// MANUAL

SERPCTRL
// Crawler

Site crawler.
Native desktop. Pay once.

A Screaming-Frog-style site crawler. Spider any site or audit a URL list, then export everything to one XLSX. Native double-click app on macOS and Windows. €50, paid once. No subscription, no renewal, no account.

// CONTENTS

01 What it is and how to run it
02 Spider mode
03 URL list mode
04 Settings
05 The XLSX output
06 Schema audit
07 Controls and tips

// WHAT IT IS AND HOW TO RUN IT

What it is and how to run it

Two jobs. Spider crawl from a URL, walking every internal link. Or check a URL list from CSV, TSV, or TXT, no spidering. Either way, you get one XLSX with one row per page. Status, indexability, title, meta, H1/H2, canonical, robots, X-Robots-Tag, word count, response time, depth, inlinks and outlinks, plus a schema audit.

Screaming Frog is the industry standard, but it's €245 a year and gates anything over 500 URLs behind the paid licence. SERPCTRL // Crawler does the core 90% for €50, paid once, in a single double-clickable app.

// PRICING

SERPCTRL // Crawler	**€50, paid once.** No renewal, no licence file, no account.
Screaming Frog	€245 a year, per seat. Free tier capped at 500 URLs.
Sitebulb	~€1,330 a year (~€110/mo). Cloud-based.
Ahrefs Site Audit	~€2,148 a year (€179/mo). Bundled in the broader plan.

// INSTALL

- **macOS.** Download the .app bundle. Drag it to Applications. Double-click. First launch may need a right-click + Open to bypass Gatekeeper.
- **Windows.** Download the .exe. Double-click. SmartScreen may warn on first launch. Click *More info*, then *Run anyway*.
- **From source.** pip install -r requirements.txt, then python main.py. Or use ./run-dev.sh on macOS.

// WHERE THE OUTPUT GOES

macOS app	~/Documents/serpctrl-crawler/
Everywhere else	./crawls/ next to the executable
filename	crawl_<domain>_<timestamp>.xlsx for spider mode, urlcheck_<domain>_<timestamp>.xlsx for list mode

// FIRST RUN IN 30 SECONDS

1. Open the app.
2. Mode = Spider from URL.
3. Paste a URL.
4. Click Start Crawl.
5. Wait. Click Open Output Folder when done.

// MODE_01

Spider mode

The default. Pick **Spider from URL**, paste a start URL, click **Start Crawl**. The crawler walks every internal link, follows them depth-first, and stops at your *Max Pages* limit.

// WHAT GETS CRAWLED

- Links from <a href> on every page, resolved against the page URL.
- Same-registered-domain only. blog.example.com and example.com count as the same site (Public Suffix List match). cdn.example.org does not.
- URLs are normalised. Fragments dropped, host lowercased, trailing slash stripped, query string kept.
- robots.txt is fetched once per host and cached. Disallowed paths are skipped and recorded as *Blocked by robots.txt* with status 0.

// WHEN TO USE IT

- Full-site audits where you want every reachable page.
- Quick crawl-budget checks. Set *Max Pages* low to estimate scope before a real run.
- Discovery. You don't have a URL list and want one.

// DEPTH

Crawl depth is recorded per page. The seed URL is depth 0, links from it are depth 1, and so on. The XLSX is sorted by depth, so the homepage and top-nav links sit at the top.

// MODE_02

URL list mode

Switch to **Check URL list (CSV)**, click **Choose file...**, pick your file. The crawler audits exactly those URLs, no spidering, no link discovery. Useful when you already know which pages you care about.

// FILE FORMATS

`.csv / .tsv`	Delimiter is auto-detected from , \t ; |. Header row is auto-detected too.
`.txt`	One URL per line. Anything not starting with http:// or https:// is ignored.
`URL column`	If a header is present, the column named url, urls, loc, address, link, page, or similar is used. Otherwise the first column that looks like URLs wins.
`encoding`	UTF-8 with or without BOM. Bad bytes are replaced rather than failing.
`dedup`	Duplicate URLs are collapsed, original order preserved.

// WHAT YOU GET BACK

The same XLSX as spider mode, with the same 28 columns. The only difference: *Inlinks* shows zero for every page (no link graph was built), and *Crawl Depth* is always 0.

When you load a list, *Max Pages* auto-bumps to match the list length if it's set lower. The default of 500 won't truncate a 2,000-URL list unless you explicitly drop it.

// WHEN TO USE IT

- Re-checking a known set of URLs after a deploy.
- Auditing pages from a sitemap export, GSC export, or hand-curated list.
- Validating a URL migration. Old URLs in, new XLSX out, look at status codes.

// SETTINGS

Settings

Four knobs above the Start button. Defaults are sane for most sites. The two you'll actually tune are **Max Pages** and **Threads**.

