,115
20	IBERDROLA	9.9820	2.36	10.0400	9.7620	23,662,983
21	INDITEX	21.2800	3.70	21.4300	20.6500	8,073,474
22	INDRA A	10.2300	2.51	10.3300	10.0100	751,632
23	INM.COLONIAL	8.3100	0.12	8.3300	8.1250	1,174,385
24	MAPFRE	1.9300	2.12	1.9300	1.9000	2,866,547
25	MELIA HOTELS	6.9520	5.24	6.9520	6.6720	1,088,057
26	MERLIN	10.7600	0.33	10.8100	10.5950	953,240
27	NATURGY	26.6200	-0.86	27.1200	26.2100	504,332
28	PHARMA MAR	67.6600	0.18	68.7000	66.2000	71,962
29	R.E.C.	17.9400	-0.19	18.0750	17.7150	1,210,224
30	REPSOL	11.7720	-1.11	12.1700	11.6220	6,530,273
31	ROVI	67.5000	-0.88	69.3000	66.7000	91,861
32	SIEMENS GAME	15.8050	1.64	15.9150	15.5350	3,032,784
33	SOLARIA	18.7000	-8.82	20.8200	18.6350	1,775,648
34	TELEFONICA	4.3560	0.97	4.4105	4.3150	15,124,860

	Turnover (€ Thousands)	Date	Time
0	31,085.53	29/03/2022	Close
1	13,798.84	29/03/2022	Close
2	21,932.87	29/03/2022	Close
3	16,912.78	29/03/2022	Close
4	7,925.93	29/03/2022	Close
5	80,913.07	29/03/2022	Close
6	24,017.71	29/03/2022	Close
7	230,269.21	29/03/2022	Close
8	43,534.96	29/03/2022	Close
9	22,548.98	29/03/2022	Close
10	188,935.83	29/03/2022	Close

11	51,841.62	29/03/2022	Close
12	90,284.50	29/03/2022	Close
13	8,218.97	29/03/2022	Close
14	23,895.70	29/03/2022	Close
15	21,090.05	29/03/2022	Close
16	22,991.42	29/03/2022	Close
17	8,383.32	29/03/2022	Close
18	13,191.02	29/03/2022	Close
19	57,773.56	29/03/2022	Close
20	233,844.18	29/03/2022	Close
21	169,682.33	29/03/2022	Close
22	7,673.55	29/03/2022	Close
23	9,684.48	29/03/2022	Close
24	5,512.04	29/03/2022	Close
25	7,439.00	29/03/2022	Close
26	10,230.53	29/03/2022	Close
27	13,461.64	29/03/2022	Close
28	4,838.85	29/03/2022	Close
29	21,687.78	29/03/2022	Close
30	77,336.88	29/03/2022	Close
31	6,218.92	29/03/2022	Close
32	47,834.78	29/03/2022	Close
33	34,254.19	29/03/2022	Close
34	66,068.65	29/03/2022	Close

1.3 Selenium

Ahora utilizando el paquete Selenium se hara el web scraping.

> Selenium es una poderosa herramienta para controlar los navegadores web a través de programas y realizar la automatización del navegador.

https://www.geeksforgeeks.org/selenium-basics-components-features-uses-and-limitations/

[55]:
```
#create webdriver for chrome
driver = webdriver.Chrome(ChromeDriverManager().install())
```

```
====== WebDriver manager ======
Current google-chrome version is 99.0.4844
Get LATEST chromedriver version for 99.0.4844 google-chrome
There is no [win32] chromedriver for browser 99.0.4844 in cache
Trying to download new driver from
https://chromedriver.storage.googleapis.com/99.0.4844.51/chromedriver_win32.zip
Driver has been saved in cache
[C:\Users\Usuario\.wdm\drivers\chromedriver\win32\99.0.4844.51]
```

Ahora iniciamos el manejo de la url a traves del webdriver de selenium:

[61]:
```
#get the url through the driver
driver.get(url_shares)
```

Se realiza la busqueda de la tabla anterior de las acciones mediante el Xpath: >"XPath is a language for selecting nodes in XML documents, which can also be used with HTML."

