

WEB SCRAPING 01.

El Web Scraping tiene que ver con el concepto de extraer o recolectar datos desde sitios web, directo desde su código HTML, con el objetivo de estructurar dichos datos en un formato que sea manejable.

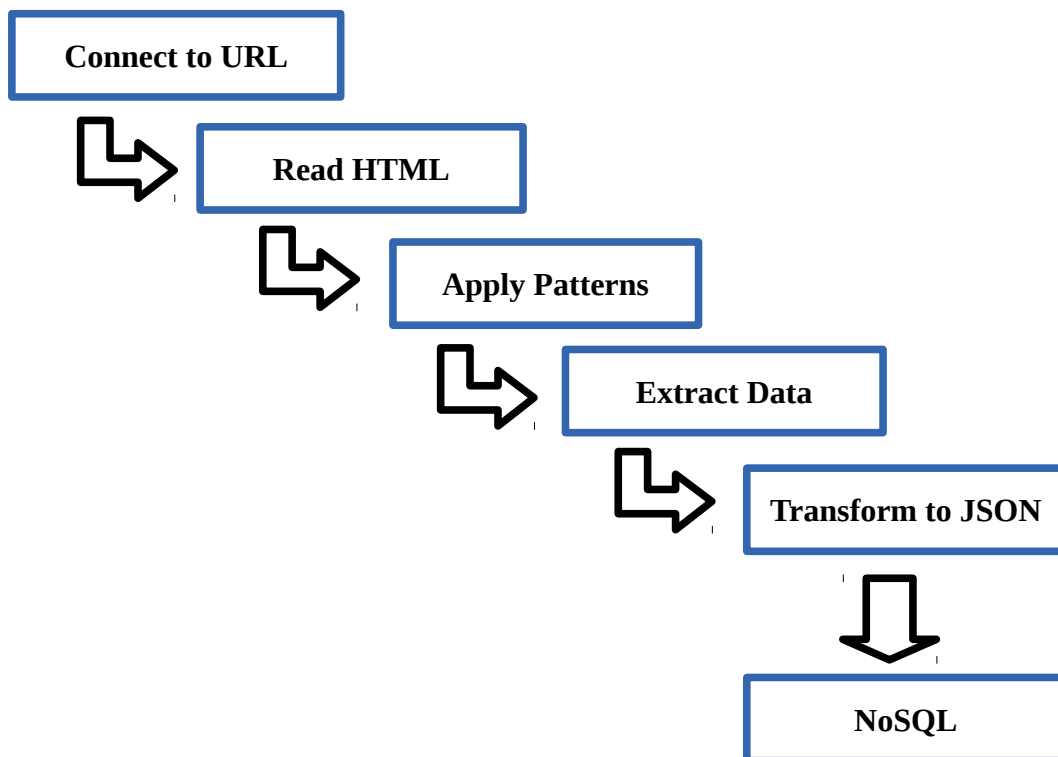
En otras palabras, es recolectar información publica relevante desde las páginas de un sitio. Información que podremos estructurar de manera tal que, a posterior se podrán utilizar para llevar a cabo análisis más interesantes.

El Web Scraping nos permitirá estructurar información publicada en sitios mediante páginas HTML (“información de dominio público”), sin tener la necesidad de utilizar una API o mecanismo que nos entregue los datos bajo un formato. En mi opinión, siempre es recomendable utilizar las API en caso de que éstas existan.

Desde un punto de vista técnico, el Web Scraping lo podemos hacer con cualquier lenguaje de programación que tenga las API's y librerías que permitan realizar las siguientes acciones:

1. Conectarse a una dirección en la web.
2. Leer el HTML de la URL especificada.
3. Identificar patrones dentro del HTML.
4. Extraer los datos relevantes y estructurarlo en un formato.
5. Persistir la información en un motor de persistencia (como una BD NoSQL, por ejemplo) para futuros análisis.

Esquemáticamente la aplicación debe ejecutar la siguiente secuencia de acciones.



