
NILS HARTMANN
https://nilshartmann.net
Claude
Code
an eigene Bedürfnisse
anpassen
ENTERJS AGENTIC | 22. OKTOBER 2026 | HTTPS://SPRING.SCHULE

NILS HARTMANN

nils@nilshartmann.net

Freiberuflicher Software-Entwickler, –Architekt, Coach, Trainer

Java, Spring, GraphQL, React, TypeScript

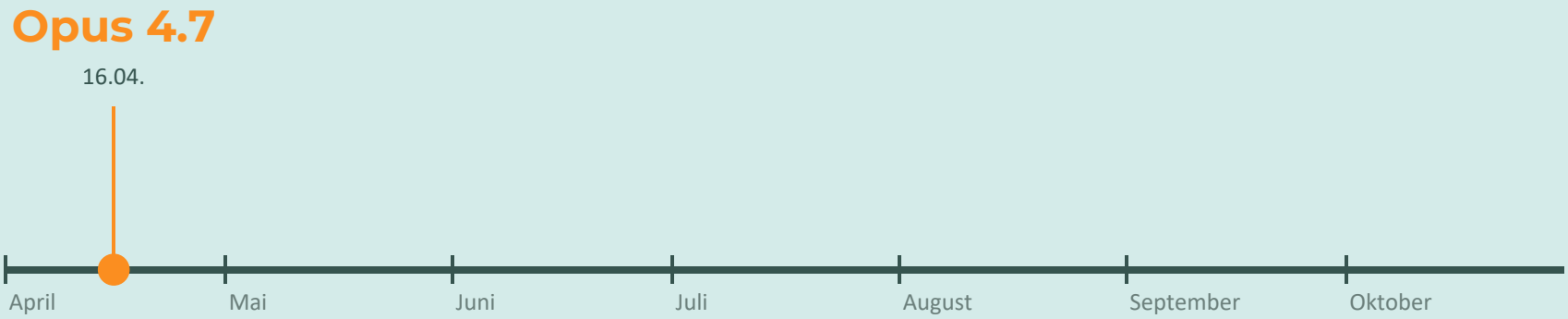


https://react.schule/video-kurs-react

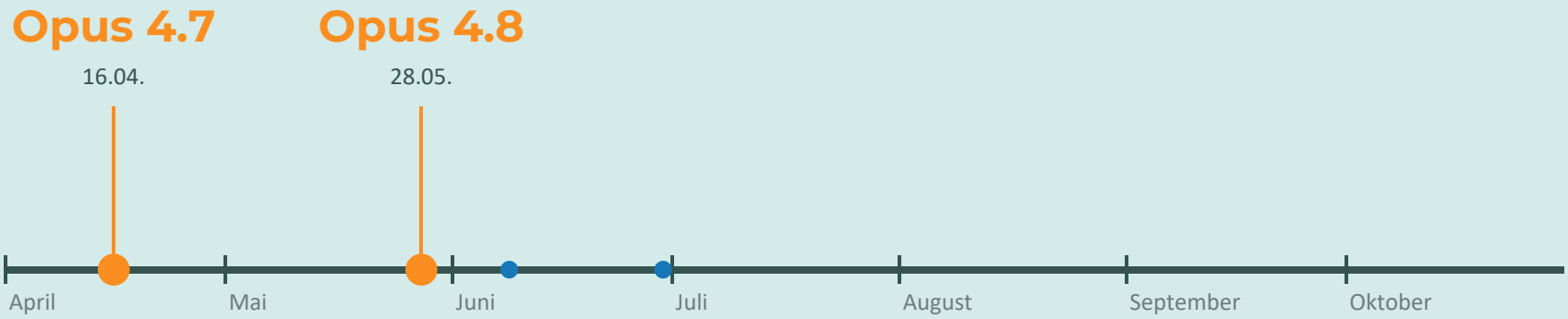


https://reactbuch.de

IN EINEM HALBEN JAHR VIER OPUS-VERSIONEN



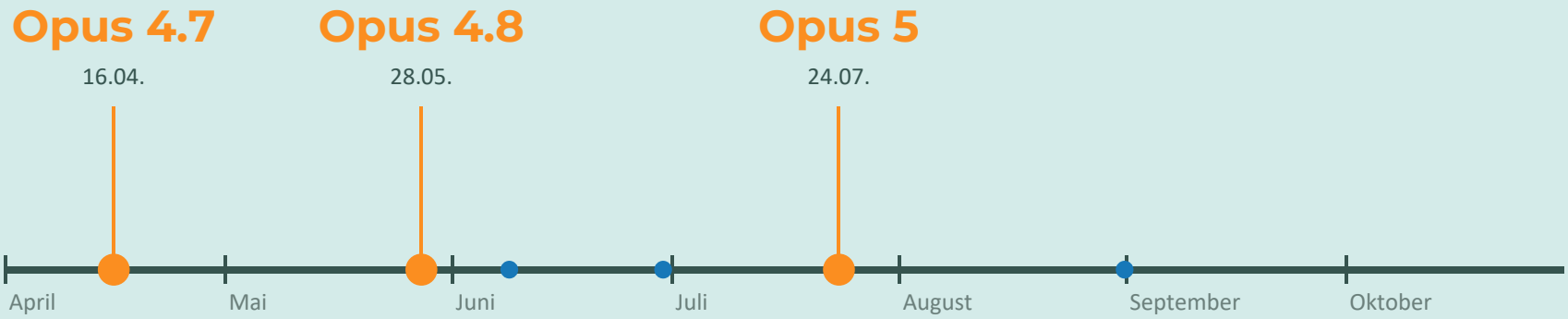
IN EINEM HALBEN JAHR VIER OPUS-VERSIONEN



Fable 5 09.06.

