Web Scraping Guide

What is Web Scraping?

Web scraping is a technique for extracting data from websites in an automated, efficient manner. This process transforms unstructured web data into a structured format, such as databases or spreadsheets, making it easier to analyze and use for various purposes, whether in B2B or B2C contexts.

Uses of Web Scraping

- Industry Statistics and Insights: Gathering data for market analysis.
- Price Comparison: Comparing prices across different online stores.
- Financial Analysis: Analyzing stock prices or cash flows.
- Academic Research: Collecting data for research purposes.

→ In our project, web scraping is used for research purposes to retrieve data from the Food and Drug Administration (FDA) website about medical devices' approval (accessdata.fda.gov/scripts/cdrh/cfdocs/cfPMN/pmn.cfm) and product recalls (accessdata.fda.gov/scripts/cdrh/cfdocs/cfRES/res.cfm).

Limitations of Web Scraping

- CAPTCHA: Used to prevent automated access.
- IP Blocking: Websites may block IP addresses suspected of being bots.
- Login Areas: Restricted access to certain sections.
- robots.txt: Files that instruct crawlers on which parts of the site can or cannot be accessed.

→ In our project, by applying the web scraping technique we did not encounter these limits.

Legal Considerations

Web scraping is generally legal when used to obtain public information. It becomes illegal if it violates copyright laws, breaks terms of service, or is used for malicious purposes like phishing or identity theft.

→ In our project, by applying the web scraping technique to public information our approach is legal and not subject to legal considerations.

Types of Web Scrapers

1. Browser Extensions: Tools that run within your web browser.
2. Downloadable Software: Programs installed on your computer.
3. Self-Built Scrapers: Custom-built tools using programming languages like Python, R, or PHP.

→ In our project, we have adapted a browser extension called “Webscraper.io” (https://webscraper.io). This is a no-code tool that is easy to use because it does not require programming experience or knowledge and the site-mapping implementation is user-friendly.

How Web Scraping Works

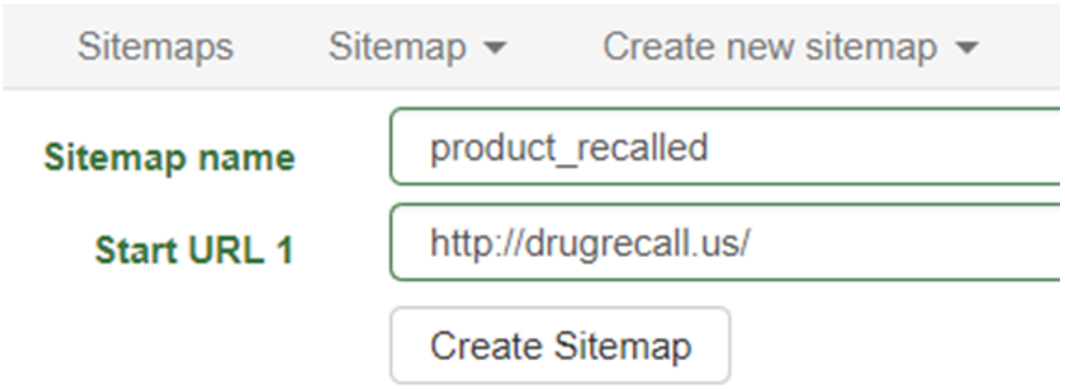
Web scraping tools work by iterating through every URL on a page coded in HTML. These tools extract the required data assuming the structure of the website remains consistent.

Making the Web Scraping using Webscraper.io

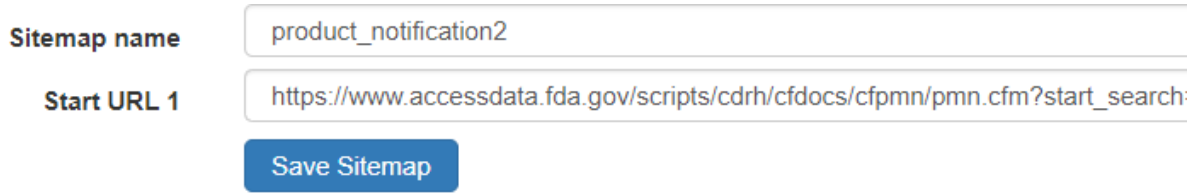
1. Setting Up:

- Create a sitemap pointing to the starting domain (either a local host or a web existing domain)

→ This is the case with a virtualization of the localhost:

