Junifye — The Publishing Engine

PUBLIFYE AS

Updated 2026-08-10

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 **both of you write** — and *you* decide how much of the writing is your own. Some authors type every word themselves in the browser editor and call on their assistant only to check a reference, quote a verse, or tidy the structure. Some dictate the substance and let it draft. Some hand over an outline and judge what comes back. The dial is yours, it can be set differently for different chapters, and it can move at any time. What does not move is who the author is: you bring the idea, the voice, and the final word — 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**
- an **EPUB 3** for Kindle, Apple Books and Kobo
- a **plain-text** edition that opens on anything
- a **press-ready print PDF** with a full-wrap cover, sized for a real printer
- 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 Books, documents, and everything around them

A Junifye title can be a **book** — cover, table of contents, numbered chapters — or a **document**, a plain title and flowing sections. One setting decides. Around the text sits the rest of the platform: guest editors and groups who write alongside you, language editions that are kept in step, a public library with author pages and searchable questions, physical print with real ISBNs, and a lifecycle designed so that published work stays published.

1.3 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 — every one of them rendered from the same single source.

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. Sub-chapter headings nest and number themselves — 3.1, then 3.1.1 — and the same numbering drives a collapsible table of contents.

2.2 A web reader

Alongside the PDFs is a **standalone web reader** — responsive, with a light/dark toggle and a switcher to the other formats. It is the easiest way to share a book by link: a reader opens it in a browser, with nothing to download. Greek and Hebrew words carry a tap-or-hover popover with the transliteration and the lexicon code; every section has its own stable anchor, so you can link straight to a single paragraph from anywhere.

2.3 EPUB and plain text

The same source also produces an **EPUB 3** — reflowable, cover included, validated with epubcheck — for Kindle, Apple Books and Kobo; and a **plain-text** edition that opens on anything at all, down to a feature phone. Both sit beside the PDFs at their own permanent URLs.

2.4 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 published book gets a short permalink and a readable title slug as well. Books in use

are kept — every read refreshes the clock — published work is pinned permanently, and only a long-abandoned private draft is eventually retired.

2.5 Multilingual, done right

Junifye typesets **Hebrew, Greek, Latin, Arabic, Farsi, Cyrillic** and the rest 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. Scripture book names, the byline, the last-updated line and the closing colophon are all rendered in the book's own language. This is the part most tools get wrong, and it is built in.

2.6 Print-ready

The same source also produces a **press-ready interior PDF** and a **full-wrap cover** — back, spine and front, with the spine width computed from the real page count — sized for Amazon KDP, Lulu, IngramSpark, or a European short-run printer. With a real ISBN the wrap carries the book's own EAN-13 barcode. 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 You set how much help you want

You can type a book into Junifye yourself, and some authors do. More often 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. Both hands are on the same book, and how much each does is your call: the same author might draft one chapter by hand and hand the next over whole. The interface is deliberately lean because the assistant can carry as much of the weight as you give it — not because you are meant to hand over all of it.

3.2 It works in the same book, beside you

Connected over MCP (the later 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. A verse picker pulls Scripture straight from Darash into a quote block, an outline view lets you reorder chapters and sections by hand, and a book-switcher jumps between your own titles without leaving the editor.

3.4 Private notes that never reach the page

Alongside the text, a book or a chapter can carry **notes** — research, sources, decisions, things still to do. They are private scratch shared between you and your assistant: never

rendered into the PDF, the reader, or any export. A note can also be marked as belonging to the whole title, so it follows every language edition of it.

3.5 Everything is versioned

Every change is **versioned** and checksum-locked. Nothing is lost; any chapter can be rolled back. You can list a chapter's history, read any earlier version, and diff two of them — or simply ask what your last edit changed, narrowed to a single section, with the words removed and added marked inline. A revert is non-destructive: the old text returns as a new current version, and the edit you reverted from stays in history. 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, subtitle, author, language, page size, margins) and an ordered list of **chapters**. Each chapter is a sequence of **blocks**. Inside a paragraph are **spans** — a bold or italic run, a link, and the Scripture primitives that make Junifye what it is.

