Scrape Any Website with Claude Code + Firecrawl

This guide walks you through how to use Firecrawl's MCP server inside Claude Code to turn any website into LLM-ready data, including structured content, screenshots, branding, and full site maps, all through natural language commands.

By: Nate Herk

I. What is Firecrawl?

Firecrawl is a web scraping tool that turns websites into clean, structured data you can use with AI. Instead of manually configuring API calls for different scraping tasks, you give Claude Code access to Firecrawl's MCP server and let it figure out which tools to use based on your natural language request.

What Firecrawl can do:

- **Scrape**, Extract all content from a single page (text, images, metadata).
- **Map**, Discover all the URLs and understand the architecture of an entire website.
- **Crawl**, Explore and extract data from multiple pages across a site.
- **Search**, Perform a web search first, then scrape the resulting pages.
- **Extract**, Pull structured data fields from pages (e.g., job titles, prices, descriptions).

Output formats include: Markdown, AI-generated summaries, links, HTML, full-page screenshots, branding info (logo, colors, typography), and raw JSON.

II. Quick Demo in the Firecrawl Playground

Before jumping into Claude Code, you can test Firecrawl directly in the browser.

1. Paste any URL into the playground.
2. Choose your format options (markdown, screenshot, branding, AI summary, links, HTML, etc.).
3. Run the scrape and review the output.

This is a great way to get familiar with what Firecrawl returns before integrating it into your workflow. The free plan gives you enough credits to experiment.

III. Setting Up Firecrawl MCP in Claude Code

Step 1: Open a New Project in VS Code

Create or open an empty project folder. Your left-hand side should be empty, and Claude Code should be open on the right.

Step 2: Get the MCP Install Command

Go to the Firecrawl Docs and find the **MCP Server** section. Look for the **Claude Code** installation command, it's a single line you'll paste into Claude Code.

Step 3: Connect to the MCP Server

Tell Claude Code:

> "Hey Claude, I want to connect to Firecrawl's MCP server. You can do that using this command. But I'm not going to give you my API key, I'm just going to put it in a .env. So if you could create that file for me, I will put my API key in there. Then go ahead and initialize and connect to Firecrawl's MCP server."

Step 4: Add Your API Key

1. Open the `.env` file Claude created.
2. Go to your Firecrawl dashboard and copy your API key.
3. Paste it into the `.env` file and save (`Ctrl + S`).
4. Close the `.env` file.

> **Why use a .env file?** It keeps your API key secure and out of the conversation history.

Step 5: Reload VS Code

Press `Ctrl + Shift + P` and type **"Developer: Reload Window"** so Claude Code can access the newly configured MCP server.

Step 6: Test It

Send a quick request to confirm everything works. Claude Code should be able to call Firecrawl tools (scrape, map, crawl, etc.) and return results.

IV. Creating a Firecrawl Cheat Sheet

Before diving into real tasks, give your project more context so Claude Code knows which Firecrawl tools to use and when.

Prompt Claude Code:

> "Create a Firecrawl cheat sheet as a markdown file in this project that you can look at. It should tell you about the different tools and how to use them."

This generates a reference file (e.g., `firecrawl-cheatsheet.md`) that includes a tool overview, usage instructions for each tool, and a decision guide for selecting the right tool for each task.

V. Setting Up the CLAUDE.md File

The `CLAUDE.md` file acts as the **system prompt** for your project, it tells Claude Code what this project is about and what resources it has access to.

Prompt Claude Code:

> "Help me set up a CLAUDE.md file for this project. This project is specifically for scraping data, extracting it, getting screenshots, crawling everything, mapping everything. You have access to the Firecrawl MCP server and the firecrawl-cheatsheet.md which explains how that MCP server works and when to use each tool."

Why separate the CLAUDE.md from the cheat sheet? The `CLAUDE.md` file stays concise and high-level. The detailed cheat sheet lives in its own file, and Claude Code knows to reference it when needed, keeping your system prompt clean without losing any context.

> **Pro Tip:** In practice, use **Plan Mode** first to brainstorm with Claude and align on file structure before implementing. The demo used bypass permissions mode for speed, but planning first leads to better results.

VI. Use Case: Scraping 1,700+ Job Listings

The Scenario

A remote job website has ~1,700 job opportunities spread across 60+ pages. The goal is to extract structured data from these listings into a format you can drop into a Google Sheet.

The Prompt

> "I found this website and I've got about 1,700 job opportunities I want to look at. But I need help using the Firecrawl MCP server to get all of these listed out. I want these as structured data so I could throw them into a Google Sheet."

What Happened

1. **Claude Code first scraped** the page to understand the website structure.
2. **Then it mapped** the site to discover all the paginated URLs.
3. **It created a plan** and asked clarifying questions:
    - How many listings to grab? (Set to 200 to save time)
    - Which data fields? (All of them)
    - Full descriptions or summaries? (Summaries)
4. **Executed the plan**, and when the initial extract returned empty results, it self-corrected and tried the Firecrawl agent instead.

The Output

A CSV file with 200 job listings containing: title, company, job type, location, salary, experience level, category, date posted, apply URL, description summary, and tags.

Key Takeaway: Agentic Self-Correction

Claude Code didn't just fail and stop, it recognized the issue, adjusted its approach, and completed the task. This is the power of agentic workflows: the AI adapts its plan in real-time when something doesn't work.

VII. Use Case: Screenshots & Branding Analysis

The Prompt

> "Please grab screenshots of this page and help me understand the branding."

The Output

- A **full-page screenshot** of the website's landing page.
- **Branding details** including: color palette, typography, spacing, components, and the logo.

This is useful for competitive analysis, design briefs, or understanding a client's existing brand before building for them.

VIII. Use Case: Mapping a Full Website

The Prompt

> "Go ahead and map out this site for me and show me what it looks like."

The Output

A complete site map showing: main pages, category pages (best sellers, coffee, instant, matcha, etc.), collections, store locations, product URLs, brew guides, and more.

Once you have the map, you can then crawl specific sections, extract product data to a database, or build further analysis on top of it.

Running Multiple Agents

You can run **two Claude Code agents simultaneously** on different tasks, for example, one scraping branding from one site while another maps a different site. They work in parallel using different Firecrawl tools.

IX. Pricing & Credits Breakdown

All three use cases (job scraping, branding/screenshots, and site mapping) used approximately **30 credits out of the 500 free credits** included with Firecrawl's free plan, just 6% of the free allotment.

Key differences between plans:

Feature	Free	Hobby	Higher Tiers

Credits	500	More	Scales up
Pages	Limited	More	Scales up
Concurrent Requests	2	5	More

Concurrent requests determine how many scraping operations can run at the same time. On the free plan, Claude Code will queue and retry requests as needed. For bulk operations, higher concurrency is useful.

> Use the link in the video description to get **10% off** your Firecrawl plan when you're ready to upgrade.

X. Project Setup Quick Reference

Step	What to Do
1. Create project folder	Open an empty folder in VS Code
2. Install Firecrawl MCP	Use the one-line command from Firecrawl docs
3. Add API key	Paste into `.env` file, save, and close
4. Reload VS Code	`Ctrl + Shift + P` → "Developer: Reload Window"
5. Create cheat sheet	Ask Claude to generate a Firecrawl tool reference
6. Create CLAUDE.md	Define the project scope and reference the cheat sheet
7. Start scraping	Use natural language to describe what you need

---

Want to connect with others building and monetizing AI automation?

Become an AIS Plus Member