**Selenium vs Scrapy: Which One Should You Choose for Web Scraping?**

Web scraping is a technique for extracting data from an online source. It provides you with structured data that can be stored in any format. This data can then be used in AI and ML algorithms. Web scraping can provide you with large volumes of clean data that are optimal for these algorithms.

There are various tools and libraries that can be used for web scraping. In this article we will focus on two of the most popular web scraping frameworks: Selenium and Scrapy. We will analyze both frameworks and then we will see which one is the best choice for your web scraping needs.

**What is Selenium?**

[Selenium](https://www.selenium.dev/) consists of a set of powerful tools and libraries that automates web browser actions. In layman’s terms this means that [Selenium](http://blazemeter.com/selenium) provides tools that can interact with browsers and can automate browser actions like click, input, select, and navigate etc. with the help of APIs and scripts. This capability can be used for testing web applications, including cross-browser testing. Selenium supports Safari, Firefox, Chrome and Internet Explorer. Developed and released as an open source tool in 2004, Selenium is widely used by many companies worldwide.

**Selenium for Web Scraping**

You must be wondering: how can a test automation tool be used for web scraping? Selenium has a webdriver component that provides web scraping features. There are various methods and objects in Selenium WebDriver that are useful for web scraping. There are:

**1. WebDriver.page\_source**

This method returns the HTML code of the page.

**2. WebDriver.title**

Gives the title of the page.

**3. WebDriver.current\_url**

Used to get the current URL of the page.

**4. Find\_elements**

Get a list of specific elements on a page. You can find an element by its name, class\_name, tag, id, xpath.

**5. Web\_Element**

To get particular data from HTML elements, Web\_Element is used. Web\_Element.text,Web\_Element,click(),web\_Element.get\_attribute(),Web\_Element.send\_keys() are few useful features in Web\_Element

**6. Is\_displayed()**

A method used to find out if an element is present on a page. It returns true if an element is present and vice versa.

**Selenium for Web Scraping: Pros and Cons**

**Selenium Advantages**

* Free and open source
* Provides multi-browser support
* Supports Linux, Windows and MAC OS
* Multiple language support like Java, c#, Python, Kotlin, Ruby, Javascript

**Selenium Disadvantages**

* Selenium WebDriver occupies system resources even for small data set
* The scraping process begins once page is fully loaded so it is slow in terms of processing
* For each browser you need to install a WebDriver component

**What is Scrapy?**

[Scrapy](https://scrapy.org/) is a web scraping and web crawling framework designed to get structured data from websites. However, Scrapy can also be used for monitoring and automated testing web applications. Scrapy was developed in 2008 by “ScrapingHub.Ltd” and is written entirely in Python. Scrapy provides an asynchronous mechanism which processes multiple requests in parallel.

**Scrapy for Web Scraping: Features**

Here’s a list of the main built-in Scrapy features that make it a powerful web scraping tool:

**1. Spiders**

Spiders are classes that define a set of instructions to scrape a particular website. These built-in customized classes provide an efficient approach for web scraping.

**2. Selectors**

Selectors in scrapy are used to select parts of an HTML document defined by XPath or CSS expressions. With selectors you can use regular expressions through the re() method.

**3. Items**

Data extracted through spiders is returned as items. The itemadapter library supports the following items: attrs objects, dictionaries, item object, data class object.

**4. Item Pipeline**

A python class that validates, cleans and stores the scraped data in a database. In addition to this it also checks for duplicates.

**5. Requests and Responses**

Requests are generated from the spider that takes the request to the end point, where the request is executed and the response object takes the issued request to spider.

**6. Link Extractors**

A powerful feature that extracts links from responses.

**Scrapy Built-in Services**

Scrapy also provides following built-in services to automate tasks when scraping:

* Logging
* Stats collection
* Sending emails
* Telnet console
* Web service

**Scrapy: Pros and Cons**

**Scrapy Advantages**

* Scrapy can extract data in different formats such as CSV, XML and JSON.
* Scrapy provides AutoThrottle features that automatically adjust the tool to the ideal crawling speed.
* Scrapy is asynchronous so it can load several pages in parallel.
* Large volumes of data can be extracted
* In terms of speed, Scrapy is fast
* Scrapy consumes little memory and CPU space

**Scrapy Disadvantages**

* Scrapy cannot handle Javascript
* The installation process varies for different operating systems
* Scrapy requires Python version 2.7.+

**Selenium or Scrapy?**

When it comes to selecting only one library, Selenium or Scrapy, the decision ultimately boils down to the nature of the use cases. Each library has its own pros and cons. Selenium is primarily a web automation tool, however, Selenium WebDrivers can also be used to scrape data from websites, if you’re already using it or you’re scraping a JS website. On the other hand, Scrapy is a powerful web-scraping framework that can be used for scraping huge volumes of data from different websites.

Let’s see some examples about when to choose each:

**Data Volumes**

Let’s say we are working on a project where we need large volumes of data from different websites. To scrape those websites we have to make multiple calls using proxies and VPNs. In addition to this we need a robust mechanism and we can’t afford delays. In such scenarios, Scrapy is an ideal choice. Using Scrapy you can easily work with proxies and VPNs. It can pull large volumes of data since it is a specialized web scraping framework.

**JavaScript Support**

To scrape data from a website that uses Javascript, Selenium is a better approach. However, you can use Scrapy to scrape JavaScript-based websites through the Splash library.

**Performance**

Scrapy is asynchronous, it executes multiple requests simultaneously. Even if a request fails or any errors happen the incoming requests aren't affected. This improves the overall speed efficiency of the process. Selenium is also robust but in case of large data volume the overall process is slow.

**Selenium vs. Scrapy Comparison Table**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Data Volumes | JavaScript Support | Performance |
| Selenium | Medium-low | JS support | Robust, slow with high data volume |
| Scrapy | High | JS support via Splash | Fast |

**Conclusion**

To conclude the above discussion I would say that both Selenium and Scrapy are powerful tools. The nature of work for which they’re originally developed is different from one another.

Selenium is an excellent automation tool and Scrapy is by far the most robust web scraping framework.

When we consider web scraping, in terms of speed and efficiency Scrapy is a better choice. While dealing with JavaScript based websites where we need to make AJAX/PJAX requests, Selenium can work better.