DOCTOR AI · SIMPLE TECH SKILLS

Claude Code Commands

The master cheat sheet. Our 60 and 100 command sheets in one, with the repeats taken out.

263
COMMANDS + TIPS

★ **Start here:** new to it? Run `/init` once, then `/help` to see everything. Type `/` to see what your version has.

Slash commands

Type these inside Claude Code. Type / to see the ones your version has.

START & BASICS

- `/help` See all commands
- `/init` Make project notes (CLAUDE.md)
- `/clear` Start fresh, keep notes
- `/compact` Shrink a long chat
- `/context` See what fills the memory
- `/resume` Open an older chat
- `/rewind` Undo code and chat
- `/export` Save the chat to a file
- `/copy` Copy the last reply
- `/recap` Get a one line summary
- `/release-notes` See what is new
- `/powerup` Learn with quick demos

MODELS & EFFORT

- `/model` Pick your model
- `/model fable` Use the strongest model
- `/effort` Set how hard it works
- `/effort auto` Go back to normal
- `/fast` Turn fast mode on or off
- `/advisor` Add a second model for advice

MEMORY & FOLDERS

- `/memory` Edit project notes
- `/add-dir` Let it read another folder
- `/cd` Move to another folder

PLAN, SHIP & REVIEW

- `/plan` Think before big edits
- `/diff` See what changed
- `/code-review` Find bugs, add --fix to apply
- `/simplify` Clean up code only
- `/security-review` Check for security holes
- `/review` Fast read of a GitHub PR
- `/ultrareview` Deep review in the cloud

MCP & EXTENSIONS

- `/mcp` Connect outside tools
- `/hooks` Run scripts on events
- `/plugin` Add command bundles
- `/reload-plugins` Refresh plugins
- `/reload-skills` Refresh skills
- `/chrome` Set up Claude in Chrome
- `/ide` Connect your editor
- `/dataviz` Help with charts
- `/claude-api` Load API help
- `/design-sync` Send your design system
- `/rename` Name this session

GOALS & AUTOMATION

- `/goal` Work until a goal is met
- `/loop` Repeat a task on a timer
- `/workflows` Watch big multi-agent runs
- `/deep-research` Research with real sources
- `/autofix-pr` Auto fix a pull request
- `/install-github-app` Run on GitHub
- `/install-slack-app` Add the Slack app

AGENTS & PARALLEL WORK

- `/agents` Make helper agents
- `/subtask` Hand off a side job
- `/background` Run session in the back
- `/fork` Copy chat to a new session
- `/branch` Try a different path
- `/batch` Split a big job across agents
- `/tasks` List background work
- `/todos` See the current to-do list

CONFIG & CONTROLS

- `/config` Open settings
- `/config theme=dark` Set one thing fast
- `/permissions` Set what it may do
- `/fewer-permission-prompts` Get fewer pop-ups
- `/keybindings` Edit shortcuts
- `/terminal-setup` Turn on Shift+Enter
- `/vim` Turn on vim mode
- `/color` Set the bar color
- `/scroll-speed` Set scroll speed
- `/focus` Show a clean view

MOVE & CONNECT

- `/teleport` Pull a web chat here
- `/remote-control` Run it from your phone
- `/desktop` Open in the desktop app
- `/mobile` Get the mobile app

ACCOUNT & FIXES

- `/status` See account and system
- `/usage` See token use
- `/login` Sign in
- `/logout` Sign out
- `/passes` Share a free week
- `/privacy-settings` Manage privacy
- `/doctor` Check and fix setup
- `/debug` Find runtime problems
- `/heapdump` Save a memory snapshot
- `/feedback` Send feedback
- `/bug` Report a bug
- `/btw` Ask a quick aside
- `/insights` See your work patterns
- `/exit` Quit the session

MORE SLASH COMMANDS

- `/cost` Token usage + cache breakdown
- `/theme` Change color theme
- `/skills` List available skills
- `/voice` Push-to-talk (20 langs)
- `/schedule` Cloud scheduled tasks

Keyboard shortcuts

Press these while Claude Code is open.

