# Claude Code – Befehlsreferenz

Quick Reference · v2.1.227

Terminal-Befehle & Optionen geprüft gegen Version 2.1.227 (inhaltsgleich mit 2.1.226, Update vom 8. August 2026 / 2.1.222). Slash-Befehle (Teil C) auf Stand 2.1.222.

Von **Admin-Cheatmeister** · **weitere Cheatsheets →**

## Über dieses Cheatsheet

Alle Terminal-Befehle, alle Optionen und alle Slash-Befehle innerhalb einer Sitzung – mit einer Erklärung, was sie tun. Teil A/B (Terminal-Befehle, Optionen, Unterbefehle) direkt aus der hier installierten Version ausgelesen (`claude --help` und die Hilfe jedes Unterbefehls), zuletzt gegen **2.1.227** geprüft (inhaltsgleich mit 2.1.226 – keine Änderungen an Flags oder Unterbefehlen gefunden). Teil C (Slash-Befehle) konnte non-interaktiv nicht nachgeprüft werden und entspricht dem Stand von Version 2.1.222 (27. Juli 2026) – neue Slash-Befehle seit 2.1.222 fehlen hier ggf. noch. Neue, entfernte und umbenannte Einträge sind markiert: **NEU** · **ENTFERNT** · **UMBENANNT**.

## Sitzung starten und fortsetzen 12

| | |
|---|---|
| `claude` | Startet eine interaktive Sitzung im aktuellen Ordner. |
| `claude "deine Frage"` | Startet mit einer ersten Nachricht. |
| `-c, --continue` | Setzt die letzte Unterhaltung in diesem Ordner fort. |
| `-r, --resume [wert]` | Setzt eine Unterhaltung per Sitzungs-ID fort oder öffnet die Auswahlliste. |
| `--fork-session` | Beim Fortsetzen eine neue Sitzungs-ID vergeben statt die alte weiterzuführen. |
| `--session-id <uuid>` | Feste Sitzungs-ID vorgeben. |
| `-n, --name <name>` | Anzeigename der Sitzung – sichtbar in Eingabezeile, /resume-Liste und Terminaltitel. |
| `--from-pr [wert]` | Sitzung fortsetzen, die zu einem Pull Request gehört. |
| `--teleport [session]` | NEU — Eine Teleport-Sitzung fortsetzen, optional per Sitzungs-ID. Ergänzt den bisherigen Slash-Befehl /teleport um ein eigenes CLI-Flag. |
| `--no-session-persistence` | Sitzung nicht auf Platte speichern (nur mit --print). |
| `-w, --worktree [name]` | Legt für die Sitzung einen eigenen Git-Worktree an. |
| `--tmux` | Erzeugt eine tmux-Sitzung für den Worktree (nur zusammen mit --worktree). Nutzt native iTerm2-Panes falls verfügbar, sonst klassisches tmux (--tmux=classic erzwingt das). |

## Fernsteuerung und Hintergrund 6

| | |
|---|---|
| `--remote-control [name]` | Interaktive Sitzung mit Fernsteuerung – erreichbar über die Claude-App. Modus für Dauerbetrieb als Dienst. |
| `--remote-control-session-name-prefix <p>` | Präfix für automatisch vergebene Sitzungsnamen (Standard: Rechnername). |
| `--bg, --background` | Startet als Hintergrund-Agent und kehrt sofort zurück; Verwaltung über claude agents. |
| `--cloud [beschreibung\|session_id\|url]` | NEU — Erstellt eine Cloud-Sitzung mit der angegebenen Beschreibung, oder hängt sich an eine bestehende per Sitzungs-ID oder claude.ai/code-URL an. |
| `--environment <environment_id>` | NEU — Erstellt eine neue Cloud-Sitzung auf einer selbst gehosteten Umgebung (ccpool_...). |
| `--brief` | Aktiviert das Werkzeug SendUserMessage, mit dem der Agent von sich aus etwas mitteilen kann. |

