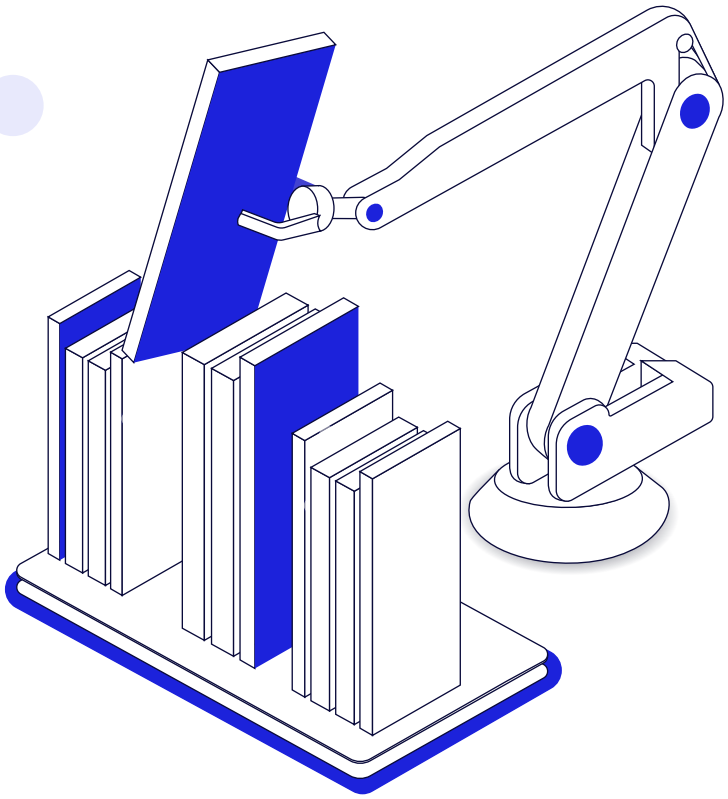
scraperapi

Web Scraping:
The Basics Explained



Who is this for?

Explore the world of web scraping: the process, the tools required, and some best practices for running a successful scraping project. This handbook will help anyone, from scraping enthusiasts to enterprise companies, adopt web scraping in their day-to-day tasks.

Index

Web Scraping: The Secret to Scalable Growth

In today's digital economy, data is the new differentiator.

Having reliable data at your disposal can give your business a competitive edge.

Let's use the eCommerce giant Amazon as an example:

Amazon leverages big data collected from the internet, and their customers' behavior, to update their product pricing approximately every ten minutes. Their pricing is set according to the general trends in the market, users' shopping patterns, and business goals—among others.

By capturing big data, Amazon can smartly offer discounts on best-selling items and, at the same time, earn large profits on less popular products. This data-driven strategy has proven fruitful as they significantly doubled their annual sales from 2018 to 2021.

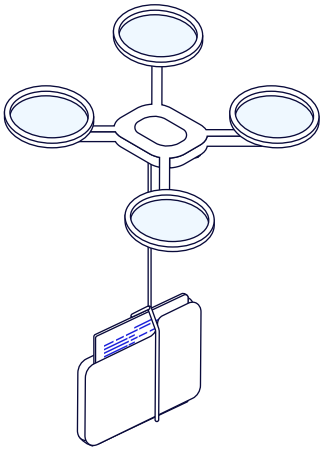
Netflix experienced similar success. They used web data acquisition to gather data about the preferences of their viewers and potential subscribers. It is no surprise that many of the Netflix Original shows are a hit, helping them to maintain a low churn rate of 2.4% from 2019 to 2021.

These two examples show that data harvesting is helpful in various businesses, regardless of the industry, type, or size. Every organization that strives to scale should leverage publicly available data and use it to its advantage. But how? How can organizations collect web data at a large scale, automatically, and within minutes? The answer is web scraping.

Three major benefits of data harvesting:

- Give insight into the market condition
- Close observation of competitors
- Deep understanding of consumer behavior

What is Web Scraping?



