**WEB SCRAPPING**

Mecanismo que nos permite extraer datos de la web de manera automática utilizando softwares. Hay paginas webs que nos proveen mecanismos formales denominados APIs que sirven para poder extraer todos sus datos, como por ejemplo Twitter.

Ventaja: La principal ventaja de usar web scrapping es que no dependo de si la pagina tiene APIs para poder extraer sus datos, debido a que con esto nosotros podemos extraer datos de casi todas las paginas webs directamente de la estructura de la página.

Desventaja: La desventaja más importante es que siempre voy a depender de la estructura visual de la pantalla de la pagina web. Si una pagina web se actualiza, el web scrapping también debe actualizarse para que siga extrayendo de manera correcta. También debemos tener en cuenta que las webs pueden detectar que se esta extrayendo datos y podemos caer en **banneos** temporales

¿Cómo funcionan las páginas webs?

Las paginas web como las vemos con sus colores y animaciones, son en realidad un conjunto de letras que forman un “lenguaje estructurado”, lo cual me permite definir una estructura ordenada y jerárquica. A este lenguaje estructurado se lo llama **HTML**, por sus siglas en ingles HyperText Markup Lenguaje

Se piensa en un HTML como si fuese un árbol, que posee ramas llamadas **tags.** El tronco principal se lo denomina **boody** y el contiene varias ramas hijas llamadas **tags** que a su vez tienen mas hijos. La rama principal llamada **boody** no puede tener otro nombre, es el único que no puede repetirse, pero los **tags** si tienen nombre que se pueden repetir, que dependen de la funcionalidad que se les va a dar. Por ejemplo

* **<p>** contiene grandes cantidades de textos
* **<a>** contiene links de otras páginas
* **<button>** contiene botones que solamente desembocan en acciones
* **<form>** contiene formularios
* **<input>** contiene cajitas de texto donde nosotros podemos escribir información. Usualmente <input> siempre es un hijo de <form>.
* **<div>** se lo usa como contenedor para cualquier cosa, texto imágenes entre otros,
* **<span>** se lo usa para textos cortos de no más de una línea,
* **<h1>** contiene títulos principales,
* **<h2>** contiene subtítulos
* **<h3>** subtítulos de subtítulos, y así sucesivamente según la importancia de los subtítulos
* **<img>** contiene imágenes
* **<li>** contiene listas de bullet points
* **<table>** contiene tablas.

DE todo esto lo importante es saber identificar que clase de jerarquía posee cada tag, y por suerte en HTML cada uno se abre y cierra en etiqueta de apertura y etiqueta de cerrado. Lo importante es que lo que se encuentre dentro de estas etiquetas son los hijos de esas etiquetas

|  |
| --- |
| <body>  <div>  <p>  Hola Mundo  </p>  </div>  </body> |

En este ejemplo tenemos una rama principal llamada <body> de donde sale una etiqueta hija llamada <div>, donde a su vez sale otra rama llamada <p> que contiene una hoja llamada “Hola Mundo”. La etiqueta principal tambien puede teer mas de una etiqueta hija, como en el siguiente ejemplo. Tambien se tiene en cuenta que se consideran a las etiquetas de <span> y a <div> como “primas” debido a que están a un mismo nivel jerárquico

|  |
| --- |
| <body>  <div>  <p> Hola Mundo </p>  </div>  <span>  ¡Hola!  </span>  </body> |

Se pueden definir características adicionales a cada tag denominada atributos que están divididos en dos partes, el nombre del atributo que en este caso es **class** y el valor del atributo, que en este caso es **contenedor**. En el siguiente caso el nombre. Con esto ya la podemos identificar a esta etiqueta y diferenciarla del resto.

|  |
| --- |
| <div class = “contenedor”>  </div> |

Los nombres de los atributos pueden ser cualquiera que se desee. Sin embargo tenemos algunos estándares como lo son:

* **CLASS**: que me ayuda a darle estilos a las etiquetas, como si quiero texto rojo, fondo blanco, entre otros
* **ID**: valor que debe ser único a lo largo de todo el HTML.
* **VALUE**: Se usa para determinar lo escrito un usuario en una caja de texto.

La **indentacion** nos ayuda a saber quien es padre de quien y cual es el orden jerárquico de la estructura del HTML y así poder diferenciarlos y armar el árbol estructural de forma más fácil.

**ESTRUCTURA CLIENTE – SERVIDOR**

Es un modelo en el que uno o más clientes realizan requerimientos que son respondidos por uno o mas servidores. Cuando apretamos ENTER en una URL lo que estamos haciendo es un requerimiento, por lo que nosotros como clientes estamos mandando un requerimiento al servidor que está detrás de la URL. Se puede conceptualizar como que estamos abriendo un túnel de conexión por donde mandamos un túnel de conexión de información llamado **request** el cual le llega al servidor, que no es mas que una computadora recibiendo requerimientos. Esta computadora la detecta, la procesa y nos devuelve literalmente el HTML de la pagina web que vamos a visualizar. Un **request** puede contener más información, como la información personal, si estamos logados o no, entre otros parámetros. No necesariamente los HTML nos devuelve toda la información cargada, puede que nos devuelva un esqueleto sin los detalles de información y que luego esta se cargue a través de nuevos requerimientos que son manejados por instancias del mismo HTML.

**INTRODUCCION A WEB SCRAPPING**

**URL**

Las URLs son secuencias de caracteres que identifican y permite localizar y recuperar una información determinada de internet. Cuando realizamos una búsqueda, Los HTML se vuelven muy largos y difíciles de entender. Esto se debe a que cuando realizamos un **request** a una URL, el servidor nos responde enviando mucha infor

**SELENIUM**

SELENIUM es una herramienta que nos permite automatizar el navegador, haciendo click en botones de páginas web, escribir textos en cuadros de dialogo y con esto hacer elementos de la pagina web y también podemos hacer web scrapping de elementos HTML.