Sonnet 5 30.06.

IN EINEM HALBEN JAHR VIER OPUS-VERSIONEN

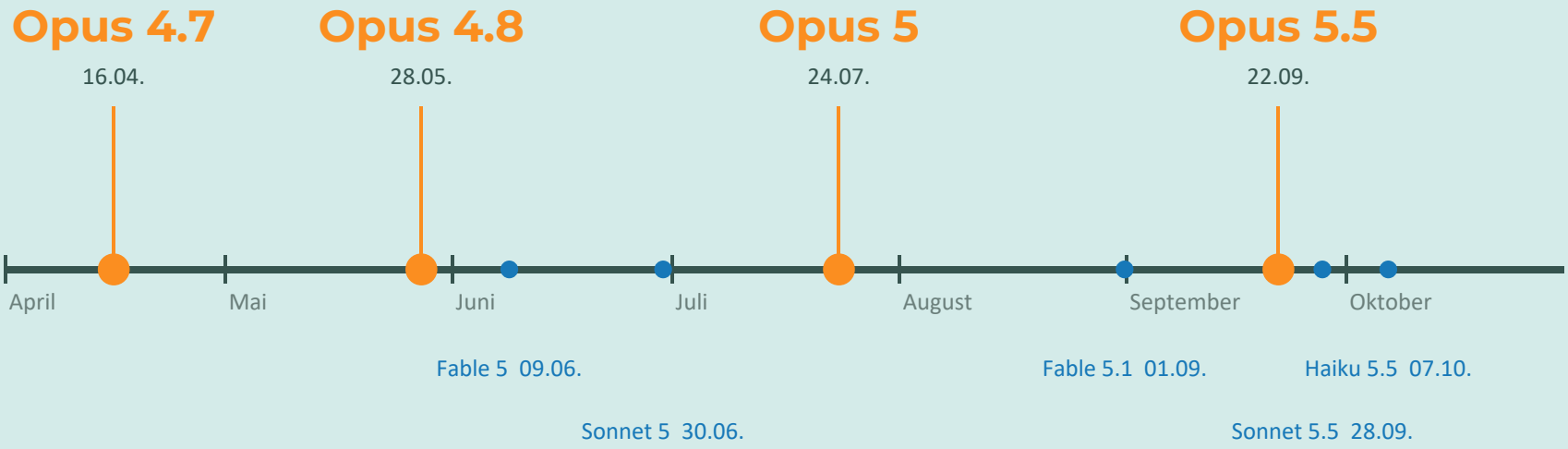


Fable 5 09.06.

Fable 5.1 01.09.

Sonnet 5 30.06.

IN EINEM HALBEN JAHR VIER OPUS-VERSIONEN

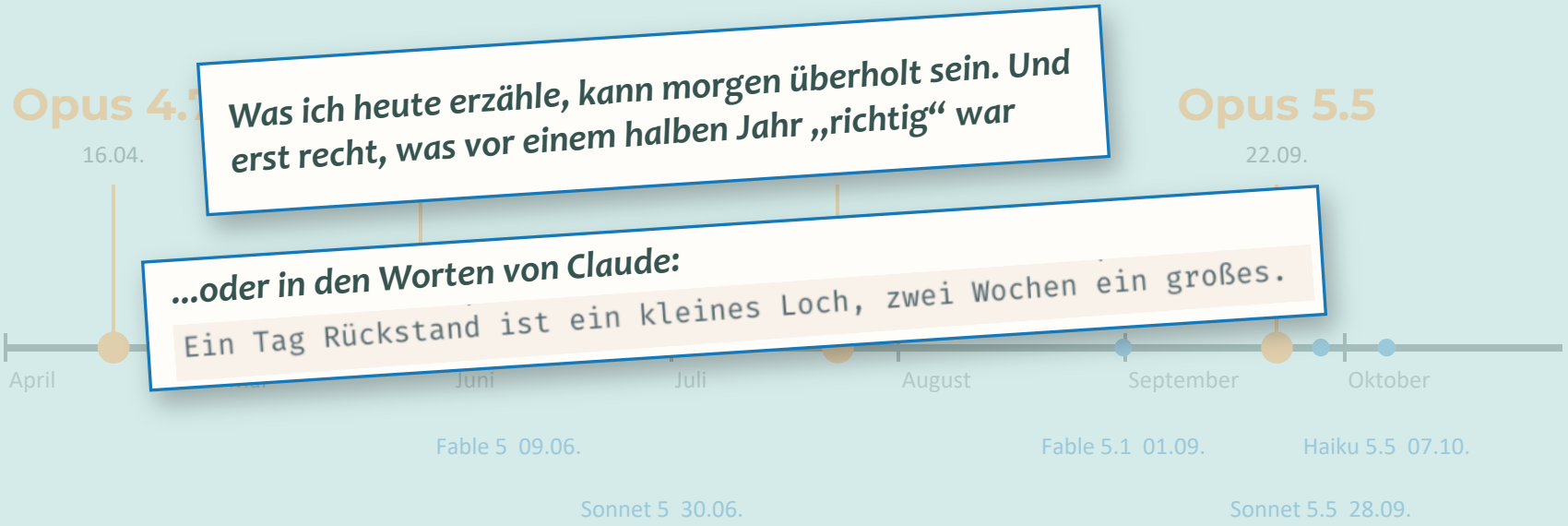


IN EINEM HALBEN JAHR VIER OPUS-VERSIONEN

Was ich heute erzähle, kann morgen überholt sein. Und erst recht, was vor einem halben Jahr „richtig“ war



