

# BIG DATA Y PYTHON

**MÁSTER EN BIG DATA**

201020

GABRIEL MARÍN DÍAZ

**hola**

# Presentación

---

---

Yo mismo

**Nombre:** Gabriel Marín Díaz

**A qué me dedico...**

- Channel Enablement Manager en Sage
- Profesor Asociado UCM

**Perfil de LinkedIn:** <https://www.linkedin.com/in/gabrielmarindiaz/>

# **CONTENIDO**

# Contenido

---

---

## Resumen

Tema 1 – Visión General

Tema 2 – Introducción a SQL

Tema 3 – Introducción al Lenguaje Python

Tema 4 – HTML y Python

Tema 5 – Big Data y Python

Tema 6 – Procesamiento Distribuido (Spark)

**Prácticas** las realizaremos con Python, MySQL, MongoDB, Apache Spark

**Arrancamos?**

**RECORDATORIO...**

# Ejercicio

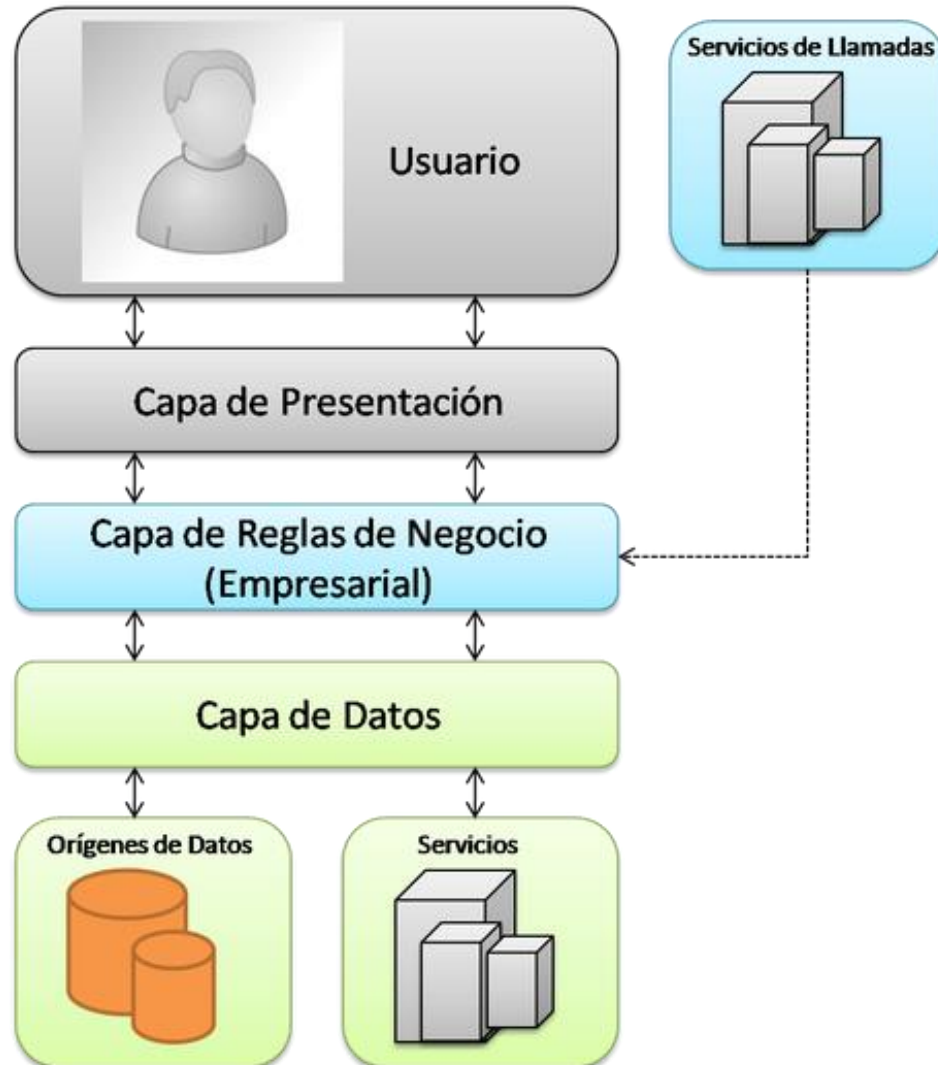
---

**OBTENER LA SERIE HISTÓRICA DE COVID 19  
EN FORMATO CSV, INTENTAR OBTENER EL  
INCREMENTO DIARIO DE CASOS  
DIAGNOSTICADOS, HOSPITALIZACIONES,  
UCI, FALLECIMIENTOS. LLEVAR EL  
RESULTADO A UN FORMATO EXCEL.**



