MCP SERVER | NO CODE | CLOUD HOSTED

HTML DOM Query Engine MCP

Extract specific data points from messy HTML code.

HTML DOM Query Engine provides precise data extraction from messy web pages. Stop feeding massive HTML payloads into your AI agent and risking token limits or hallucination. This MCP lets you pass a raw webpage string and a CSS selector, instantly pulling out exactly the text or attributes (like image URLs or prices) you need. It's fast, memory-efficient parsing for reliable scraping.

F Quality Score 3.6/100

html-parsing | css-selectors | data-extraction | web-automation | dom-manipulation

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/html-dom-query-engine

SCAN TO VIEW

The infrastructure that powers AI agents in the real world.

Vinkius connects AI to the world's software through secure, enterprise-grade infrastructure — enabling real-world execution at scale, built on the Model Context Protocol (MCP).

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the cloud infrastructure where AI agents connect to the software your business already runs. We handle the hosting, the security, the credentials, the uptime — you get agents that actually do things.

We operate the world's largest managed MCP catalog. Major SaaS platforms, CRMs, databases, and cloud providers — running, monitored, production-ready. This MCP server is hosted and maintained by the Vinkius Cloud for AI Agents.

> *The agent doesn't manage credentials, doesn't manage uptime, doesn't manage security. Vinkius does.*
>
> — Architecture principle

Four Pillars of the Vinkius Runtime

01 — Security by design

Credentials stay encrypted at rest via AES-256. The AI agent never touches raw keys — they're injected into a sandboxed V8 isolate at runtime. Actions are logged, and connections have an emergency kill switch.

02 — Built on MCP Fusion

This MCP server was built with **MCP Fusion**, the open-source framework (Apache 2.0) that powers the entire Vinkius catalog. Schema-as-firewall strips undeclared fields, compiled PII redaction runs at zero overhead, and cryptographic lockfiles produce git-diffable audit trails.

03 — Deterministic observability

Eight immutable metrics per endpoint: request volume, p95 latency, error rate, active connections, cost attribution. A live payload feed logs every tool call with mutation detection.

04 — Autonomous operations

Servers are deployed, monitored, and patched autonomously. New capabilities and security patches ship weekly. Zero-downtime deployments ensure continuous availability across all managed MCP servers.

AES-256	Ed25519	24h TTL	V8 Isolate
Encryption at rest	PKI vault signatures	Ephemeral session keys	Sandboxed execution

One Token. Instant Access.

Every MCP server on Vinkius is accessed through a **Connection Token**. Tokens are generated in the cloud dashboard and produce a unique MCP endpoint URL. Paste this URL into any MCP-compatible client — no SDK required.

A single token can serve **multiple AI clients simultaneously**, or you can issue separate tokens per client for granular access control. Each token tracks its own request count, last activity timestamp, and can be individually enabled or revoked.

MCP ENDPOINT

```
https://edge.vinkius.com/{token}/mcp
```

Claude · Cursor · VS Code · Windsurf · Grok · Gemini

Security Is the Architecture

Security in Vinkius is not a feature — it's the foundation of the runtime. The gateway enforces multiple independent protection layers between AI agents and third-party APIs.

01 — Ed25519 PKI Vault

Every workspace has an Ed25519 Master Key. Session keys are generated ephemerally (24h TTL) and signed by the Master Key. Credentials never leave the vault boundary.

02 — V8 Isolate Sandboxing

Tool code runs inside isolated-vm V8 isolates with 64 MB memory caps and per-request timeouts. No filesystem access, no network access except through the SSRF-guarded fetch bridge.

03 — SSRF Guard

All outbound HTTP requests are DNS-resolved and validated before execution. Private IP ranges (10.x, 172.16-31.x, 192.168.x, AWS metadata 169.254.x) are blocked at the network layer.

04 — DLP & PII Redaction

A ResponseGuard pipeline intercepts every tool response. Configurable redaction patterns strip sensitive fields (emails, SSNs, card numbers) before data reaches the AI agent.

