



Maestría en Explotación de Datos
y Descubrimiento del Conocimiento
Universidad de Buenos Aires

Web scraping

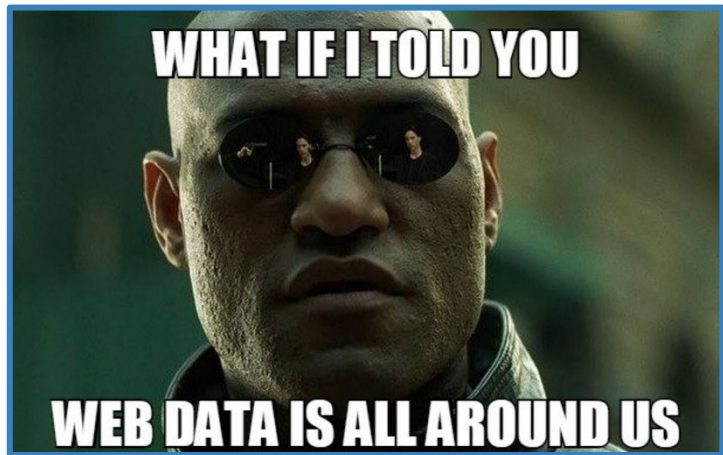


Gustavo Juantorena [[contacto](#)] * 26/08/21



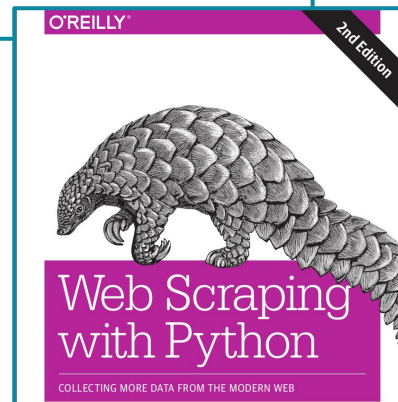
Hoja de ruta

1. ¿Qué es el web scraping?
2. ¿Qué son las APIs?
3. ¿Cuándo conviene usar cada cosa?
4. Breve introducción a las tecnologías web necesarias
5. Web Scraping con Python básico
6. Web scraping "avanzado"
7. Práctica en Google Colab



¿Qué es el web scraping?

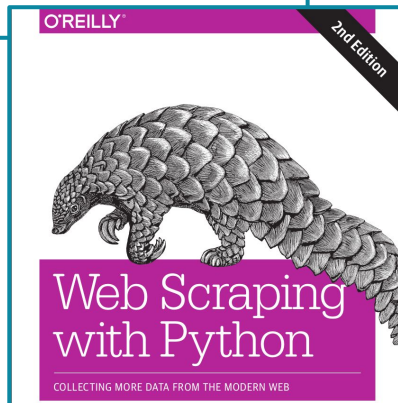
La práctica de **recopilar datos** a través de cualquier medio que no sea un programa que interactúa con una API o un humano que usa un navegador web. **Normalmente mediante un programa automatizado que consulta un servidor web**, solicita datos (generalmente en forma de HTML y otros archivos que componen las páginas web) y luego analiza esos datos para extraer la información necesaria.



¿Qué es el web scraping?

La práctica de **recopilar datos** a través de cualquier medio que no sea un programa que interactúa con una API o un humano que usa un navegador web. **Normalmente mediante un programa automatizado que consulta un servidor web**, solicita datos (generalmente en forma de HTML y otros archivos que componen las páginas web) y luego analiza esos datos para extraer la información necesaria.

El **web crawling** o indexación, se utiliza para indexar la información de la página mediante bots también conocidos como crawlers (lo que hacen los motores de búsqueda). Se trata de ver una página como un todo e indexarla. Cuando un bot rastrea un sitio web, recorre todas las páginas y todos los enlaces, hasta la última línea del sitio web, en busca de **CUALQUIER información**.



Disclaimer

Legality and Ethics of Web Scraping

Emergent Research Forum (ERF)

