

■ // CLAUDE CODE · HANDOFF-GUIDE

Handoff statt **/compact**

Wenn der Kontext vollläuft, drückst du **/compact** — und danach fehlen genau die Details, die du gebraucht hättest. Das Zwei-Datei-System, mit dem jede neue Session sauber da weitermacht, wo du aufgehört hast.

ZWEI-DATEI-SYSTEM

COPY-VORLAGE

SESSIONSTART-HOOK

3 FAUSTREGELN

Worum es geht: `/compact` fasst deinen ganzen Verlauf zu einer Zusammenfassung zusammen — unwiederbringlich, und meist fällt raus, was du als Nächstes gebraucht hättest. Ein **expliziter Handoff** startet die nächste Session sauber, mit Ziel und nächstem Schritt statt mit altem Ballast.

1 Warum `/compact` das Problem nicht löst

`/compact` ist gut, um Platz zu sparen — aber schlecht, um über Sessions hinweg den Faden zu halten. Beim Zusammenfassen fällt raus:

- **Exakte Code-Details** und Fehlerausgaben aus früheren Schritten
- **Pfad-spezifische Regeln**, bis die passende Datei erneut gelesen wird
- Die **Sackgassen** wandern als „Kontext“ mit, statt sauber wegzufallen

UNTERSCHIED MERKEN

`/clear` startet frischen Kontext — die alte Session bleibt über `--resume` abrufbar. `/compact` überschreibt den Verlauf mit einer Zusammenfassung.

2 Das Zwei-Datei-System

Zwei Dateien, klar getrennt nach Funktion:

- **CLAUDE.md — was dauerhaft gilt.** Wird bei *jeder* Session automatisch geladen und überlebt sogar `/compact`. Hier stehen Architektur, Regeln, Constraints — alles, was immer gilt.
- **HANDOFF.md — wo genau wir weitermachen.** Die Session-Erzählung: Ziel, Stand, Entscheidungen, was zu vermeiden ist, nächster Schritt. Die nächste Session liest sie zuerst.

MERKSATZ

CLAUDE.md gibt die Lage des Landes, HANDOFF.md die Erzählung, wo du warst. Zusammen hat eine frische Session fast denselben Stand wie du am Ende der letzten.

3 Die Vorlage zum Kopieren

Am Session-Ende ausfüllen (oder von Claude ausfüllen lassen) und als `HANDOFF.md` speichern:

```
# HANDOFF — [Task-Name]

## Ziel
[Ein Satz: Endzustand, nicht Tätigkeit.]

## Stand
- Fertig: [was steht]
- Offen: [was noch fehlt]
- Gescheitert: [was nicht ging — inkl. Grund]

## Wichtige Änderungen
- [Datei / Bereich -> was geändert wurde]

## Nicht anfassen / Sackgassen
- [Ansätze, die schon durchgefallen sind]

## Nächster Schritt
[Eine konkrete erste Aktion für die nächste Session.]
```

4 Drei Faustregeln aus der Praxis

- **Bei ~60 % Kontext kompaktieren**, nicht erst bei 90 % — je später, desto mehr Detail geht verloren.
- **Sessions unter ~120k Input-Tokens halten** — lieber sauber übergeben als überfüllt weiterschleppen.
- **Immer erst den Handoff schreiben, dann schließen.** Nie umgekehrt.

5 Handoff automatisch laden (SessionStart-Hook)

Was ein `SessionStart`-Hook nach stdout schreibt, landet direkt in Claudes Kontext. So liest jede neue Session die `HANDOFF.md` ohne dein Zutun. In `.claude/settings.json`:

```
{
  "hooks": {
    "SessionStart": [
      { "hooks": [
        { "type": "command",
          "command": "[ -f ./HANDOFF.md ] && cat ./HANDOFF.md || true" } ] }
    ]
  }
}
```

Willst du den Kontext gezielt **nach dem Kompaktieren** wieder einspeisen, setzt du den Matcher auf `compact` — genau dafür ist er da (offiziell dokumentiert; Source-Werte: `startup`, `resume`, `clear`, `compact`, `fork`).

6 Zurück in die alte Session

- `claude --continue` — knüpft an die letzte Session im aktuellen Ordner an
- `claude --resume` — zeigt einen Session-Picker

Beide lesen `CLAUDE.md` neu ein. Frisch getippte Chat-Anweisungen von damals sind aber weg — deshalb gehört alles Wichtige in `CLAUDE.md` oder `HANDOFF.md`.

■ // VORM SESSION-ENDE

Die Handoff-Checkliste

Fünf Häkchen — dann geht zwischen den Sessions nichts verloren.

- ✓ **CLAUDE.md** enthält alles Dauerhafte (Architektur, Regeln, Constraints)
- ✓ **HANDOFF.md** mit Ziel, Stand, Sackgassen, nächstem Schritt
- ✓ Bei ~60 % **Kontext** kompaktiert, nicht erst bei 90 %
- ✓ **Optional:** SessionStart-Hook lädt `HANDOFF.md` automatisch
- ✓ **Nächste Session:** `HANDOFF.md` zuerst, dann weiterarbeiten

■ // BLEIB DRAN

**Nie wieder von vorn
erklären. Dein Kontext lebt
jetzt in zwei Dateien.**

Folg mir für Claude-Code-Workflows ohne Hype — ehrliche Setups,
keine Buzzwords.

@ki.borish