4.2 The blocks you can use

- **headings**, which nest by relative depth and number themselves — 3.1, then 3.1.1 — in the PDF and in the collapsible table of contents
- **paragraphs**, **lists** (ordered or unordered, and resumable when a quote interrupts the numbering), and **footnotes**
- **Bible quotes**, and **general quotes** carrying an author, a citation, and an optional link on the attribution
- **tables**, **images**, and **vector SVG figures** that stay live in the reader and the EPUB and rasterise crisply for print
- **stat blocks** — labelled percentage rings, one or two columns, plain or coloured
- **math**, inline or displayed

4.3 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.

4.4 The glossary

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

4.5 Questions a book answers

A book can also be attached to **questions** — the real questions a reader might ask, each with its own public page and stable link. Link a question to the book, or to the exact chapter that answers it, add a short teaser, and the book gains a discovery footprint: readers arrive at the question, and the question sends them into your text.

4.6 Publishing, vetting, and permanence

A book is private until you publish it — the URL works for anyone holding it, but the book stays out of the public library and only you can edit 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. An admin makes the final call, and only an approved book is **listed** publicly. A rejection comes back with its reason, so you can fix it and submit again.

Publishing is **permanent**, and deliberately so. A published book cannot be deleted by its owner, it is pinned against retention, and its text freezes as canonical. You get **ten self-service revision windows**: open one, edit, close it, and the new text refreezes as the canonical version. Past that, an operator has to step in. Any ISBN assigned to the book is final. Submit only work you are ready to stand behind — and listed or not, the owner always keeps the permanent link and the downloads.

Chapter 5

Writing With Other People

Junifye is built for more than one author. There are four ways to let someone else into a book, and they differ in exactly one thing: how much control comes with the access.

5.1 Guest editors

Invite another Junifye user as a **guest editor** and they gain content access to that one book — chapters, sections, prose. It appears in their own library and their book-switcher, so it is genuinely theirs to work on. What they never gain is control: publication, branding, print setup, deletion, and the guest list itself all stay with the owner. Invitations go by name through a native user directory, and any user may opt out of appearing in it. A guest can leave on their own initiative; nobody can add themselves.

5.2 Groups

A **group** is a named set of Junifye users. Attach the group to a book and every member becomes a content editor of it at once — resolved live, so adding somebody to the group grants access the same moment, and removing them takes it away again. Membership is by consent: you invite, they accept or decline.

A group has an owner and any number of admins; it can be renamed, transferred, and dissolved, and members can be removed or leave. A group can also be set to **autoshare**, so every new book any of its members creates is attached to it from the moment it exists — the natural arrangement for a team that writes everything together.

5.3 Wiki mode

A published book can additionally be marked **publicly editable**: any authenticated user may then edit it, the classic wiki arrangement. It is owner-settable and reversible, and it requires the book to be published first. To keep abandoned wikis from drifting, a public group-editable book left untouched past thirty days locks itself; the owner can reopen it, and the clock restarts.

5.4 Editorial review

Separate from the publication pipeline there is an opt-in **editorial workflow**. Mark a book *ready* and it appears in an editor's queue; name the editor you would like, or assign a specific Junifye user — which also grants them edit access, so the hand-off is a single step. When they have been through the book they mark it *approved*. It is a guide, not a lock: anyone with edit rights may still edit or sign off. Not every book needs this. It is there for the ones that have a real editor.

5.5 Handing a link to a human

Not everyone has an AI assistant wired to Junifye. For those people the owner can mint a **web-editor link** gated by a four-digit PIN — the url and the PIN shared separately, the PIN typed by hand rather than pasted. Four wrong entries disable the link permanently, and minting a fresh one takes a moment. The link locks itself after an idle window, and can be revoked outright at any time.

Chapter 6

Editions and Translations

One title, several languages — kept honest rather than merely duplicated.

6.1 Linking editions

Any book can be declared a **translation of** another. The two then behave as language editions of a single title: the reader offers a language picker, the library card shows which languages exist, and every edition lists its siblings with their own links and downloads. A reader who lands on the Norwegian edition is one click from the English one.

6.2 Keeping editions in step

Translations drift. You fix a paragraph in one language and forget the other, and six months later the two books quietly say different things. Junifye tracks a **reconciled baseline** for every chapter in every edition and reports the drift: which chapters have moved in which language, a same-language diff of exactly what changed since the baseline, and the full current text of both sides so the lagging edition can be rewritten against it. When the editions agree again you mark the chapter reconciled, and the baseline advances.

There is deliberately no cross-language diff — comparing Norwegian prose against English prose line by line tells you nothing. What you get instead is: *this* side changed, *here* is what changed in it, and here is the other side as it stands.