Vlad Krotov

Murray State University
vkrotov@murraystate.edu

Leiser Silva

University of Houston
lsivla@uh.edu

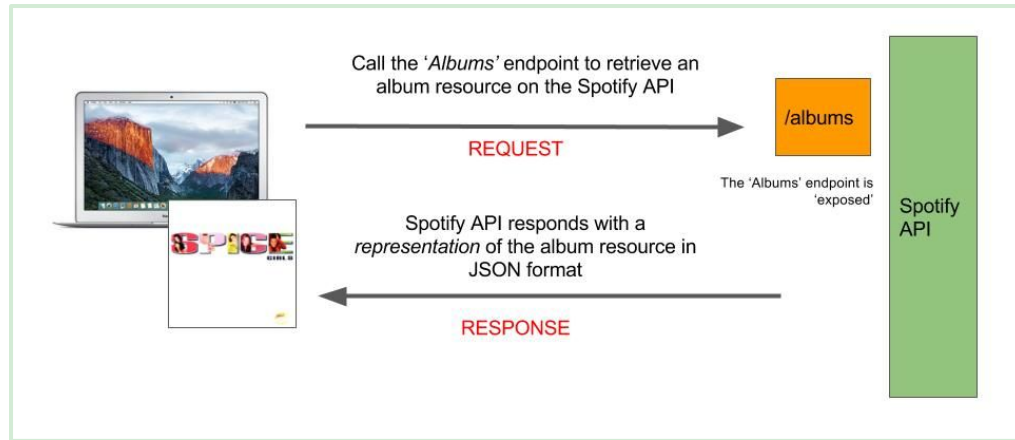
- Is Web Crawling or Web Scraping explicitly prohibited by the website's "terms of use" policy?
- Is the website's data explicitly copyrighted?
- Does the project involve illegal or fraudulent use of the data?
- Can crawling and scraping potentially cause material damage to the website or Web server hosting the website?
- Can the data obtained from the website compromise individual privacy?
- Can the data obtained from the website reveal confidential information about operations of the organizations providing data or the company owning the website?
- Can the project requiring the Web data potentially diminish the value of the service provided by the website?

Fuente:

https://www.researchgate.net/profile/Vlad-Krotov/publication/324907302_Legality_and_Ethics_of_Web_Scraping/links/5aea622345851588dd8287dc/Legality-and-Ethics-of-Web-Scraping.pdf

API (Interfaz de programación de aplicaciones)

Conjunto de funciones, métodos y protocolos de comunicación que ofrece un programa para ser utilizado por otro.



Nos provee el contenido con un determinado formato y especifica las limitaciones (ej: cantidad de pedidos al servidor).

Ejemplos de APIs



Developer Platform

Products ▾Use cases ▾Docs ▾Community ▾

Updates ▾Support ▾

Documentation

Search the docs

Twitter API

Getting started ▾TutorialsTools and librariesMigrateAPI reference index

the new Twitter API **v2**

Fundamentals ▾Tweets ▾Users ▾

API reference index

This resource is the Twitter API-specific API reference index. If you are looking for a list of endpoints from the entire platform, including Twitter Ads API and Labs, please visit the [platform API Reference Index](#).

Twitter API v2

Tweets

Filtered stream

- GET /2/tweets/search/stream
- GET /2/tweets/search/stream/rules
- POST /2/tweets/search/stream/rules

Hide replies

- PUT /2/tweets/:id/hidden

Likes

- DELETE /2/users/:id/likes/tweet_id
- GET /2/tweets/:id/liking_users



WIKIPEDIA
The Free Encyclopedia

Wikipedia API

Wikipedia-API is easy to use Python wrapper for [Wikipedias'](#) API. It supports extracting texts, sections, links, categories, translations, etc from Wikipedia. Documentation provides code snippets for the most common use cases.

build **passing** docs **passing** test coverage **96%** **python** **v0.5.4** **Py Versions** stars **255**

Installation

This package requires at least Python 3.4 to install because it's using IntEnum.

```
pip3 install wikipedia-api
```

Usage

Goal of `Wikipedia-API` is to provide simple and easy to use API for retrieving informations from Wikipedia. Below are examples of common use cases.