Web scraping is a method for extracting large amounts of data from the internet. This intelligent automated approach gathers anything from prices to product specifications, property listings, and other publicly available data. The results can be presented in structured file formats: XML or JSON.

Put simply, web scraping can be compared to “copy-pasting” content from websites, but it differs in the process and the tools needed to perform the action. As you can imagine, data scraping requires a web scraper and a few lines of code to function. Some common programming languages and libraries used include Python BeautifulSoup and Python Scrapy.

Furthermore, unlike manual copy-pasting, a web scraper can harvest information from thousands of URLs by queuing requests in bulk. This scalable solution eliminates any human intervention during the scraping process—saving you a lot of time and manual labor.

But Is Web Scraping Legal?

One general concern around web scraping is whether or not it’s legal.

No government has passed laws explicitly legalizing or de-legalizing web scraping thus far (2022). Therefore, we can only make strong assumptions based on case law about web scraping activity (e.g., HiQ vs. LinkedIn) and other data-related regulations.

We know that web scraping itself is legal—but it can be illegal depending on what type of data you scrape and how you scrape it. In general, you can legally scrape the internet as long as:

- The data is publicly available
- You don’t scrape private information
- You don’t scrape copyrighted data
- You don’t need to create an account and log in to access the website, OR you have read and fully understand the Terms and Conditions (T&Cs)

Types of Data Explained

Private information

Private or personal data is any data that could be used to directly or indirectly identify an individual. This includes, but isn't limited to, email addresses, medical data, user names, IP addresses, and banking information.

Different legal jurisdictions have different regulations about personal data, but it is generally illegal for anyone to obtain, store and/or use someone's personal data without their consent.

When scraping personal data from a website, it is likely that you don't have the permission of the data owners to extract their information. Consequently, scraping private information could be considered illegal. However, if you don't extract personal data, you are likely safe to keep scraping.

Copyrighted Data

Copyrighted data is owned by businesses or individuals with explicit rights over its reproduction and capture. Some examples of copyrighted data are articles, videos, pictures, music, and databases.

Scraping copyrighted data can be lawfully tricky. The scraping process itself isn't technically illegal—it's what you do with the data that could make the action against the law. For example, do you plan to replicate the scraped article entirely or only use snippets of it? Is the data factual (names, prices, features, etc.) or copyrighted?

For ethical data scraping, you should only scrape some of the available data and don't replicate the organizational structure of the original database.

*Note: Every legal jurisdiction has its own regulations governing personal, copyrighted, and database data, as well as the legal protections they give to the data owner. For instance, the personal data of EU citizens is protected by GDPR. The equivalent of GDPR in the USA is CCPA, which only protects the personal data of California residents. Therefore, it is important to understand the rules of the legal jurisdiction you are scraping in.

Logins and T&Cs

Many websites ask users to create an account and log in to access the website.

If this is the case, you should examine the T&Cs you agreed to when you created the account. Most websites state in their T&Cs that they forbid any scraping activity from their sites—and if you agree to this, you acknowledge that data scraping is illegal.

As a rule of thumb, you should always assume that logging into a site and scraping is illegal (unless you've read through the T&Cs carefully).

So, is web scraping illegal? It isn't if you follow specific rules. Double-check your web scraping plans to ensure that you conduct a legal and ethical data extraction process.

The information given is provided for informational purposes only. Please seek legal advice if you're in doubt about your web scraping project to ensure you're not scraping the web illegally.

The Standard Sync Web Scraping Process

There are two main components of a web scraper, the web crawler and the web scraper itself.

