SIDEBAR AI

Project Sidebar Content Cheat Sheet

Sidebar

AI · Project Sidebar Content Cheat Sheet

The four kinds of Sidebar page and how each is built. From `rules/sidebar_rules.md` (SB1-SB11, LN, MT) and `rules/cheat_sheet_series.md`. Everything lives under `content/sidebar/<subject>/`.

THE FOUR TYPES

`Links`	`/links <topic>`. Wrapper `.links-chapter`, sky-blue `#38bdf8`. No PDF. File `links_<topic>.html`.
`My Tools`	`/my-tools <name>`. Interactive tool from a YAML spec. No PDF. File `<name>.html`.
`Sidebar lesson`	`sidebar: <topic>`. Wrapper `.sidebar-lesson`, amber `#d97706` / `#fbbf24`. File `sidebar_<topic>.html`.
`Cheat sheet`	`cheat sheet <topic>`. Wrapper `.cheat-sheet`, emerald `#059669` / `#6ee7b7`. File `cheat_sheet_<topic>.html`.

SUBJECTS AND BANNER (SB1, SB2)

`subject folders`	ai, drinks, football, history, linux, operating-systems, programming, web-development. The folders are the mapping.
`no fit`	Reuse a subject from the content tree; ask only if genuinely ambiguous.
`browsing pages`	Generated automatically. Never hand-write a subject index.
`Category: Sidebar`	Always. Then `Subcategory:` (subject), `Chapter:` (page title), `File:` and the date line.
`Topic: line`	Included only for Sidebar lessons.

SIDEBAR LESSON (SB3)

`generated directly`	No outline, no numbering, no link to other lessons.
`research`	Search rather than recall for anything current, a named person's claim, or a product.
`structure`	Course-name and breadcrumb lead-in, intro on why it merits a lesson, body sections, honest cross-references, typically 3 hands-on exercises, closing Quick Reference box.
`exercise solutions`	`solutions/sidebar_<topic>_exercise<N>.txt`.
PDF	`py scripts/python/build_sidebar_lesson_pdf.py <Subject> <file.html> "<Title>"`

CHEAT SHEET (SB6-SB8)

`framing`	A 1-2 sentence line, no full intro.
`category cards`	A 2-3 column grid; each card is a short heading plus `command - one-line description` rows. Cards never split when printed.
`no extras`	No exercises, no Quick Reference, no copy buttons. At most one closing tip for a real gotcha.
`verify`	Anything version-specific or dated is checked by search or by running it. The sheet says what was and was not verified.
PDF	`py scripts/python/build_cheat_sheet_pdf.py <Subject> <file.html> "<Title>"`

CHEAT SHEET BATCHES (SB9)

`cheat-sheet-<series>_1-4`	Builds sheets 1 to 4 of the named series, in order, then gives one combined report.
`cheat-sheet-<series>_5`	A single number builds one sheet.
`series file`	`rules/cheat_sheet_series.md`: topic, scope and a `done` date for each sheet.
`batch size`	About 4-5 sheets. New topics go at the end so numbers stay stable.
`range too long`	If the range runs past the list or a topic is unclear, Claude stops and asks.

LINKS (LN)

`modes`	`/links <topic>` creates; `update` adds links. If a page exists and no mode is given, Claude asks. It never overwrites silently.
`every run`	Claude asks whether you have a file of extra links for the topic.
`Links.md`	Scanned read-only for relevant entries. Shortened `lnkd.in` links are resolved to find the real title.
`real links only`	Found by search and confirmed by fetching. A URL is never constructed.
`sections, in order`	Documentation, Downloads, Free Online Courses, LinkedIn Learning. Empty ones are omitted.
`LinkedIn titles`	Taken from the URL slug, hyphens to spaces, title case. A `/paths/` URL gets the prefix "Learning Path: ".
`update mode`	Only genuinely new links, de-duplicated by URL. Nothing removed or reordered.

MY TOOLS (MT)

`spec`	`my-tools/<name>.yaml` is the only input. `name` and `description` are required; `example` is optional.
`no spec`	Claude says so and asks you to create one (by hand or with `tool-builder.html`). It never guesses.
`fragment`	No `<!DOCTYPE>`, `<html>`, `<head>`, `<body>`, `<meta>` or `<title>`; a `<script>` is fine.
`CSS scoping`	Every rule under the tool's own wrapper class. Never a bare element selector or `*`.
`palette`	Background `#0f1117`, surfaces `#1a1d27` / `#161b22`, borders `#30363d`, text `#c9d1d9`.
`Load Example`	A button wherever the tool has inputs, inferred from the description if there is no `example`.
`updates`	No incremental mode: a tool is regenerated wholesale from a revised YAML.

TRACKING (SB11)

`sidebar-pages.md`	One shared log: one entry per page with type, subject, file path, created date and every update.
`when`	Updated in the same step as generating or updating any Sidebar page.
`how`	Append or edit the one entry. Never read the whole file; search it.

SNAPSHOT

Snapshot of the rule files as at 2026-11-08, summarised by reading them. Nothing was run to test it. If this sheet and a rule file ever disagree, the rule file wins, and the sheet needs regenerating.