**CONTEXTO**

# Arquitectura en capas



**1. Capa de Presentación:** Interacción entre el usuario y el software. Puede ser tan simple como un menú basado en líneas de comando o tan complejo como una aplicación basada en formas. Su principal función es mostrar información al usuario, interpretar los comandos de este y realizar algunas validaciones simples de los datos ingresados.

**2. Capa de Reglas de Negocio (Empresarial):** También denominada Lógica de Dominio, esta capa contiene la funcionalidad que implementa la aplicación. Involucra cálculos basados en la información dada por el usuario, datos almacenados y validaciones. Controla la ejecución de la capa de acceso a datos y servicios externos.

**3. Capa de Datos:** Esta capa contiene la lógica de comunicación con otros sistemas que llevan a cabo tareas por la aplicación. Para el caso de aplicaciones empresariales, está representado por una base de datos, que es responsable del almacenamiento persistente de información. Esta capa debe abstraer completamente a las capas superiores (negocio) del dialecto utilizado para comunicarse con los repositorios de datos (PL/SQL, Transact-SQL, etc.).

# Índice

☐ Lectura de Ficheros

☐ Web Scraping

☐ Uso de APIs

☐ Mongo DB

☐ Procesamiento Distribuido con SPARK

☐ Visualización de Resultados

**ANTES UNA COSA...**

# MySQL y Python

---

Vamos a ver en el siguiente enlace cómo interaccionan MySQL con Python.

Crearemos una BD en MySQL con el nombre de bdPython y a partir de ahí empezamos...



Click me!

**CONTINUAMOS CON PYTHON...**  
**WEB SCRAPING**

# Web Scraping

---

Existen tres posibilidades proporcionadas por las páginas web para ofrecernos información:

- a) Ficheros ya preparados que pueden descargarse directamente.
- b) El sitio ofrezca acceso a sus datos a través de un protocolo API-REST, al que podemos hacer peticiones sobre datos concretos, siendo el formato JSON o XML.
- c) La información forme parte de la propia página web, en un formato accesible para “humanos” que la visitan, pero complicada para un programa. Python proporciona una biblioteca para poder acceder a estos datos, es lo que llamamos ***web scraping***.

# Web Scraping

---

Dentro de esta última posibilidad, tenemos a su vez dos posibilidades de acceso a los datos:

- a) En la primera, la información está disponible simplemente accediendo a la URL. Emplearemos la biblioteca ***BeautifulSoup***.
- b) En otros casos la página requerirá de información por nuestra parte antes de mostrarnos los datos. En este caso utilizaremos la biblioteca ***Selenium***.



# Web Scraping

---

| Fuentes Web de Datos                | Biblioteca Python |
|-------------------------------------|-------------------|
| Ficheros incluidos en la página web | requests, csv...  |
| API-REST                            | Requests          |
| Datos que forman parte de la página | BeautifulSoup     |
| Datos que requieren interacción     | Selenium          |

# Web Scraping

## FREE PYTHON TOOLS

|  |  |
|--|--|
| <b>Pandas</b> - used for data analysis                     | <b>NumPy</b> - multidimensional arrays                     |
| <b>SciPy</b> - algorithms to use with numpy                | <b>Matplotlib</b> - data visualization tool                |
| <b>HDF5</b> - used to store and manipulate data            | <b>PyTables</b> - used for managing HDF5 datasets          |
| <b>Jupyter</b> - research collaboration tool               | <b>IPython</b> - powerful shell                            |
| <b>HDFS</b> - C/C++ wrapper for Hadoop                     | <b>Pymongo</b> - MongoDB driver                            |
| <b>SQLAlchemy</b> - Python SQL Toolkit                     | <b>Redis</b> - Redis' access libraries                     |
| <b>pyMySQL</b> - MySQL connector                           | <b>Scikit-learn</b> - used for machine learning algorithms |
| <b>Theano</b> - deep learning with neural networks         | <b>Keras</b> - high-level neural networks API              |
| <b>Lasagne</b> - build and train neural networks in Theano | <b>Bokeh</b> - data visualization tool                     |
| <b>Seaborn</b> - data visualization tool                   | <b>Dask</b> - data engineering tool                        |
| <b>Airflow</b> - data engineering tool                     | <b>Luigi</b> - data engineering tool                       |
| <b>Elasticsearch</b> - data search engine                  | <b>SymPy</b> - symbolic math                               |
| <b>PyBrain</b> - algorithms for ML                         | <b>Pattern</b> - natural language processing               |