IN EINEM HALBEN JAHR VIER OPUS-VERSIONEN



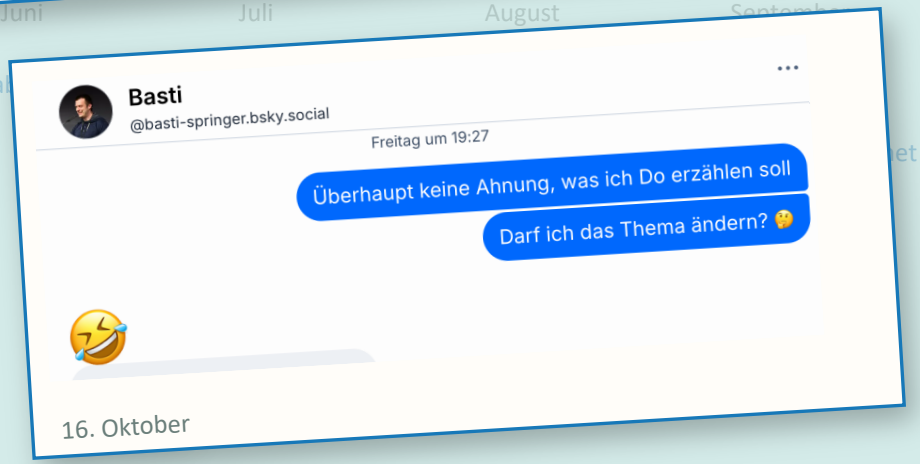
IN EINEM HALBEN JAHR VIER OPUS-VERSIONEN

Was ich heute erzähle, kann morgen überholt sein. Und erst recht, was vor einem halben Jahr „richtig“ war

...oder in den Worten von Claude:

Ein Tag Rückstand ist ein kleines Loch, zwei Wochen ein großes.

Opus 4.7
16.04.
Opus 5.5
22.09.
April
Juni
Juli
August
Oktober
Haiku 5.5 07.10.
et 5.5 28.09.





NILS HARTMANN
https://nilshartmann.net

Claude Code

~~an eigene Bedürfnisse~~ Einsatzbericht

EIN HALBES JAHR CLAUDE CODE

- Hier die Fakten, damit wir über dasselbe reden.

(Claude Code)

EIN HALBES JAHR CLAUDE CODE

- `Hier die Fakten, damit wir über dasselbe reden.` (Claude Code)

Betrachtungszeitraum: April bis Oktober 2026

EIN HALBES JAHR CLAUDE CODE

- `Hier die Fakten, damit wir über dasselbe reden.` (Claude Code)

Betrachtungszeitraum: April bis Oktober 2026

Daten: Prompt-Historie und Claude Code Sessions, aufbereitet von Claude Code

> ***„Sieh dir mal alle meine vorherigen Claude Sessions an. Prüf mal, ob du darin Themen für einen Vortrag auf dem EnterJS Agentic Day findest. Nicht: Arbeiten mit TanStack Query, sondern auf der Meta-Ebene.“***
>
> erster Prompt zu diesem Vortrag

MEINE EINSATZGEBIETE

Entwickeln

- Code schreiben, reviewen, ...
- eigene Werkzeuge bauen

Vorträge und Workshops

- Ideen sammeln, planen, umsetzen

Lernen

- neue APIs, Frameworks, etc.
- Claude Code selbst lernen

Im Projekt

- Konzepte bewerten, Tickets schreiben
- programmiert wird dort zeitweise kaum

> ***„At this point, it is safe to say that coding is largely solved.“***
>
> Boris Cherny, Entwickler von Claude Code, Mai 2026

Meine Verwendung

- nur Claude Code im Terminal
- claude.ai für Recherchen
- die Claude-App auf dem Handy für Remote Control (58 ×, davon 41 im Oktober)
- keine Extension für VS Code oder IntelliJ, kein Claude Desktop

- IntelliJ/VS Code und Git Client: Diff lesen, debuggen, ausprobieren

Meine Verwendung

- nur Claude Code im Terminal
- claude.ai für Recherchen, Claude Design für Design
- die Claude-App auf dem Handy für Remote Control (58 ×, davon 41 im Oktober)
- keine Extension für VS Code oder IntelliJ, kein Claude Desktop

- IntelliJ/VS Code und Git Client: Diff lesen, debuggen, ausprobieren

- Prompt werden länger und länger...

Ohne tippen...

- Prompts werden länger und länger...

Ohne tippen...

- Prompts werden länger und länger... VoiceInk für Sprachnachrichten



Ohne tippen...

