Junifye — The Publishing Engine

PUBLIFYE AS

Updated 2026-07-18

Contents

Chapter 1

What Junifye Is

This is a getting-started guide: the one place that takes you — or your AI — from nothing to publishing with Junifye. It is a starting point, not an exhaustive reference, and it grows as Junifye does — new features are added here as they ship.

Junifye is the *publishing* half of a two-part pipeline. The line is simple: **Darash studies Scripture; Junifye turns that study into a finished, published book.**

It is not a word processor, and it is not a "generate a book" button. In Junifye, an **AI assistant is the author** — it composes the book through small, validated tools, while you stay the author who directs, judges, and approves. You bring the idea, the outline, the voice; your assistant does the typing and works the typesetting machinery; and what comes out the other end is a real book.

1.1 What "a book" means here

From a single source, Junifye produces:

- a **typeset PDF**, in both a light and a dark edition
- a clean, responsive **web reader**
- all at a **permanent link**, stamped with your name

You never touch LaTeX or HTML or fiddle with fonts. You and your AI work in plain, structured content; Junifye renders it into a volume that looks professionally typeset — including **Hebrew, Greek, and right-to-left scripts** done properly.

1.2 Honest about what it is

"Markup into a typeset PDF" is not a new idea — many tools do it. What makes Junifye worth using is the *integration*: an AI authors it over MCP through self-correcting tools; it has first-class **Scripture primitives** (quote a verse, cite a Strong's number, drop in a Hebrew or Greek word); and it is **fed by Darash**, so the scriptural content in your book is verifiable, not invented. Take those away and it is just another markup-to-PDF pipeline. Together, they make it the natural *output* side of Darash.

Chapter 2

What You Get

When your book is finished, Junifye gives you a small, durable set of things.

2.1 Two PDF editions

Every book renders as a **light** PDF and a **dark** PDF from the same source — the dark edition is genuinely dark-themed for comfortable screen reading, not just an inverted page. Both are typeset by Tectonic, a modern LaTeX engine, so they look like real books: proper margins, a title page, a table of contents, and a footer on every page.

2.2 A web reader

Alongside the PDFs is a **standalone web reader** — responsive, with a light/dark toggle and a switcher to the PDF editions. It is the easiest way to share a book by link: a reader opens it in a browser, with nothing to download.

2.3 A permanent link

Each book lives at a **stable URL** that does not change as you edit — Junifye re-renders behind it whenever the content changes, so the link always serves the current version. A book is kept as long as it is in use (and on the paid tier a book can be marked permanent so it is kept for good); a long-untouched draft is eventually retired.

2.4 Multilingual, done right

Junifye typesets **Hebrew, Greek, Latin, Arabic and Farsi** with the correct fonts and direction — right-to-left books really run right-to-left, with Latin inclusions handled cleanly — and the reverse works too: a Hebrew or Greek verse dropped into an English book keeps its own script and direction, a whole Hebrew passage right-aligning on its own. This is the part most tools get wrong, and it is built in.

2.5 Print-ready

The same source also produces a **print-ready PDF** sized for Lulu and other print-on-demand services, so a book that began as a link can become a physical volume.

Chapter 3

Writing With Your AI

The heart of Junifye is the working loop between you and your assistant.

3.1 Your AI does the legwork

You do not type a book into Junifye. You *direct* one. You tell your assistant what you want — a chapter on a theme, a study of a passage, a rewrite in a plainer voice — and it composes the content through Junifye's tools while you read, react, and steer. The interface is deliberately lean, because the AI carries the weight.

3.2 It works in the same book, beside you

Connected over MCP (the next chapters cover how), your assistant authors directly into the live book: adding chapters, writing paragraphs, quoting Scripture, building the glossary. You watch the book take shape and say "tighten that," "add a section here," "this paragraph is wrong" — and it revises.

3.3 A browser editor for hands-on edits

When you want to touch the text yourself, Junifye has **Scriptorium**, a clean browser editor. You can open a single section, edit it, and watch the published reader update in place — no reload, no export step. Editor and reader stay paired.

3.4 Everything is versioned

Every change is **versioned** and checksum-locked. Nothing is lost; any chapter can be rolled back. You and your AI can work without clobbering each other — Junifye refuses a write that would overwrite an edit it has not seen, and retries cleanly. You stay the author; the machinery just keeps your work safe.

Chapter 4

From Manuscript to Published Book

A short tour of how a Junifye book is built, and how it becomes public.

4.1 The building blocks

A **book** holds metadata (title, author, language, page size) and an ordered list of **chapters**. Each chapter is a sequence of **blocks** — a heading, a paragraph, a list, a table, a Bible quote, a general quote. Inside a paragraph are **spans** — a bold or italic run, a link, and the Scripture primitives that make Junifye what it is.