05 — Cryptographic Audit Trail

Every request is signed into a SHA-256 hash chain with Ed25519 signatures. Events form a tamper-proof, SIEM-exportable forensic record.

06 — Honeytoken Trap System

Phantom credentials are injected into isolated environments. If a honeytoken is used outside Vinkius infrastructure, the server is quarantined instantly.

Emergency Kill Switch

EU AI Act Art. 14(1) Compliant

The kill switch is an **emergency halt** mechanism — not a simple toggle. When triggered, it executes three actions atomically:

01 — Server deactivated

The MCP server is immediately taken offline across the entire cluster.

02 — All tokens revoked

Every connection token is invalidated. Total lockout — reconnection blocked until new tokens are issued.

03 — WebSocket connections killed

Active connections terminated via Redis pubsub broadcast. Propagates to every runtime node in the cluster.

Full Visibility. Zero Guesswork.

The Vinkius cloud dashboard includes a full MCP Governance suite — real-time analytics and security controls for production AI operations.

Control Plane

KPI dashboard with request volume, latency, success rate, token consumption, and AI-generated operational briefings.

Agent Activity

Which AI clients are connecting, how often, and what they're doing — real-time session tracking.

FinOps

Cost tracking per tool, payload compression savings, budget optimization signals, and consumption trends.

Tool Health

Slowest and most error-prone tools, with actionable root-cause insights and performance baselines.

Firewall & DLP

PII redaction activity, sensitive data protection counters, and security event timeline.

Incident Log

Error trends, failure rates, status-code breakdowns, and forensic audit trail access.

> Get started at **cloud.vinkius.com** — connect your AI agent in under 60 seconds.

HTML DOM Query Engine MCP

1 tools available
Cloud-hosted on Vinkius

When you run into a huge e-commerce page—say, one with thousands of lines of HTML—and you only care about three things, like the product price and all the gallery images, passing that whole raw code block to your agent is bad news. It wastes tokens and often confuses the AI.

This MCP fixes that. You feed it the messy HTML alongside a specific CSS selector. The engine handles the heavy lifting of parsing the page structure, isolating only the data you asked for. You get back clean text or attributes directly, without any surrounding junk code. This capability is built on reliable native runtimes and makes scraping predictable.

Connecting this MCP through Vinkius gives your agent a dedicated tool to handle web data extraction cleanly. It means your workflow doesn't crash when it hits complex, poorly structured websites; it just gets the numbers or links you need.

Core Capabilities

01 — Extracting text content

It pulls out visible text from a web element identified by its CSS selector.

02 — Retrieving attributes

You can grab specific data points associated with an element, like the 'src' of an image or the 'href' of a link.

03 — Parsing complex selectors

The tool supports advanced CSS queries (e.g., targeting elements only inside another container) for pinpoint accuracy.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/html-dom-query-engine** — connect your AI agent in three steps.

01 You pass the raw HTML content of a webpage and specify exactly what you're looking for using a standard CSS selector string.

02 The MCP engine processes the entire payload, running the query against the DOM structure to locate all matches.

03 Your agent receives only the clean data—either the requested text or list of attributes—ready for immediate use.

The bottom line is you get structured data out of unstructured HTML without overloading your AI client's context window.

Built For

SEO analysts, web developers, and research associates need this. If your job involves reading the source code or scraping public websites for specific details, you know how painful it is to copy-paste giant blocks of HTML into an agent just to get one piece of text.

SEO Analyst

They use this to extract structured data like product names or category links from large site maps, ensuring their AI agent doesn't miss any necessary metadata.

Data Scientist

They rely on it when building web scraping pipelines that need predictable access to specific elements (e.g., a stock ticker or an article author) across dozens of different sites.

Content Auditor

They use this to validate page structure, checking if required attributes like canonical URLs or social share links are present on every single page type.

What Changes When You Connect

01 Saves tokens. Instead of dumping gigabytes of raw web content into your agent, this MCP processes the heavy lifting outside the LLM, keeping your context window clean and efficient.