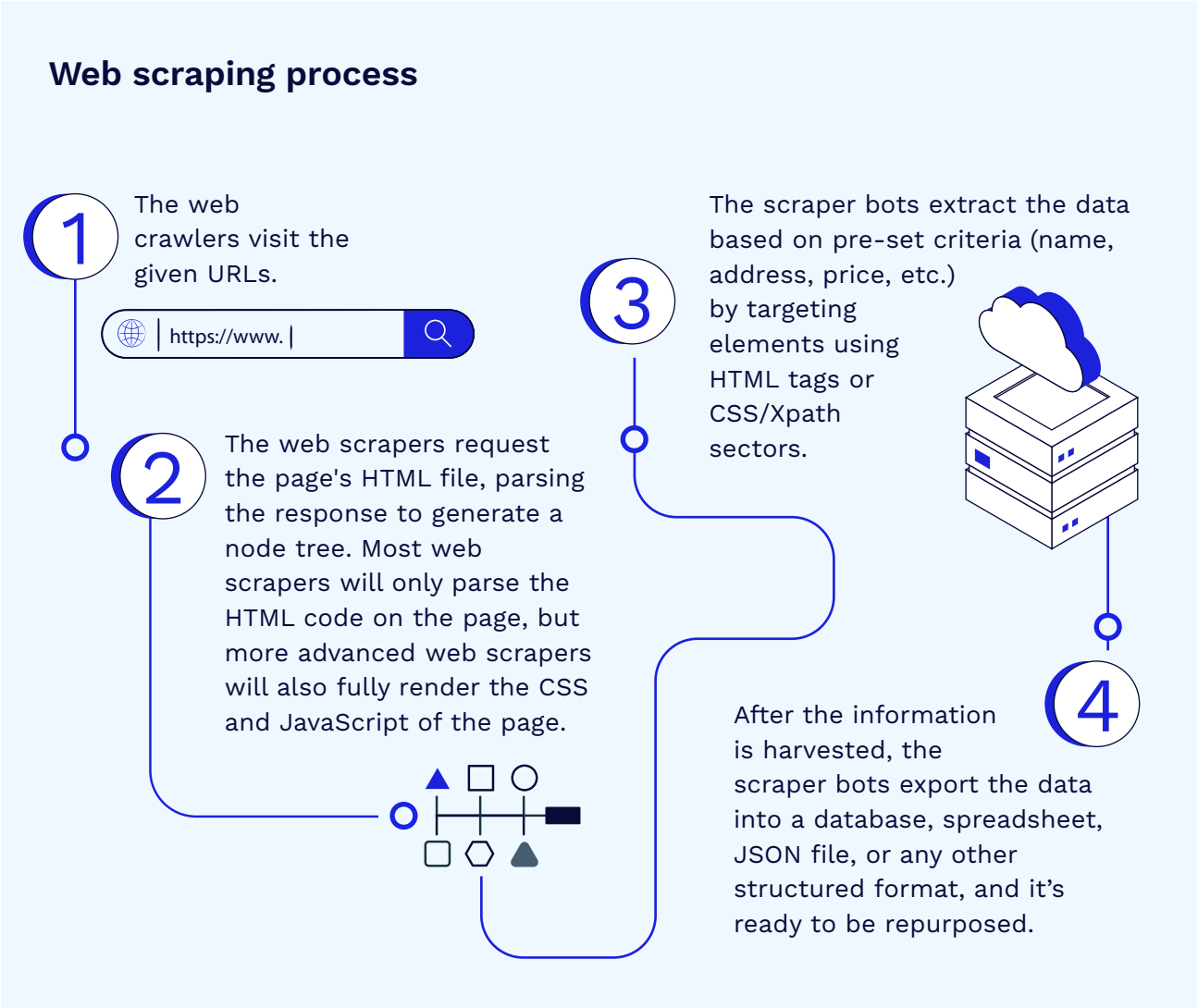
- **Web crawlers**
  The web crawler works similarly to a search engine bot. It crawls a list of URLs and catalogs the information. Then, it visits all the links it can find within the current and subsequent pages until it hits a specified limit or there are no more links to follow.