6.3 Translating without breaking the markup

A chapter reads out as flat, round-trippable source in which the Hebrew, Greek, Strong's and Scripture-reference spans appear as tidy, self-contained islands. A translator — human or AI — rewrites the prose around them and leaves the islands untouched. The structure, the citations, and the typesetting all survive the translation intact.

6.4 What localises itself

Some things never need translating at all. Scripture book names are rendered in the book's own language, the byline word, the last-updated line and the closing colophon

follow suit, and the page furniture — reading direction, body font, page-number placement — flips for a right-to-left edition without anyone having to ask for it.

Chapter 7

Print, Covers and ISBN

A link is a fine place for a book to live. Sometimes it needs to be an object you can hold.

7.1 Printer presets

Setting a book up for print is one call. Pick a **printer preset** — Amazon KDP, Lulu and IngramSpark at the common US trims, or a European short-run house such as Drukātava at A5 — and Junifye persists that geometry as the book's own; afterwards, regenerating the print is a single bare call.

The preset carries everything the printer actually cares about: the trim size, the bleed rule (KDP's text interiors want an exact trim with *no* bleed; Lulu and IngramSpark want the standard eighth of an inch; most European printers want five millimetres), the printer's own gutter ladder so the binding margin grows with the page count, and the page-count bounds it will accept. A custom trim in inches or millimetres is there for anything unusual, and print mode switches the interior to asymmetric inner and outer margins and a true single-plate black.

7.2 The press-ready report

Every print render comes back with a report you can act on: the trim size, the page count, whether that count is even and inside the printer's bounds, whether the black is true DeviceGray, whether the file carries a PDF/X output intent, whether every font is embedded, and any image that falls below 300 DPI. Problems are named, not hidden — the point of the report is that you find them before the printer does.

7.3 The wrap cover

Alongside the interior, Junifye generates a **full-wrap cover**: back, spine and front as one PDF, with the spine width computed from the book's real page count. Your uploaded cover art is placed if you have one and a typeset front is composed if you do not, the description goes on the back, and the barcode zone is handled properly — with a **print ISBN** on the book the wrap prints that number's EAN-13, and without one the KDP presets reserve the white keep-out that KDP's own barcode needs. The digital ISBN Junifye

assigns is not the one that prints: a printed edition carries its own. Interior plus wrap is the complete upload.

7.4 Real ISBNs

Junifye can assign a genuine, pool-allocated **ISBN-13** from a legally registered block. It appears on the colophon page and as the publisher recorded in the EPUB — it is the number of the **digital** edition. A printed edition needs a different ISBN of its own: that one is yours to obtain and you record it as ‘print_isbn‘, because Publifye issues digital ISBNs only. Taking one makes Publifye the publisher of that book, and Norwegian legal deposit follows from that — which is the real constraint here, not the supply of numbers. So it is default-denied and multiply gated: the book must be published and yours, the owner on the paid plan, with room inside their book cap, and the author must accept the publisher terms for that book — one ISBN per book, final once made. Every refusal names the exact next step rather than simply saying no.

7.5 Covers, logos and share cards

A book can carry a **cover image**, stored unmodified at print grade, shown at the top of its web reader and available at a stable URL for the print wrap. An uploaded cover is vetted by vision for a legible title; one without title text is composed into a proper title band for the EPUB, so no edition ever ships without a real cover.

Separately, an author can upload a **brand logo** — a church, an imprint, an organisation — which replaces the default initials monogram on the social share card generated for every one of their books. The two never mix: the cover belongs to the book, the logo belongs to the author.

Chapter 8

Being Found

A published book joins a public library, and the library is built to be searched — by people and by machines.

8.1 The library and search

Listed books appear in the public library with their covers, languages and reading time. Search matches titles and authors always, and — when you ask for it — the body text of every book. It folds case, accents and diacritics as it goes: *dap* finds *dåp*, *jose* finds *José*, and unpointed Hebrew matches pointed text. A hit inside a book tells you which chapter it landed in and shows the surrounding snippet, and the chapter id it hands back is a deep link straight to that spot.

8.2 Author pages

Every author can have a **public page** at their own handle: a display name — their own, or a church or organisation — a portrait or logo, a blurb, and every listed book they have written. Pages are opt-in: hidden until published, hidden again on request, and a hidden page leaves the sitemap as well as the index. The published ones are collected in a public author index that links back into the library.