Importing

```
import wikipediaapi
```

How To Get Single Page

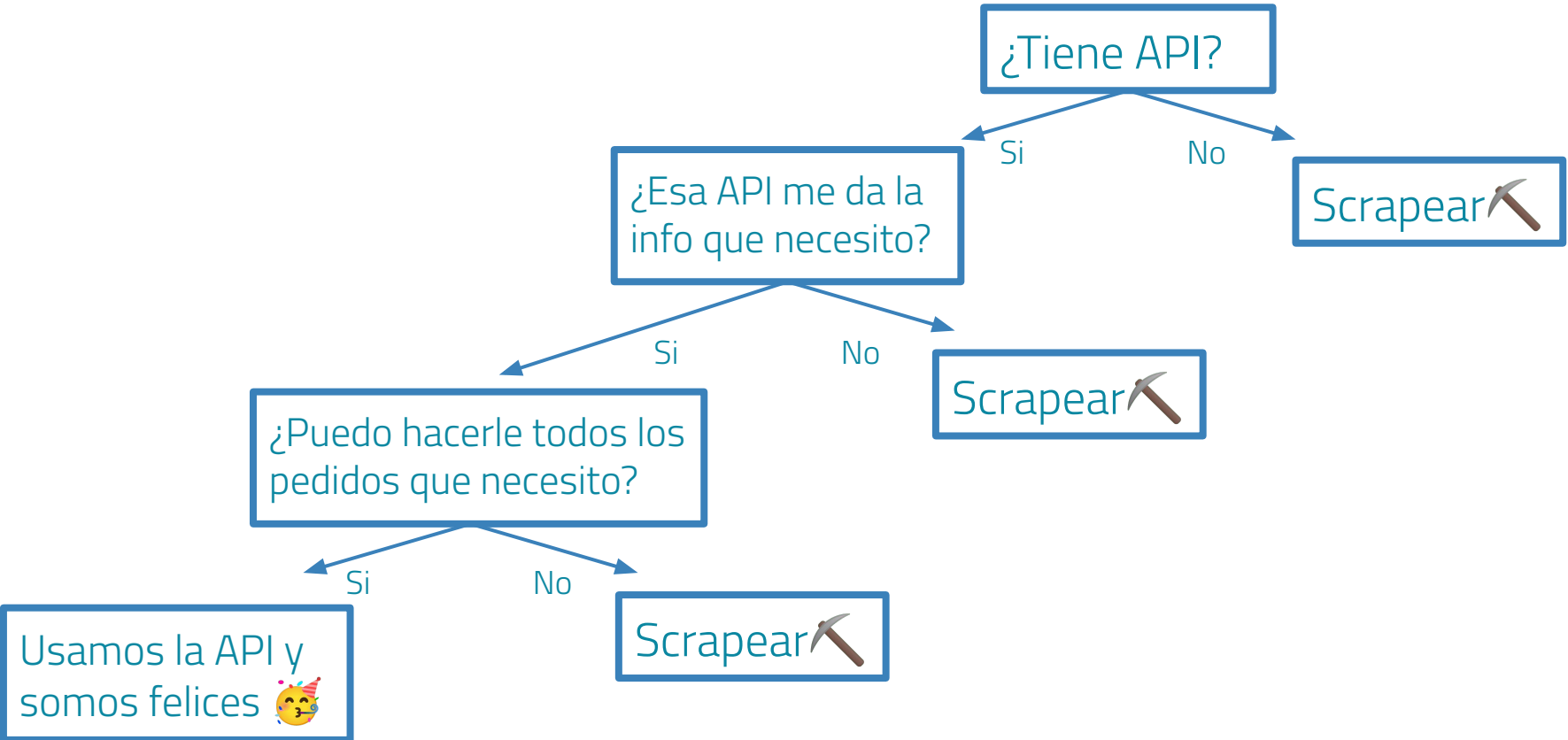
Getting single page is straightforward. You have to initialize `Wikipedia` object and ask for page by its name. It's parameter language has be one of [supported languages](#).

```
import wikipediaapi
wiki_wiki = wikipediaapi.Wikipedia('en')
page_py = wiki_wiki.page('Python_(programming_language)')
```

API vs Web scraping (I)

Web scraping	API
Extraer información de un sitio web usando un programa informático.	Proveer acceso a los datos de una aplicación, sistema operativo u otro servicio.
Mismo objetivo: Acceder a los datos del sitio web	

API vs Web scraping (II)



Conceptos básicos sobre la web (I)

- **HTTPS** (Protocolo seguro de transferencia de hipertexto): Protocolo de mensajería que permite a los navegadores web comunicarse con los servidores web (donde se almacenan los sitios web).

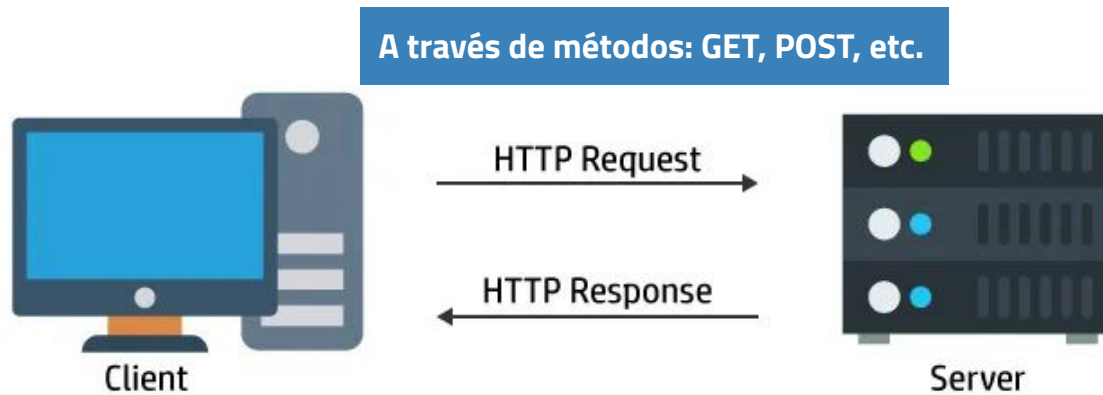


Fig: HTTP Protocol

Conceptos básicos sobre la web (II)