# Web Scraping

---

**VAMOS A REALIZAR LA LECTURA DE DATOS  
DE LA SIGUIENTE URL...**

**<http://www.mambiente.munimadrid.es/opendata/horario.txt>**

**PROPORCIONA LOS DATOS DE LAS  
ESTACIONES DE MEDICIÓN DE LA CALIDAD  
DEL AIRE DE MADRID**

Click me!

# Web Scraping

## EJERCICIO 1

Y COMO SABÉIS MUCHO DE PYTHON... OS PROPONGO EL SIGUIENTE EJERCICIO PARA HACER AHORA.

- Las columnas 0, 1 y 2 identifican la estación meteorológica.
- Las columnas 3, 4 y 5 identifican el valor medido (12 es óxido de nitrógeno).
- Las columnas 6, 7 y 8 identifican el año, mes y día de la medición.
- Las columnas 9 – 56 identifican el valor en cada hora del día, van por parejas, el primer número indica la medición y el segundo es “V” si es un valor válido o “N” en caso contrario. En la práctica, el primer valor “N” corresponde con la hora actual.

Usaremos esta información para mostrar los datos en formato de gráfico que indique la evolución de la contaminación por óxido de nitrógeno en un día a partir de los datos de la primera estación “28079008”.  
**Tenemos 30 minutos para realizar el ejercicio.**

# Web Scraping

---

## EJERCICIO 1

# SOLUCIÓN...

Tenemos 30 minutos para realizar el ejercicio.

Click me!