## Modell, Aufwand, Kontext und Kosten 6

| | |
|---|---|
| `--model <modell>` | Modell für diese Sitzung. Kurzname (opus, sonnet, fable) oder vollständiger Name. |
| `--fallback-model <modell>` | Ausweichmodell(e) bei Überlastung, kommagetrennt in Reihenfolge (nur mit --print). |
| `--effort <stufe>` | Denkaufwand: low, medium, high, xhigh, max. |
| `--autocompact <auto\|tokens>` | Wie voll der Kontext werden darf, bevor automatisch zusammengefasst wird: auto oder ein Wert zwischen 100k und 1M Tokens. |
| `--max-budget-usd <betrag>` | Kostenobergrenze für den Lauf (nur mit --print). Sehr nützlich für Cronjobs. |
| `--betas <...>` | Beta-Header für API-Anfragen (nur für API-Key-Nutzer). |

## Nicht-interaktiv – für Skripte und Cronjobs 9

| | |
|---|---|
| `-p, --print` | Antwort ausgeben und beenden. Achtung: entfällt die Vertrauensabfrage für den Ordner. |
| `--output-format <f>` | text (Standard), json (ein Ergebnis) oder stream-json (laufend). |
| `--input-format <f>` | text (Standard) oder stream-json für laufende Eingaben. |
| `--include-partial-messages` | Teilnachrichten mitstreamen, sobald sie eintreffen. |
| `--include-hook-events` | Hook-Ereignisse in den Ausgabestrom aufnehmen. |
| `--forward-subagent-text` | Text und Gedanken von Unteragenten mit durchreichen. |
| `--replay-user-messages` | Eingehende Nutzernachrichten zur Bestätigung zurückspiegeln. |
| `--json-schema <schema>` | Ausgabe gegen ein JSON-Schema prüfen – erzwingt strukturierte Antworten. |
| `--prompt-suggestions` | Erzeugt nach jeder Runde einen Vorschlag für die nächste Eingabe. |

## Rechte und Sicherheit 7

| | |
|---|---|
| `--permission-mode <modus>` | manual, acceptEdits, auto, dontAsk, plan oder bypassPermissions. |
| `--allowedTools <...>` | Erlaubte Werkzeuge, z. B. "Bash(git *) Edit". Der saubere Weg für automatische Läufe. |
| `--disallowedTools <...>` | Ausdrücklich verbotene Werkzeuge. |
| `--tools <...>` | Welche eingebauten Werkzeuge überhaupt zur Verfügung stehen. "" schaltet alle ab. |
| `--dangerously-skip-permissions` | Überspringt alle Rechteprüfungen. Nur für abgeschottete Umgebungen ohne Internetzugang. |
| `--allow-dangerously-skip-permissions` | Macht diesen Modus verfügbar, ohne ihn einzuschalten. |
| `--add-dir <ordner...>` | Zusätzliche Ordner für den Dateizugriff freigeben. |

## Anweisungen und Agenten 6

| | |
|---|---|
| `--system-prompt <text>` | Ersetzt den Systemprompt vollständig. |
| `--append-system-prompt <text>` | Hängt etwas an den Standard-Systemprompt an. |
| `--agent <agent>` | Agent für diese Sitzung; überschreibt die Einstellung. |
| `--agents <json>` | Eigene Agenten direkt als JSON definieren. |
| `--file <spez...>` | Dateien beim Start bereitstellen, Format datei-id:pfad. |
| `--exclude-dynamic-system-prompt-sections` | Verschiebt maschinenabhängige Angaben in die erste Nachricht – verbessert Prompt-Cache-Wiederverwendung. |