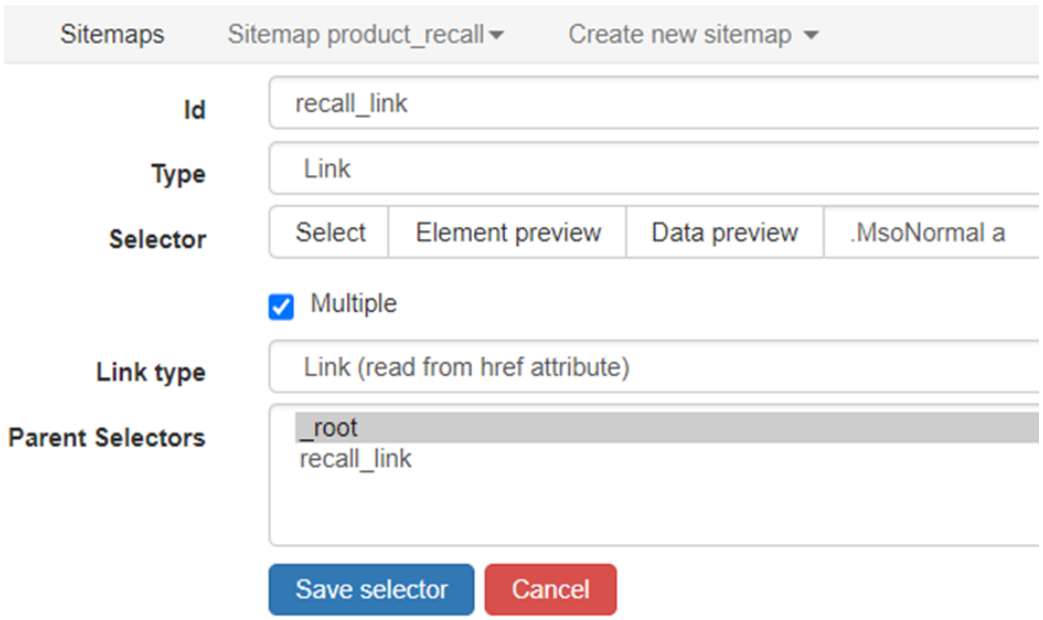


→ This is the case with an existing website:



- Define selectors to target specific data fields in the HTML code

→ This is an example of how to build a selector. Selectors can be of several types, we used principally ‘text’ and ‘link’ data type:



→ This is a sample of selectors for a sitemap:

ID	Selector	Actions
device_classification_name	td td td tr:contains('Device Classification Name') a	Element preview Data preview Edit Delete
510k_number	td td td tr:contains('510(k) Number') td	Element preview Data preview Edit Delete
device_name	td td td tr:contains('Device Name') td	Element preview Data preview Edit Delete
applicant	td td tr[valign]:contains('Applicant') > td	Element preview Data preview Edit Delete
applicant_contact	td td td tr:contains('Applicant Contact') td	Element preview Data preview Edit Delete
correspondent	td td tr[valign]:contains('Correspondent') > td	Element preview Data preview Edit Delete
correspondent_contact	td td td tr:contains('Correspondent Contact') td	Element preview Data preview Edit Delete
regulation_number	td td td tr:contains('Regulation Number') a	Element preview Data preview Edit Delete
classification_product_code	td td td td td a	Element preview Data preview Edit Delete

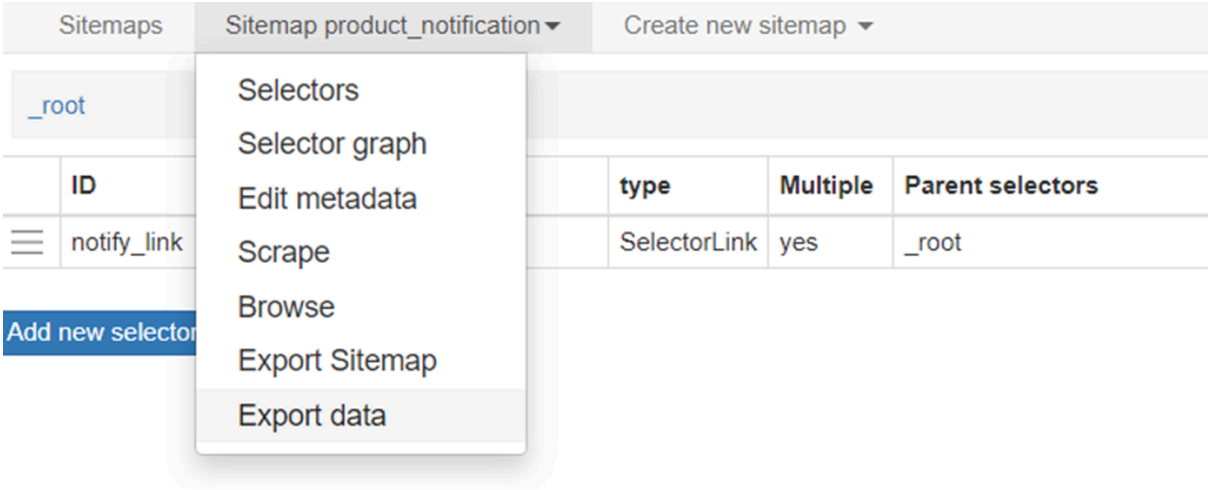
2. In the case of a local webpage, you have to change 'localhost' to a Domain Name:
   - Redirect the website name on your computer (hosts file in C:\Windows\System32\drivers\etc).

```
127.0.0.1       drugrecall.us
127.0.0.1       www.drugrecall.us
```