HTTP Status Codes		
Éxito	Errores del cliente	Errores del servidor
Level 200 (Success) 200 : OK 201 : Created 203 : Non-Authoritative Information 204 : No Content	Level 400 400 : Bad Request 401 : Unauthorized 403 : Forbidden 404 : Not Found 409 : Conflict 429: Too many requests	Level 500 500 : Internal Server Error 503 : Service Unavailable 501 : Not Implemented 504 : Gateway Timeout 599 : Network timeout 502 : Bad Gateway

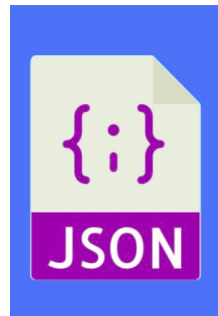
Fuente: <https://gabicuesta.blogspot.com/2019/01/http-status-codes.html>

Conceptos básicos sobre la web (III)



comma-separated values

```
album, año, ranking
The White Stripes, 1999, 2
De Stijl, 2000, 3
Lo mejor del amor, 1996, 1
```



Javascript Object Notation

```
[
  {
    "album": "The White Stripes",
    "año": 1999,
    "ranking": 2
  },
  {
    "album": "De Stijl",
    "año": 2000,
    "ranking": 3
  },
  {
    "album": "Lo mejor del amor",
    "año": 1996,
    "ranking": 1
  }
]
```

<https://csvjson.com/csv2json>

Conceptos básicos sobre la web (III)



comma-separated va

XML



Javascript Object Notation

```
album, año, ranking
The White Stripes, 1999, 2
De Stijl, 2000, 3
Lo mejor del amor, 1996, 1
```

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <element>
    <album>The White Stripes</album>
    <year number="true">1999</year>
    <ranking number="true">2</ranking>
  </element>
  <element>
    <album>De Stijl</album>
    <year number="true">2000</year>
    <ranking number="true">3</ranking>
  </element>
  <element>
    <album>Lo mejor del amor</album>
    <year number="true">1996</year>
    <ranking number="true">1</ranking>
  </element>
</root>
```

```
"The White Stripes",
99,
: 2
```

```
"De Stijl",
00,
: 3
```

```
"Lo mejor del amor",
96,
: 1
```

<https://csvjson.com/csv2json>

<https://www.utilities-online.info/xmltojson>

Conceptos básicos sobre la web (IV)

- **HTML, CSS y JavaScript son los tres lenguajes principales con los que está hecho la parte de la web que vemos.**



Conceptos básicos sobre la web (IV)

- HTML, CSS y JavaScript son los 3 lenguajes principales con los que estan hecho la parte de la web que vemos.



Estructura



Funcionalidad



Estilo



HTML: Lenguaje de marcado de hipertexto

CSS: Hojas de estilo en cascada

```
<!DOCTYPE html>
<html>
  <head>
    <title>Mi primer página</title>
  </head>
  <body>

    <h1>Título</h1>
    <h2 style='color:red;'> Subtitulo en rojo</h2>
    <p>Primer párrafo.</p>
    <hr>

    <h3>Gatito</h3>
    <img style='width: 100px;'
src='https://i.pinimg.com/originals/d4/8d/07/d48d074f8c
9ec8448612822295686754.jpg' >

  </body>
</html>
```

Título

Subtitulo en rojo

Primer párrafo.

Gatito



HTML: Lenguaje de marcado de hipertexto

CSS: Hojas de estilo en cascada

```
<!DOCTYPE html>
<html>
  <head>
    <title>Mi primer página</title>
  </head>
  <body>

    <h1>Título</h1>
    <h2 style='color:red;'> Subtitulo en rojo</h2>
    <p>Primer párrafo.</p>
    <hr>

    <h3>Gatito</h3>
    <img style='width: 100px;'
src='https://i.pinimg.com/originals/d4/8d/07/d48d074f8c
9ec8448612822295686754.jpg' >

