# Inkra

User Manual — Version 0.1

Inkra is a native Markdown editor for macOS built with SwiftUI and AppKit. Write in Markdown with real-time syntax highlighting, see a live preview with KaTeX formulas, Mermaid diagrams and code blocks, and use the built-in AI assistant for writing tasks like proofreading, shortening or translating. Inkra runs in the macOS sandbox and supports 20+ languages.

## Key Features

- **Native editor** — NSTextView with TextKit 2 and AST-based syntax highlighting
- **Live preview** — WKWebView with KaTeX, Mermaid and highlight.js
- **AI assistant** — Chat panel with Claude, OpenAI, Ollama and Inkra KI
- **Writing actions** — Proofread, shorten, expand, translate and more
- **File management** — Sidebar with file tree and document outline
- **Export** — HTML and PDF with full formatting
- **MCP server** — External automation via inkra-mcp-server

## Contents



Export documents as HTML or PDF.

7. Settings

   Configure theme, font, editor modes and debug API.

8. API & MCP

   Debug server, MCP server and external automation.

9. FAQ

   Answers to common questions about Inkra — formats, AI, privacy and more.

**Download as PDF**

Handbuch (Deutsch) Manual (English)

# Getting Started

## Installation

Download the DMG file from the product page and drag Inkra to your Applications folder. Alternatively, Inkra is available on the Mac App Store.

**System requirements:** macOS 14.0 (Sonoma) or later, Apple Silicon or Intel.

## First Launch

On first launch, Inkra opens an empty document. The interface has three areas:

- **Sidebar** (left) — file tree or document outline
- **Editor** (center) — Markdown text editor with syntax highlighting
- **Preview / Chat** (right) — live preview or AI chat panel

The status bar at the bottom shows word count, character count and document status.

## Your First Document

1. Write Markdown text in the editor
2. Open the preview to see the rendered result
3. Save with **Cmd+S** or via the menu

Inkra detects changes automatically and shows the status in the title bar.

# Editor

## Syntax Highlighting

The editor is built on a native NSTextView with TextKit 2. Markdown syntax is highlighted in real time — headings, bold, italic, links, code blocks and lists each receive distinct formatting. Highlighting is performed via an AST walk using the swift-markdown framework.

## Focus Mode

Focus mode dims everything except the current paragraph. Inactive paragraphs are faded so you can concentrate on the text you are writing. Enable it via **View → Focus Mode** or the toolbar.

## Wide Mode

By default, text width is limited to a comfortable reading column. Wide mode uses the full window width — useful for tables or long code blocks.

## Supported Markdown Syntax

| Element | Syntax |
| --- | --- |
| Headings | `# H1` through `###### H6` |
| Bold | `**text**` |
| Italic | `*text*` |
| Inline code | `` `code` `` |
| Code block | ` ```language ` |
| Link | `[Text](URL)` |
| Image | `![Alt](URL)` |
| List | `- Item` or `1. Item` |

| Element | Syntax |
| --- | --- |
| Blockquote | `> Text` |
| Table | `\| A \| B \|` |
| Horizontal rule | `---` |
| Task list | `- [ ] Task` |
| KaTeX formula | `$formula$` or `$$formula$$` |
| Mermaid diagram | ```` ```mermaid ```` |

# Preview

## Live Preview

The preview renders your Markdown text as HTML in real time. It runs inside a WKWebView and updates on every editor change via incremental DOM updates through JavaScript injection — no full page reload.

## KaTeX Formulas

Mathematical formulas are rendered with KaTeX. Inline formulas with `$E = mc^2$`, block formulas with `$$...$$`. KaTeX supports a large subset of LaTeX syntax.

## Mermaid Diagrams

Code blocks with the language `mermaid` are rendered as diagrams. Flowcharts, sequence diagrams, Gantt charts, class diagrams and other Mermaid diagram types are supported.

## Code Highlighting

Code blocks in the preview receive syntax highlighting via highlight.js. Language detection uses the identifier after the triple backticks (e.g. ```` ```swift ````). Many programming languages are supported.

# AI Assistant

## Chat Panel

The AI chat panel opens to the right of the editor. You can ask questions, get text suggestions or request revisions. Responses are streamed — you see the text as it is generated.

