> Crawl4AI Documentation (v0.4.3b2)

Home

Quick Start

Search

Crawl4AI: Open-Source LLM-Friendly Web Crawler & Scraper

Crawl4AI is the #1 trending GitHub repository, actively maintained by a vibrant community. It delivers blazing-fast, AI-ready web crawling tailored for large language models, AI agents, and data pipelines. Fully open source, flexible, and built for real-time performance, Crawl4AI empowers developers with unmatched speed, precision, and deployment ease.

> Note: If you're looking for the old documentation, you can access it here.

Quick Start

Here's a quick example to show you how easy it is to use Crawl4AI with its asynchronous capabilities:

```
import asyncio
from crawl4ai import AsyncWebCrawler

async def main():
    # Create an instance of AsyncWebCrawler
    async with AsyncWebCrawler() as crawler:
        # Run the crawler on a URL
        result = await crawler.arun(url="https://crawl4ai.com")

        # Print the extracted content
        print(result.markdown)

# Run the async main function
asyncio.run(main())
```

What Does Crawl4AI Do?

Crawl4AI is a feature-rich crawler and scraper that aims to:

1. Generate Clean Markdown: Perfect for RAG pipelines or direct ingestion into LLMs.
2. Structured Extraction: Parse repeated patterns with CSS, XPath, or LLM-based extraction.
3. Advanced Browser Control: Hooks, proxies, stealth modes, session re-use—fine-grained control.
4. High Performance: Parallel crawling, chunk-based extraction, real-time use cases.
5. Open Source: No forced API keys, no paywalls—everyone can access their data.

Core Philosophies: - Democratize Data: Free to use, transparent, and highly configurable.
- LLM Friendly: Minimally processed, well-structured text, images, and metadata, so AI models can easily consume it.

Documentation Structure

To help you get started, we've organized our docs into clear sections:

- Setup & Installation
  Basic instructions to install Crawl4AI via pip or Docker.
- Quick Start
  A hands-on introduction showing how to do your first crawl, generate Markdown, and do a simple extraction.
- Core
  Deeper guides on single-page crawling, advanced browser/crawler parameters, content filtering, and caching.
- Advanced
  Explore link & media handling, lazy loading, hooking & authentication, proxies, session management, and more.
- Extraction
  Detailed references for no-LLM (CSS, XPath) vs. LLM-based strategies, chunking, and clustering approaches.
- API Reference

Find the technical specifics of each class and method, including `AsyncWebCrawler`, `arun()`, and `CrawlResult`.

Throughout these sections, you'll find code samples you can copy-paste into your environment. If something is missing or unclear, raise an issue or PR.

How You Can Support

- Star & Fork: If you find Crawl4AI helpful, star the repo on GitHub or fork it to add your own features.
- File Issues: Encounter a bug or missing feature? Let us know by filing an issue, so we can improve.
- Pull Requests: Whether it's a small fix, a big feature, or better docs—contributions are always welcome.
- Join Discord: Come chat about web scraping, crawling tips, or AI workflows with the community.
- Spread the Word: Mention Crawl4AI in your blog posts, talks, or on social media.

Our mission: to empower everyone—students, researchers, entrepreneurs, data scientists—to access, parse, and shape the world's data with speed, cost-efficiency, and creative freedom.

Quick Links

- GitHub Repo
- Installation Guide
- Quick Start
- API Reference
- Changelog

Thank you for joining me on this journey. Let's keep building an open,

democratic approach to data extraction and AI together.

Happy Crawling!
— Unclecode, Founder & Maintainer of Crawl4AI

Site built with MkDocs and Terminal for MkDocs.