  </body>
</html>
```

HTML TAG

CSS

HTML ATTRIBUTE

Título

Subtitulo en rojo

Primer párrafo.

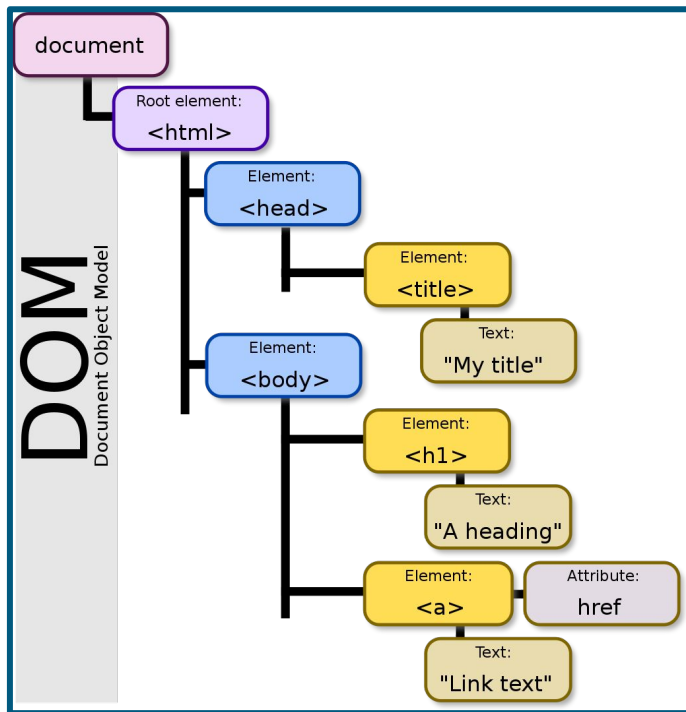
Gatito



[Ir a probar el código](#)

DOM (Document Object Model)

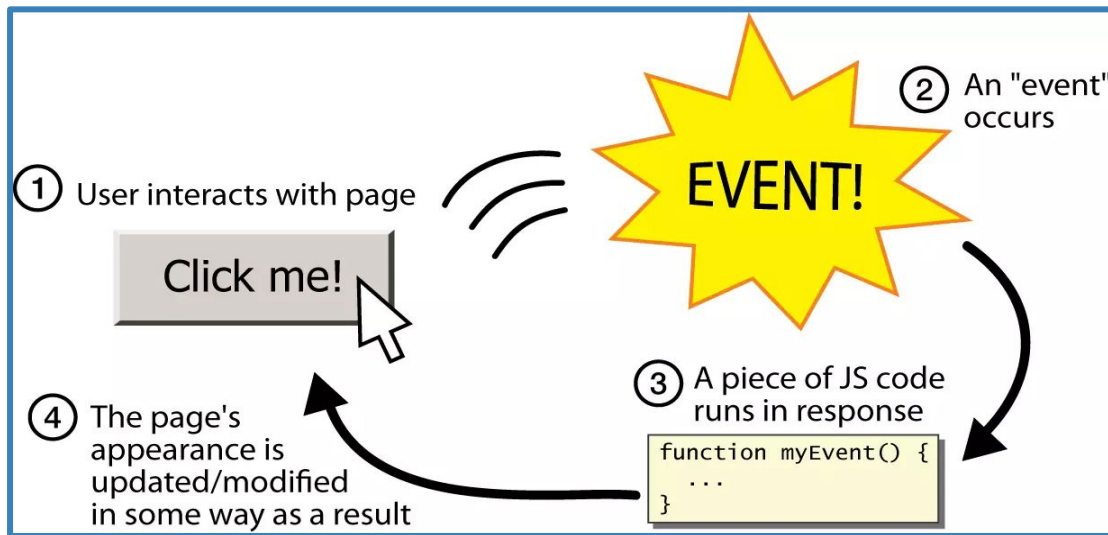
Interfaz independiente del lenguaje que trata un documento XML o HTML como una estructura de árbol



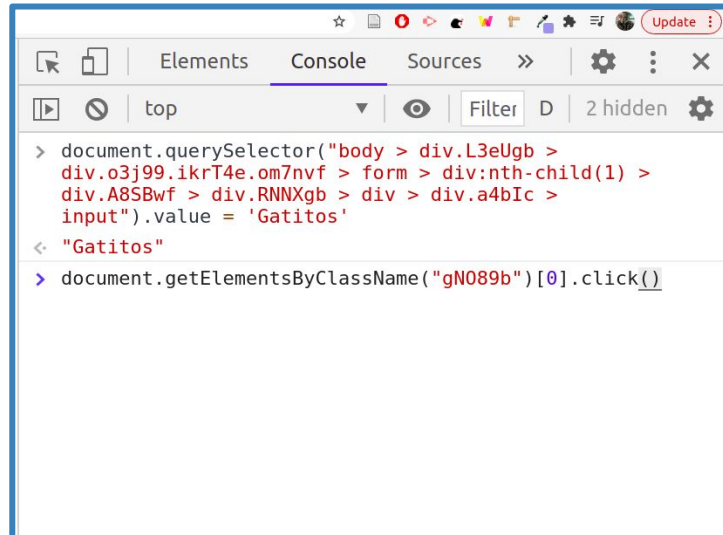
Fuente: https://en.wikipedia.org/wiki/Document_Object_Model
Autor: Birger Eriksson

JavaScript

- Permite usar un paradigma de programación orientado a eventos (entre otros).
- Más del 97% de las páginas web lo usan para generar comportamientos interactivos.



<https://itzone.com.vn/en/article/event-in-javascript/>