## Erweiterungen: MCP, Plugins, IDE, Chrome 7

| | |
|---|---|
| `--mcp-config <...>` | MCP-Server aus JSON-Dateien oder -Zeichenketten laden. |
| `--strict-mcp-config` | Nur die so geladenen MCP-Server verwenden, alle anderen ignorieren. |
| `--plugin-dir <pfad>` | Plugin aus einem Ordner oder einer .zip laden, nur für diese Sitzung. |
| `--plugin-url <url>` | Plugin von einer URL laden, nur für diese Sitzung. |
| `--ide` | Beim Start automatisch mit der IDE verbinden, wenn genau eine verfügbar ist. |
| `--chrome / --no-chrome` | Die Chrome-Anbindung ein- oder ausschalten. |
| `--disable-slash-commands` | Schaltet alle Skills ab. |

## Einstellungen, Diagnose, Sonstiges 10

| | |
|---|---|
| `--settings <datei\|json>` | Zusätzliche Einstellungen aus Datei oder JSON-Zeichenkette laden. |
| `--setting-sources <quellen>` | Welche Einstellungsquellen gelten: user, project, local. |
| `--safe-mode` | Startet ohne jede Anpassung (CLAUDE.md, Skills, Plugins, Hooks, MCP …). Zum Eingrenzen einer kaputten Konfiguration. |
| `--bare` | Minimalmodus: überspringt Hooks, LSP, Plugin-Abgleich, Auto-Gedächtnis und automatische CLAUDE.md-Suche. |
| `-d, --debug [filter]` | Debug-Modus, optional gefiltert, z. B. api,hooks. |
| `--debug-file <pfad>` | Debug-Protokoll in eine Datei schreiben. |
| `--verbose` | Ausführliche Ausgabe, überschreibt die Einstellung. |
| `--ax-screen-reader` | Ausgabe für Screenreader: flacher Text, keine Rahmen, keine Animationen. |
| `-v, --version` | Versionsnummer ausgeben. |
| `-h, --help` | Hilfe anzeigen. |

## Unterbefehle (claude <befehl>) 21

| | |
|---|---|
| `claude agents` | Hintergrund-Agenten verwalten. Mit --json auch für Skripte. |
| `claude auth login\|logout\|status` | Anmeldung verwalten und den Anmeldestatus anzeigen. |
| `claude auto-mode config\|defaults\|reset\|critique` | Auto-Modus prüfen, Voreinstellungen anzeigen, eigene Regeln bewerten lassen oder alles zurücksetzen. |
| `claude doctor` | Prüft die Installation. Für vollen Check mit Reparatur: /doctor in der Sitzung. |
| `claude import [quelle]` | Konfiguration eines anderen KI-Coding-Agenten übernehmen – codex oder gemini. Mit --dry-run nur anzeigen, was übernommen würde, mit --yes ohne Rückfrage. |
| `claude install [ziel]` | Native Version installieren – stable, latest oder eine bestimmte Version. |
| `claude update` | Auf Aktualisierungen prüfen und installieren. upgrade ist dasselbe. |
| `claude mcp add\|add-json\|get\|list\|login\|logout\|remove` | MCP-Server hinzufügen, anzeigen, anmelden und entfernen. |
| `claude mcp add-from-claude-desktop` | MCP-Server aus Claude Desktop übernehmen (nur Mac und WSL). |
| `claude mcp reset-project-choices` | Setzt alle genehmigten und abgelehnten projektbezogenen (.mcp.json-)Server dieses Projekts zurück. |
| `claude mcp serve` | Startet Claude Code selbst als MCP-Server. |

| | |
|---|---|
| `claude plugin install\|uninstall\|list` | Plugins installieren, entfernen, auflisten. |
| `claude plugin enable\|disable\|update` | Plugins ein- und ausschalten oder aktualisieren (Neustart nötig). |
| `claude plugin init\|validate\|tag\|eval` | Eigenes Plugin anlegen, prüfen, versionieren und testen. |
| `claude plugin details <name>` | Zeigt die Bestandteile eines Plugins und den geschätzten Token-Verbrauch. |
| `claude plugin prune\|autoremove` | Entfernt automatisch installierte Abhängigkeiten, die nicht mehr gebraucht werden. |
| `claude plugin marketplace` | Marktplätze verwalten. |
| `claude project purge [pfad]` | Löscht den gesamten Projektzustand: Verläufe, Aufgaben, Dateihistorie, Konfigurationseintrag. |
| `claude setup-token` | Langlebigen Anmelde-Token einrichten (Abo erforderlich). |
| `claude ultrareview [ziel]` | Mehragenten-Code-Review in der Cloud für den aktuellen Branch oder einen Pull Request. |
| `claude gateway` | Betreibt das Gateway für Anmeldung und Telemetrie im Unternehmensumfeld. |