8.3 Questions

Questions are the other door in. Each question — "What does the Bible say about baptism?" — gets a public page, a stable slug, free-form topic tags, and one or more books linked as its answers, ranked, each with a short teaser while the full answer stays in the book. A question with no answers yet is simply an open question waiting for one. Delete or heavily edit the chapter that answered a question and it is flagged for review rather than left quietly wrong.

8.4 Deep links

Every section of every book has a stable anchor that survives edits, so one book can link into a precise paragraph of another — or a question can point at the exact passage that

answers it. The permalink, the title slug and the raw UUID form all resolve the same anchor.

8.5 Machine-readable by design

The library ships a sitemap, structured JSON-LD on every book, a generated social share card per title, and an llms.txt that describes the corpus to AI agents that come looking. Changed pages are pushed to search engines through IndexNow rather than waiting to be crawled.

Chapter 9

Safety, Ownership and the Long Run

Publishing is a promise to a reader. These are the parts that keep it.

9.1 Nothing is lost

Every chapter edit is a new version carrying a checksum, an author and a timestamp. History, diffs and non-destructive reverts are always available, and a write that would overwrite an edit the writer has not yet seen is refused outright rather than merged badly.

9.2 Deleting, and un-deleting

Deleting a private book puts it in a **36-hour trash window**. It disappears from every listing immediately and can be restored intact until the window closes, after which it is purged for good. A published book cannot be deleted by its owner at all — removing published work takes an operator, on purpose.

9.3 Freezing and revision windows

Publication freezes a book's text as canonical. The owner gets ten self-service **revision windows** — open one, edit, close it, and the new text refreezes — and past that an operator must intervene. An admin can also freeze any book outright, overriding every other permission on it. Reading is never affected by a freeze; only writing is.

9.4 Provenance

Every book records its **origin**: *original* — the owner's own authored work — or *transcript*, captured external material such as somebody else's recorded talk. A transcript is source material: quote it with attribution, never present it as your own writing. Junifye enforces that rather than trusting it. A transcript cannot be published at all — the attempt is refused at every entry point, including the admin one — and the tag travels with the book through every listing.

9.5 Taking something back

If something has to come down, an admin can unpublish it and then **rotate its share links**. The short code always changes, and by default the content UUID is re-minted too, so every previously shared reader and PDF URL stops resolving, the old artifacts are purged from disk, and the book re-renders under a new identity. Copies already downloaded cannot be recalled, and Junifye says so plainly rather than implying a takedown is total.

9.6 Durability and access

Books live in Redis with their rendered artifacts on disk and a mirror in S3; an operator dashboard compares all three and reports exactly where they have drifted apart. Reading a book in the browser is always free and always ungated. An instance-wide setting can require a free reader account before a PDF or EPUB *download* — it is off by default, and it never touches the reader itself.

Chapter 10

Connecting Junifye to Your AI

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

10.1 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.

10.2 Start with the house style

Before writing anything, call ‘house_style‘. It is the one-stop authoring guide: what Junifye considers a publishable book, how a book differs from a document, the Scripture conventions, a worked example chapter, and the complete block-and-span grammar. The vetter judges on those same grounds, so reading it first is the cheapest way to pass review. ‘source_syntax‘ returns the grammar on its own.

10.3 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, the reader, the EPUB and the plain-text edition. There is a compact, round-trippable **source form** for that content — read one block, a whole chapter, or the whole book in a single call, edit the text, and push it back. A read starts with ‘book_outline‘, which reports every chapter’s size and block count and carries no text at all, so the assistant knows what it is about to pull before it pulls it.

10.4 Big books travel over REST

Whole-book export and import do not go through MCP. Ask for an export and you get a one-time ticket: download the book as an editable bundle — a flat source and an id-independent checksum per chapter, plus a merkle root for the book. Edit it offline and POST it back, and the server runs a conflict-safe three-way gate per chapter and returns

a manifest — applied, skipped-unchanged, conflict, rejected-invalid, unmatched, untouched. Chapters that changed on the server since your export are reported as conflicts, never clobbered. A malformed chapter is rejected atomically. Nothing is ever deleted.

10.5 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.

10.6 The call shape

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

The next chapter shows the source form itself, block by block; the one after it lists the authoring tools, grouped by what they build.