GENERAL CONTROLS

- `Ctrl C` Cancel input/generation
- `Ctrl D` Exit session
- `Ctrl L` Redraw screen
- `Ctrl O` Toggle verbose/transcript
- `Ctrl R` Reverse search history
- `Ctrl G` Open prompt in editor
- `Ctrl X E` Open in editor (alias)
- `Ctrl B` Background running task
- `Ctrl T` Toggle task list
- `Ctrl V` Paste image chip
- `Ctrl X K` Kill background agents
- `Esc Esc` Rewind or summarize

MODE SWITCHING

- `Shift Tab` Cycle permission modes
- `Alt P` Switch model
- `Alt T` Toggle thinking
- `Alt O` Toggle fast mode

INPUT & PREFIXES

- `\Enter / Ctrl J` Newline
- `/` Slash command
- `!` Direct bash
- `@` File mention + autocomplete

SESSION PICKER

- `Up / Down` Navigate sessions
- `Left / Right` Expand / collapse
- `P` R / Preview | Rename | Search
- `A` All projects | B Current branch

TRANSCRIPT MODE (CTRL+O)

- `Ctrl E` Toggle show all
- `Q / Esc` Exit transcript

Terminal

Type these in your shell, before Claude Code opens.

START AND RUN

- `claude` Start a session
- `claude -c` Continue last session
- `claude -r` Resume a session
- `claude -p` Run with no chat UI
- `claude update` Get the newest version
- `claude doctor` Check your install
- `claude agents` Watch background agents
- `claude mcp` Set up tools
- `claude mcp add` Add one tool server
- `claude mcp list` List your tools
- `claude config` Change settings
- `claude --model` Start with a model
- `claude --effort` Start with an effort
- `claude --bg` Run a background job
- `claude --exec` Run a shell command
- `claude --debug` Turn on debug logs
- `claude --add-dir` Allow another folder
- `claude --version` Show the version

CORE COMMANDS

- `claude "query"` With prompt

HEADLESS (SDK)

- `--output-format json` Structured output
- `--max-budget-usd 5` Cost cap
- `cat file | claude -p` Pipe input

KEY FLAGS

- `-w` Git worktree
- `-n / --name` Session name
- `--agent` Use a specific agent
- `--allowedTools` Pre-approve tools
- `--console` Auth via Anthropic Console
- `--verbose` Verbose output
- `--bare` Minimal headless (no hooks/LSP)
- `--channels` Permission relay / MCP push
- `--permission-mode` plan / default / bypassPermissions
- `--dangerously-skip-permissions` Skip all prompts

Power user reference

Settings, files, and tips for when you want more control.

THINKING & EFFORT

- `"ultrathink"` Max effort for current turn

CONTEXT MANAGEMENT

- `Auto-compact` Fires at ~95% capacity
- `1M context` Opus 4.6 (Max / Team / Ent)
- `CLAUDE.md` Survives compaction!

GIT WORKTREES

- `--worktree name` Isolated branch per feature
- `isolation: worktree` Agent in own worktree
- `sparsePaths` Checkout only needed dirs

VOICE MODE

- `Space (hold)` Record, release to send

20 languages: EN, ES, FR, DE, CZ, PL...

SCHEDULING & REMOTE

- `/rc` Remote control bridge
- `--remote` Web session on claude.ai

AUTO MODE DENIED

- `/permissions → Recent` Retry denied action with R

MCP: ADD SERVERS

- `--transport http` Remote HTTP (recommended)
- `--transport stdio` Local process
- `--transport sse` Remote SSE

MCP: SCOPES

- `Local` ~/.claude.json (you only)
- `Project` .mcp.json (shared / VCS)
- `User` ~/.claude.json (global)

MCP: MANAGE

- `claude mcp serve` Run Claude Code as MCP server
- `maxResultSizeChars` Up to 500K via _meta override

CLAUDE.MD LOCATIONS

- `./CLAUDE.md` Project (team-shared)
- `~/.claude/CLAUDE.md` Personal (all projects)
- `/etc/claude-code/` Managed (org-wide)

RULES & IMPORT

- `.claude/rules/*.md` Project rules
- `~/.claude/rules/*.md` User rules
- `paths: frontmatter` Path-specific rules
- `@path/to/file` Import in CLAUDE.md

AUTO MEMORY

~/.claude/projects/<proj>/memory/
MEMORY.md + topic files, auto-loaded
Max: 25KB / 200 lines per file

PERMISSION MODES

`default`	Prompts for every action
`acceptEdits`	Auto-accept file edits
`plan`	Read-only, no writes
`dontAsk`	Deny unless explicitly allowed
`bypassPermissions`	Skip all permission checks