## Providers

| Provider | API Key | Description |
|---|---|---|
| Inkra KI | License key | Dedicated endpoint, no own API key needed (EUR 9.90/month) |
| Claude | Yes | Anthropic's Claude models |
| OpenAI | Yes | GPT-4o and other models |
| Ollama | No | Local LLMs, all data stays on your Mac |

Select the provider under **Settings → AI**. API keys are stored securely in the macOS Keychain.

## Writing Actions

Select text in the editor and choose an action from the context menu or toolbar:

| Action | Description |
|---|---|
| Proofread | Check spelling and grammar |
| Shorten | Reduce text to essentials |
| Expand | Elaborate on the text |
| Translate | Translate into another language |
| Summarize | Extract key points |

| Action | Description |
| --- | --- |
| Change tone | Make more formal or informal |

The result appears in the chat panel. From there you can insert it directly into the editor.

# File Management

## Sidebar

The sidebar offers two views via tabs at the top:

- **Files** — file tree of the opened folder with Markdown files
- **Outline** — document outline based on headings in the current file

Clicking an outline entry scrolls the editor to the corresponding heading.

## File Tree

Open a folder via **File → Open Folder**. The file tree shows all Markdown files in the folder and its subfolders. Click a file to open it in the editor.

## Document Status

The status bar shows the current document state:

- **Word count** — total number of words in the document
- **Character count** — total number of characters
- **Modified** — whether the document has unsaved changes

# Export

## HTML Export

Via **File → Export → HTML**, the document is exported as a self-contained HTML file. KaTeX formulas, Mermaid diagrams and code highlighting are included in the exported file.

## PDF Export

Via **File → Export → PDF**, the document is exported as a PDF file. The layout matches the preview — including formulas, diagrams and code blocks.

# Settings

## Editor

- **Font** — Choose the editor font
- **Font size** — Adjust the editor font size
- **Theme** — Light or dark mode for the editor
- **Focus mode** — Dim inactive paragraphs
- **Wide mode** — Full window width for the text area

## AI

- **Provider** — Inkra KI, Claude, OpenAI or Ollama
- **API key** — Key for the selected provider
- **Endpoint** — Custom endpoint for OpenAI-compatible APIs
- **License key** — For Inkra KI (monthly subscription)

## Debug API

Under advanced settings you can enable the debug server. This starts a local TCP server with token authentication for external automation.

- **Port** — TCP port for the debug server
- **Token** — Authentication token for access

# API & MCP

## Debug Server

The debug server is a local TCP server (NWListener) that can be enabled in settings. It provides a JSON-based API for external automation.

**Authentication:** Every request must include the configured token in the `Authorization: Bearer <token>` header.

| Endpoint | Description |
|---|---|
| `GET /health` | Check server availability |
| `GET /document` | Get current document |
| `POST /document` | Set document content |
| `GET /status` | Editor status (word count, file, etc.) |

## MCP Server

The MCP server (`inkra-mcp-server`) ships as a standalone CLI tool inside the app bundle. It implements the Model Context Protocol and communicates with the main app via XPC. AI assistants like Claude Code can access the editor through it.

Configuration is done through the MCP settings of the respective AI client. The server starts automatically when a client connects.

## Automation

External scripts and tools can access the editor via the debug server. Typical use cases:

- Read and write document content programmatically
- Query editor status (open file, word count)
- Integration into CI/CD pipelines or documentation workflows

- Control by AI assistants via MCP

# FAQ

## Which file formats does Inkra support?

Inkra works with Markdown files (.md). HTML and PDF are available for export. The preview renders KaTeX formulas, Mermaid diagrams and syntax highlighting for many programming languages.

## Is my text sent to a server?

Only when you use the AI assistant. The selected text is sent to the chosen provider. With Ollama, all data stays local. Without AI usage, no data leaves your Mac.

## What is Inkra KI?

Inkra KI is an optional monthly subscription (EUR 9.90/month) that provides a dedicated AI endpoint. No API key needed — AI features are available immediately.

## Can I use my own AI providers?

Yes, Inkra supports Claude, OpenAI, Ollama and any OpenAI-compatible endpoint. API keys are stored in the macOS Keychain.

## Does Inkra run sandboxed?

Yes, Inkra uses the macOS App Sandbox with file access and network entitlements. File access goes through the system security dialogs.

## How does the MCP integration work?

The MCP server (inkra-mcp-server) ships as a standalone process inside the app bundle and communicates with the main app via XPC. AI assistants like Claude Code can access the editor through it.