- **Web scrapers**
  After the web crawler visits the dedicated web pages, the web scraper will collect the data. An integral element of a web scraper called 'data locators' will find, select, and collect the targeted data from the HTML file of a website at scale, without being blocked.

In layman's terms, this is how web crawling feeds into sync scraping: once data is crawled, it can be harvested. When the first scraping request is complete, you can begin the next task.

Of course, the purpose of your scraping needs will always determine the type of scraper and method/s you use. Depending on your timeline and the volume of data collection you need, you may face challenges when you try to use a standard sync scraper to complete multiple tasks. Why? Because you're bound to a limited response (timeouts) and the need to re-submit tasks.

If you use an asynchronous scraper service, you can scrape at scale without these problems. It requires less coding and less infrastructure needed to build or maintain on your side. This speedy, modern method allows you to submit a large batch of requests simultaneously—still working to achieve the highest reachable success rate.
Once the job is done, you'll be notified.



Different Types of Web Scrapers

Self-built vs. pre-built web scrapers

Anyone can build a web scraper. But to build one, you'll need advanced knowledge of some programming languages and their corresponding scraping libraries (Python's Beautiful Soup or JavaScript's Cheerios). Another option is to download a pre-built web scraper and customize it according to your needs.

Cloud-based vs. local-based web scrapers

The main difference between cloud- and local-based scrapers is the location of the scraping process. Since the former runs the tasks on the cloud, it won't affect the performance of your computer, and you don't need to operate a costly server infrastructure. But on the contrary, the latter runs the scraping requests directly on your computer and, therefore, could slow down the system.

Browser extension vs. software web scrapers

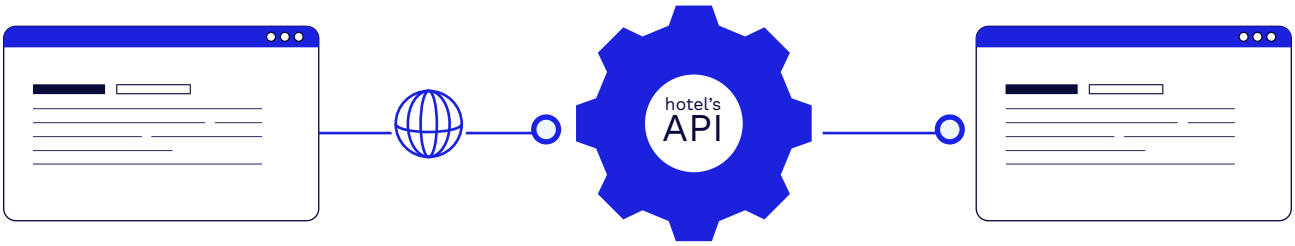
Although browser scrapers are easier to install, their features are limited to the browser you're using. That being said, software web scrapers may have more extensive features but aren't optimal for large-scale data extraction.

What About a Scraping API?

Before explaining what a scraping API is, let's talk about an API first.

Application Programming Interface (API) works as an intermediary between two software or websites. Each API is assigned unique protocols that allow them to communicate with each other. This also means that an API only works with a system that can accommodate it.

For example: Imagine you want to book a room at a hotel using a third-party online travel website. The travel site will connect your request to check the room availability and relay your booking order via the hotel's API. An API makes the interaction between the travel and hotel's websites direct and seamless.



Third-party online travel website

Hotel's website

Now, a scraping API, like the ScraperAPI tool, is then a combination of a web scraper and an API. It acts as the middleman between your computer and the websites you extract data from.

The biggest advantage of using a scraping API is that your chances of being blocked are less. Many scraping API solutions offer built-in features that prevent your scraping request from being detected as a malicious activity. These features include proxy management, IP rotation, CAPTCHA bypass, and custom headers.

Common Challenges of Web Scraping

Many websites stop web scrapers from accessing their data.