```
document.querySelector("body > div.L3eUgb > div.o3j99.ikrT4e.om7nvf >  
form > div:nth-child(1) > div.A8SBwf > div.RNNXgb > div > div.a4bIc >  
input").value = 'Gatitos'
```

```
document.getElementsByClassName("gN089b")[0].click()
```

Xpath (XML Path Language)

Para aprender más sobre el tema:
[The W3Schools XPath Tutorial](#)

- Es un lenguaje que permite construir expresiones que recorren y procesan un documento XML.
- La idea es parecida a las expresiones regulares para seleccionar partes de un texto sin atributos (plain text)

The screenshot shows a web browser displaying the Wikipedia article "Historia de la moda". The article text is visible on the left, and the right sidebar contains a list of "Actualidad" (Current events) and "Fallecimientos" (Deaths). The Chrome DevTools console is open on the right, showing the XPath query `//*[@id="main-tfa"]` selected. The query is highlighted in blue, and a blue arrow points to it from the text "Xpath query" on the right. The console also shows the results of the query, including the `body` element and its `style` property.

Xpath
query

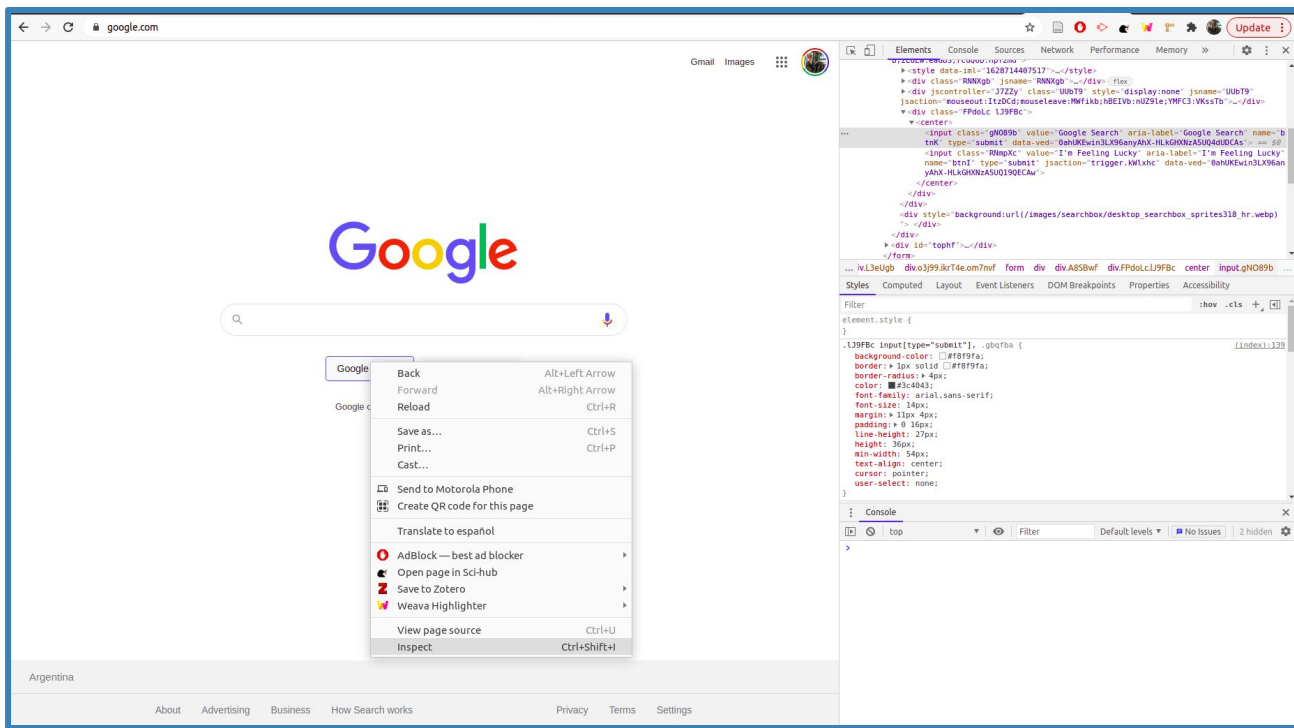
CSS selectors

Para aprender más sobre el tema:
[CSS Selectors](#)

- Los selectores CSS se utilizan para "buscar" (o seleccionar) los elementos HTML y aplicarles estilo.
- Podemos aprovecharlos para seleccionar lo que queramos scrapear.