- Prompts werden länger und länger... VoiceInk für Sprachnachrichten



VoiceInk funktioniert prima, aber...

„Clot“ *„Cloud“*

...ausgerechnet „Claude“ erkennt es fast nie richtig

EIN BILD SAGT MEHR ALS 1000 WORTE

Screenshots verwenden

EIN BILD SAGT MEHR ALS 1000 WORTE

Screenshots verwenden

- Korrekturen im Design: die Stelle zeigen, die nicht stimmt
- Fehler in der eigenen Anwendung zeigen, statt sie nachzuerzählen
- eine fremde Oberfläche erklären lassen, Bild für Bild

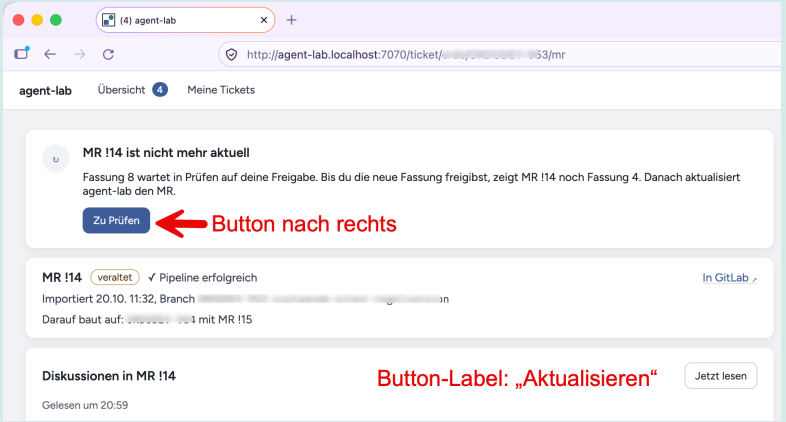
```
● Der Screenshot zeigt einen Fehler, den ich sonst nicht gefunden hätte: input {
  w-full } im Stylesheet zieht auch die Checkbox auf die volle Breite. Das repariere
  ich im Startpunkt-Commit, wo die Regel steht.

  Rebased onto $NEU, read 1 file, ran 7 shell commands

● Die Checkbox hat jetzt die richtige Größe, aber der Abstand zum Text fehlt. Ich
  prüfe, woran das liegt:

  Read 1 file, ran 3 shell commands

● Der Screenshot stimmt jetzt. Ich trage die Korrektur dorthin, wo sie hingehört,
```



EIN BILD SAGT MEHR ALS 1000 WORTE

Screenshots verwenden

- Korrekturen im Design: die Stelle zeigen, die nicht stimmt
- Fehler in der eigenen Anwendung zeigen, statt sie nachzuerzählen
- eine fremde Oberfläche erklären lassen, Bild für Bild

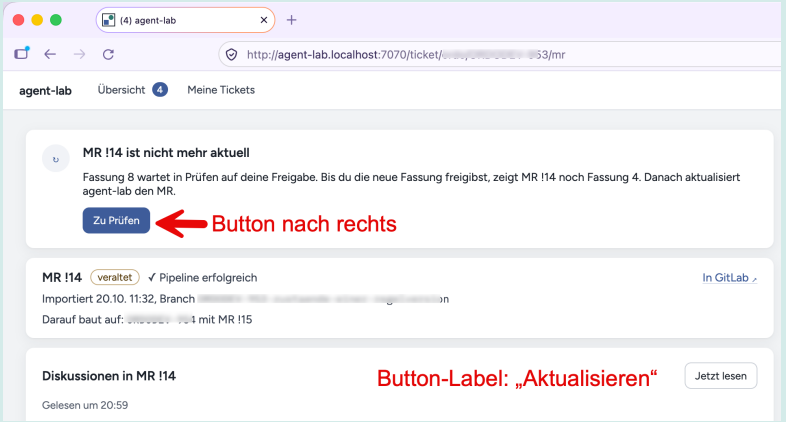
```
● Der Screenshot zeigt einen Fehler, den ich sonst nicht gefunden hätte: input {
  w-full } im Stylesheet zieht auch die Checkbox auf die volle Breite. Das repariere
  ich im Startpunkt-Commit, wo die Regel steht.

  Rebased onto $NEU, read 1 file, ran 7 shell commands

● Die Checkbox hat jetzt die richtige Größe, aber der Abstand zum Text fehlt. Ich
  prüfe, woran das liegt:

  Read 1 file, ran 3 shell commands

● Der Screenshot stimmt jetzt. Ich trage die Korrektur dorthin, wo sie hingehört,
```



Mitnahme **Zeigt Claude, was ihr seht, statt es zu beschreiben**

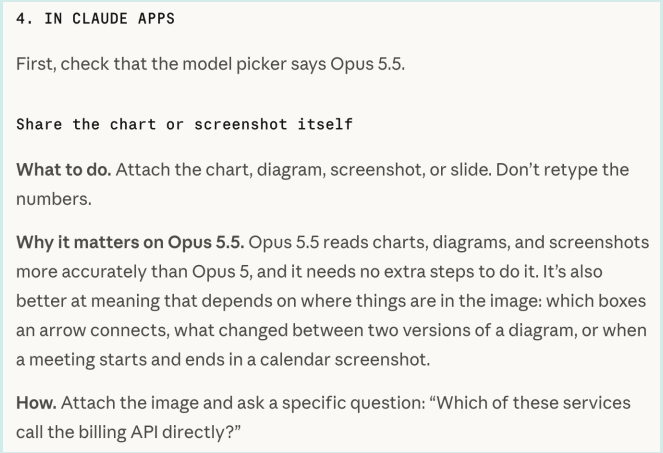
EIN BILD SAGT MEHR ALS 1000 WORTE

Screenshots verwenden

- Korrekturen im Design: die Stelle zeigen, die nicht stimmt
- Fehler in der eigenen Anwendung zeigen, statt sie nachzuerzählen
- eine fremde Oberfläche erklären lassen, Bild für Bild