Max Pages	Spider stops at this number. List mode auto-bumps it to match the list. Default **500**. For a full-site audit, set it well above your expected page count.
Threads	Parallel HTTP workers. Default **5**. Raise carefully. Most servers handle 10 fine, small VPSes get angry above that.
Delay (ms)	Pause between requests *per worker*. Default **0**. Set to 100–250ms for sites you don't own. Set to 500ms+ for the polite version.
Respect robots.txt	On by default. Off means the crawler ignores Disallow rules. Only turn it off on your own properties.

// MATH WORTH KNOWING

Effective request rate = **threads ÷ delay**. Five threads with 200ms delay = 25 req/s. Ten threads with no delay = however fast your CPU and the target server can keep up.

// ROBOTS.TXT BEHAVIOUR

- Fetched once per host, cached for the run.
- Disallowed URLs are skipped and recorded with status 0 and the marker *Blocked by robots.txt*. They still appear in the XLSX, so you can see what was excluded.
- If robots.txt doesn't exist or returns 4xx/5xx, everything is allowed.

// THE XLSX OUTPUT

The XLSX output

One sheet, frozen header row, AutoFilter on, sorted by crawl depth then URL. 28 columns. Open in Excel, Google Sheets, or Numbers and filter your way around.

// COLUMN MAP

Identity	URL, Status Code, Status, Redirect URL, Content Type.
Indexability	Indexability (Indexable / Non-Indexable), Indexability Status (Noindex, Canonicalised, Client Error, Blocked by robots.txt, etc.).
On-page	Title 1 + length, Meta Description + length, H1-1, H1-2, H2-1, H2-2, Canonical, Meta Robots, X-Robots-Tag, Word Count.
Performance	Size (bytes), Response Time (ms), Crawl Depth, Inlinks, Outlinks.
Schema	Schema Count, Schema Types, Schema Status, Schema Issues. Full breakdown on the next page.

// INDEXABILITY LOGIC

Computed from the same signals Screaming Frog uses. Status code, meta robots, X-Robots-Tag, and canonical-vs-self. A page is **Non-Indexable** when any of these say so, with the reason in the *Indexability Status* column.

// INLINKS / OUTLINKS

Outlinks is the count of unique internal links found on the page. **Inlinks** is the count of pages on the site that link to this URL. Inlinks is computed after the crawl finishes, so the column populates in the export, not live in the log.

// SCHEMA AUDIT

Schema audit

Every <script type="application/ld+json"> on the page is parsed, @graph wrappers are unwrapped, and each block is validated. Four columns land in the XLSX.

`Schema Count`	How many JSON-LD blocks the page contains.
`Schema Types`	Comma-separated @type values, in document order.
`Schema Status`	**NONE** (no JSON-LD), **OK**, **WARNINGS**, **CRITICAL**, or **PARSE ERROR**. The worst severity across all blocks wins.
`Schema Issues`	Compact [C] critical and [W] warning summary, one per finding. Filter this column to find pages with broken schema fast.

// WHAT GETS VALIDATED

Type-specific checks for the rich-result types Google actually surfaces. Article, Product, FAQPage, BreadcrumbList, Recipe, Event, LocalBusiness, VideoObject, JobPosting, Organization. Each gets bespoke rules. Required fields, ISO date and duration formats, TitleCase enforcement on @type, HTML inside string properties, relative URLs, and bare filenames.

// WHAT IT DELIBERATELY MISSES

The audit reads the **server-rendered** HTML the crawler fetched. JSON-LD that's injected by JavaScript at runtime won't appear here. That's the point. AI crawlers (GPTBot, ClaudeBot, PerplexityBot, CCBot) don't render JS either, so what this audit sees is what they see. If you need your schema cited in AI Overviews, it has to be in the SSR document.

// CONTROLS AND TIPS

Controls and tips

// BUTTONS

Start Crawl	Green button. Disables once a crawl is running.
Pause / Resume	Pauses workers without killing them. Safe for long crawls when you need to free up bandwidth.
Stop	Red button. Requests a graceful stop. The XLSX still gets written with everything crawled so far.
Open Output Folder	Shortcut to the export directory. Works after the crawl finishes.

// LIVE STATUS

- Progress bar fills as *processed / max pages* (spider) or *processed / list size* (list mode).
- Current URL shown below the progress label.
- Log box is colour-coded. Green for 2xx, yellow for 3xx, red for 4xx/5xx and errors.

// TIPS THAT SAVE YOU TIME

- For huge sites, run with low *Max Pages* first to estimate scope. Then bump it up for the real run.
- Don't crank threads on small servers. 5 to 10 is usually fine. Above that, watch for 5xx errors creeping into the log.
- If CSV detection misbehaves, save your URL list as a plain .txt with one URL per line. That path is the simplest and never fails.
- Filter the XLSX by *Schema Status* = CRITICAL to find broken schema across the whole site in one click.
- Filter by *Indexability* = Non-Indexable to find every page Google won't index, and the reason why.

// END OF MANUAL

€50 once. Native, no quota, no renewal. The crawl runs on your machine. The XLSX is yours.