

Claude Code Cheatsheet

Daily essentials for maximum productivity

v3.16.0 | January 2026

How to Install

```
npm install -g @anthropic/clause-code  
# Verify installation  
which clause && clause --version
```

Health Check: `clause doctor && clause mcp list`

Essential Commands

Command	Action
/help	Contextual help
/clear	Reset conversation
/compact	Free up context
/status	Session state + context usage
/context	Detailed token breakdown
/plan	Enter Plan Mode (no changes)
/execute	Exit Plan Mode (apply changes)
/model	Switch model (sonnet/opus/opusplan)
/exit	Quit (or Ctrl+D)

Keyboard Shortcuts

Shortcut	Action
Shift+Tab	Cycle permission modes
Esc × 2	Rewind (undo)
Ctrl+C	Interrupt
Ctrl+R	Retry last operation
Ctrl+L	Clear screen (keeps context)
Tab	Autocomplete
Shift+Enter	New line
Ctrl+D	Exit

IDE Shortcuts: VS Code `Alt+K` | JetBrains `Cmd+Option+K`

File References

```
@path/to/file.ts      → Reference a file  
@agent-name           → Call an agent  
!shell-command        → Run shell command
```

Features Méconnues (But Official!)

Feature	Since	What It Does
Tasks API	v2.1.16	Persistent task lists with dependencies
Background Agents	v2.0.60	Sub-agents work while you code
TeammateTool	Experimental	Multi-agent coordination (unstable)
Session Forking	v2.1.19	Rewind + create parallel timeline
LSP Tool	v2.0.74	Code intelligence (go-to-def, refs)

Pro tip: These aren't "secrets"—they're in the [CHANGELOG](#). Read it!

Permission Modes

Mode	Editing	Execution
Default	Asks	Asks
Auto-accept	Auto	Asks
Plan Mode	✗	✗

Shift+Tab to switch modes

Memory & Settings (2 levels)

Level	Path	Scope	Git
Project	.claude/	Team	✓
Personal	~/.claude/	You (all projects)	✗

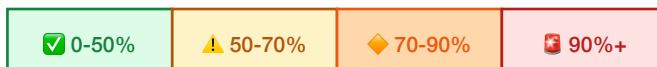
Priority: Project overrides Personal

.claude/ Folder Structure

```
.claude/
  └── CLAUDE.md          # Local memory
  (gitignored)
  └── settings.json       # Hooks (committed)
  └── settings.local.json # Permissions (not
                           committed)
  └── agents/             # Custom agents
  └── commands/           # Slash commands
  └── skills/             # Knowledge modules
```

Context Management (CRITICAL)

Statusline: Model: Sonnet | Ctx: 89.5k | Ctx(u): 56.0%



Watch ctx(u): → >70% = /compact , >85% = /clear

Sign	Action
Short responses	/compact
Frequent forgetting	/clear
>70% context	/compact
Task complete	/clear

Context Recovery Commands

Command	Usage
/compact	Summarize and free context
/clear	Fresh start
/rewind	Undo recent changes
claude -c	Resume last session
claude -r <id>	Resume specific session

Plan Mode & Thinking Depth

Feature	Activation	Usage
Plan Mode	Shift+Tab × 2 or /plan	Explore without modifying
OpusPlan	/model opusplan	Opus for planning, Sonnet for execution

Extended thinking uses prompt keywords (not CLI flags):

Prompt Keyword	Thinking Depth	Use For
"Think"	Standard	Multi-component analysis
"Think hard"	Deep	Architectural decisions
"Ultrathink"	Maximum	Critical redesign

Typical Workflow

1. Start session → claude
2. Check context → /status
3. Plan Mode → Shift+Tab × 2 (for complex tasks)
4. Describe task prompt → Clear, specific
5. Review changes → Always read the diff!
6. Accept/Reject → y/n
7. Verify → Run tests
8. Commit → When task complete
9. /compact → When context >70%

Quick Prompting Formula

WHAT: [Concrete deliverable]
WHERE: [File paths]
HOW: [Constraints, approach]
VERIFY: [Success criteria]

Example:

Add input validation to the login form.
WHERE: src/components/LoginForm.tsx
HOW: Use Zod schema, show inline errors
VERIFY: Empty email shows error, invalid format shows error

MCP Servers

Server	Purpose
Serena	Indexation + session memory + symbol search
mgrep	Semantic search by intent
Context7	Library documentation
Sequential	Structured reasoning
Playwright	Browser automation
Postgres	Database queries

Serena memory: write_memory() / read_memory() / list_memories()

Check status: /mcp

CLI Flags Quick Reference

Flag	Usage
-p "query"	Non-interactive mode (CI/CD)
-c / --continue	Continue last session
-r / --resume <id>	Resume specific session
--model sonnet	Change model
--add-dir .. /lib	Allow access outside CWD
--permission-mode plan	Plan mode
--dangerously-skip-permissions	Auto-accept (use carefully)
--debug	Debug output

Anti-patterns

✗ Don't

- Vague prompts
- Accept without reading
- Ignore warnings
- Skip permissions
- Negative constraints only

✓ Do

- Specify file + line with @references
- Read every diff
- Use /compact at 70%
- Never in production
- Provide alternatives

Cost Optimization

Model	Use For	Cost
Haiku	Simple fixes, reviews	\$
Sonnet	Most development, Architecture, complex tasks	\$
OpusPlan	Plan (Opus) + Execute (Sonnet)	\$\$

Quick Decision Tree

Simple task	→ Just ask Claude
Complex task	→ TodoWrite to plan first
Risky change	→ Plan Mode first
Repeating task	→ Create agent or command
Context full	→ /compact or /clear
Need docs	→ Use Context7 MCP

Deep analysis → Use extended thinking prompts

The Golden Rules

- 1 Always review diffs before accepting
- 2 Use /compact before context gets critical (>70%)
- 3 Be specific in requests (WHAT, WHERE, HOW, VERIFY)
- 4 Plan Mode first for complex/risky tasks
- 5 Create CLAUDE.md for every project
- 6 Commit frequently after each completed task
- 7 Know what's sent — prompts, files, MCP results → Anthropic

Common Issues Quick Fix

Problem	Solution
"Command not found"	Check PATH, reinstall npm global
Context too high (>70%)	/compact immediately
Slow responses	/compact or /clear
MCP not working	claude mcp list , check config
Permission denied	Check settings.local.json

Health Check:

```
which claude && claude doctor && claude mcp list
```

Author: Florian BRUNIAUX | [Méthode Aristote](#) | Written with Claude

Version 3.16.0 | January 2026