Chapter 11

The Source Form, By Example

Everything an assistant writes goes in through one compact source form: read it back, edit the text, push it in. This chapter is that form shown rather than described. ‘source_syntax‘ returns the same grammar as a tool call, and ‘house_style‘ returns it with the editorial rules attached.

11.1 The one rule

A blank line separates blocks, and no block ever contains one. That is the whole document model. A paragraph break is a new paragraph block — never an empty line inside a paragraph. Junifye refuses the second kind outright, because a blank line inside a block would swallow half of it on the next read.

The first line of a block picks its type: a line starting with a hash is a heading, a line starting with a colon directive is that kind of block, and anything else is a paragraph.

11.2 Headings

```
# The Word made flesh
```

Heading level is *relative nesting depth*, not a font size. The first heading in a chapter is the top sub-chapter whatever level you wrote, and a deeper one nests under it. The reader numbers them — 3.1, then 3.1.1 — in the PDF and in the collapsible table of contents, with no holes.

11.3 A paragraph, and the spans inside it

```
The Gospel opens with a claim. John reaches for \[g:G3056:logos:λόγος\]
— a word his readers already owned — and hands it a person. Compare
\[ref:John 1:1|ESV\]; \*note\* the \*\*shift\*\* at verse 14, and read \[a:the pro-
logue|https://example.org/prologue\] beside it.
```

The spans, in full:

- *x* is italic, **x** is bold

h:H7965:shalom:שָׁלוֹם and [g:G3056:logos:λόγος] are the FULL original-language form — Strong's index, transliteration, original script, as one tidy unit. Always prefer it: it lets a reader who cannot read the script still say the word.

h:שלום and [g:λόγος] are the bare-script form. Avoid it unless the book's own language *is* Hebrew or Greek.

H7965 or [G3056] alone cites a Strong's number and nothing else

l:Hello is a Latin island inside a right-to-left book

ref:John 3:16 is a Scripture reference; [ref:John 3:16|ESV] tags the translation

a:visible text|url is a link, as in the paragraph above; the target must be http, https or mailto, and a bare URL dropped into prose is rejected rather than rendered

- \ escapes any of \ * [ ]

11.4 Quoting Scripture

:bible John 1:14 | ESV And the Word became flesh and dwelt among us.

The reference comes first, the translation tag after the pipe is optional, and the body lines are the verse text. One quotation per block — commentary belongs in its own paragraph after it, never inside the quote.

11.5 A quotation from anyone else

:quote Athanasius | On the Incarnation | https://example.org/incarnation
He became what we are, that he might make us what he is.

Author, citation and URL are each optional and pipe-separated; the URL, when present, links the attribution line.

11.6 Lists

:list unordered - what the Word was - who the Word became

Use ':list ordered' for numbers. Every list block counts from 1 on its own, so a numbered run that a quotation interrupts must resume with ':list ordered start=6' — otherwise the continuation silently restarts at 1.

11.7 Tables

:table | Two witnesses | Verse | Claim | | — | — | | John 1:1 | The Word was God |

The caption after the pipe is optional, and the separator row marks the header row above it.

11.8 Percentage rings

:stat color=scale columns=2 | 73 | Occurrences in John | of 331 across the New Testament |

One ring per row: the value, a label, an optional description, and an optional colour of its own.

11.9 The rest, in one line each

- ':footnote' — a footnote; the body lines are spans
- ':math display' or ':math inline' — the next line is LaTeX
- ':image <asset-id> w=0.85 | caption' — a raster image
- ':figure <asset-id> w=0.85 | caption' — a vector SVG figure

You never invent an asset id. 'image_upload_begin' and 'figure_upload_begin' hand back a one-time URL, the bytes travel over REST, and the block is placed in the chapter for you.

11.10 Pushing it back

One block goes back with ‘block_set_source‘, a whole chapter with ‘chapter_set_source‘, and a chapter write is atomic: if any block is malformed the entire chapter is rejected, nothing changes, and the error names the block that broke. ‘chapter_get_source‘ returns exactly these blocks joined by the blank lines that separate them — so what you read is what you write.