# Web Scraping

## DATOS QUE FORMAN PARTE DE UNA PÁGINA WEB

Lo que vemos en el navegador es el resultado de interpretar el código HTML de la página. Un ejemplo:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Título de ejemplo</title>
  </head>
  <body>
    <div id="date"> Fecha 08/03/2020 </div>
    <div id="content"> Esta es mi página WEB.... </div>
  </body>
</html>
```

# Web Scraping

## DATOS QUE FORMAN PARTE DE UNA PÁGINA WEB

`<head> ... </head>`: describe la cabecera del documento.

`<body> ... </body>`: es el contenido en sí de la página (lo que vamos a examinar).

Dentro del elemento `<body>` existirán otros elementos, que a su vez pueden contener otros elementos...

Vamos a grabar el código anterior en un fichero con el nombre ejemplo.html, a continuación cargaremos la página como un fichero de texto normal, y la mostraremos mediante la biblioteca *BeautifulSoup*.

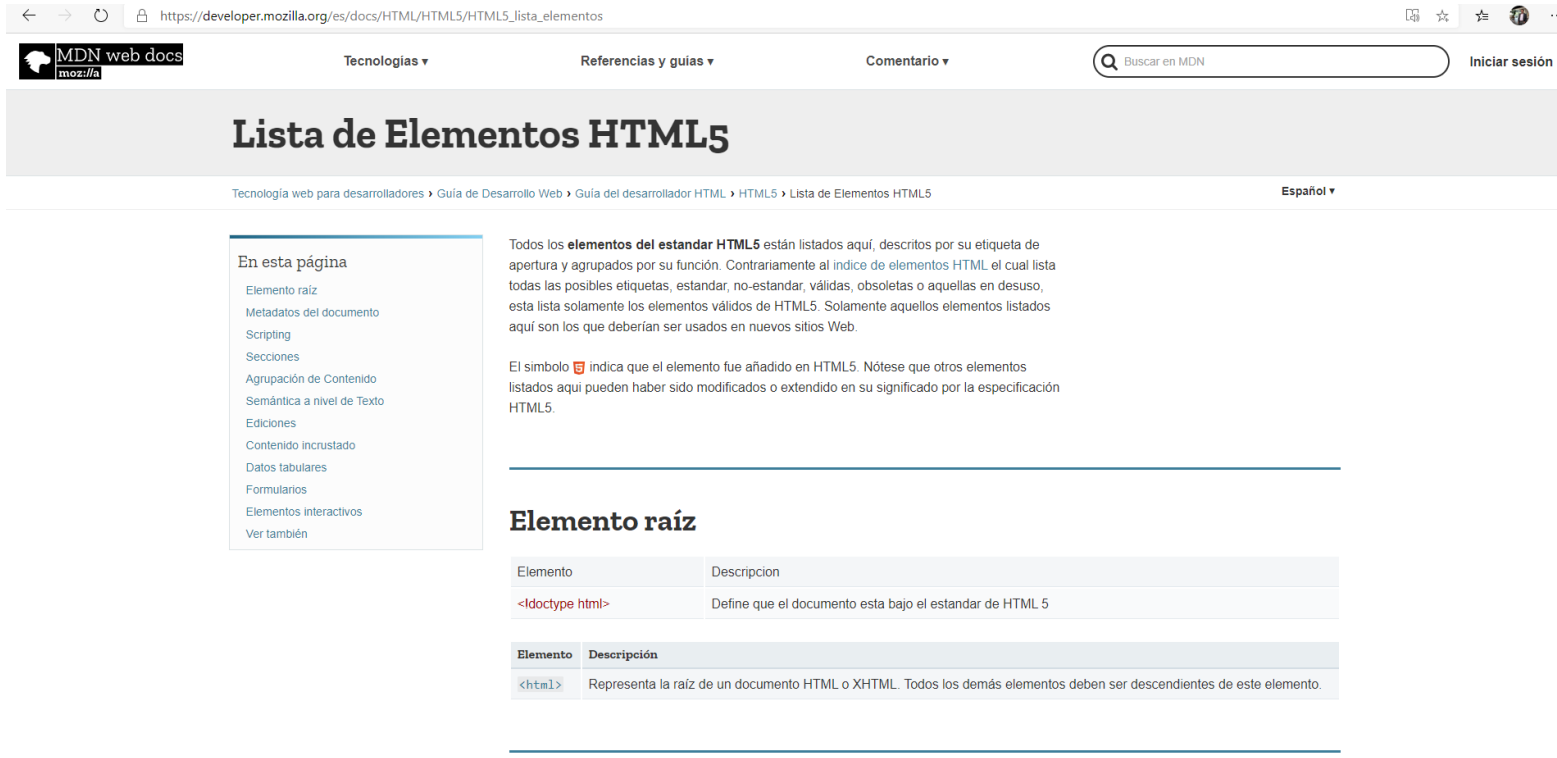


Click me!

# Web Scrapping

## DATOS QUE FORMAN PARTE DE UNA PÁGINA WEB

Para hacer web scrapping debemos identificar los elementos fundamentales que se pueden encontrar en un documento *html*... Lista de elementos:



The screenshot shows the MDN web docs page for 'Lista de Elementos HTML5'. The page has a navigation bar with 'Tecnologías', 'Referencias y guías', and 'Comentario'. A search bar and 'Iniciar sesión' link are also present. The main heading is 'Lista de Elementos HTML5'. Below it, a breadcrumb trail reads: 'Tecnología web para desarrolladores > Guía de Desarrollo Web > Guía del desarrollador HTML > HTML5 > Lista de Elementos HTML5'. A language selector shows 'Español'. On the left, a sidebar titled 'En esta página' lists various topics. The main content area explains that the page lists standard HTML5 elements and their functions. It also mentions that a red square icon indicates elements added in HTML5. Below this, a section titled 'Elemento raíz' contains a table with two rows: one for '<!doctype html>' and one for '<html>'. The table has columns 'Elemento' and 'Descripción'.

| Elemento        | Descripción   |
|-----------------|---|
| <!doctype html> | Define que el documento esta bajo el estandar de HTML 5 |

| Elemento | Descripción  |
|----------|--|
| <html>   | Representa la raíz de un documento HTML o XHTML. Todos los demás elementos deben ser descendientes de este elemento. |





# Web Scraping

## EJERCICIO 2

Supongamos que estamos desarrollando una aplicación en la que en cierto punto es fundamental conocer la fecha y hora oficiales. Podríamos obtener esta información directamente desde el ordenador, pero no queremos equivocarnos, así que nos conectaremos al BOE.

[https://www.boe.es/informacion/hora\\_oficial.php](https://www.boe.es/informacion/hora_oficial.php)

Que nos da la hora oficial en la península. Vamos a ver cómo podemos extraer esta información...

Venga que tenemos 20 minutos para resolverlo...

# Web Scraping

---

## EJERCICIO 2

# SOLUCIÓN...

Tenemos 20 minutos para realizar el ejercicio.

Click me!

# Web Scraping

## DATOS QUE REQUIEREN DE INTERACCIÓN

*BeautifulSoup* es una biblioteca excelente, ayuda a recuperar de forma cómoda datos de aquellas páginas que nos ofrecen directamente la información buscada. La idea es sencilla, nos descargamos el código html de la página y a continuación se navega a través de ella.

A menudo nos encontramos con páginas que nos solicitan información que debemos completar antes de mostrarnos el resultado deseado, exigen cierta interacción por nuestra parte. Esto no lo podemos hacer con *BeautifulSoup*.

Utilizaremos para estos casos la biblioteca *Selenium*.

# Web Scraping

## DATOS QUE REQUIEREN DE INTERACCIÓN

Voy a utilizar para estos ejercicios el editor por defecto de Python IDLE.

... Deberemos primero instalar el paquete Selenium...

