

	Ribbon collapse review ...	3af9275 · 2 hours ago
	M7/M14: comment ba...	2 weeks ago
	ci: add nix check work...	3 days ago
	fix(macos): skip wgpu ...	2 hours ago
	Russian proofing: spel...	4 days ago
	Italian: fixes from a na...	2 hours ago
	Ribbon collapse review...	2 hours ago
	Ribbon collapse review...	2 hours ago
	macOS packaging: stri...	4 days ago
	xtask assets: fail when...	4 days ago
	M0.1-M0.6: workspac...	2 weeks ago
	M14: shellcheckrc so ...	2 weeks ago
	Progress docs: target...	last week
	i18n(pl): citation style ...	9 hours ago
	Progress docs: target...	last week
	fix(ui): offer Save As fo...	3 hours ago
	Release 0.8.0	16 hours ago
	M0.1-M0.6: workspac...	2 weeks ago
	M0.1-M0.6: workspac...	2 weeks ago
	refactor(pdf): reconcil...	3 days ago
	#1963: wordcraft-cli r...	2 hours ago
	Comments: a draft sur...	2 hours ago
	M0.1-M0.6: workspac...	2 weeks ago
	M0.1-M0.6: workspac...	2 weeks ago

ARTCRAFT

WordCraft

Writing and document design; an open-source, clean-room reimplementation of Microsoft Word, rebuilt in pure Rust.

A fast, open-source word processor with the Word workflow you already know: the ribbon, styles, tables, track changes, references and mail merge. It reads and writes .docx, runs natively on macOS, Windows, Linux and BSD, and in the browser via WebAssembly.

By the *ArtCraft* team.

written in

Rust

runs on

macOS · Windows · Linux · BSD · Web

license

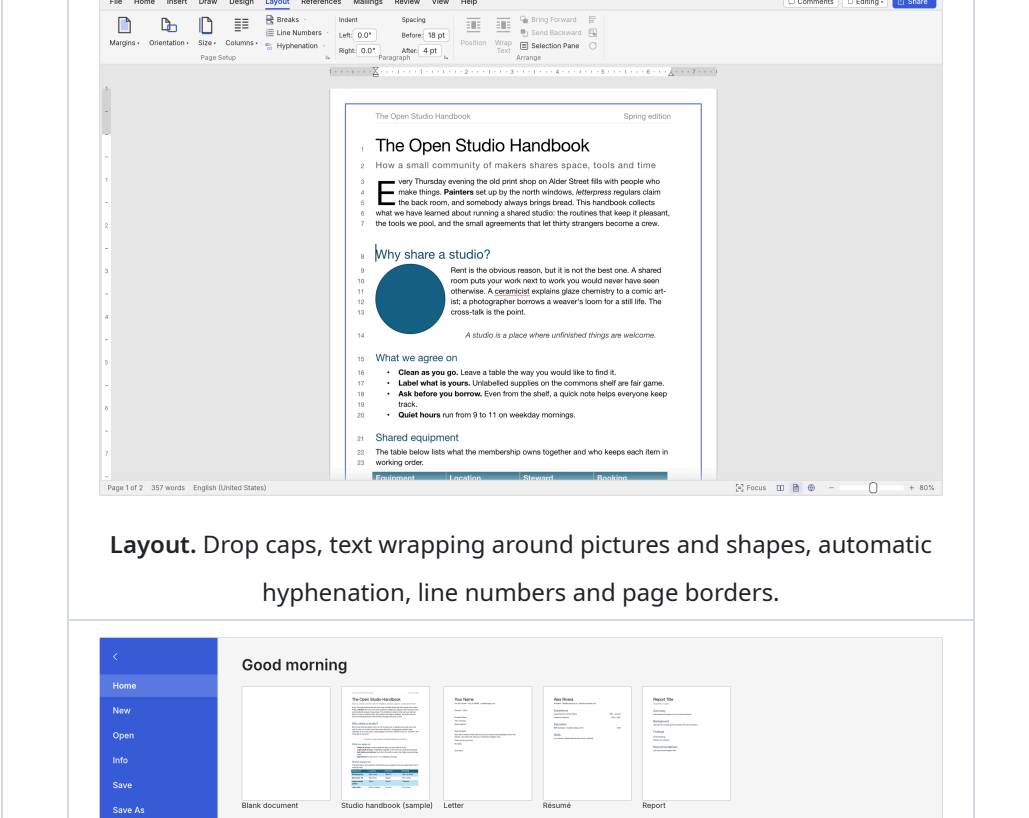
MIT / Apache-2.0

agents

MCP · CLI

[JOIN US ON DISCORD](#)

[WordCraft on getarcraft.com](#) · [ArtCraft](#) · [All Crafting Apps](#)



The Open Studio Handbook, WordCraft's built-in sample: the ribbon, the live Styles gallery, rulers and the Navigation pane.