https://www.browserstack.com/guide/find-element-by-xpath-in-selenium

[62]:
```
#search bt xpath
table_rows = driver.find_elements(By.XPATH, "/html/body/div[1]/table/tbody/
↪tr[4]/td[2]/div[1]/form/div[6]/table/tbody/tr")
```

Iteramos nuevamente por la tabla para extraer los datos:

[63]:
```
#iteration for gather data

new_rows = []
for row in table_rows[1:]:
    element = []
    values = row.find_elements(By.XPATH, "td")
    for value in values:
        element.append(value.text)
    new_rows.append(element)
```

Se convierten los datos extraidos en un dataframe :

[64]:
```
#save to dataframe
df_shares = pd.DataFrame(new_rows[1:], columns=rows[0])
df_shares
```

[64]:

	Name	Last	% Dif.	High	Low	Volume
0	ACERINOX	10.2700	1.73	10.3000	10.1000	1,350,393
1	ACS	24.9300	0.93	25.1000	24.7500	879,936
2	AENA	150.2000	2.18	150.6000	147.5500	113,071
3	ALMIRALL	11.7500	-5.62	12.5300	11.7100	666,050
4	AMADEUS	60.7000	7.28	61.0400	57.0200	1,359,484
5	ARCELORMIT.	29.9750	-1.38	30.7800	29.2150	798,387
6	B.SANTANDER	3.2485	5.47	3.2630	3.1140	71,880,085
7	BA.SABADELL	0.8016	7.02	0.8040	0.7512	55,519,643
8	BANKINTER	5.4500	6.78	5.4600	5.2060	4,206,500
9	BBVA	5.5010	6.71	5.5530	5.2080	35,106,942
10	CAIXABANK	3.1630	4.56	3.1640	3.0350	16,581,598
11	CELLNEX	44.0300	0.99	44.3100	43.5400	2,057,550
12	CIE AUTOMOT.	21.4200	7.10	21.4200	20.1200	393,235
13	ENAGAS	19.7700	-2.51	20.4000	19.6550	1,205,706
14	ENDESA	19.1600	1.19	19.2300	18.9150	1,103,494
15	FERROVIAL	24.5900	1.70	24.8100	24.1500	936,605
16	FLUIDRA	27.8000	4.91	28.0000	26.7500	302,745
17	GRIFOLS CL.A	15.6250	1.99	15.6750	15.3950	847,625

18	IAG	1.7660	5.65	1.7790	1.6870	33,210,115
19	IBERDROLA	9.9820	2.36	10.0400	9.7620	23,662,983
20	INDITEX	21.2800	3.70	21.4300	20.6500	8,073,474
21	INDRA A	10.2300	2.51	10.3300	10.0100	751,632
22	INM.COLONIAL	8.3100	0.12	8.3300	8.1250	1,174,385
23	MAPFRE	1.9300	2.12	1.9300	1.9000	2,866,547
24	MELIA HOTELS	6.9520	5.24	6.9520	6.6720	1,088,057
25	MERLIN	10.7600	0.33	10.8100	10.5950	953,240
26	NATURGY	26.6200	-0.86	27.1200	26.2100	504,332
27	PHARMA MAR	67.6600	0.18	68.7000	66.2000	71,962
28	R.E.C.	17.9400	-0.19	18.0750	17.7150	1,210,224
29	REPSOL	11.7720	-1.11	12.1700	11.6220	6,530,273
30	ROVI	67.5000	-0.88	69.3000	66.7000	91,861
31	SIEMENS GAME	15.8050	1.64	15.9150	15.5350	3,032,784
32	SOLARIA	18.7000	-8.82	20.8200	18.6350	1,775,648
33	TELEFONICA	4.3560	0.97	4.4105	4.3150	15,124,860

