SIDEBAR AI

Project Programming Lessons Cheat Sheet

Sidebar

AI · Project Programming Lessons Cheat Sheet

How standalone programming lessons and their exercise chapters are generated. From `rules/programming_lesson_rules.md`: rules P8 and PL1-PL3. Python is the established case.

PROMPTS (PL1, PL2)

`<language>: <topic>`	e.g. `python: for loops`. A standalone lesson, generated directly with no outline step, plus its exercise chapter.
`... no ex / no exercises`	Lesson only.
`... ex only / exercises only`	Exercise chapter only, e.g. to add one to an older lesson.
`matching`	Based on the trailing phrase; small punctuation differences are fine. If genuinely ambiguous, Claude asks.
`other languages`	No lesson exists yet, so Claude asks about storage before generating the first one.

LESSON STRUCTURE, IN ORDER (P8)

`1 .section-intro`	Intro paragraph framing the topic.
`2 .java-box`	A comparison callout (for Python: coming from Java / JavaScript).
`3 .fastapi-box`	Only when the topic genuinely connects to FastAPI or web work. Omitted otherwise.
`4 .concept-grid`	Cards for the core pieces, each with a short code example.
`5 .example-block`	Realistic, slightly longer examples in context.
`6 .ref-table`	Quick-reference table of syntax and operations.
`7 .tip-box`	Closing gotchas, specific traps rather than generic warnings.

BANNER AND STYLING

`banner`	`PYTHON LESSON: <Topic>`, a `Level:` line, then the Date Created / Date Updated line last.
`self-contained`	The lesson embeds its own `<style>` block, wrapper `.py-lesson`.
`highlight classes`	`.cm .kw .st .nu .fn .cl .op`.
`palette`	Blue `#4f8ef7`, green `#7ecfa0`, amber `#ffd580`, purple `#7c5cbf`, red `#f08080`.
`no numbers`	No chapter or course numbers in the banner or filename.

AUDIENCE AND TONE

`reader`	Someone who already knows Java and JavaScript well.
`approach`	Actively contrast Python with both, rather than explaining Python alone.
`size`	A big topic may take two lessons. Say so rather than cramming.
`other languages`	Adapt the comparison callout to the relevant background instead of forcing Java/JavaScript.

FILES

`python_lesson_<topic>.html`	In `programming/general-purpose-languages/python/python-lessons/`; lowercase, underscores.
`pdfs/`	Sibling folder for the PDFs.
`PDF command`	`py scripts/python/build_python_lesson_pdf.py <file.html> "<Title>"`
`solutions/`	Sibling folder for the exercise `.txt` solutions.

EXERCISE CHAPTER (PL3)

`python_lesson_<topic>_exercises.html`	A separate fragment, not appended to the lesson.
`banner`	`PYTHON LESSON EXERCISES: <Topic>` plus the date line.
`styling`	The same lesson-family CSS as the paired lesson, so the two read as one unit.
`at least 10 exercises`	In a `.challenges-grid` of `.challenge-card`s: title, problem, and a `→ Solution` link opening a new tab.
`depth`	Meaty problems native to the theme, not one-line fill-ins and not forced reskins of another exercise.
`intro`	A short `.section-intro` saying it is the companion set.

SOLUTION FILES (PL3)

`python_lesson_<topic>_exercise<N>.txt`	One per exercise.
`layout`	Title line, a `====` divider, the solution code, then `Output:` where relevant.
`closing section`	`WHY THIS WORKS AS AN ANSWER` with a `---` divider.
`links`	Solution links are rewritten for the live site by the build, whatever path the source has.

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. The rule file also holds a backlog of exercise chapters still to be written; that list is not repeated here.