From a technical perspective, scraping can spike traffic and break a website server down. From a business and legal perspective, some websites don't want anyone to extract their data and/or want to protect their users' sensitive information from illegal scrapers. To defend themselves, many site owners set up anti-scraping mechanisms to filter out traffic that seems artificial (doesn't come from human users but programmed bots). When a scraping bot is detected, the system will block the bot, the IP will be blacklisted, and the scraping project will be terminated.

Here are some popular anti-scraping methods applied to prevent web scrapers:

- **Requests frequency and patterns detector**
  It measures X number of requests every Y seconds from Z IP address, and blocks anything that doesn't look human-like (e.g., too frequent hits).
- **CAPTCHAs blocker**
  Usually, either a text-based, image-based or social media login CAPTCHA blocker.
- **Honeypots**
  A trap is set up to detect web scraping activity. There are many types of honeypots, and one of them is the hidden fields on a web form. This field doesn't need to be accessed or completed by human users, but web scrapers usually fill out every field to collect the data, including this one. Once the form is submitted, the scraping activity will be flagged.

Web Scraping vs. Data Mining

Data mining is often confused with web scraping—but they are, in fact, two different processes.

Data mining is the process of sorting through large amounts of data using software, statistical methods, and algorithms to find patterns and anomalies. This means that data mining doesn't involve data extraction. Instead, it only organizes and analyzes raw data into valuable business knowledge.

On the other hand, web scraping is the practice of extracting information from websites and repurposing it into other applications and formats. It collects raw data, which is later used for data mining.

Web scraping use cases	Data mining use cases
Data collection for machine learning	Find anomalies and patterns in data sets
Lead generation for marketing and sales	Analyze user behavior data for marketing to improve segmentation, optimize campaigns, and create customer loyalty plans
Find harmful content associated with a company's brand (reputation management)	Apply process mining to find bottlenecks and reduce operational costs
Collect price and product data for price comparison websites and eCommerce companies	Mining prospects' data to find sales opportunities and cross-sells opportunities
Scrape search engine result pages for SEO purposes (e.g., Twitter and forum data for sentiment analysis)	Analyze student profiles, classes, time spent, etc., for educational institutions to improve the study framework

Top Five Businesses That Use Web Scraping

Web scraping is used for various applications, particularly tasks that require hours of manual work or hefty budgets. By scraping data, you can make better, more informed business decisions.

The following are just a few examples of web scraping's endless applications.

eCommerce and comparison sites (price monitoring and market research)

Price comparison sites and online retailers need to regularly update millions of product prices based on market trends. However, it is impossible to extract information at such a massive scale using conventional methods. This is where web scraping comes in. Web scraping enables price comparison sites to update their database automatically. Similarly, having access to competitors' behavior allows online stores to offer unbeatable prices and improve their product selections.

Marketing (lead generation and social media listening)

Marketing agencies or consultants can utilize web scraping to generate leads. They can collect publicly available contact details of potential clients from LinkedIn, Google Business Profiles, or local wanted listings. They know who matches their ideal customer profile and is looking for specific services—turning cold outreach into an effective move.

Additionally, scraping brand mentions and hashtags on social media is a powerful way to discover up-and-coming trends and your brand's sentiment. For example, you can create campaigns that follow the latest trends, converse with your audience to increase engagement, or spot angry comments involving your brand worth responding to.

Recruitment (market research and skills extraction)

Networking platforms (LinkedIn) and employment websites (Indeed) are goldmines for recruiters. They can extract data to source qualified candidates (based on education, experience, etc.) and analyze the job market to find job opportunities or compare salaries.

Real estate (geotargeting, pricing analysis, and listings)

Web scraping can fetch any data points about a property based on the defined factors (locations, price, building specifications, and more). Real estate agents can then adopt this information to create a listing, support the proposed price, or better position an offer.

How to Use a Web Scraping API Effectively

Web scraping comes with challenges—but it doesn't mean you can't work around them.

Follow these best practices to ensure you don't hit any roadblocks with your project.