Zu beachten

- Das Bild bleibt bis zum Ende der Session im Kontext
- Keine Secrets auf dem Screenshot



4. IN CLAUDE APPS

First, check that the model picker says Opus 5.5.

Share the chart or screenshot itself

What to do. Attach the chart, diagram, screenshot, or slide. Don't retype the numbers.

Why it matters on Opus 5.5. Opus 5.5 reads charts, diagrams, and screenshots more accurately than Opus 5, and it needs no extra steps to do it. It's also better at meaning that depends on where things are in the image: which boxes an arrow connects, what changed between two versions of a diagram, or when a meeting starts and ends in a calendar screenshot.

How. Attach the image and ask a specific question: "Which of these services call the billing API directly?"

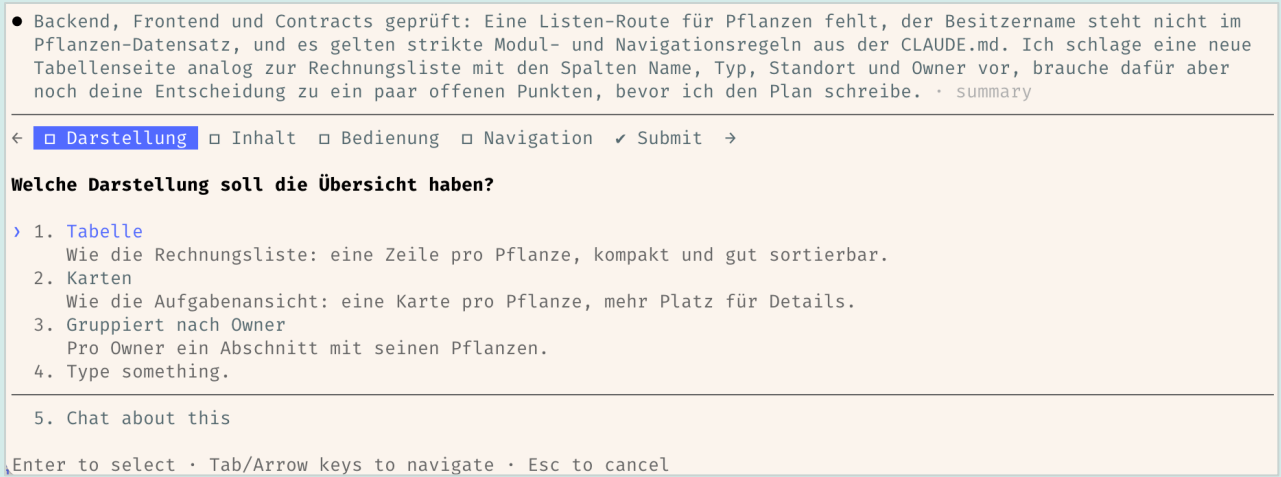
https://claude.dev/blog/getting-the-most-out-of-opus-5-5/

Mitnahme **Zeigt Claude, was ihr seht, statt es zu beschreiben**

RÜCKFRAGEN

ERLEBT

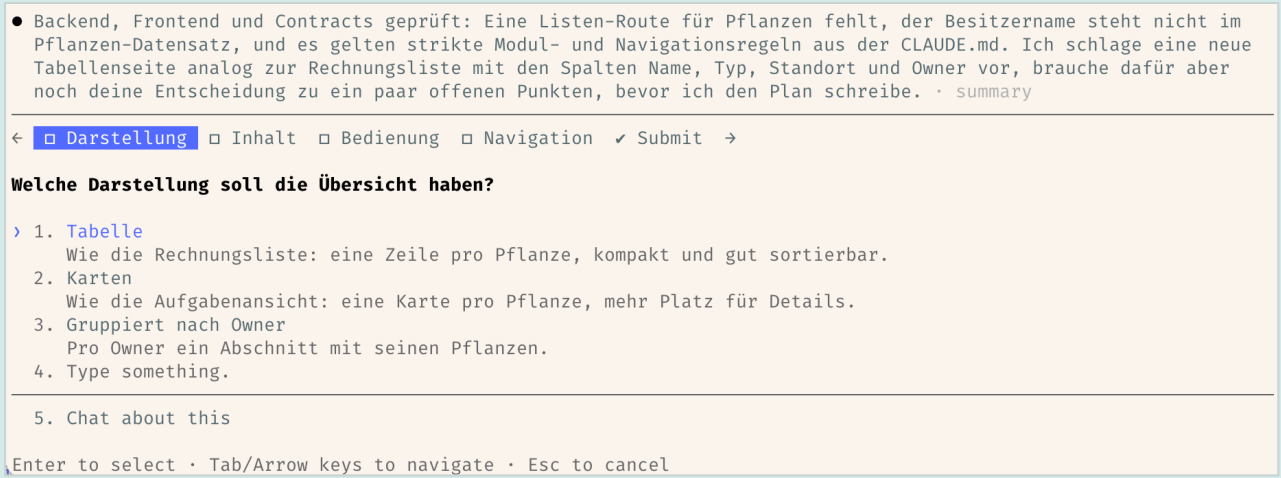
Claude stellt Rückfragen gern über ein Auswahl-Tool: drei Optionen, eine anklicken.