Chapter 12

The Authoring Tool Reference

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

12.1 Books and chapters

- **book_create** / **book_get** / **book_list** / **book_search** / **book_set** — create, read, browse, and set a book’s metadata, render settings, provenance, visibility and life-cycle, one key at a time.
- **book_outline** — every chapter’s size and block count, with no text; the cheap planning view before a read.
- **book_get_source** — many chapters’ round-trippable source in one call, paged to fit.
- **book_replace** — find-and-replace a literal string across every chapter, whole-word by default, with a dry run first.
- **book_export_begin** / **book_import_begin** — download the whole book as an editable bundle and push it back through a conflict-safe three-way gate.
- **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.
- **chapter_history** / **chapter_version_get** / **chapter_diff** / **chapter_revert** — the version trail: list it, read a version, see exactly what an edit changed, undo it non-destructively.

12.2 Blocks

- **block_list** — the chapter’s blocks with their ids; each id doubles as the reader’s stable anchor.

- **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_figure / block_add_bible_quote / block_add_general_quote / block_add_stat** — add a block of a given kind.
- **block_set_table / block_set_stat / stat_add_item** — edit the block kinds that are not plain prose.
- **list_add_item / list_move_item / list_delete_item** — work inside a list block.
- **block_move / block_delete** — reorder or remove a block.

12.3 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.
- **span_move / span_delete** — reorder or remove a span.

12.4 Media and branding

- **image_upload_begin / figure_upload_begin** — a raster image, or a sanitised vector SVG figure; both stream over REST and land as a block in a chapter.
- **book_cover_upload_begin / book_logo_upload_begin** — this book's cover art and its logo.
- **owner_logo_upload_begin** — your account-wide brand logo, used on the social share card of every book you own.

12.5 Glossary and notes

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

- **note_set** / **note_get** / **note_list** / **note_patch** / **note_delete** — private authoring notes, never rendered into the book.

12.6 Working with other people

- **guest_find_user** / **guest_add** / **guest_list** / **guest_remove** — guest editors on one book.
- **group_create** / **group_invite** / **group_accept** / **group_decline** / **group_list** / **group_get** / **group_rename** / **group_member_remove** / **group_leave** / **group_admin_set** / **group_autoshare_set** / **group_transfer** / **group_invite_cancel** / **group_delete** — groups and their membership.
- **book_group_add** / **book_group_remove** — attach or detach a group as editors of a book.
- **editorial_set** / **editorial_list** — the opt-in ready/approved review workflow, and an editor's own queue.
- **edit_session_create** / **edit_session_revoke** — a PIN-gated browser-editor link for a human who has no assistant.

12.7 Editions and questions

- **book_set_translation_of** — link a book as a language edition of an original.
- **sync_status** / **sync_resolve** — cross-edition drift, and marking a chapter reconciled.
- **question_add** / **question_set** / **question_get** / **question_list** / **question_delete** — the public questions your books answer.
- **question_link** / **question_unlink** / **book_questions** / **chapter_questions** — attach answers, and see a book's or a chapter's discovery footprint.

12.8 Print, ISBN and your public face

- **print_set** / **print_get** — configure a printer preset and render the press-ready interior plus wrap cover; read the status and the press report.
- **book_isbn_assign** — assign a real, pool-allocated ISBN-13 to a published book.

- **author_page_get / author_page_set / authors_list** — your opt-in public author page, and the public index of everyone else's.
- **user_contact_get / user_contact_set_field** — your own contact details.

12.9 Publishing and lifecycle

- **publish_request** — submit the book for review. Publication is permanent; confirm with the human first.
- **vet_status** — the current verdict, the reason if it was rejected, and whether the book is listed.
- **book_revision_open / book_revision_close** — the ten self-service revision windows on a published book.
- **book_delete / book_restore** — the 36-hour trash window. Restoring is the human's decision, never the assistant's.

12.10 Admin

- **admin_book_list / admin_book_delete / admin_book_transfer / admin_author_summary** — the catalogue, and per-owner rollups.
- **vet_list / vet_decide / vet_submit_result / admin_cover_vet** — the review queue and its verdicts, including the vision check on uploaded covers.
- **book_freeze / book_unfreeze / rotate_share_links** — lock a book against all edits; retire links that have already been shared.
- **admin_rerender / admin_pin_published / admin_download_gate** — force a re-render, pin published books against retention, toggle the instance-wide download gate.
- **admin_s3_status / admin_s3_list / admin_s3_sync / admin_s3_verify / admin_s3_restore** — the durability mirror and its drift report.
- **isbn_quota_get / isbn_quota_set** — a user's ISBN allowance.
- **indexnow_run / indexnow_status** — search-engine notification.

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.

A gift of support — Stripe

Scan to read this book online