	Turnover (€ Thousands)	Date	Time
0	13,798.84	29/03/2022	Close
1	21,932.87	29/03/2022	Close
2	16,912.78	29/03/2022	Close
3	7,925.93	29/03/2022	Close
4	80,913.07	29/03/2022	Close
5	24,017.71	29/03/2022	Close
6	230,269.21	29/03/2022	Close
7	43,534.96	29/03/2022	Close
8	22,548.98	29/03/2022	Close
9	188,935.83	29/03/2022	Close
10	51,841.62	29/03/2022	Close
11	90,284.50	29/03/2022	Close
12	8,218.97	29/03/2022	Close
13	23,895.70	29/03/2022	Close
14	21,090.05	29/03/2022	Close
15	22,991.42	29/03/2022	Close
16	8,383.32	29/03/2022	Close
17	13,191.02	29/03/2022	Close
18	57,773.56	29/03/2022	Close
19	233,844.18	29/03/2022	Close
20	169,682.33	29/03/2022	Close
21	7,673.55	29/03/2022	Close
22	9,684.48	29/03/2022	Close
23	5,512.04	29/03/2022	Close
24	7,439.00	29/03/2022	Close
25	10,230.53	29/03/2022	Close
26	13,461.64	29/03/2022	Close
27	4,838.85	29/03/2022	Close
28	21,687.78	29/03/2022	Close

```
29              77,336.88  29/03/2022  Close
30               6,218.92  29/03/2022  Close
31              47,834.78  29/03/2022  Close
32              34,254.19  29/03/2022  Close
33              66,068.65  29/03/2022  Close
```

Se extrae informacion adicional referente al titulo y la clase de datos de la tabla:

[67]:
```
#get page title and classname
TituloPag = driver.find_element(By.CLASS_NAME, "TituloPag").text
Ctr = driver.find_element(By.CLASS_NAME, "Ctr").text
TituloPag + ' ' + Ctr
```

[67]:
```
'Session Prices IBEX 35®'
```

Se salvan los datos en un archivo csv:

[68]:
```
#save data to file
filename= './{}_{}_Selenium.csv'.format(TituloPag.replace(" ","),Ctr.
 ↪replace(r"®","").replace(r" ",""))
df_shares.to_csv(filename, index=False)
```

Se cierra y termina el driver:

[69]:
```
#close file
driver.quit()
```

1.4 Nivel 2

1.4.1 - Ejercicio 2

Documenta en un word tu conjunto de datos generado con la información que tienen los distintos archivos de Kaggle.

[53]:
```
#new document
document = Document()

# Heading
document.add_heading('IBEX 35 Shares', 0)

# Context
document.add_heading(' CONTEXT: ', level = 2)
document.add_paragraph('This document shows the shares listed on the Spanish␣
 ↪market. The IBEX 35 is the main reference stock market index of the Spanish␣
 ↪stock market prepared by Bolsas y Mercados Españoles.', style = 'List␣
 ↪Bullet').runs[0].font.size =Pt(10)


# Content
document.add_heading(' CONTENT: ', level = 2)
```

```
document.add_paragraph('Nombre: Share\'s name', style = 'List Bullet').runs[0].
 ↪font.size = Pt(10)
document.add_paragraph('Últ: Latest share price update', style = 'List Bullet').
 ↪runs[0].font.size = Pt(10)
document.add_paragraph('% Dif: Share price difference', style = 'List Bullet').
 ↪runs[0].font.size = Pt(10)
document.add_paragraph('Máx: The maximum share price of the session', style =␣
 ↪'List Bullet').runs[0].font.size = Pt(10)
document.add_paragraph('Mín: The minimum share price of the session', style =␣
 ↪'List Bullet').runs[0].font.size = Pt(10)
document.add_paragraph('Volumen: Share volume', style = 'List Bullet').runs[0].
 ↪font.size = Pt(10)
document.add_paragraph('Efectivo (miles €):  Company cash', style = 'List␣
 ↪Bullet').runs[0].font.size = Pt(10)
document.add_paragraph('Fecha: Daily date', style = 'List Bullet').runs[0].font.
 ↪size = Pt(10)
document.add_paragraph('Hora: Hour ', style = 'List Bullet').runs[0].font.size␣
 ↪= Pt(10)


# Table
table = document.add_table(rows = 1, cols = len(df_acciones.columns), style =␣
 ↪'TableGrid')
hdr_cells = table.rows[0].cells

rows = []
for child in table_tag.children: #iterate child element
    element = []
    if child != "\n":
        for i in child:
            if i != "\n":
                element.append(i.text)
        rows.append(element)

paragraph = document.add_paragraph()
paragraph.paragraph_format.line_spacing_rule = WD_LINE_SPACING.EXACTLY

document.add_page_break()

document.save('data_shares.docx')
```