## Sitzung und Kontext 20

| | |
|---|---|
| `/clear` | Neue Unterhaltung mit leerem Kontext beginnen. Alias: /new, /reset. |
| `/compact` | Kontext freimachen, indem das bisherige Gespräch zusammengefasst wird. |
| `/autocompact` | Einstellen, wie voll der Kontext werden darf, bevor automatisch zusammengefasst wird. |
| `/context` | Zeigt die aktuelle Kontextauslastung als farbiges Raster. |
| `/resume` | Zu einer früheren Unterhaltung zurückkehren. |
| `/rewind` | Code und Gespräch auf einen Prüfpunkt zurücksetzen. |
| `/branch` | Das Gespräch an dieser Stelle verzweigen, um eine andere Richtung auszuprobieren. |
| `/fork` | Das Gespräch in eine neue Hintergrundsitzung kopieren. |
| `/btw` | Eine kurze Zwischenfrage stellen, ohne den Gesprächsverlauf zu belasten. |
| `/subtask` | Eine Nebenaufgabe an einen Unteragenten geben, der zurückmeldet. |
| `/tasks` | Laufende Hintergrundarbeit dieser Sitzung anzeigen. |
| `/background` | Die Sitzung ablösen und als Hintergrund-Agent weiterlaufen lassen. Alias: /bg. |
| `/stop` | Eine Hintergrundsitzung stoppen; Gesprächsverlauf und Worktree bleiben erhalten. |
| `/daemon` | Hintergrunddienste und Routinen verwalten. |
| `/goal` | Ein Ziel setzen – Claude arbeitet über mehrere Runden weiter, bis es erreicht ist. |
| `/recap` | Sofort eine einzeilige Zusammenfassung der Sitzung erzeugen. |
| `/alias` | Eigene Befehlskürzel anlegen oder auflisten. |
| `/export` | Das Gespräch als reinen Text ausgeben. |
| `/copy` | Die letzte Antwort in die Zwischenablage kopieren. |
| `/exit` | Beenden. Alias: /quit. |

## Code, Git und Qualität 12

| | |
|---|---|
| `/diff` | Interaktive Ansicht der nicht übernommenen Änderungen. |
| `/review` | Schnelle, rein lesende Durchsicht eines Pull Requests. |

| | |
|---|---|
| `/code-review` | Prüft den aktuellen Änderungsstand auf Fehler und Aufräummöglichkeiten. |
| `/security-review` | Prüft die Änderungen auf Sicherheitslücken. |
| `/simplify` | Schlägt Vereinfachungen im Code vor. |
| `/verify` | Prüft den Code, bevor er ausgeliefert wird. |
| `/plan` | In den Planungsmodus wechseln, bevor etwas Größeres geändert wird. |
| `/ultraplan` | Claude Code im Web erstellt einen bearbeitbaren Plan, den du dann freigibst. |
| `/init` | Legt eine CLAUDE.md als Projektleitfaden an. |
| `/batch` | Große Änderungen parallel über eine ganze Codebasis orchestrieren. |
| `/autofix-pr` | Startet eine Cloud-Sitzung, die den Pull Request überwacht und Korrekturen nachschiebt. |
| `/install-github-app` | Die Claude-GitHub-App für ein Repository einrichten. |