The image shows a screenshot of the Wikipedia homepage in Spanish. The browser's developer tools are open, displaying the 'Elements' panel. A blue box highlights the element with the ID 'main-tfa' in the HTML tree. A blue arrow points from the text 'CSS selector' to this highlighted element. The 'Styles' panel shows the default styles for this element, including a width of 1000px and a height of 720px. The main content area of the page is visible, showing the 'Historia de la moda' article and a list of 'Actualidad' (Current events) items.

Uso de herramientas de desarrollador en el navegador



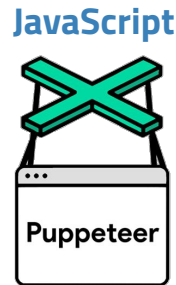
Practiquemos ...

¿¿¿Y el Python???



🐍 Web scraping con Python

- Python no es la única tecnología con la cual podemos hacer scraping (ej: R, JavaScript, Java, C++, etc)

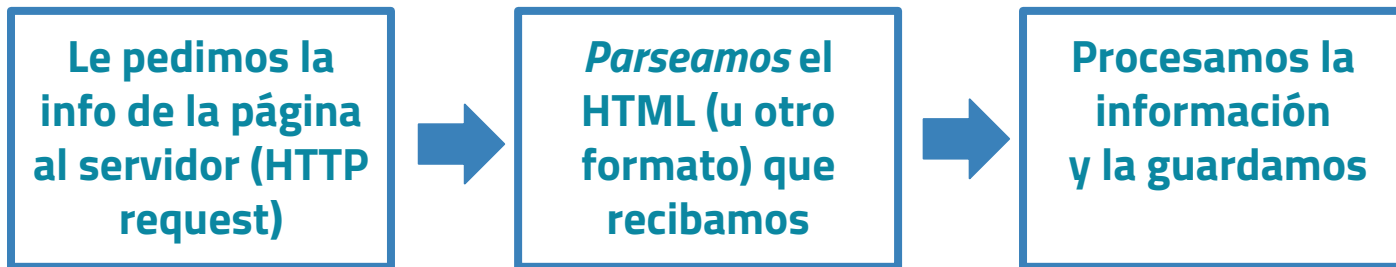


- Python nos puede ayudar en muchas cosas:
 - Pedidos HTTP (urllib, requests)
 - Parseo de la información (Beautiful Soup)
 - Automatización
 - Control de excepciones
 - etc

“Parsear” la información

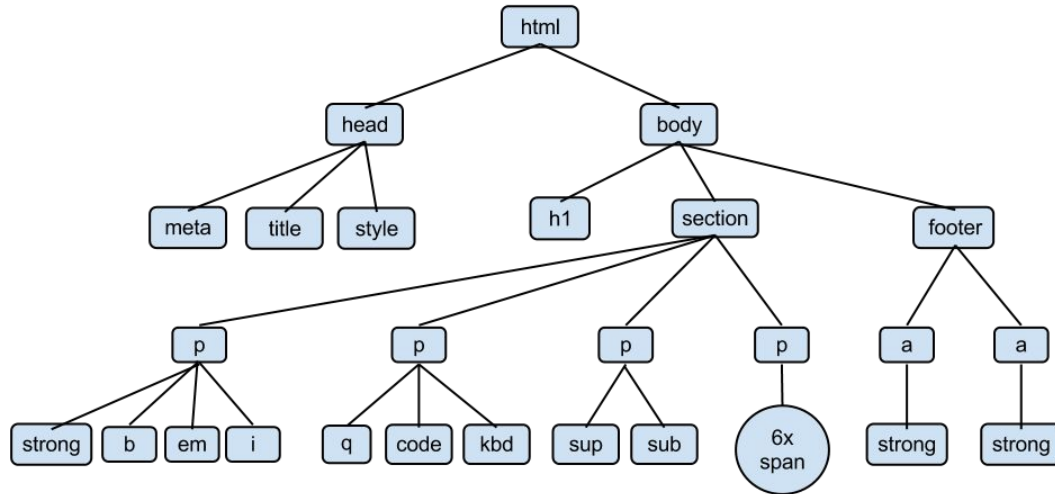
- Dividir un texto en sus componentes y describir sus roles sintácticos.
- El “parseo” de un documento HTML es básicamente tomar código HTML y extraer información relevante como el título de la página, párrafos en la página, encabezados en la página, enlaces, texto en negrita, etc.

Pipeline web scraping



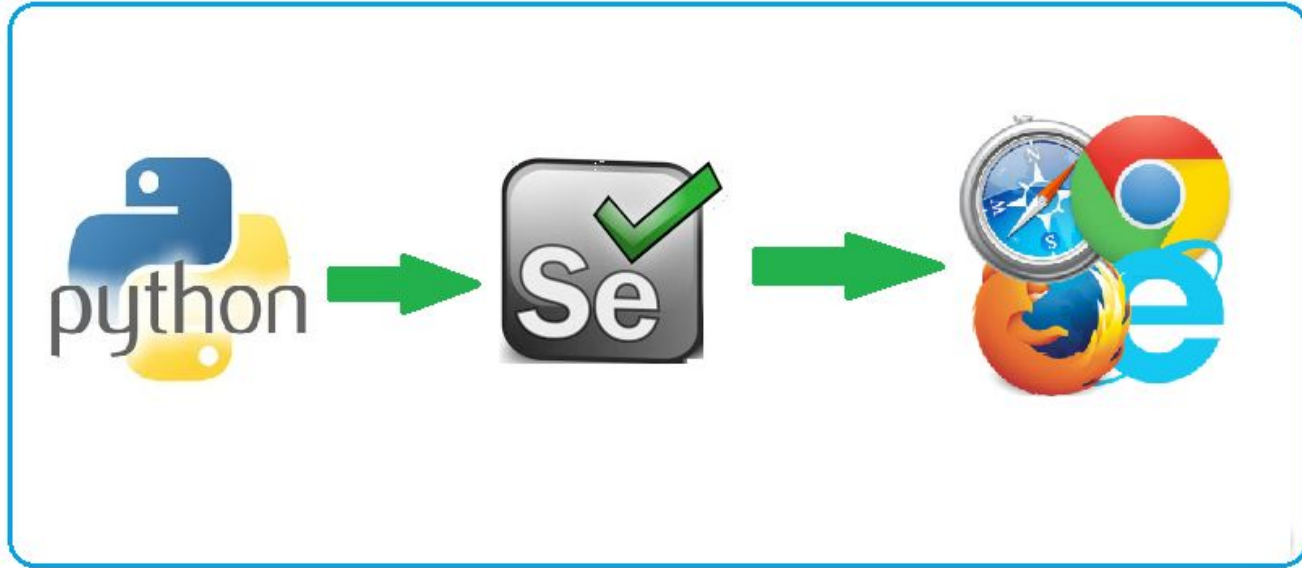
Parseando con Python: BeautifulSoup

- BeautifulSoup es una librería de Python para web scraping
- Se usa para extraer los datos de archivos HTML y XML. Crea un árbol de análisis a partir del código fuente de la página que se puede utilizar para extraer datos de forma jerárquica y más legible.