RÜCKFRAGEN

ERLEBT

Claude stellt Rückfragen gern über ein Auswahl-Tool: drei Optionen, eine anklicken.



Ich antworte lieber frei.

Dann kann ich eine Option wählen und ergänzen.

Und mit einer Empfehlung sehe ich, wohin Claude tendiert.

„Bitte verwende nicht das AskUserQuestion Tool, sondern stellt mir die Fragen in Textform."

ich, neunmal getippt in vier Tagen 😮‍💨

RÜCKFRAGEN

DARAUS FOLGT

Anpassung in globaler CLAUDE.md:

```
## Rückfragen
Bei Rückfragen niemals das Tool AskUserQuestion verwenden.
Rückfragen immer als Freitext stellen, eine Frage pro Turn.
Nenne bewertete Optionen und gib eine Empfehlung mit Begründung.
```

Mitnahme **Legt fest, wie Claude euch fragt: als Freitext, eine Frage, mit Empfehlung**

RÜCKFRAGEN

DARAUS FOLGT

Anpassung in globaler CLAUDE.md:

```
## Rückfragen
Bei Rückfragen niemals das Tool AskUserQuestion verwenden.
Rückfragen immer als Freitext stellen, eine Frage pro Turn.
Nenne bewertete Optionen und gib eine Empfehlung mit Begründung.
```

Die Grenze von „eine pro Turn“

- Machmal stehen viele Fragen an, z.B. in einer Plan-Session
- Eine pro Turn hieß dann: 38 von 62 Prompts waren „ok“, „A“ oder „B“. 😮‍💨

Mitnahme **Legt fest, wie Claude euch fragt: als Freitext, eine Frage, mit Empfehlung**

RÜCKFRAGEN

DARAUS FOLGT

Fragen und Antworten stehen im Dokument, nicht im Chat

```
## 3.2 Planen und Ausführen trennen
Zwei Dateien, zwei Sessions, oft zwei Modelle.

> ? Geht das auch ohne eigenen Workflow?
> !! Ja. Plan in eine Datei schreiben lassen,
>    neue Session starten, auf die Datei zeigen.


## 3.3 Modell-Wahl
Claude empfiehlt ein Modell.

> ? Erst im Chat besprechen, dann anpassen
```

Beispiel, nachgestellt

- Ich gehe das ganze Dokument durch und schreibe meine Fragen mit **?** an die Stelle
- Claude beantwortet alle in einem Rutsch, mit **!!** an derselben Stelle, oder ändert das Dokument gleich
- Einzelne Fragen bekommen den Zusatz: erst im Chat besprechen
- Umgekehrt schreibt Claude seine Rückfragen als **?** an die Stelle

RÜCKFRAGEN

DARAUS FOLGT

Fragen und Antworten stehen im Dokument, nicht im Chat

```
## 3.2 Planen und Ausführen trennen
Zwei Dateien, zwei Sessions, oft zwei Modelle.

> ? Geht das auch ohne eigenen Workflow?
> !! Ja. Plan in eine Datei schreiben lassen,
>    neue Session starten, auf die Datei zeigen.


## 3.3 Modell-Wahl
Claude empfiehlt ein Modell.

> ? Erst im Chat besprechen, dann anpassen
```

Beispiel, nachgestellt

- Ich gehe das ganze Dokument durch und schreibe meine Fragen mit **?** an die Stelle
- Claude beantwortet alle in einem Rutsch, mit **!!** an derselben Stelle, oder ändert das Dokument gleich
- Einzelne Fragen bekommen den Zusatz: erst im Chat besprechen
- Umgekehrt schreibt Claude seine Rückfragen als **?** an die Stelle

Mitnahme **Bei langen Dokumenten: Fragen an die Stelle schreiben, nicht in den Chat**

SPRACHE

ERLEBT

Deutsche Sprache, schwere Sprache

SPRACHE

ERLEBT

Deutsche Sprache, schwere Sprache

```
Der Testlauf lief sauber durch – Überschriften, Codeblock, Sonderzeichen,
leere Zwischenablage und Aufräumen funktionieren wie gewünscht. Nur bei einer
einzigen Zeile steht fälschlich „1 Zeilen“ statt „1 Zeile“, das behebe ich
jetzt.
```



SPRACHE

ERLEBT

Deutsche Sprache, schwere Sprache

Der Testlauf lief sauber durch – Überschriften, Codeblock, Sonderzeichen, leere Zwischenablage und Aufräumen funktionieren wie gewünscht. Nur bei einer einzigen Zeile steht fälschlich „1 Zeilen“ statt „1 Zeile“, das behebe ich jetzt.



