Claude Code cheat sheet

Flags, commands, shortcuts and file locations for everyday work.

Verified 2026-10-13

Start and resume sessions

`claude`	Start an interactive session
`claude "query"`	Start with an initial prompt
`claude -c`	Continue the most recent conversation in this directory
`claude -r "<session>"`	Resume by ID or name, or pick one
`claude -n "<name>"`	Name the session for a later resume
`--fork-session`	With `-c` or `-r`: new session ID
`claude -w <name>`	Work in an isolated git worktree
`--model <alias|name>`	Set the model, such as `sonnet`
`--permission-mode <mode>`	Start in a given permission mode
`claude --bg "<task>"`	Start a background session; monitor it with `claude agents`

Headless and scripting

`claude -p "query"`	Run once, print the answer, exit
`cat file | claude -p "…"`	Pipe content in on stdin (10MB cap)
`--output-format <format>`	`text` (default), `json`, `stream-json`
`--json-schema '<schema>'`	JSON validated against your schema
`--allowedTools "<rule>" …`	Tools that run without a prompt, such as `"Bash(git diff *)"`
`--max-turns <n>`	Limit the number of agentic turns
`--max-budget-usd <n>`	Stop at an estimated spend
`--bare`	Skip hooks, skills, plugins, MCP servers, auto memory and CLAUDE.md
`claude -c -p "query"`	Continue the last conversation

JSON output carries `result`, `session_id` and `total_cost_usd`. Exit code 0 means success.

Settings files

highest precedence first

`managed-settings.json`	Managed: set by your organization
`--settings <file-or-json>`	Command line: this session only
`.claude/settings.local.json`	Project local: only you, kept out of git
`.claude/settings.json`	Shared project: commit it for the team
`~/.claude/settings.json`	User: you, in every project

List keys such as `permissions.allow` merge across files. `/status` shows which files loaded; `claude doctor` reports invalid entries.

`.mcp.json`	Project-scoped MCP servers; manage them with `/mcp`

In-session commands

`/init`	Generate a starting CLAUDE.md
`/clear`	New conversation with empty context
`/compact`	Summarize the conversation to free context
`/context`	Visualize current context usage
`/resume`	Resume a conversation or open the picker
`/branch`	Branch the conversation at this point
`/rewind`	Rewind code, conversation or both
`/model`	Switch model; saved as your default
`/plan`	Enter plan mode from the prompt
`/permissions`	Manage allow, ask and deny rules
`/memory`	Edit CLAUDE.md files, toggle auto memory
`/config`	Open settings, or set `key=value` directly
`/usage`	Session cost and plan usage limits
`/diff`	Review changes in your working tree
`/btw`	Side question that stays out of history
`/code-review`	Review the current diff for bugs

Keyboard and input

`Esc`	Interrupt Claude; work so far is kept
`Esc Esc`	Open the rewind menu (on an empty prompt)
`Shift+Tab`	Cycle permission modes
`Ctrl+C`	Interrupt, or clear the input when idle
`Ctrl+O`	Toggle the transcript viewer
`Ctrl+R`	Reverse-search command history
`Ctrl+B`	Background the running command
`Ctrl+G`	Edit the prompt in your text editor
`Ctrl+V`	Paste an image from the clipboard
`Ctrl+J`	New line, in any terminal
`!`	Shell mode: run a command directly
`@`	Mention a file path, with autocomplete

Checkpoints and rewind

Created	At every prompt that starts a turn; the 100 most recent are kept
Restore	Code and conversation, or either alone
Summarize	From here or up to here; files untouched
Not tracked	Bash file changes, most subagent edits, edits made outside the session

Permission modes

what runs without asking

`default`	Reads only. Shown as Manual
`acceptEdits`	Reads, file edits, common filesystem commands
`plan`	Reads, plus classifier-approved commands
`auto`	Everything, with background safety checks
`dontAsk`	Reads and pre-approved tools; the rest is denied
`bypassPermissions`	Everything. Isolated containers and VMs only

`Shift+Tab` cycles `default`, `acceptEdits` and `plan`, then `bypassPermissions` and `auto` when enabled. Deny rules block in every mode. From v2.1.283 the built-in starting mode for interactive terminal sessions is `auto`; set `permissions.defaultMode` to change it.

Plan mode

Allows	Reads, shell commands to explore, a written plan
Blocks	Edits to your source until you approve the plan
Commands	Reviewed by the classifier, or prompted when outside the read-only set
Not enforced	In an interactive terminal session with bypass permissions available

Hooks

Configure in	`hooks` in a settings file; `/hooks` lists them
`SessionStart`	A session begins or resumes
`UserPromptSubmit`	A prompt is submitted, before processing
`PreToolUse`	Before a tool call runs; can block it
`PostToolUse`	After a tool call succeeds
`Stop`	Claude finishes responding
Input	Event data as JSON on stdin
Exit codes	0: no objection. 2: block the action where the event allows it; stderr is the reason

Memory and CLAUDE.md

`./CLAUDE.md`	Project instructions, shared through git
`./CLAUDE.local.md`	Personal project notes; add to `.gitignore`
`~/.claude/CLAUDE.md`	Your preferences, in every project
`.claude/rules/*.md`	Topic rules, optionally scoped by `paths:`
`@path/to/file`	Import into CLAUDE.md, up to four hops deep
`MEMORY.md`	Auto memory index Claude writes; the first 200 lines or 25KB load