   - Redirect Apache to the website address in the installation directory (httpd-vhosts.conf).

```
<VirtualHost *:80>
    ServerName www.drugrecall.us
    ServerAlias drugrecall.us
    DocumentRoot C:\Bitnami\wappstack-7.4.25-0\apache2\htdocs
</VirtualHost>
```

3. Start the web scraper:
   - After creating Sitemap and Selectors, the scraping process can be initialized

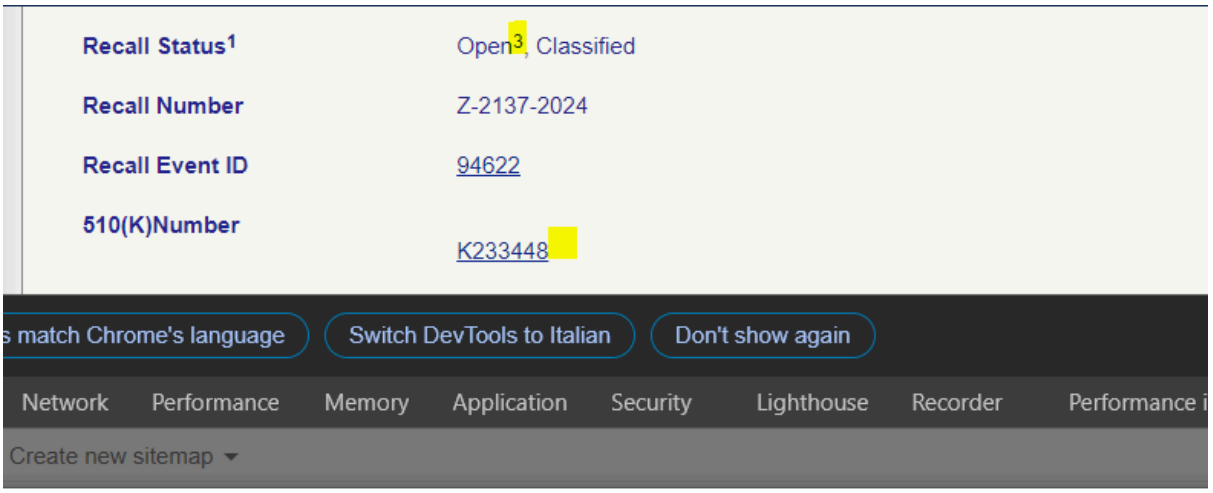


Once scraping is complete, data can be exported from the tool to an Excel spreadsheet or a .csv file.

Final Tips for Effective Web Scraping

- Know What to Search For: Identify the specific data fields you need.
→ Focusing on the data of interest reduces the time needed for web scraping.

- Analyze Page Source: Understand the HTML structure (elements like tr/td/a).
→ In our process of web scraping, irrelevant elements have been added to the useful data because these are part of the HTML structure of the page (see example displayed in the figure below). To clean the data, in a second step, these elements were deleted.



posted	recall_status	recall_number	recall_event_id	510k_number	product_classification	product
12,	Open3, Classified	Z-2137-2024	94622	K23344824	Shunt, central nervous system and components	JXG

- Respect Crawl Rate: Follow guidelines in robots.txt to avoid being blocked.

→ The FDA website does not allow the switch to another web page within 30 seconds. FDA's robots.txt is placed at the root domain (https://www.fda.gov/robots.txt):

```
#For all other crawlers
User-agent: *
Crawl-Delay: 30 # wait 30 seconds before starting a new URL request default=30
Disallow: /health  # don't crawl healthcheck
```

- Sitemap Portability: Export and import sitemaps between different computers using JSON.

→ Direct export is only possible for existing and available websites. In the case of local sites, changes in the starting URL are required.