Beide Gruppen liegen an geteilter Ahnenschaft 😐

die Blackbox bekommt ein Fenster. 🤔

SPRACHE

ERLEBT

Deutsche Sprache, schwere Sprache

Der Testlauf lief sauber durch – Überschriften, Codeblock, Sonderzeichen, leere Zwischenablage und Aufräumen funktionieren wie gewünscht. Nur bei einer einzigen Zeile steht fälschlich „1 Zeilen“ statt „1 Zeile“, das behebe ich jetzt.



Beide Gruppen liegen an geteilter Ahnenschaft 😐

Das Risiko hat zwei Türen, und ein Test sollte beide zuhalten: 🤓

Was der Plansitzung an Nachzug im Auszug begegnen wird 🤔

ist, nicht, was implementiert uns ist

die Blackbox bekommt ein Fenster. 🤔

SPRACHE

ERLEBT

Deutsche Sprache, schwere Sprache

Der Testlauf lief sauber durch – Überschriften, Codeblock, Sonderzeichen, leere Zwischenablage und Aufräumen funktionieren wie gewünscht. Nur bei einer einzigen Zeile steht fälschlich „1 Zeilen“ statt „1 Zeile“, das behebe ich jetzt.



Beide Gruppen liegen an geteilter Ahnenschaft 😐

Das Risiko hat zwei Türen, und ein Test sollte beide zuhalten: 🤓

D wie Entscheidung, weil E beim Epic und S bei der Story liegt. 🫣

Was der Plansitzung an Nachzug im Auszug begegnen wird 🤔

die Blackbox bekommt ein Fenster. 🤔

SPRACHE

ERLEBT

Deutsche Sprache, schwere Sprache

Der Testlauf lief sauber durch – Überschriften, Codeblock, Sonderzeichen, leere Zwischenablage und Aufräumen funktionieren wie gewünscht. Nur bei einer einzigen Zeile steht fälschlich „1 Zeilen“ statt „1 Zeile“, das behebe ich jetzt.



Beide Gruppen liegen an geteilter Ahnenschaft 😐

Das Risiko hat zwei Türen, und ein Test sollte beide zuhalten: 🤓

D wie Entscheidung, weil E beim Epic und S bei der Story liegt. 🫣

Was der Plansitzung an Nachzug im Auszug begegnen wird 🤔

ist, nicht, was implementiert uns is
die Blackbox bekommt ein Fenster. 🤔

Bevor ich etwas umbaue, sehe ich mir an, welche Commits sich an denselben Dateien vergriffen haben. 😬

SPRACHE

ERLEBT

Deutsche Sprache, schwere Sprache

Der Testlauf lief sauber durch – Überschriften, Codeblock, Sonderzeichen, leere Zwischenablage und Aufräumen funktionieren wie gewünscht. Nur bei einer einzigen Zeile steht fälschlich „1 Zeilen“ statt „1 Zeile“, das behebe ich jetzt. 👍

Beide Gruppen liegen an geteilter Ahnenschaft 😐

Das Risiko hat zwei Türen, und ein Test sollte beide zuhalten: 🤓

Die eine didaktische Warze der klassischen Variante, 🤯

D wie Entscheidung, weil E beim Epic und S bei der Story liegt. 🙈

Was der Plansitzung an Nachzug im Auszug begegnen wird 🤔

die Blackbox bekommt ein Fenster. 🤔

Bevor ich etwas umbaue, sehe ich mir an, welche Commits sich an denselben Dateien vergriffen haben. 😬

SPRACHE

ERLEBT

Deutsche Sprache, schwere Sprache

„Warum sprichst du immer von Naht und Kante?“

ich, im Mai

Der Testlauf lief sauber durch – Überschriften, Codeblock, Sonderzeichen, leere Zwischenablage und Aufräumen funktionieren wie gewünscht. Nur bei einer einzigen Zeile steht fälschlich „1 Zeilen“ statt „1 Zeile“, das behebe ich jetzt.

Beide

Eko hat zwei Türen, und ein Test sollte beide zuhalten:

Die eine didaktische Warze der klassischen Variante,

D wie Entscheidung, weil E beim Epic und S bei der Story liegt.

Was der Plansitzung an Nachzug im Auszug begegnen wird

die Blackbox bekommt ein Fenster.

Bevor ich etwas umbaue, sehe ich mir an, welche Commits sich an denselben Dateien vergriffen haben.

SPRACHE

ERLEBT

Deutsche Sprache, schwere Sprache

„Warum sprichst du immer von Naht und Kante?“

ich, im Mai

Aktuell 25 Regeln mit gut 1.6000 Wörtern

CLAUDE.md, im Oktober



Der Testlauf lief sauber durch – Überschriften, Codeblock, Sonderzeichen, leere Zwischenablage und Aufräumen funktionieren wie gewünscht. Nur bei einer einzigen Zeile steht fälschlich „1 Zeilen“ statt „1 Zeile“, das behebe ich jetzt.

Beide

eide zuhalten:



Die eine

D wie Entscheid

beim Epic und S bei der Story liegt.



Was der Plansitzung an Nachzug im Auszug begegnen wird



die Blackbox bekommt ein Fenster.



Bevor ich etwas umbaue, sehe ich mir an, welche Commits sich an denselben Dateien vergriffen haben.