```
Python -m pip install selenium
```

La segunda parte consiste en instalar el cliente del navegador, yo usaré Google Chrome

<https://github.com/SeleniumHQ/selenium/wiki/ChromeDriver>

Se descomprime el fichero y en la variable Path del sistema se incorpora el acceso al directorio...



Click me!

# Web Scraping

## DATOS QUE REQUIEREN DE INTERACCIÓN

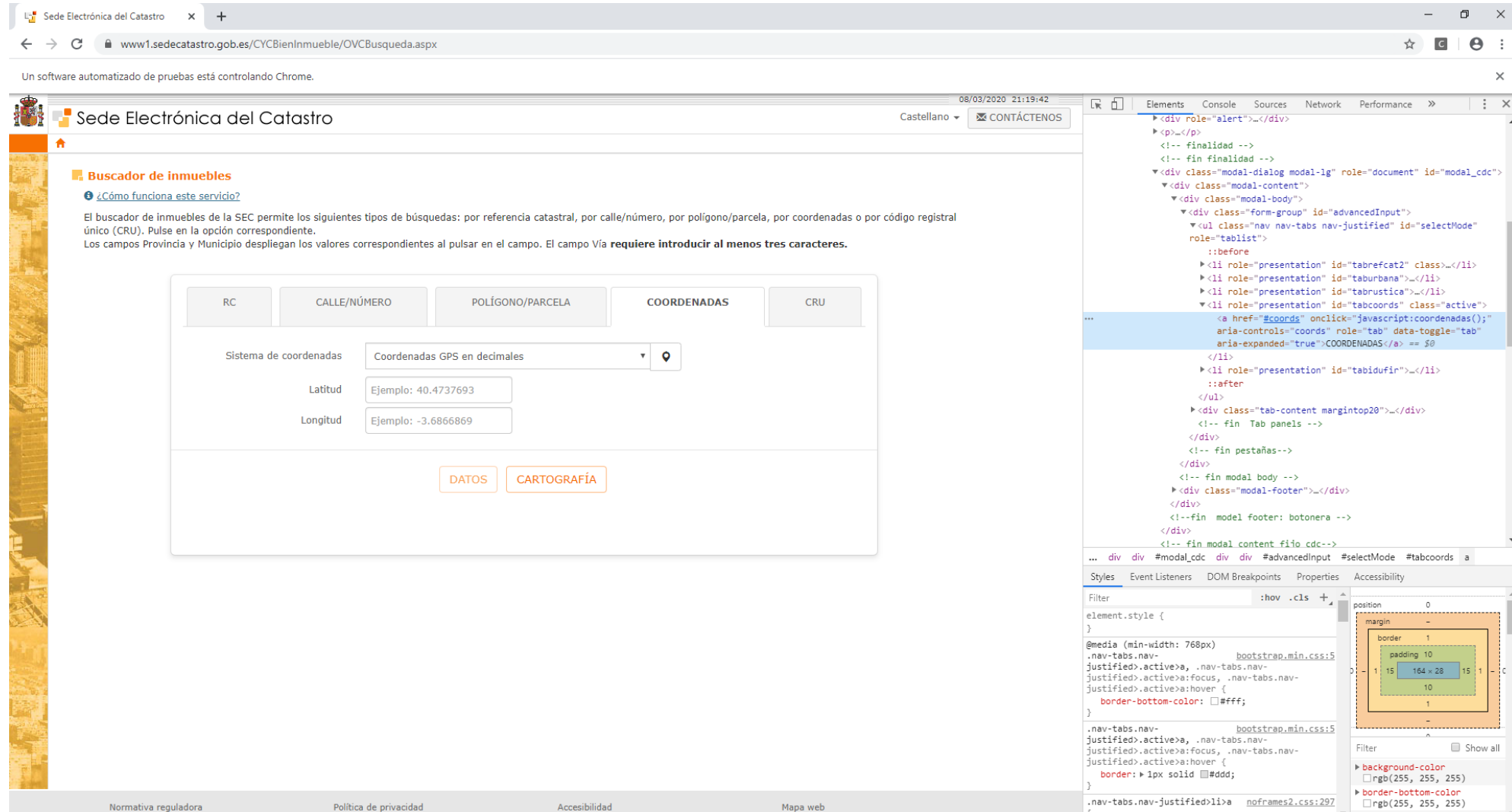
A continuación ejecutamos el siguiente código:

```
>>> from selenium import webdriver  
>>> driver=webdriver.Chrome()  
>>> url="https://www1.sedecatastro.gob.es/CYCBienInmueble/OVCBusqueda.aspx"  
>>> driver.get(url)
```

Si todo es correcto se mostrará la página del catastro. A continuación nos situamos en la pestaña COORDENADAS y pulsamos con el botón derecho la opción de inspeccionar.

# Web Scraping

## DATOS QUE REQUIEREN DE INTERACCIÓN



The screenshot shows a web browser displaying the 'Sede Electrónica del Catastro' website. The page title is 'Sede Electrónica del Catastro' and the URL is 'www1.sedecatastro.gob.es/CYCBienInmueble/OVCBusqueda.aspx'. The page content includes a search bar with tabs for 'RC', 'CALLE/NÚMERO', 'POLÍGONO/PARCELA', 'COORDENADAS', and 'CRU'. The 'COORDENADAS' tab is selected, showing a form for entering coordinates. The form includes a dropdown for 'Sistema de coordenadas' (set to 'Coordenadas GPS en decimales'), and input fields for 'Latitud' (Ejemplo: 40.4737693) and 'Longitud' (Ejemplo: -3.6866869). Below the form are buttons for 'DATOS' and 'CARTOGRAFÍA'.

Overlaid on the right side of the browser window is the Chrome DevTools 'Elements' panel. It shows the HTML structure of the page, specifically a modal dialog with a tabbed interface. The selected element is a tab with the role 'tab' and the text 'COORDENADAS'. The 'Styles' panel on the right shows the default Bootstrap styles for this element, including a border and padding.

# Web Scraping

## DATOS QUE REQUIEREN DE INTERACCIÓN

A continuación ejecutamos el siguiente código:

```
>>> coord = driver.find_element_by_id("tabcoords")  
>>> coord.click()
```

Ya estamos listos para introducir las coordenadas, para ello repetimos el mismo proceso, nos situamos en la casilla de latitud y pulsamos el botón derecho seleccionando la opción de inspeccionar.

# Web Scraping

## DATOS QUE REQUIEREN DE INTERACCIÓN

The screenshot displays the 'Sede Electrónica del Catastro' website. The main section is the 'Buscador de inmuebles' (Property Searcher). It includes a header with the site name and a 'CONTÁCTENOS' button. Below the header, there's a section titled '¿Cómo funciona este servicio?' (How this service works) explaining the search criteria: by cadastral reference, street/number, polygon/parcel, coordinates, or registration code (CRU). It also mentions that the 'Provincia' and 'Municipio' fields are dropdowns, and the 'Vía' field requires at least three characters.

The search form has five tabs: 'RC', 'CALLE/NÚMERO', 'POLÍGONO/PARCELA', 'COORDENADAS', and 'CRU'. The 'COORDENADAS' tab is active. It contains a 'Sistema de coordenadas' (Coordinate system) dropdown set to 'Coordenadas GPS en decimales'. Below this are input fields for 'Latitud' (Latitude) with the example '40.4737693' and 'Longitud' (Longitude) with the example '-3.6866869'. At the bottom of the form are two buttons: 'DATOS' and 'CARTOGRAFÍA'.

On the right side, the browser's developer tools are open, showing the 'Elements' panel. It displays the HTML structure of the form, highlighting the 'input' element for the latitude field. The 'Styles' panel is also open, showing the default styles for the 'input' element, including 'font-size: 12px', 'padding: 3px 6px', and 'height: 28px'. A visual representation of the element's box model is shown, with a width of 173px and a height of 26px.



# Web Scraping

## DATOS QUE REQUIEREN DE INTERACCIÓN

A continuación ejecutamos el siguiente código:

```
>>> lat = driver.find_element_by_id("ctl00_Contenido_txtLatitud")  
>>> lon = driver.find_element_by_id("ctl00_Contenido_txtLongitud")
```

Ahora podemos introducir la latitud y la longitud utilizando el método `send_keys()`.

```
>>> latitud = "28.2723368"  
>>> longitud = "-16.6600606"  
>>> lat.send_keys(latitud)  
>>> lon.send_keys(longitud)
```

# Web Scraping

## DATOS QUE REQUIEREN DE INTERACCIÓN

Un software automatizado de pruebas está controlando Chrome.

Sede Electrónica del Catastro

Castellano [CONTÁCTENOS](#)

**Consulta y certificación de Bien Inmueble**




[Volver](#)

[CARTOGRAFÍA](#)

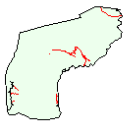
[CONSULTA DESCRIPTIVA Y GRÁFICA](#)

[IMPRIMIR DATOS](#)

**DATOS DESCRIPTIVOS DEL INMUEBLE**

|   |  |
|---|--|
| Referencia catastral  | 38026A035000010000EI   |
| Localización  | Polígono 35 Parcela 1 PARQUE NACINAL DEL TEIDE MONTE. LA OROTAVA (S.C. TENERIFE)   |
| Clase   | Rústico  |
| Uso principal   | Agrario  |
| Superficie construida  | 3.704 m <sup>2</sup>   |
| Año construcción  | 1965   |

**PARCELA CATASTRAL**

 Parcela construida sin división horizontal

|                    |  |
|--------------------|--|
| Localización       | Polígono 35 Parcela 1 PARQUE NACINAL DEL TEIDE MONTE. LA OROTAVA (S.C. TENERIFE) |
| Superficie gráfica | 73.631.526 m <sup>2</sup>  |

**CONSTRUCCIÓN**

| Uso principal | Escalera | Planta | Puerta | Superficie m <sup>2</sup> | Tipo Reforma | Fecha Reforma |
|---------------|----------|--------|--------|---------------------------|--------------|---------------|
| AGRARIO       |          |        |        | 40                        |              |               |
| OCIO HOSTEL.  | 00       | 01     |        | 328                       |              |               |
| OCIO HOSTEL.  |          | 00     | 02     | 360                       |              |               |
| OCIO HOSTEL.  |          | 00     | 03     | 265                       |              |               |
| VIVIENDA      |          | 00     | 01     | 309                       |              |               |
| ALMACEN       |          | 00     | 01     | 117                       |              |               |

# Web Scraping

## EJERCICIO 3

Para que se muestre la página anterior es necesario hacerlo por código...

- Introducir las líneas de código necesarias para ir a esta página.
- Investigar cómo podemos volver a una página anterior.
- Localizar elementos, investigar además de la función `find_element_by_id()`, otros atributos que pueden ser utilizados por Selenium.
- Utilizar `find_element_by_XXXXX` (cualquier otro atributo) para repetir el ejercicio anterior.
- Revisar cómo se puede utilizar XPath para acceder a una ruta mediante consultas.
- Investigar el funcionamiento del componente "/" en XPath.
- Investigar el funcionamiento del componente "\*" en XPath.
- Investigar el funcionamiento del componente "." en XPath.
- Investigar el funcionamiento del componente "//" en XPath.
- Investigar el funcionamiento de los filtros [...] en XPath.

Todo esto aplicarlo al ejemplo del catastro.

# Web Scraping

---

DATOS QUE REQUIEREN DE INTERACCIÓN

PRESENTAR EL TRABAJO HECHO ...  
TENEMOS 10 MINUTOS PARA CADA UNO

# **EJERCICIOS**

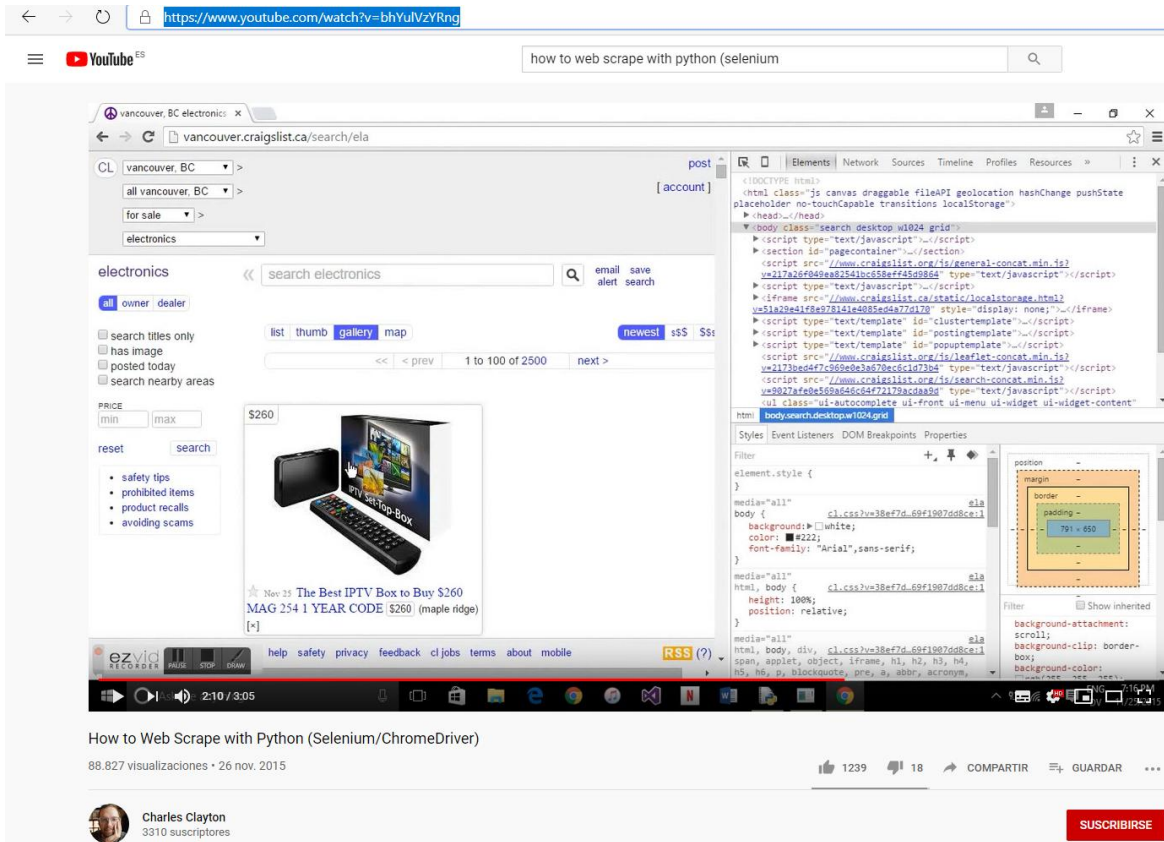
# Ejercicio

---

**OBTENER LA SERIE HISTÓRICA DE COVID 19  
EN FORMATO CSV, INTENTAR OBTENER EL  
INCREMENTO DIARIO DE CASOS  
DIAGNOSTICADOS, HOSPITALIZACIONES,  
UCI, FALLECIMIENTOS. LLEVAR EL  
RESULTADO A UN FORMATO EXCEL.**

# Ejercicios

Revisar el siguiente vídeo... hacer lo mismo con una página web de compras en español, tenemos una hora para hacerlo y presentarlo.



How to Web Scrape with Python (Selenium/ChromeDriver)

88.827 visualizaciones • 26 nov. 2015

Charles Clayton  
3310 suscriptores

SUSCRIBIRSE

Click me!

# !Muchas Gracias!

---

GABRIEL MARÍN DÍAZ  
LCDO. CIENCIAS FÍSICAS UCM

[www.linkedin.com/in/gabrielmarindiaz/](http://www.linkedin.com/in/gabrielmarindiaz/)