1.5 Nivel 3

1.5.1 - Ejercicio 3

Elige una página web que quieras y realiza web scraping mediante la librería Scrapy. > **Scraping es una técnica, la cual podemos utilizar, para hacer barridos de web completas.** Su

arquitectura basada en Pipelines, Schedulers, Spiders y Downloaders permite al desarrollador tener control sobre todo el proceso de Scraping.

https://www.theninjacto.xyz/Web-Scraping-Using-Python-in-Jupyter-notebooks/

Se crea un pipeline sencillo para guardar los elementos encontrados en un JSON file:

[3]:
```
class JsonWriterPipeline(object):

    def open_spider(self, spider):
        self.file = open('quoteresult.jl', 'w')

    def close_spider(self, spider):
        self.file.close()

    def process_item(self, item, spider):
        line = json.dumps(dict(item)) + "\n"
        self.file.write(line)
        return item
```

Se define la clase spider de la cual se va a realizar el crawling de la web para obtener los datos.

[4]:
```
class QuotesSpider(scrapy.Spider):
    name = "quotes"
    start_urls = [
        'http://quotes.toscrape.com/page/1/',
        'http://quotes.toscrape.com/page/2/',
    ]
    custom_settings = {
        'LOG_LEVEL': logging.WARNING,
        'ITEM_PIPELINES': {'__main__.JsonWriterPipeline': 1}, # Used for␣
↪pipeline 1
        'FEED_FORMAT':'json',                                 # Used for␣
↪pipeline 2
        'FEED_URI': 'quoteresult.json'                        # Used for␣
↪pipeline 2
    }

    def parse(self, response):
        for quote in response.css('div.quote'):
            yield {
                'text': quote.css('span.text::text').extract_first(),
                'author': quote.css('span small::text').extract_first(),
                'tags': quote.css('div.tags a.tag::text').extract(),
            }
```

Se inicia el crawler:

[26]:
```
process = CrawlerProcess({
    'USER_AGENT': 'Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)'
})

process.crawl(QuotesSpider)
process.start()
```

```
2022-04-03 13:27:47 [scrapy.utils.log] INFO: Scrapy 2.6.1 started (bot:
scrapybot)
2022-04-03 13:27:47 [scrapy.utils.log] INFO: Versions: lxml 4.6.3.0, libxml2
2.9.10, cssselect 1.1.0, parsel 1.6.0, w3lib 1.21.0, Twisted 22.2.0, Python
3.8.8 (default, Apr 13 2021, 15:08:03) [MSC v.1916 64 bit (AMD64)], pyOpenSSL
20.0.1 (OpenSSL 1.1.1k  25 Mar 2021), cryptography 3.4.7, Platform
Windows-10-10.0.19041-SP0
2022-04-03 13:27:47 [scrapy.crawler] INFO: Overridden settings:
{'LOG_LEVEL': 30,
 'USER_AGENT': 'Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)'}
```

```
---------------------------------------------------------------------------
ReactorAlreadyInstalledError                Traceback (most recent call last)
<ipython-input-26-d8e819c2a04b> in <module>
      3 })
      4
----> 5 process.crawl(QuotesSpider)
      6 process.start()

~\anaconda3\lib\site-packages\scrapy\crawler.py in crawl(self,␣
↪crawler_or_spidercls, *args, **kwargs)
    203                 'The crawler_or_spidercls argument cannot be a spider␣
↪object, '
    204                 'it must be a spider class (or a Crawler object)')
--> 205         crawler = self.create_crawler(crawler_or_spidercls)
    206         return self._crawl(crawler, *args, **kwargs)
    207

~\anaconda3\lib\site-packages\scrapy\crawler.py in create_crawler(self,␣
↪crawler_or_spidercls)
    236         if isinstance(crawler_or_spidercls, Crawler):
    237             return crawler_or_spidercls
--> 238         return self._create_crawler(crawler_or_spidercls)
    239
    240     def _create_crawler(self, spidercls):

~\anaconda3\lib\site-packages\scrapy\crawler.py in _create_crawler(self,␣
↪spidercls)
    311         if isinstance(spidercls, str):
    312             spidercls = self.spider_loader.load(spidercls)
--> 313         return Crawler(spidercls, self.settings, init_reactor=True)
```

```
    314
    315     def start(self, stop_after_crawl=True, install_signal_handlers=True :

~\anaconda3\lib\site-packages\scrapy\crawler.py in __init__(self, spidercls,␣
 ↪settings, init_reactor)
     80             else:
     81                 from twisted.internet import default
---> 82                 default.install()
     83             log_reactor_info()
     84         if reactor_class:

~\anaconda3\lib\site-packages\twisted\internet\selectreactor.py in install()
    192     from twisted.internet.main import installReactor
    193
--> 194     installReactor(reactor)
    195
    196


~\anaconda3\lib\site-packages\twisted\internet\main.py in installReactor(reacto )
     30
     31     if "twisted.internet.reactor" in sys.modules:
---> 32         raise error.ReactorAlreadyInstalledError("reactor already␣
 ↪installed")
     33     twisted.internet.reactor = reactor
     34     sys.modules["twisted.internet.reactor"] = reactor

ReactorAlreadyInstalledError: reactor already installed
```