Set a timeout to a minimum of 60 seconds

Set your timeout to at least 60 seconds. If you set a shorter timeout period, the connection will be cut off on your end, but the API will keep retrying the requests until the 60-second timeout is met. Because the API returns successful requests, these requests will still be counted against your monthly limit.

Only set custom headers when needed

Don't set custom headers unless you need to. This will prevent any performance drop and keep you safe from header inspection from the website's server.

Send requests to an HTTPS version of a website

Always send your requests to the HTTPS version of a website to avoid duplicate requests. If you send a request to the HTTP version, it will redirect to the HTTPS, and the server will read this as two requests—increasing the chance of being flagged as a scraper.

Avoid using sessions unless necessary

Your session pool is much smaller than the main proxy pools. Because of this, the session pool can quickly get burnt out if overused by a single user. We suggest using this feature only if you need to use one proxy for multiple requests.

Manage your concurrency properly

Making parallel requests may equal faster scraping times but cost you more concurrency sessions. This is especially problematic when you handle a large number of distributed scrapers. We recommend setting a central cache (like Redis) to ensure all your scrapers stay within your plan's concurrency limits.

Only use JavaScript rendering if needed

JavaScript requests take longer to process. As a result, this reduces the number of retries you can make internally before returning a failed response. Configure your scraper to stay under the 3-requests-per-second burst limit.

Use premium proxies as a backup

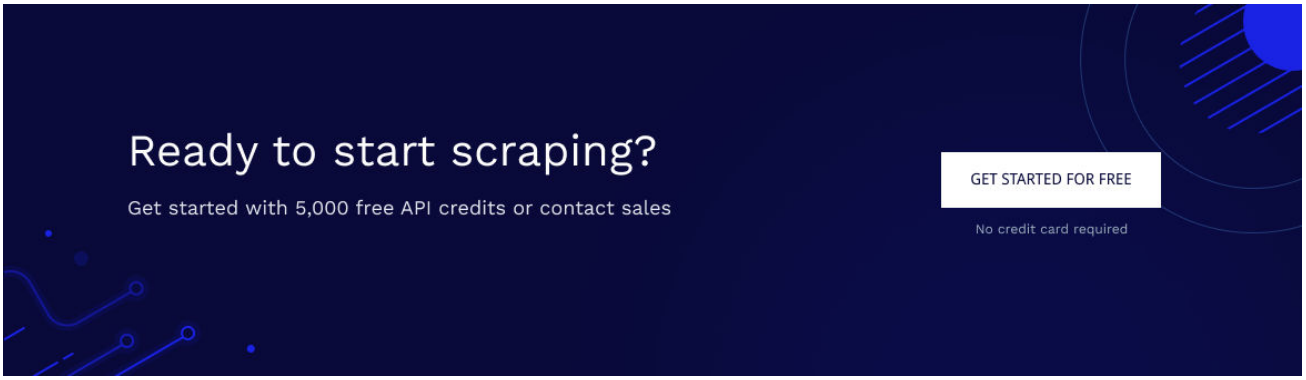
Premium proxies are expensive, but it is wise to have one as a backup. To minimize the expense, only set your script to use the premium proxy if the requests still fail after multiple retries with the standard proxy.

Verify if you need geotargeting before running your scraper

Websites like Amazon and Google return different information depending on where the request comes from. If you need data from a specific geographic location, you'll need to geotarget the request using the "country_code" query parameter.

Getting Started with Web Scraping

Quality data is powerful and fundamental to any business worth its salt. It is the best way to gain insights into the market at scale, within minutes, and in auto-pilot mode. However, we know that web scraping can be demanding, and choosing a reliable and robust web scraping tool is vital to overcome technical difficulties. ScraperAPI is an all-around web scraping solution that handles proxies, browsers, and CAPTCHAs—enabling you to acquire data from any website without getting blocked.



www.scraperapi.com