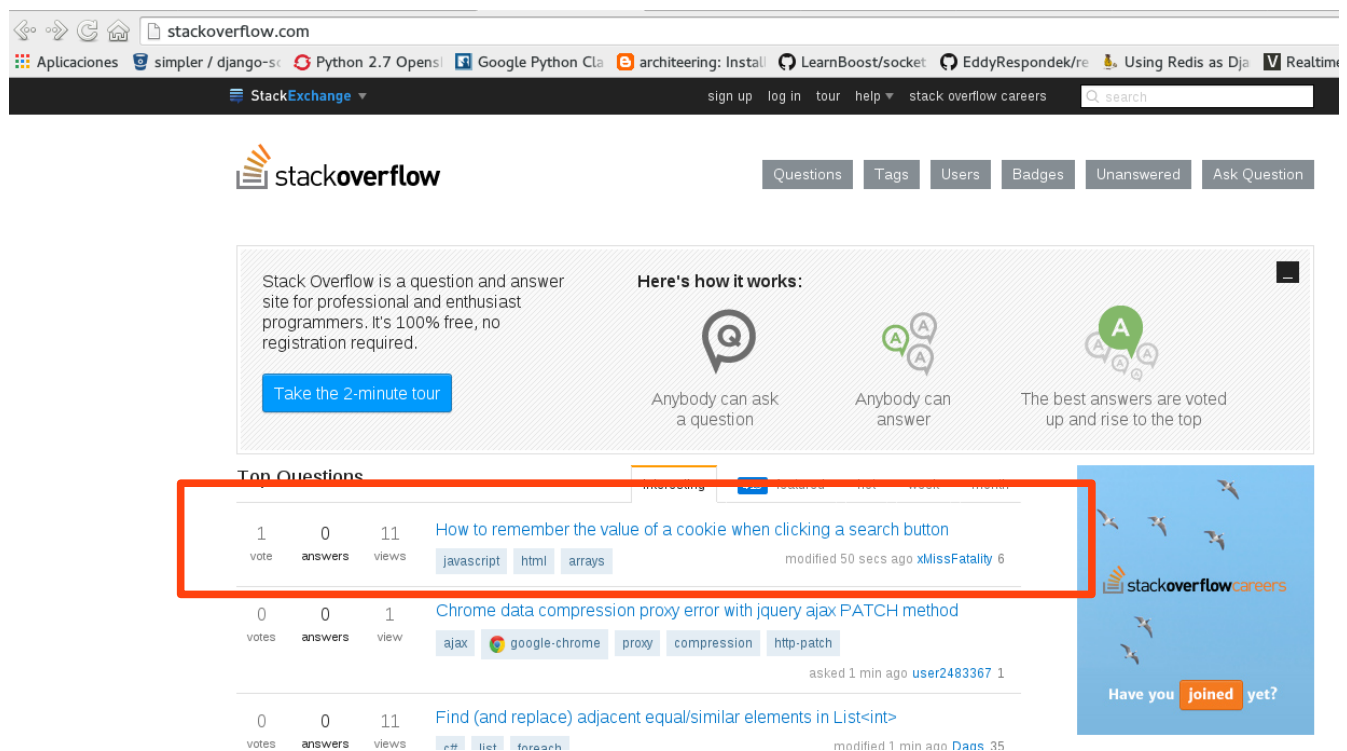
SCRAPING STACKOVERFLOW.

El siguiente ejemplo está desarrollado en Python utilizando las librerías y módulos propios del lenguaje, los módulos principales que se utilizan son:

1. **Módulo `httplib`**, el cual nos permite conectarnos a una URL y leer las páginas HTML a través de HTTP/HTTPS..
2. **Módulo `re`**, el cual nos permite aplicar patrones de búsqueda mediante el uso de expresiones regulares, permitiendonos encontrar y extraer los datos relevantes desde el código HTML.
3. **Módulo `json`**, para formatear la representación de los datos en formato JSON.
4. **Módulo `datetime`**, para registrar la hora en que se ejecuta el web scraping sobre el sitio.

La aplicación desarrollada extrae las preguntas publicadas en <http://stackoverflow.com/>; transformando el HTML que “muestra” las preguntas, junto con los votos (votes), respuestas (answers), las vistas (views) y los tags (asociados a las preguntas) al formato JSON.

A contiinucación se muestra la página del sitio stackoverflow.com y como la información relativa a las preguntas son estructuradas bajo un formato JSON.



```
{
  "answers": "0",
  "id_question": "28653551",
  "question": "How to remember the value of a cookie when clicking a search button",
  "tags": [
    "javascript",
    "html",
```

```
    "arrays"  
  ],  
  "timestamp": "2015-02-21 23:27:48.146811",  
  "views": "11",  
  "votes": "1"  
}
```

Nota: El *timestamp* es puesto por la aplicación para indicar en que momento se realiza el proceso de extracción y transformación.

La aplicación genera como salida una lista con todas las preguntas (junto con su metadata) que están publicadas en la url <http://stackoverflow.com/>

El código de la aplicación y como se utiliza se puede ver en la siguiente URL:

<https://github.com/rancavil/laboratory/tree/master/scraping>

Descripción de las funciones.

Función `fetch_page()`:

La función `fetch_page` realiza la conexión al sitio www.stackoverflow.com mediante el uso de la librería Python `httplib`, recuperando la página HTML principal, retornandola como un String de caracteres.

Función `parse_html()`:

Esta función recibe como parámetro el HTML recuperado por la función `fetch_page()` y lo empieza a analizar mediante la aplicación de patrones de búsqueda definidos con expresiones regulares.

Se utiliza una expresión regular junto con la función `finditer()` del módulo de Python `re`, que permite encontrar, dentro del HTML las preguntas.

El patrón de búsqueda, junto con el uso de la función `re.finditer()`, devuelve una lista con todos los extractos de código HTML que cumple con dicho patrón y que corresponde a las preguntas publicadas en el sitio.

En cada iteración se llama a la función `extract_data_2_json()` la cual extrae los datos desde el trozo de HTML que corresponde a la pregunta.

Función `extract_data_2_json()`:

La función recibe como parámetro un fragmento del código HTML correspondiente a la pregunta, lo analiza y extrae, mediante el uso de patrones de búsquedas definidos a través de expresiones regulares, los datos que se requieren y crea una representación en formato JSON (diccionario de Python).

Retorna un objeto diccionario de Python.

Función find_all_tags():

Esta función busca todos los tags relativos a una pregunta, retornando una lista con los tags encontrado.

Finalmente, la información estructurada en un formato JSON se puede almacenar en una base de datos NoSQL, tal como MongoDB, para efectuar posteriores análisis. Como por ejemplo, responder a preguntas tales como: *¿Cuales hna sido las tecnologías y temáticas más consultadas?*.