mdBook Markdown books

CONCEPTS

SUMMARY.md decides which chapters exist and their order; a Markdown file not listed there is not part of the book.

src/*.md chapters
↓ `SUMMARY.md`
Preprocessors links, index
↓ `mdbook build`
Renderers html, custom
↓ `writes`
book/ static site

Install with cargo install mdbook, or grab a prebuilt binary from the GitHub releases.

CREATE AND BUILD

`mdbook init`
Create a book in the current directory

`mdbook init <dir> --title "<title>"`
Create a book in a new directory

`mdbook init --force`
Skip the title and .gitignore prompts

`mdbook build`
Render the book to book/

`mdbook build <dir>`
Build a book in another directory

`mdbook build -o`
Build, then open in the browser

`mdbook build -d <dir>`
Write output to a different directory

`mdbook clean`
Delete the rendered output

PREVIEW

`mdbook serve`
Serve and rebuild on every change

`mdbook serve -o`
Serve and open the browser

`mdbook serve -p <port>`
Use another port (default 3000)

`mdbook serve -n <host>`
Bind to another hostname

`mdbook watch`
Rebuild on change without a server

TEST AND TOOLING

`mdbook test`
Compile and run all Rust code blocks

`mdbook test -c <chapter>`
Test one chapter by name or path

SUMMARY.MD

`[Intro](intro.md)`
Prefix chapter, unnumbered, before the list

`- [Chapter](chapter.md)`
Numbered chapter

`    - [Sub](sub.md)`
Indent four spaces for a sub-chapter

`[Title]()`
Draft chapter with no file yet

`# Part Title`
Group the chapters that follow

Use one list marker (- or *) throughout; mixing them breaks parsing.

BOOK.TOML

`[book] title = "<title>"`
Book title

`[book] authors = ["<name>"]`
Author list

`[book] src = "<dir>"`
Source directory (default src)

`[build] build-dir = "<dir>"`
Output directory (default book)

`[build] create-missing = false`
Fail on files missing from SUMMARY

`[rust] edition = "2021"`
Edition for Rust snippets (default 2015)

MARKDOWN AND CODE

`{{#include <file>}}`
Insert a whole file

`{{#include <file>:2:10}}`
Insert lines 2 to 10

```` ```rust,ignore ````
Highlight, but skip the test

`# let x = 1;`
Hide a Rust line (still compiled)