CUSTOM SKILL LOCATIONS

`.claude/skills/<n>/`	Project skills
`~/.claude/skills/<n>/`	Personal skills

SKILL FRONTMATTER

`description`	Auto-invocation trigger
`allowed-tools`	Skip permission prompts
`model`	Override model for skill
`effort`	Override effort level
`paths: [globs]`	Path-specific (YAML list)
`context: fork`	Run in subagent
`$ARGUMENTS`	User input placeholder
`${CLAUDE_SKILL_DIR}`	Skill's own directory
`!`cmd``	Dynamic context injection
`plugin bin/`	Ship executables for Bash tool

BUILT-IN AGENTS

`Explore`	Fast read-only (Haiku)
`Plan`	Research for plan mode
`General`	Full tools, complex tasks
`Bash`	Terminal separate context

AGENT FRONTMATTER

`permissionMode`	default / acceptEdits / plan / dontAsk
`memory:`	user | project | local
`background: true`	Background task
`maxTurns`	Limit agentic turns
`initialPrompt`	Auto-submit first turn
`SendMessage`	Resume agents (replaces resume)
`@agent-name`	Mention named subagents

CONFIG FILES

`~/.claude/settings.json`	User settings
`.claude/settings.json`	Project (shared)
`.claude/settings.local.json`	Local only
`~/.claude.json`	OAuth, MCP, state
`.mcp.json`	Project MCP servers
`managed-settings.d/`	Drop-in policy fragments

KEY SETTINGS

`modelOverrides`	Map model picker to custom IDs
`autoMemoryDirectory`	Custom memory dir
`worktree.sparsePaths`	Sparse checkout dirs
`sandbox.failIfUnavailable`	Exit if sandbox fails
`hooks: if`	Conditional hooks
`PermissionDenied`	Hook: auto-mode denial
`showThinkingSummaries`	Opt-in, off by default
`hooks: "defer"`	Pause headless, resume later
`disableSkillShellExec`	Block !cmd in plugins
`forceRemoteSettingsRefresh`	Fail-closed startup

KEY ENV VARS

`ANTHROPIC_API_KEY`	Your API key
`ANTHROPIC_MODEL`	Override default model
`CLAUDE_CODE_EFFORT_LEVEL`	low/medium/high/max/auto
`MAX_THINKING_TOKENS`	0 = off
`CLAUDE_CODE_SUBPROCESS_ENV_SCRUB`	Strip creds from subprocesses
`CLAUDE_STREAM_IDLE_TIMEOUT_MS`	Streaming watchdog (default 90s)
`CLAUDE_CODE_NO_FLICKER`	Alt-screen rendering (=1)
`MCP_CONNECTION_NONBLOCKING`	Skip MCP wait in -p mode
`CLAUDECODE`	Detect CC shell (=1)

BEDROCK & ENTERPRISE

`Bedrock wizard`	Interactive AWS setup on login
`CLAUDE_CODE_MAX_OUTPUT_TOKENS`	Default: 32K
`CLAUDE_CODE_DISABLE_CRON`	Disable scheduled tasks
`ANTHROPIC_CUSTOM_MODEL_OPTION`	Custom /model entry

EFFORT LEVELS

`low`	Fast, simple tasks
`medium`	Default balanced mode
`high`	Complex multi-step tasks
`max`	Hardest problems, most tokens
`auto`	Claude picks based on task

MODELS (ALT+P)

`Haiku`	Fastest, cheapest
`Sonnet`	Balanced quality + speed
`Opus`	Highest quality, 1M context

WHEN TO USE PLAN MODE

Refactoring large codebases
Risky or irreversible changes
When you want to review before execution

CONTEXT BEST PRACTICES

CLAUDE.md survives /compact
Use /btw for quick side questions
@filename auto-completes file paths
Use /diff to review all changes after a session

USEFUL COMBOS

`claude -c --model opus`	Resume with Opus
`/plan then Shift Tab`	Review before any action
`!ls -la`	Run bash without leaving chat
`@CLAUDE.md`	Reference your own rules inline

THINGS TO DO FIRST

1. Run /init to create your CLAUDE.md
2. Set --max-budget-usd to avoid surprises
3. Add CLAUDE.md to every project root
4. Use /doctor if anything feels broken
5. Check /cost after long sessions

WORKTREE WORKFLOW

Use --worktree for parallel feature work
Each worktree = isolated branch + agent
Use /batch to auto-create worktrees

PRO TIPS

Use /insights to analyze session history
Run /voice for hands-free coding sessions
Use /schedule for cloud recurring tasks
Pipe files in: cat file | claude -p

Want this built for you? We build your whole AI business system. You own it.
Checked against Claude Code docs, July 2026. Commands change. Type / to see yours.
simpletechskills.com/crm