02 Guarantees precision. By requiring a CSS selector, you tell the system exactly where to look (e.g., `.product-title`), minimizing the chance of irrelevant data being pulled in.

03 Handles attributes easily. Need all image sources? You don't have to parse them manually; this tool lets your agent grab every `src` or `href` attribute from a specified selector group.

04 Stops hallucination. Because the extraction happens via native code, the results are deterministic and factual, unlike when an LLM tries to guess data from raw HTML.

05 Supports complex targeting. You can use advanced selectors like `#main .price:nth-child(2)` to hit elements that only appear sometimes or in a specific order.

Real-World Applications

Collecting product link lists

An SEO analyst needs all the image URLs for a gallery. Instead of reading through thousands of lines just to find the `src` attributes, they run their agent with this MCP and specify `.gallery img`. The agent instantly gets a clean list of every single source URL.

Extracting pricing data

A researcher is compiling price comparisons across several competitor websites. They pass the raw HTML for each page to their agent, use this MCP with the selector `.price-display`, and consistently retrieve only the accurate dollar amounts.

Auditing documentation structure

A developer needs to find all internal links on a help page. They feed the HTML into the MCP and query for `a[href*='/help/']`. The agent returns only the relevant link texts and URLs, perfect for building an index.

Extracting headers or titles

A content curator needs to pull just the main title of several articles from a directory listing. They use the MCP with `h1` as the selector, and their agent gets back only the clean text for every matching article headline.

Patterns to Avoid

Passing raw HTML blocks

✗ AVOID

The developer copies a 15KB snippet of source code containing images, scripts, and main content into their agent prompt and asks it to 'find the price.' The LLM struggles with the noise, often hallucinating or getting confused by script tags.

✓ INSTEAD

Use `query_dom`. Pass the raw HTML block and use a specific selector like `.product-price` as input. The engine isolates *only* the data you want, ignoring all surrounding code.

Asking for generalized content

✗ AVOID

The user asks their agent to 'tell me what this webpage is about' using raw HTML. The agent spends massive tokens summarizing garbage and fails to give a concise answer.

✓ INSTEAD

If you need specific data, use `query_dom` with the selector for that element (e.g., `.article-summary`). If you just want general context, pass the text content of a known wrapper tag instead.

Manual scraping and copy/paste

✗ AVOID

The user has to open 50 web pages manually, right-click on the data point they need (like an image URL), and copy it into a spreadsheet. This is slow and error-prone.

✓ INSTEAD

Feed all 50 HTML payloads into your agent through Vinkius and let the MCP run `query_dom` for the attribute you want, like `img[src]`. You get results in bulk.

The Right Fit

Use this MCP when your primary goal is structured data extraction from unstructured HTML. If you know *what* element you are looking for (e.g., 'the price', 'all links'), and you can identify it using a CSS selector, this tool is perfect. It's fast, reliable, and saves tokens.

Do NOT use this if your task requires complex reasoning or interpretation of context that isn't tied to visible HTML elements. For instance, if you need the agent to 'summarize the tone' or 'explain the implications of X,' then a general text processing tool is better. If you just need to pull out data points—text, attributes, lists of URLs—this MCP and its `query_dom` tool are your best bet.

Copy-Pasting Web Code Into Your AI Agent

Right now, when you need a specific piece of information from a website, the process is tedious. You copy the URL, paste it into your agent, and watch it struggle to parse thousands of lines of raw HTML, complete with script tags, comments, and background CSS that means nothing to you. The result is often an expensive hallucination or a token limit error.

With this MCP, you stop sending garbage data. You give the engine the messy code and the precise address—the selector—of what you want. Your agent gets back only clean text or links; the rest of the web page vanishes.

The HTML DOM Query Engine gives you predictable, targeted element values.

You no longer have to waste time manually inspecting elements in your browser just to find a CSS selector. You write the selector once and use it repeatedly across multiple pages or data sets. This capability keeps your workflow moving without manual validation.