### Modell, Leistung, Kosten 8

| | |
|---|---|
| `/model` | Modell wechseln und als Voreinstellung speichern. |
| `/effort` | Denkaufwand einstellen. |
| `/fast` | Schnellmodus ein- oder ausschalten (kein kleineres Modell, nur schnellere Ausgabe). |
| `/usage` | Aktuellen Verbrauch anzeigen. Alias: /cost. |
| `/usage-credits` | UMBENANNT — Nutzungskontingent verwalten oder beim Admin anfragen, wenn ein Limit erreicht ist. Vormals /extra-usage – der alte Name verweist jetzt hierher. |
| `/insights` | Bericht über die eigenen Claude-Code-Sitzungen erzeugen. |
| `/voice` | Sprachmodus ein- oder ausschalten. |
| `/powerup` | Claude-Code-Funktionen anhand kurzer interaktiver Lektionen entdecken. |

### Einstellungen und Rechte 21

| | |
|---|---|
| `/config` | Einstellungen öffnen. Alias: /settings. |
| `/permissions` | Freigaberegeln festlegen. |
| `/privacy-settings` | Datenschutzeinstellungen einsehen und ändern. |
| `/memory` | Die CLAUDE.md-Gedächtnisdateien bearbeiten. |
| `/hooks` | Hook-Konfiguration für Werkzeug-Ereignisse ansehen. |
| `/keybindings` | Die Datei mit den Tastenkürzeln öffnen. |
| `/theme` | Das Farbschema wechseln. |
| `/tui` | Die Terminal-Darstellung festlegen (Standard oder Vollbild). |
| `/color` | Farbe der Eingabezeile für diese Sitzung setzen. |
| `/focus` | Fokusansicht umschalten. |
| `/brief` | Kurzantwort-Modus umschalten. |
| `/add-dir` | Einen weiteren Arbeitsordner freigeben. |
| `/cd` | Die Sitzung in einen anderen Arbeitsordner umziehen. |
| `/advisor` | Zweitmeinung eines weiteren Modells ein- oder ausschalten. |
| `/agents` | ENTFERNT — Als Slash-Befehl entfernt – das Programm markiert ihn ausdrücklich als (removed). Unteragenten werden jetzt direkt über .claude/agents/ angelegt und verwaltet. |
| `/mcp` | MCP-Verbindungen und deren Anmeldung verwalten. |
| `/skills` | Verfügbare Skills auflisten. |

| | |
|---|---|
| `/skill-doctor` | Zeigt, welche geladenen Skills ungenutzt sind und unnötig Kontext kosten. |
| `/reload-skills` | Während der Sitzung neu oder geändert hinzugekommene Skills übernehmen. |
| `/reload-plugins` | Ausstehende Plugin-Änderungen in der laufenden Sitzung aktivieren. |
| `/auto-mode-setup` | Auto-Modus einrichten und anpassen – Umgebungskontext plus eigene Regeln. |

## Geräte, Verbindung, Anmeldung 15

| | |
|---|---|
| `/remote-control` | Diese lokale Sitzung von einem anderen Gerät aus weiterführen. |
| `/teleport` | Eine Web-Sitzung in dieses Terminal holen. |
| `/desktop` | Die Sitzung in der Desktop-App fortsetzen. Alias: /app. |
| `/mobile` | QR-Code für die Mobil-App anzeigen. Alias: /ios, /android. |
| `/web` | claude.ai/code im Browser öffnen. |
| `/web-setup` | Claude Code auf dem Web mit dem eigenen GitHub-Konto einrichten. |
| `/remote-env` | Die Standardumgebung für Cloud-Agenten wählen. |
| `/ide` | IDE-Anbindung verwalten und Status anzeigen. |
| `/chrome` | Einstellungen der Chrome-Anbindung. |
| `/design` | Zugriff des Claude-Agenten auf eigene Design-Projekte gewähren oder entziehen (samt /design-consent, /design-login, /design-revoke). |
| `/setup-bedrock` | Anmeldung, Region oder Modellzuordnung für Amazon Bedrock neu einrichten. |
| `/setup-vertex` | Anmeldung, Projekt, Region oder Modellzuordnung für Google Vertex AI neu einrichten. |
| `/login` | Anmelden. |
| `/logout` | Abmelden. |
| `/install-slack-app` | Die Claude-App für Slack einrichten. |