Fuente: <https://medium.com/miloooproject/python-simple-crawling-using-beautifulsoup-8247657c2de5>

Web Scraping “avanzado” / web crawling



Fuente: <https://medium.com/swlh/get-started-with-selenium-webdriver-in-under-5-minutes-f9b91e2e9539>

Selenium

- Es una herramienta de testing y automatización que tiene una API para Python (entre otros lenguajes)
- No fue pensado específicamente para web scraping ni web crawling, pero gracias al sistema cliente/servidor **Web Driver** permite utilizar un navegador de forma local o en remoto.
 - Esto nos da acceso a un navegador “headless” (sin interfaz gráfica) con el que podemos recorrer la web.
- ¿Qué viene a solucionar respecto a lo que vimos antes?
 - Páginas dinámicas
 - Scrolleo infinito
 - Completar formularios, autenticación, pop ups, captcha, etc ...

Referencias

<https://github.com/GEJ1/web-scraping-python>

Web scraping con Python

Recursos básicos para iniciarse en web scraping usando Python.

Clase dictada por [Gustavo Juantorena](#) como docente invitado en la materia **Text Mining** de la [Maestría en Explotación de Datos y Descubrimiento de Conocimiento](#) de la Universidad de Buenos Aires. El contenido aquí expuesto excede lo que se vió en a clase, pero me parece que está bueno que lo tengan como referencia para profundizar los contenidos.

- [Slides de la clase](#)
- [Notebook de la clase](#)

Recursos

- [Web general](#)
- [Web scraping general](#)
- [Beautiful Soup](#)
- [Selenium](#)
- [Scrapy](#)
- [Libros](#)

Ahora si: ¡Vamos a Colab!