The difference is control. You move from guessing what data an agent might pull out, to demanding exactly what you need with absolute certainty.

HTML DOM Query Engine: 1 Tool Available

Use this tool to parse raw web page code and deterministically pull out specific data points using standard CSS selectors.

#	TOOL	DESCRIPTION
01	query_dom	Passes a raw HTML string and a CSS query to extract the matching text content or attributes from the web element.

See It in Action

Real prompts you can use once this MCP is connected to your AI agent through Vinkius Cloud.

U Extract the text from `.product-price` from this 5,000 line HTML file.

✅ **Matches Found:**
1. `$149.99`

U Extract all image source URLs (`src`) from the `.gallery img` selector.

✅ **Matches:** Extracted 12 `src` attributes successfully.

U Get the text inside the `<h1>` tag.

✅ **Matched Text:** 'Welcome to our API documentation.'

Frequently Asked Questions

01 **How do I use the HTML DOM Query Engine MCP for image URLs?**

You pass the raw HTML and use `query_dom` with a selector like `.gallery img`. The tool will then return all the source (`src`) attributes found on those specific image elements.

02 **Is the HTML DOM Query Engine MCP faster than just sending the whole page?**

Yes. By running the parsing in a native runtime, it skips processing massive amounts of junk data that would bog down your agent's context window and slow down response time.

03 **What if I want to extract text from an ID selector?**

You simply use `#your-specific-id` as the CSS query. The engine will target that element directly and return its clean, visible text content.

04 **Can this MCP handle very long HTML pages?**

Absolutely. It's designed to parse large payloads efficiently, making it ideal for scraping entire documentation sections or massive e-commerce product listings.

05 **Does the HTML DOM Query Engine MCP only support text extraction?**

No, it supports attributes too. You can query not just the text inside an element, but also its associated attributes like `href` or `data-id`.

Go Live in 60 Seconds

Get your connection token from **cloud.vinkius.com**, then paste the endpoint URL into any MCP-compatible client.

YOUR MCP ENDPOINT

```
https://edge.vinkius.com/[TOKEN]/mcp
```

CLIENT	WHERE TO CONFIGURE
Claude AI	Profile → Customize → Connectors → "+" → Add custom connector → Paste endpoint
Cursor	Settings → Features → MCP Servers → "+ Add New MCP Server" → Type: SSE → Paste endpoint
VS Code	Ctrl/Cmd+Shift+P → "MCP: Add Server" → add `"html-dom-query-engine": { "url": "…" }`
Windsurf	MCP Settings → `mcp_settings.json` → Add endpoint URL
ChatGPT	Settings → Tools & plugins → Add MCP server → Paste endpoint
Gemini	Extensions → Add MCP Server → Paste endpoint URL

ASK AN AI ABOUT THIS

Let your preferred AI explain this MCP server

- Ask ChatGPT
- Ask Claude
- Ask Perplexity
- Ask Gemini
- Ask Grok

READY TO CONNECT

HTML DOM Query Engine is live on Vinkius Cloud.

Get your connection token, paste it into your AI agent, and start building. No SDK. No deployment. Just results.

Start at cloud.vinkius.com →

vinkius.com · support@vinkius.com

INDEPENDENT PLATFORM DISCLAIMER

Vinkius is an independent platform and is not affiliated with, endorsed by, sponsored by, verified by, or otherwise authorized by HTML DOM Query Engine. All third-party trademarks, logos, and brand names are the property of their respective owners. Their use in this document is strictly for informational purposes to identify service compatibility and interoperability.

DOCUMENT INFORMATION

Generated	June 2026
MCP Server	HTML DOM Query Engine MCP
Server ID	`019e388d-2960-72c4-8ff4-287d2dfb0d70`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE

This document is generated automatically by the Vinkius PDF Engine. Content reflects the MCP server configuration at the time of generation and may change as updates are deployed. For the most current information, visit vinkius.com/mcp/html-dom-query-engine.