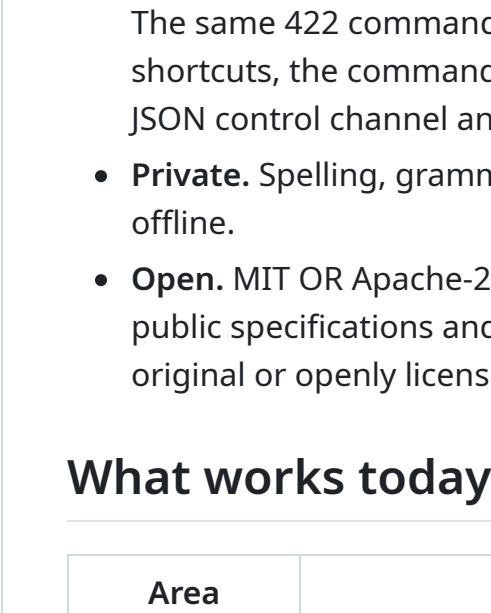
Note

ArtCraft is a community of artists from all walks of life. Painters, photographers, filmmakers, illustrators, designers, animators, hobbyists, and people who've picked up a pencil last week. If you make things, you're one of us. [Come say hi on Discord](#).

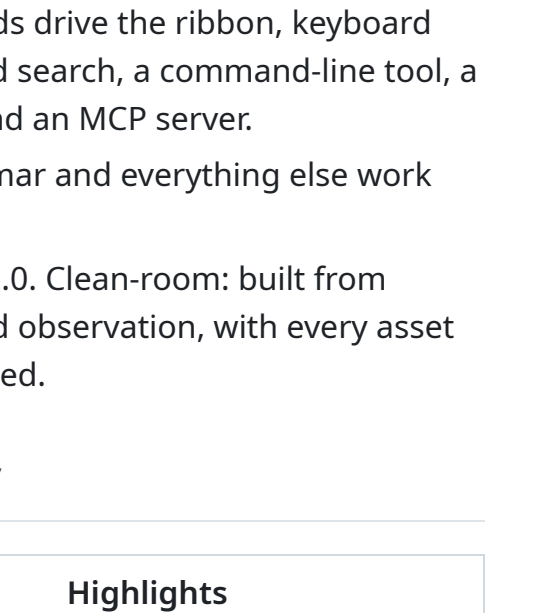
[A tour](#) · [Why WordCraft](#) · [What works today](#) · [Quick start](#) · [For agents](#) · [Architecture](#) · [Roadmap](#) · [Downloads](#) · [The Crafting Apps](#) · [License and credits](#)

A tour

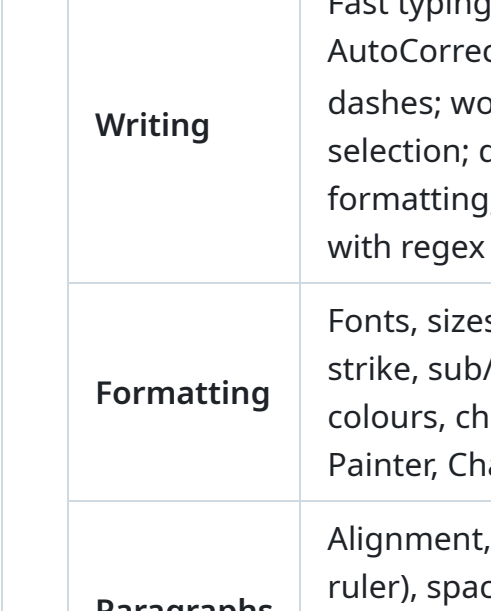
Every screenshot below is WordCraft itself, rendered offscreen by its own UI test harness (`cargo run -p wordcraft-ui-egui --example ui_shot`).



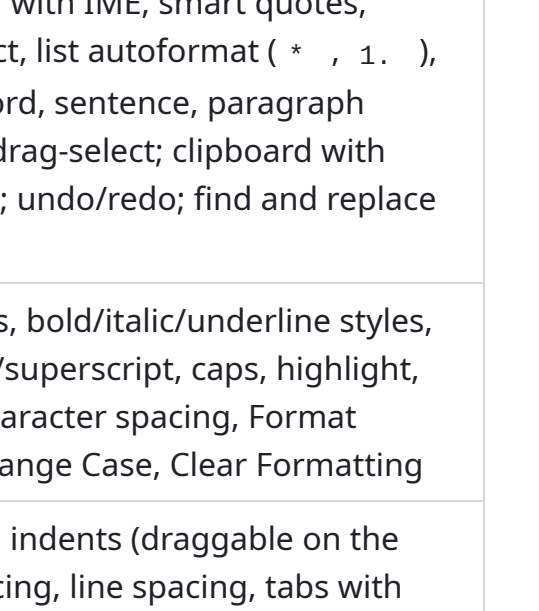
Review. Track changes, comment balloons in the margin, accept and reject, spelling and grammar as you type.



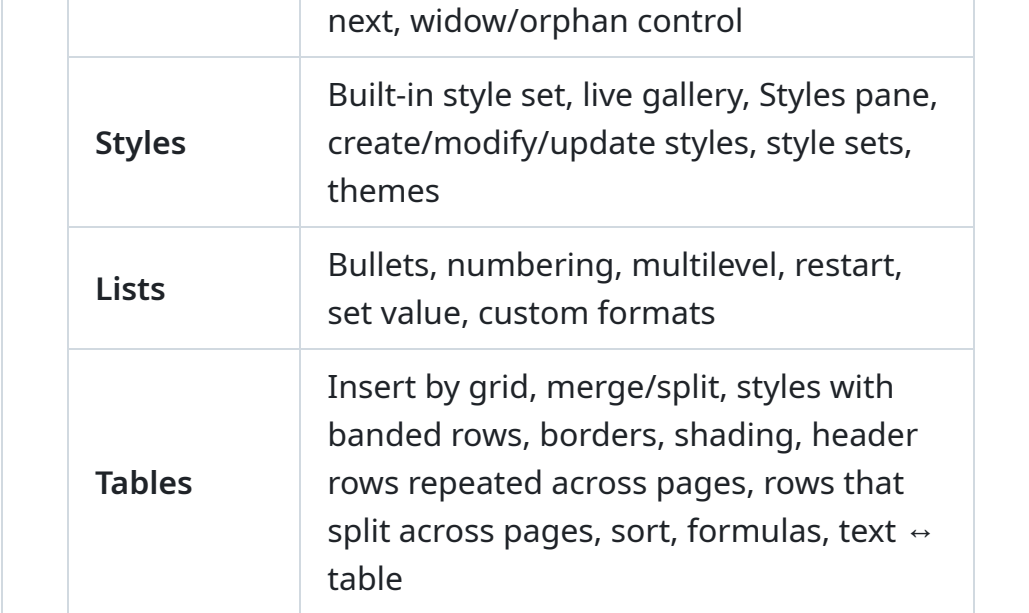
References. Tables of contents, footnotes, citations in APA, MLA, Chicago or IEEE, index and captions.



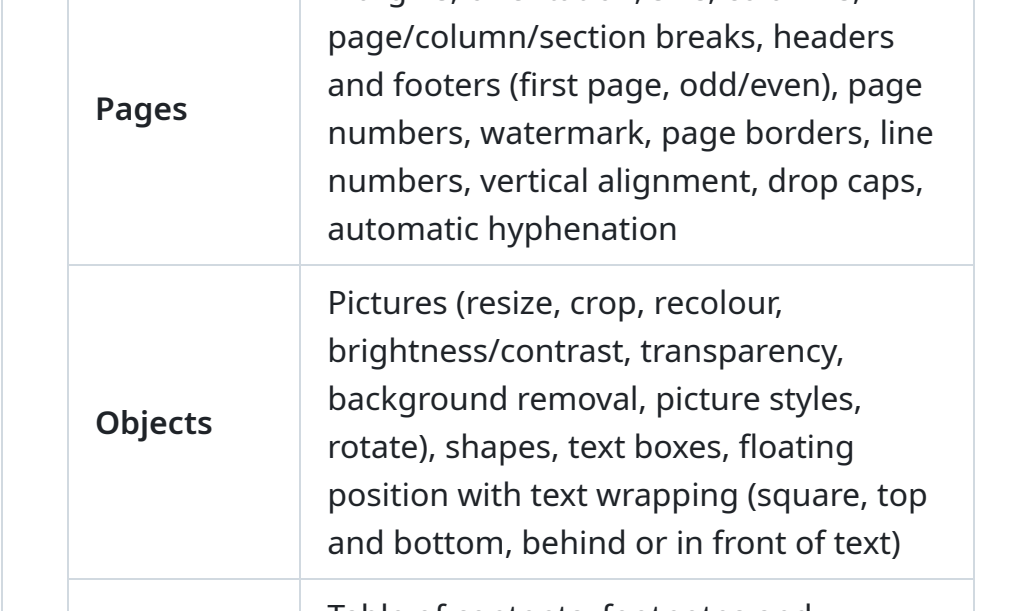
Design. Themes, style sets, paragraph spacing, watermarks, page colour and borders.



Dark mode with formatting marks, and the Insert tab: tables, pictures, shapes, links, headers, footers, fields and symbols.



Layout. Drop caps, text wrapping around pictures and shapes, automatic hyphenation, line numbers and page borders.



Good morning
File. Start from a template, open recent documents, edit properties, export to PDF and other formats.

Why WordCraft

- **Familiar.** Word's ribbon tabs, groups, shortcuts and behaviour: Enter continues a list, Tab demotes it, Ctrl/⌘+B bolds the word under the caret, the Styles gallery previews styles live, F4 repeats, F7 checks spelling, F8 extends the selection.
- **Your files.** Opens and saves .docx (OOXML), and also .odt, .rtf, .html, .md, .tex (LaTeX), .txt; opens Word 97-2003 .doc; exports PDF with real, selectable text, links and bookmarks.
- **Fast.** Paragraph layout is cached, so typing in a 188-page document re-lays it out in about 1.4 ms; pages render on demand.
- **Everywhere.** One Rust codebase for macOS, Windows, Linux, BSD and the web. No Electron, no Tauri: native [egui](#) on the GPU.
- **Built for agents.** Every action is a command with an id. The same 422 commands drive the ribbon, keyboard shortcuts, the command search, a command-line tool, a JSON control channel and an MCP server.
- **Private.** Spelling, grammar and everything else work offline.
- **Open.** MIT OR Apache-2.0. Clean-room: built from public specifications and observation, with every asset original or openly licensed.

What works today

Area	Highlights
Writing	Fast typing with IME, smart quotes, AutoCorrect, list autoformat (* · 1.), dashes; word, sentence, paragraph selection; drag-select; clipboard with formatting; undo/redo; find and replace with regex
Formatting	Fonts, sizes, bold/italic/underline styles, strike, sub/superscript, caps, highlight, colours, character spacing, Format Painter, Change Case, Clear Formatting
Paragraphs	Alignment, indents (draggable on the ruler), spacing, line spacing, tabs with leaders, borders, shading, keep with next, widow/orphan control
Styles	Built-in style set, live gallery, Styles pane, create/modify/update styles, style sets, themes
Lists	Bullets, numbering, multilevel, restart, set value, custom formats
Tables	Insert by grid, merge/split, styles with banded rows, borders, shading, header rows repeated across pages, rows that split across pages, sort, formulas, text ↔ table
Pages	Margins, orientation, size, columns, page/column/section breaks, headers and footers (first page, odd/even), page numbers, watermark, page borders, line numbers, vertical alignment, drop caps, automatic hyphenation
Objects	Pictures (resize, crop, recolour, brightness/contrast, transparency, background removal, picture styles, rotate), shapes, text boxes, floating position with text wrapping (square, top and bottom, behind or in front of text)
References	Table of contents, footnotes and endnotes, citations and bibliography (APA, MLA, Chicago, IEEE), captions, table of figures, cross-references, index, table of authorities
Review	Spelling and grammar with suggestions (English and Polish; by each run's language), thesaurus, word count, comments in margin balloons or a pane, track changes, accept/reject, compare documents, restrict editing, accessibility checker, document inspector
Mailings	Mail merge from CSV, merge fields, address block, greeting line, rules, preview, finish to a document; envelopes and labels
View	Print layout, web layout, draft, read mode, focus, zoom, one/multiple pages, page width, Navigation pane, rulers, gridlines, dark mode; interface in English, 简体中文, 繁體中文, 日本語, Português (Brasil), Español, Українська, Срpski / Srpski, Eesti, Türkçe, Русский, Italiano, Deutsch, Polski, Čeština, Magyar, Français, Български, Tiếng Việt, 한국어 or Svenska (follows the system language by default)
Files	.docx read/write (opens in Word), Word 97-2003 .doc import, PDF export, .odt, .rtf, .html, .md, .tex, .txt import/export, page images

The honest picture, area by area, is in [ROADMAP.md](#), the [parity assessment](#) and the generated [feature checklist](#).

Quick start

```
git clone https://github.com/storytold/wordcraft
cd wordcraft
cargo run --release -p wordcraft -- --sample
cargo run --release -p wordcraft -- report.docx
```

Command line:

```
wordcraft-cli convert report.docx report.pdf
wordcraft-cli list report.docx
wordcraft-cli inspect report.docx
wordcraft-cli run --template sample \
--cmd 'select.text={"text":"Membership"}' --cmd
wordcraft-cli run --script job.json
```

Web: `cd apps/wordcraft-web && trunk serve`, then open <http://127.0.0.1:8771/sample>.

For agents: CLI and MCP

WordCraft was designed to be driven by people *and* by AI agents.

```
claude mcp add wordcraft -- wordcraft-cli mcp
# confine the agent to folders: wordcraft-cli mcp
wordcraft --control 7981 &
claude mcp add wordcraft-app -- wordcraft-cli mcp
```

Tools include `list_commands`, `execute`, `batch`, `type_text`, `select_text`, `inspect_document`, `render_page`, `save_document`, and, with a running app, `screenshots`, `click` and `key`. Agents can check their work through `inspect_document` without screenshots. See [docs/mcp.md](#) and the [control protocol](#). Macros record any sequence of commands and play it back (`tools.recordMacro`, `tools.macros`).

Logs

The desktop app writes its log records to standard error and to `logs/wordcraft.log` next to its preferences: on Linux

```
$XDG_CONFIG_HOME/wordcraft/logs/ (by default
~/ .config/wordcraft/logs/ ), on macOS
~/Library/Application Support/WordCraft/logs/ , on Windows
%APPDATA%\WordCraft\logs\ . A start from a desktop menu or the Dock has no terminal, so this file is what to attach to a bug report: unreadable parts of a .docx, pictures or fonts a PDF export had to leave out and panics the command guard recovered from land there. Each launch moves the previous log to wordcraft.1.log (and that one to wordcraft.2.log ), so the log of a run that crashed survives the next start; a log stops growing at 16 MiB. --version writes no file, and runs with WORDCRAFT_NO_PREFS (agents' test runs) log to standard error only. Document text is never logged; a panic message leaves out the text it quotes.
```

By default WordCraft's own crates log at `info` and everything else at `warn`. `RUST_LOG` replaces that with `env_logger`-style directives, for example `RUST_LOG=debug`, `RUST_LOG=info,wgpu_core=warn`; a directive ending in `*` covers every target starting with it (`wordcraft*=debug`). The logger is `apps/wordcraft/src/logging.rs`.

Architecture

Layer	Crate	Job
L0	wordcraft-geom	units and measurements
L1	wordcraft-doc , wordcraft-fonts , wordcraft-proof	document model and editing; fonts and shaping; spelling, grammar, hyphenation
L2	wordcraft-layout , wordcraft-docx , wordcraft-formats	line breaking, pagination, tables, notes, hit testing; OOXML; ODT/RTF/HTML/Markdown/TXT
L3	wordcraft-render , wordcraft-pdf	rasteriser (vello_cpu); PDF (craft-pdf, Krilla)
L4	wordcraft-engine	session, undo, 422 commands, Word feature catalog
L5	wordcraft-mcp	MCP server
L6	wordcraft-ui-egui	the Word-style front end (swappable)
apps	wordcraft-cli , wordcraft-web	desktop, command line, browser

`cargo xtask ci` runs formatting, clippy, ~800 tests, the asset-attribution check, the layering check and the wasm build. Contributor and agent instructions: [AGENTS.md](#).

Roadmap

WordCraft is in **alpha**. It covers 88% of Word's ribbon features with commands today; counting depth, file fidelity, stability and the rest, we estimate it is about 60% of the way to replacing all of Word for real work, about 55% for a typical professional's weekly work and about 63% for everyday essentials. Beta needs testing against real-world .docx files, keeping charts and SmartArt through a round trip, pagination closer to Word's, a stability sweep and native printing. Details and estimates: [ROADMAP.md](#) and [where we fall short](#).

Downloads

Download WordCraft from GitHub: the [latest release](#) has every build listed below, and [all releases](#) has earlier versions and their notes. `<ver>` in the file names is the version number, and `SHA256SUMS.txt` lists a checksum for every file.

Windows

Build	Installer	Portable
x64 (64-bit Intel/AMD)	wordcraft-<ver>-windows-x64.msi	wordcraft-<ver>-windows-x64-portable.zip
arm64 (Snapdragon and other ARM PCs)	wordcraft-<ver>-windows-arm64.msi	wordcraft-<ver>-windows-arm64-portable.zip
x86 (32-bit)	wordcraft-<ver>-windows-x86.msi	wordcraft-<ver>-windows-x86-portable.zip

Installers and executables are code-signed.

If the app doesn't open on Windows: the desktop app initializes only DirectX 12 by default. Letting wgpu also create an OpenGL instance can crash some graphics drivers (AMD's `ati606xx.dll`) before the window appears, so the app would flash in Task Manager and quit. `WGPU_BACKEND` overrides the default for troubleshooting (for example `dx12` or `vulkan`). In PowerShell, from the folder containing the executable:

```
$env:WGPU_BACKEND = "vulkan"
& .\wordcraft.exe
Remove-Item Env:WGPU_BACKEND
```

An explicit `gl` override can bring the driver crash back on affected systems. The macOS, Linux and web backend defaults are unchanged.

The **32-bit** build uses OpenGL even when a DirectX 12 graphics card is present. On 32-bit Windows, creating a DirectX 12 graphics pipeline fails immediately and the app exits before the window appears. `WGPU_BACKEND=vulkan` (or `dx12`) still overrides that.

If the app quits right after its window appears with "Parent device is lost" in the log: WordCraft compiles DirectX 12 shaders with FXC, the compiler built into Windows, so it needs no `dxcompiler.dll` or `dxil.dll` and ignores any copy another program put on `PATH` (#603). `WGPU_DX12_COMPILER { fxc , dxc , auto }` overrides this; with `dxc`, a matching `dxcompiler.dll` and `dxil.dll` must sit next to `wordcraft.exe`. `WGPU_BACKEND = "vulkan"` also avoids DirectX 12 altogether.

If the app is slow on an older Windows PC: when DirectX 12 finds no graphics card with a driver for it (for example Intel 4th-generation graphics on Windows 11), it only offers Microsoft's software renderer, which draws every frame on the processor. In that case WordCraft uses OpenGL instead, without setting anything. The log file records the choice (lines starting with `graphics:`). `WGPU_BACKEND` still overrides it, e.g. `$env:WGPU_BACKEND = "gl"` or `"dx12"`.

macOS

Build	File	Notes
App, universal (Apple silicon + Intel)	wordcraft-<ver>-macos-universal.dmg	Signed and notarized
Command-line tool, universal	wordcraft-cli-<ver>-macos-universal.zip	Signed and notarized

Linux

Format	x86_64	aarch64 (/
AppImage	wordcraft-<ver>-linux-x86_64.AppImage	wordcraft-linux-aarch64.AppImage
Flatpak	wordcraft-<ver>-linux-x86_64.flatpak	wordcraft-linux-aarch64.flatpak
Debian/Ubuntu	wordcraft-<ver>-linux-x86_64.deb	wordcraft-linux-aarch64.deb
Fedora/RHEL/openSUSE	wordcraft-<ver>-linux-x86_64.rpm	wordcraft-linux-aarch64.rpm
Tarball	wordcraft-<ver>-linux-x86_64.tar.gz	wordcraft-linux-aarch64.tar.gz

RISC-V (riscv64): `wordcraft-<ver>-linux-riscv64.tar.gz`, a tarball cross-compiled for glibc 2.39+ (Ubuntu 24.04, Debian 13 and newer).

Wayland: on a Wayland desktop the app runs through XWayland when it is available, because winit (the windowing library) supports dragging files into the window only on X11. To use the native Wayland backend instead (for example for sharper text with fractional scaling, at the cost of file drag-and-drop), start it with `WORDCRAFT_WAYLAND=1`:

```
WORDCRAFT_WAYLAND=1 wordcraft
```

Arch Linux (AUR). Community members maintain three AUR packages; the project doesn't publish to the AUR itself. Install one of them:

Package	What it installs	Choose it when
wordcraft-bin	The release tarball above, prebuilt (x86_64, aarch64)	You want the latest release without compiling (quickest)

wordcraft	The latest release, built from source with cargo	You prefer to build releases yourself
wordcraft-git	The current main branch, built from source	You want unreleased changes and accepst an occasional broken build
For example yay -S wordcraft-bin or paru -S wordcraft-bin .		

Community packages

These packages are maintained by community members, not by the WordCraft team.

On Gentoo, the [snakebyte overlay](#) packages the Linux release as app-office/wordcraft-bin (prebuilt) and app-office/wordcraft (built from source):

```
eselect repository add snakebyte git https://gitl
emaint sync -r snakebyte
echo 'app-office/wordcraft-bin ~amd64' >> /etc/p
emerge --ask app-office/wordcraft-bin
```

FreeBSD

Build	File
x86_64	wordcraft-<ver>-freebsd-x86_64.tar.gz

Web (WebAssembly)

Build	File	Notes
Static site	wordcraft-web-<ver>.zip	Runs in a modern browser; host it on any static server

The Crafting Apps

WordCraft is one of the **Crafting Apps**: free, open-source creative tools from the [ArtCraft](#) team, each written from scratch in Rust and each able to stand on its own.

	App	What it's for	Code	Learn more
·	PhotoCraft	Image editing: layers, masks, type and real PSD files	GitHub	Website
·	VectorCraft	Vector illustration	GitHub	Website
·	FilmCraft	Video editing, color and sound	GitHub	Website
·	LightCraft	Photo library and raw development	GitHub	Website
·	PdfCraft	Reading, organizing and protecting PDFs	GitHub	Website
·	EffectCraft	Motion graphics and visual effects	GitHub	Website
·	DesignCraft	Page layout and publishing	GitHub	Website
·	WordCraft	Writing and document design · you are here	GitHub	Website

And [ArtCraft](#) itself, our AI image and video studio for artists who want real control.



Come make things with us

Our Discord is where artists of every kind hang out: people who paint, shoot, draw, cut film, set type, and people still figuring out what they like to make. Share what you're working on, ask for help, tell us what's broken, or tell us what you wish these tools could do. Whatever your medium and however long you've been at it, you're welcome here.

[discord.gg/artcraft](#) · [getartcraft.com](#) · [The Crafting Apps](#) · [WordCraft](#)

License and credits

WordCraft is dual-licensed under [MIT](#) or [Apache-2.0](#), at your option. Copyright (c) 2026 ArtCraft Team and the WordCraft contributors. Required notices are in [NOTICE](#).

Bundled fonts, icons, images and other assets keep their own open licenses; each one is listed with its author, source and license in [ATTRIBUTION.md](#).

The English spelling dictionary and hyphenation come from Grady Ward's public-domain Moby Hyphenator II word list. Polish spelling reads the Hunspell p1_PL dictionary installed on the system (not bundled), as do Russian, Portuguese, Czech, Italian, Croatian and Turkish spelling (how to install them: [docs/spelling-dictionaries.md](#)), and Polish hyphenation the TeX patterns by Hanna Kolodziejska, Bogusław Jackowski and Marek Ryćko (MIT option). The sample documents and templates are original text written for WordCraft.

The ArtCraft name, wordmark and logos in [docs/brand/](#) are trademarks of the ArtCraft Team and are not covered by this license. They may be used only unmodified, and only as part of this repository and WordCraft, under [docs/brand/LICENSE-brand.txt](#). Forks and modified versions must remove them.

Microsoft and Microsoft Word are trademarks of the Microsoft group of companies. Adobe, Photoshop, Illustrator, Premiere Pro, Lightroom, Acrobat, After Effects and InDesign are trademarks or registered trademarks of Adobe Inc. in the United States and/or other countries. WordCraft is an independent, open-source project and is not affiliated with, sponsored by or endorsed by Microsoft Corporation or Adobe Inc.; these names are used only to describe the workflows it is compatible with.



Made by the [ArtCraft](#) team and community.

