Headless Browser Automation (Playwright, Puppeteer, Selenium)

Browser Automation Frameworks

The foundation technique used by most scrapers. A browser renders Maps, executes JS, and extracts from the DOM.

Playwright (Most Popular)

Microsoft's library. Used by gosom/google-maps-scraper, HasData, many commercial tools. Supports Chromium, Firefox, WebKit.

- Languages: Python, Node.js, Java, .NET
- Anti-detect: stealth plugins available
- Parallel tabs: 10+ tabs in one browser, ~1.7s/URL
- Codegen: record browser interactions, auto-generate scraper code

Puppeteer (Node.js)

Google's Node.js browser automation. `puppeteer-extra-plugin-stealth` provides 17 evasion modules.

- Best stealth ecosystem (17 modules)
- XHR interception via `Network.requestWillBeSent`
- Caveat: anti-bot companies study the stealth package

Selenium (Legacy)

Original framework. `undetected-chromedriver` patches for detection evasion.

- Languages: Python, Java, C#, Ruby, JS
- Larger fingerprint, easier to detect
- Still used by HasData and Zubdata scrapers

Extraction Strategies

Strategy	Token Cost	When to Use
CSS selectors (`querySelectorAll`)	~52/item	Known structure — default choice
`aria-label` attributes	~52/item	More resilient — accessibility attrs are stabler than CSS classes
`body.innerText`	~5K/page	Discovery — learn structure once, then switch
Network/XHR interception	Minimal	Capture protobuf responses directly — best approach
Accessibility tree (filtered)	~28K/page	Find buttons, forms, interactive elements
Screenshot	~132K	CAPTCHA solving, visual debugging only

Revision #1
Created 2026-06-05 23:48:41 UTC by Ben Adrian Sarmiento
Updated 2026-06-05 23:48:41 UTC by Ben Adrian Sarmiento