4.2 Scripture primitives

These are first-class, not afterthoughts:

- a **Bible-quote** block sets a verse apart with its reference
- a **Hebrew or Greek word** span shows the word in full — its transliteration, its Strong's index, and the original script as one unit, like šālôm שָׁלוֹם[H7965]; a bare **Strong's** span shows just the index
- **cross-reference** and **glossary** spans link a word to other passages, or to the book's own glossary appendix

Because these are structured rather than hand-typed LaTeX, Junifye typesets them consistently and — paired with Darash — can verify them.

4.3 The glossary

Each book can carry a **glossary**: define a term once, link to it from anywhere, and Junifye assembles a Glossary appendix automatically.

4.4 Publishing and vetting

A book is private until you publish it. When you request publication, Junifye runs an **automated review** — a background AI session reads the whole book and, for any scriptural

claim, verifies it against Darash, while applying a content standard: clean content, and faithful, accurate Scripture. Only a book that passes is **listed** publicly. Listed or not, the owner always keeps the permanent link and the downloads.

Chapter 5

Connecting Junifye to Your AI

This chapter and the next are for the AI assistant; a technical reader will follow them easily.

5.1 The endpoint

Junifye speaks MCP at junifye.publifye.pro/mcp — a JSON-RPC 2.0 endpoint, authenticated with an API key sent as the ‘X-API-Key‘ header (or a PubHub bearer token). Access comes with **Junifye Access**, or with Darash Access, which includes it.

5.2 The authoring model: canonical JSON

The assistant never writes LaTeX or HTML. It writes **canonical JSON** through atomic tools, and one renderer turns that single source into the PDFs and the reader. There is a compact, round-trippable **source form** for chapters — read it with ‘chapter_get_source‘, edit the text, push it back with ‘chapter_set_source‘ — and you can call ‘source_syntax‘ at any time to get the exact block-and-span grammar, so the assistant never has to guess the format.

5.3 Checksum-locked, self-correcting

Mutations are **checksum-locked server-side**: the assistant never supplies a checksum, and if a write would overwrite an unseen edit, Junifye rejects it and the assistant retries once against the current version. Every mutation is validated as it is written, so a malformed block is caught at the point it is made — with an error that names the field and the fix.

5.4 The call shape

{”jsonrpc”:”2.0”,”id”:1,”method”:”tools/call”,”params”:{”name”:”chapter_get_source”,”arguments”:{”chapter_id”:”idcAbc123”}}}

The next chapter lists the authoring tools, grouped by what they build.

Chapter 6

The Authoring Tool Reference

The authoring surface, grouped by purpose. Call ‘tools/list‘ for the authoritative current set, and ‘source_syntax‘ for the block-and-span grammar.

6.1 Books and chapters

- **book_create** / **book_get** / **book_list** / **book_set** — create, read, and set a book’s metadata.
- **book_replace** — find-and-replace a literal string across every chapter (preview first).
- **chapter_create** / **chapter_list** / **chapter_set_title** / **chapter_move** / **chapter_delete** — manage the chapter list.
- **chapter_get_source** / **chapter_set_source** — read or replace a whole chapter as round-trippable source. The fastest way to author.

6.2 Blocks

- **block_list** — the chapter’s blocks with their ids.
- **block_get_source** / **block_set_source** — read or replace one block as text.
- **block_patch_text** — surgically swap an exact snippet inside a block, without re-sending the whole thing.
- **block_add_paragraph** / **block_add_heading** / **block_add_list** / **block_add_table** / **block_add_image** / **block_add_bible_quote** / **block_add_general_quote** / **block_add_stat** — add a block of a given kind (stat = labelled percentage rings).
- **block_move** / **block_delete** — reorder or remove a block.

6.3 Spans (inside a paragraph, footnote, or list item)

- **span_add_text** / **span_add_emph** / **span_add_strong** — plain, italic, and bold runs.

- **span_add_hebrew / span_add_greek / span_add_latin / span_add_strongs** — original-language words and Strong's citations.
- **span_add_bref / span_add_link** — a Scripture reference or a hyperlink.

6.4 Glossary

- **dict_add / dict_update / dict_list / dict_delete / dict_reorder** — the per-book glossary that feeds the Glossary appendix.

6.5 Publishing

- **publish_request** — submit the book for automated review.
- **vet_status** — the current verdict and whether the book is listed.
- **admin_rerender** — force an immediate re-render (admin), returning the fresh URLs.

Every tool validates its input and returns a clear, recoverable error when something is wrong — so the assistant can author confidently, correct itself from the error text alone, and never silently produce a broken book.

How this was made

This is the author's own work. It was composed with **junifye** and an AI assistant, drawing its Scripture and original-language work (Greek, Hebrew, cross-references) from **Darash** (Hebrew *darash*, "to seek, inquire, study"), a tool and reference work for the Bible in its original languages.

Both are **MCP** tools you can use from Claude Desktop or any AI assistant. Study the Word in its original tongues with Darash, read the Scriptures in **Bibleread**, and browse the public library.

A gift of support — Stripe

Scan to read this book online