Por ultimo se salvan los datos en un dataframe:

[9]:
```
quotes = pd.read_json('quoteresult.jl', lines=True)
quotes
```

[9]:

	text	author	\
0	"The world as we have created it is a process …	Albert Einstein	
1	"It is our choices, Harry, that show what we t…	J.K. Rowling	
2	"There are only two ways to live your life. On…	Albert Einstein	
3	"The person, be it gentleman or lady, who has …	Jane Austen	
4	"Imperfection is beauty, madness is genius and…	Marilyn Monroe	
5	"Try not to become a man of success. Rather be…	Albert Einstein	
6	"It is better to be hated for what you are tha…	André Gide	
7	"I have not failed. I've just found 10,000 way…	Thomas A. Edison	
8	"A woman is like a tea bag; you never know how…	Eleanor Roosevelt	
9	"A day without sunshine is like, you know, nig…	Steve Martin	
10	"This life is what you make it. No matter what…	Marilyn Monroe	
11	"It takes a great deal of bravery to stand up …	J.K. Rowling	
12	"If you can't explain it to a six year old, yo…	Albert Einstein	

```
13  "You may not be her first, her last, or her on…          Bob Marley
14  "I like nonsense, it wakes up the brain cells…           Dr. Seuss
15  "I may not have gone where I intended to go, b…       Douglas Adams
16  "The opposite of love is not hate, it's indiff…         Elie Wiesel
17  "It is not a lack of love, but a lack of frien… Friedrich Nietzsche
18  "Good friends, good books, and a sleepy consci…          Mark Twain
19  "Life is what happens to us while we are makin…      Allen Saunders
```

```
                                                 tags
0            [change, deep-thoughts, thinking, world]
1                                [abilities, choices]
2      [inspirational, life, live, miracle, miracles]
3                  [aliteracy, books, classic, humor]
4                        [be-yourself, inspirational]
5                         [adulthood, success, value]
6                                      [life, love]
7       [edison, failure, inspirational, paraphrased]
8                    [misattributed-eleanor-roosevelt]
9                             [humor, obvious, simile]
10  [friends, heartbreak, inspirational, life, lov…
11                               [courage, friends]
12                          [simplicity, understand]
13                                           [love]
14                                        [fantasy]
15                               [life, navigation]
16  [activism, apathy, hate, indifference, inspira…
17  [friendship, lack-of-friendship, lack-of-love,…
18    [books, contentment, friends, friendship, life]
19  [fate, life, misattributed-john-lennon, planni…
```

Nota: Despues de encontrar dificultades para hacer el web scraping de algunas paginas conocidas, me he decidido por utilizar una web que se utiliza regularmente con fines de estudio del web scraping.

[ ]: