Web Scraping

Introducción, HTML

Mucha información está simplemente accesible en Internet. En este pequeño script vemos que es fácil acceder a ella, recopilarla, buscarla, procesarla.

El método más usado por los humanos para acceder a esa infomración de Internet es navegando mediante un navegador. El *Web Scraping* es la extracción de información que hay en sitios web, mediante programas de software

El contenido de las páginas web está escrito en archivos HTML. Veamos uno como ejemplo. En él, la información se organiza jerárquicamente: los distintos bloques de información están contenidos unos en otros, y se marcan con etiquetas. Mejor que cualquier explicación es verlo examinando el código fuente de cualquier página web. Empezamos con un archivo html muy pequeño:

In [1]:
```
pequenno_codigo_html = """
<html>
    <head>
        <title>
            A web page
        </title>
    </head>
    <body>
        <p id="author">Joel Grus</p>
        <p id="subject">Data Science</p>
    </body>
</html>
"""

print(pequenno_codigo_html)
```

```
<html>
    <head>
        <title>
            A web page
        </title>
    </head>
    <body>
        <p id="author">Joel Grus</p>
        <p id="subject">Data Science</p>
    </body>
</html>
```

BeautifulSoup

Empezamos por ver cómo se puede manejar un fragmento de texto html cualquiera, tanto si viene de Internet o si lo tenemos en nuestro equipo.

Para manejar la estructura de un archivo html, vamos a usar la librería BeautifulSoup. Esto requiere instalar el paquete bs4. Veamos las operaciones básicas.

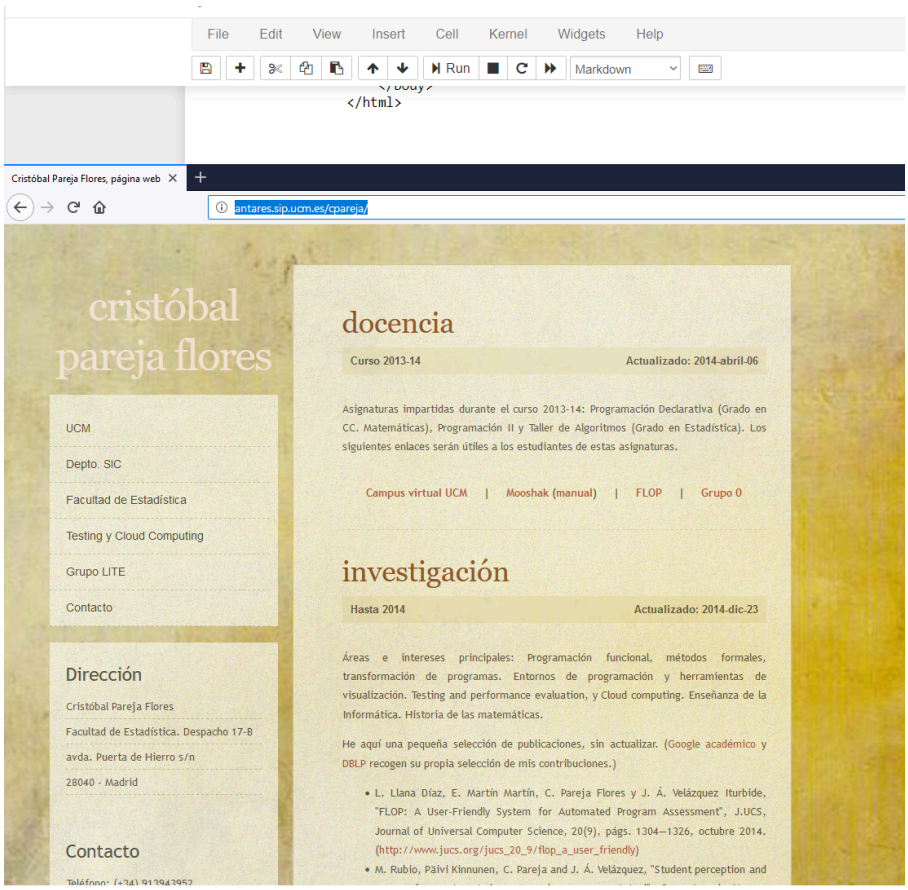
In [2]:
```
from bs4 import BeautifulSoup

soup_pequenno_codigo = BeautifulSoup(pequenno_codigo_html, "lxml")
primer_parrafo = soup_pequenno_codigo.find('p')

print(primer_parrafo.text)
print("-----------------------")
print(primer_parrafo.text.split())
print(primer_parrafo["id"])
todos_los_parrafos = soup_pequenno_codigo.find_all('p')
print(todos_los_parrafos)
print(todos_los_parrafos[0].text)
```

```
Joel Grus
-----------------------
['Joel', 'Grus']
author
[<p id="author">Joel Grus</p>, <p id="subject">Data Science</p>]
Joel Grus
```

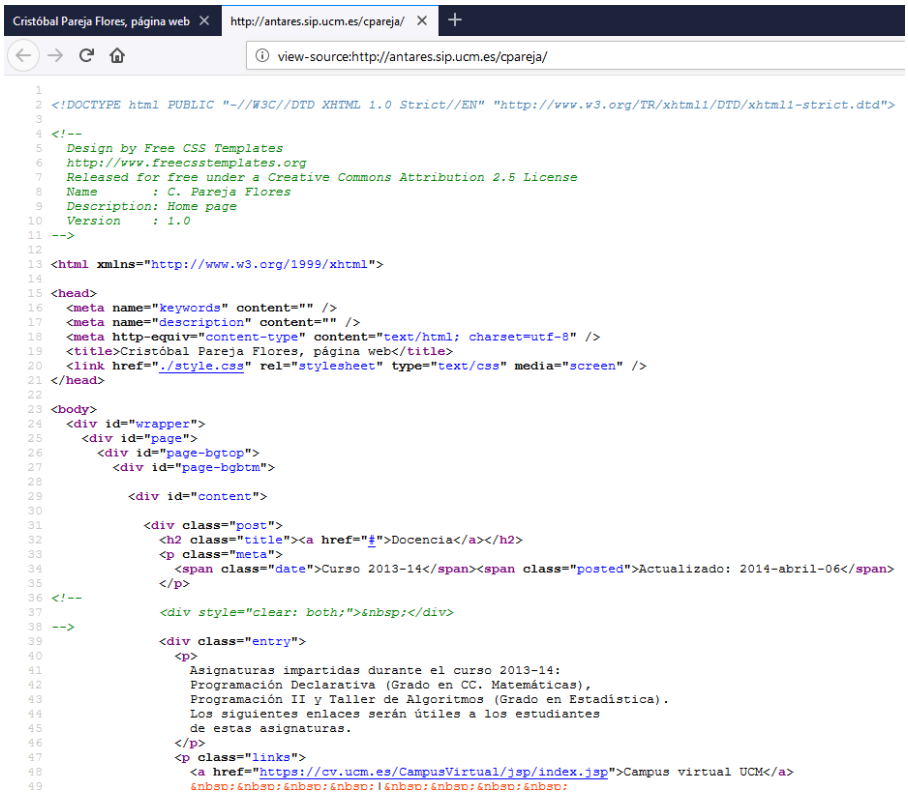
Vamos ahora con una página web real:

`http://antares.sip.ucm.es/cpareja/`

Examínala. Verás lo siguiente:



Cada navegador ofrece sus propio modo de examinar el codigo fuente en html. En el mío, pulsando el botón derecho se pued seleccionar la opción *Ver el código fuente de la página*, y tenemos lo siguiente:




```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!--
  Design by Free CSS Templates
  http://www.freecsstemplates.org
  Released for free under a Creative Commons Attribution 2.5 License
  Name        : C. Pareja Flores
  Description: Home page
  Version     : 1.0
-->

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
  <meta name="keywords" content="" />
  <meta name="description" content="" />
  <meta http-equiv="content-type" content="text/html; charset=utf-8" />
  <title>Cristóbal Pareja Flores, página web</title>
  <link href="./style.css" rel="stylesheet" type="text/css" media="screen" />
</head>

<body>
  <div id="wrapper">
    <div id="page">
      <div id="page-bgtop">
        <div id="page-bgbtm">

          <div id="content">

            <div class="post">
              <h2 class="title"><a href="#">Docencia</a></h2>
              <p class="meta">
                <span class="date">Curso 2013-14</span><span class="posted">Actualizado: 2014-abril-06</span>
              </p>
<!--
              <div style="clear: both;"> </div>
-->
              <div class="entry">
                <p>
                  Asignaturas impartidas durante el curso 2013-14:
                  Programación Declarativa (Grado en CC. Matemáticas),
                  Programación II y Taller de Algoritmos (Grado en Estadística).
                  Los siguientes enlaces serán útiles a los estudiantes
                  de estas asignaturas.
                </p>
                <p class="links">
                  <a href="https://cv.ucm.es/CampusVirtual/jsp/index.jsp">Campus virtual UCM</a>
                      |    
```

Para descargar páginas html de Internet, necesitamos la librería `requests` de Python.

Esta página que acabamos de abrir es más larga y completa que la del ejemplo anterior, así que una vez descargado su código fuente no vamos a mostrarla entera; nosotros optamos por ver únicamente su principio y su final:

In [3]:
```
import requests
requests.packages.urllib3.disable_warnings() # Prueba sin esta opción

mi_url = "https://antares.sip.ucm.es/cpareja/"
mi_codigo_html = requests.get(mi_url, verify=False).text   # Prueba sin
print(mi_codigo_html[:250])
print(".....................................")
print(mi_codigo_html[-250:])
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w
3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!--
  Design by Free CSS Templates
  http://www.freecsstemplates.org (http://www.freecsstemplates.org)
  Released for free under a Creative Commons Attribution 2.5 L
.....................................
  ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'h
ttp://www') + '.google-analytics.com/ga.js';
    var s = document.getElementsByTagName('script')[0]; s.parentNode.in
sertBefore(ga, s);
  })();

</script>
</body>

</html>
```

Y ahora vamos a usar funciones de la librería para extraer distintos tipos de elementos de dicha página.

In [4]:
```
mi_pag_web = BeautifulSoup(mi_codigo_html, "lxml")
print(mi_pag_web.find('head'))
print("-----------------------")
print(mi_pag_web.find('p'))
```

```
<head>
<meta content="" name="keywords"/>
<meta content="" name="description"/>
<meta content="text/html; charset=utf-8" http-equiv="content-type"/>
<title>CristÃ³bal Pareja Flores, pÃ¡gina web</title>
<link href="./style.css" media="screen" rel="stylesheet" type="text/cs
s"/>
</head>
-----------------------
<p class="meta">
<span class="date">Curso 2013-14</span><span class="posted">Actualizad
o: 2014-abril-06</span>
</p>
```

In [5]:
```
print(len(mi_pag_web.find_all('p')))
print("-----------------------")
print(mi_pag_web.find_all('p')[8])
print("-----------------------")
print(mi_pag_web.find_all('p', {'class': 'meta'}))
print("-----------------------")
anclas_o_hiperenlaces = mi_pag_web.find_all('a')
print(len(anclas_o_hiperenlaces))
print(anclas_o_hiperenlaces[3])
print(anclas_o_hiperenlaces[3].get("href"))
print(anclas_o_hiperenlaces[3].get_text())
```

```
22
-----------------------
<p class="meta">
<span class="date">2015</span><span class="posted">Actualizado: 2015-ab
ril-6</span>
</p>
-----------------------
[<p class="meta">
<span class="date">Curso 2013-14</span><span class="posted">Actualizad
o: 2014-abril-06</span>
</p>, <p class="meta">
<span class="date">Hasta 2014</span><span class="posted">Actualizado: 2
014-dic-23</span>
</p>, <p class="meta">
<span class="date">2015</span><span class="posted">Actualizado: 2015-ab
ril-6</span>
</p>, <p class="meta">
<span class="date">Hasta 2014</span><span class="posted">Actualizado: 2
014-abril-6</span>
</p>]
-----------------------
46
<a href="http://eprints.ucm.es/8705/1/CUADERNO_DE_TRABAJO_8.pdf">manual
</a>
http://eprints.ucm.es/8705/1/CUADERNO_DE_TRABAJO_8.pdf (http://eprints.
ucm.es/8705/1/CUADERNO_DE_TRABAJO_8.pdf)
manual
```

Vemos que los contenidos están estructurados en distintos contenedores identifidados por distintas etiquetas en el documento `html` . Para acceder a estos contenidos, se han de usar funciones de la librería, de la manera que se ha mostrado, y con ellos se pueden extraer distintos tipos de elementos de la página y posteriormente procesarlos.

Referencias:

Hay multitud de referencias a las funciones y métodos disponibles en esta librería, y hay multitud de ejemplos en Internet. Damos únicamente un par de dichas referencias.

1. Del libro "Data Science from Scratch", de Joel Grus, capítulo 9 ("Getting Data"), el apartado "Scraping the Web".
2. La página oficial de documentación de esta librería: https://pypi.org/project/beautifulsoup4/