## Hilfe und Fehlersuche 8

| | |
|---|---|
| `/help` | Hilfe und verfügbare Befehle anzeigen. |
| `/status` | Status der aktuellen Sitzung. |
| `/doctor` | Vollständiger Systemcheck, der Probleme erkennt und beheben kann. Alias: /checkup. |
| `/debug` | Debug-Protokollierung für diese Sitzung einschalten. |
| `/heapdump` | Speicherabbild schreiben – für die Suche nach hohem Speicherverbrauch. |
| `/upgrade` | Claude Code aktualisieren. |
| `/feedback` | Rückmeldung zum Produkt senden. Alias: /share. |
| `/team-onboarding` | Teamkollegen anhand eines aus der eigenen Nutzung erstellten Leitfadens einarbeiten. |

## Mitgelieferte Skills 6

| | |
|---|---|
| `/claude-api` | Lädt die Referenz zur Claude-API und zu Managed Agents. |
| `/dataviz` | Gestaltungsrichtlinien für Diagramme und Dashboards. |
| `/deep-research` | Fächert Websuchen auf, prüft Quellen gegeneinander und schreibt einen belegten Bericht. |
| `/design-sync` | Überträgt das React-Designsystem des Projekts zu Claude Design. |
| `/fewer-permission-prompts` | Durchsucht die Verläufe nach häufigen lesenden Aufrufen und legt eine Freigabeliste an. |

| | |
|---|---|
| `/loop` | Führt eine Eingabe wiederholt aus, solange die Sitzung offen bleibt. Alias: /proactive. |

## Dauerbetrieb als Dienst, per App erreichbar

```
# im systemd-Dienst, eingepackt in ein
Pseudo-Terminal:
/usr/bin/script -qec
"$HOME/.local/bin/claude --remote-control -
-name vps-main" /dev/null
```

Das Pseudo-Terminal ist zwingend: Ohne Terminal bleibt Claude bei jeder Rückfrage stehen und kann die Antwort auch nicht speichern.

## Ein Lauf ohne Rückfragen, für Cronjobs

```
timeout 1800 "$HOME/.local/bin/claude" -p -
-model sonnet \
  --allowedTools "Bash(git *),Read" \
  --max-budget-usd 2 \
  "Deine Aufgabe hier"
```

`-p` heißt einmalig abarbeiten statt Gespräch. `--allowedTools` begrenzt, was er dabei darf, und `--max-budget-usd` deckelt die Kosten. Im Cron unbedingt den vollen Pfad verwenden – `~/.local/bin` steht dort nicht im Suchpfad.

## Wenn etwas nicht mehr funktioniert

```
claude doctor                    #
Installation prüfen
claude --safe-mode               # ohne jede
eigene Anpassung starten
claude -d api,hooks              # gezielt
mitprotokollieren
```

`--safe-mode` ist der schnellste Weg zur Antwort auf die Frage, ob eine eigene Konfiguration das Problem verursacht: Startet er damit sauber, liegt es an einer Anpassung.

## Kurz nachschlagen

```
claude --help                    # alle
Optionen
claude mcp --help                # Hilfe zu
einem Unterbefehl
```

## Die drei, die man wirklich braucht

`claude update` hält das Programm aktuell, `claude doctor` sagt dir bei Problemen, was nicht stimmt, und `claude mcp list` zeigt, welche externen Dienste angebunden sind.

## Zwei Optionen mit Bedacht

`--dangerously-skip-permissions` hebt sämtliche Rechteprüfungen auf und ist nur für abgeschottete Umgebungen ohne Internetzugang gedacht. Für den normalen Fernsteuerungsbetrieb wird es nicht gebraucht. Und `claude project purge` löscht den gesamten Projektzustand einschließlich aller Gesprächsverläufe – das lässt sich nicht rückgängig machen.