SPRACHE

ERLEBT

Deutsche Sprache, schwere Sprache

„Warum sprichst du immer von Naht und Kante?“

ich, im Mai

Aktuell 25 Regeln mit gut 1.6000 Wörtern

CLAUDE.md, im Oktober

Da die Datei noch wachsen kann, solltest du vor dem Vortrag noch einmal zählen.

CLAUDE, 21. Oktober

SPRACHE

Sprache wurde mit Opus 5.5 deutlich besser, keine schiefen Bilder mehr, kein übertriebener Enthusiasmus mehr...

Beide Gruppen liegen an geteilter Ahnenschaft

Das Risiko hat zwei Türen, und ein Test sollte beide zuhalten:

Die eine didaktische Warze der klassischen Variante,

D wie Entscheidung, weil E beim Epic und S bei der Story liegt.

Was der Plansitzung an Nachzug im Auszug begegnen wird

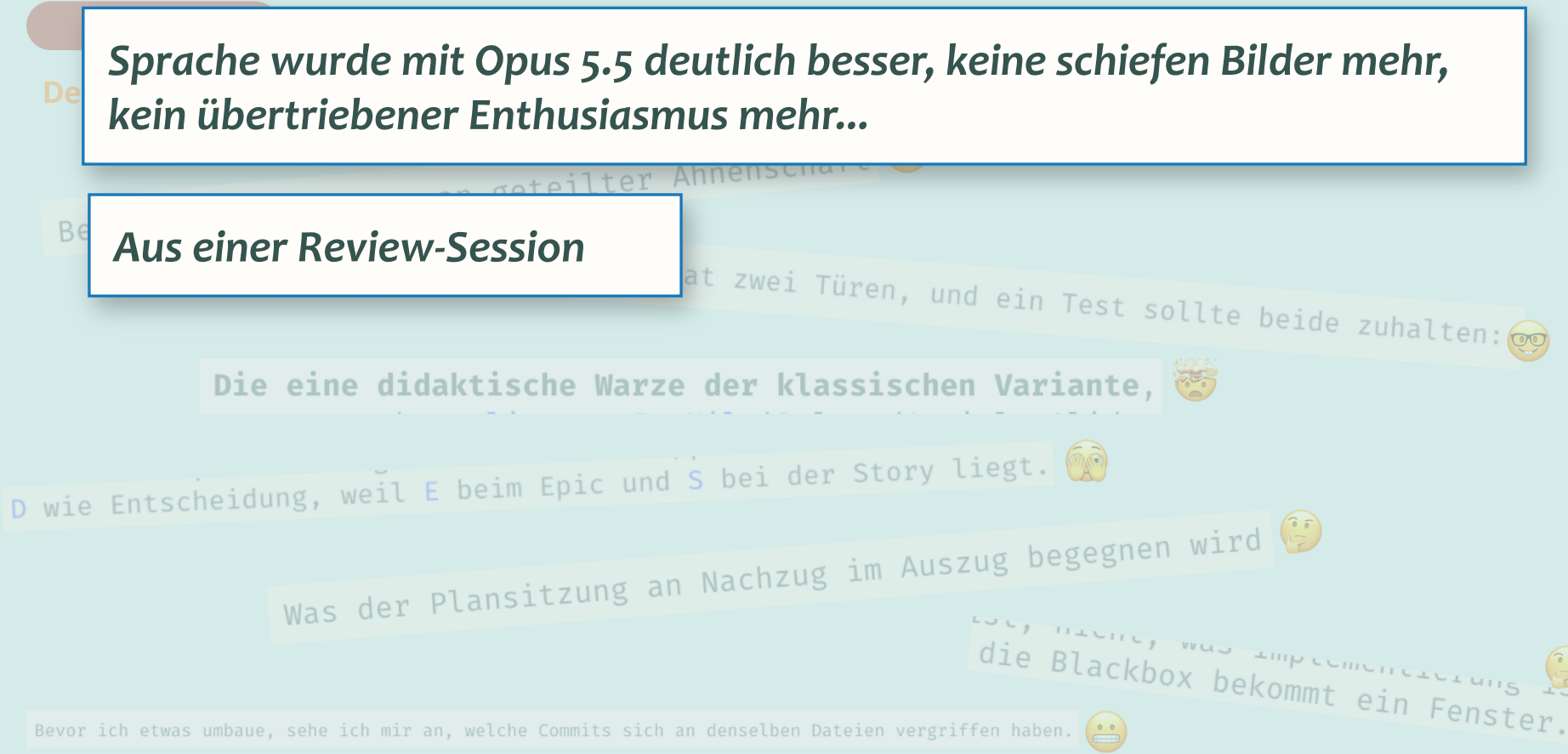
die Blackbox bekommt ein Fenster.

Bevor ich etwas umbaue, sehe ich mir an, welche Commits sich an denselben Dateien vergriffen haben.

SPRACHE

Sprache wurde mit Opus 5.5 deutlich besser, keine schiefen Bilder mehr, kein übertriebener Enthusiasmus mehr...

Aus einer Review-Session



SPRACHE

Sprache wurde mit Opus 5.5 deutlich besser, keine schiefen Bilder mehr, kein übertriebener Enthusiasmus mehr…

Aus einer Review-Session

Das sind no-regrets-Änderungen. Die t.sh ist die niedlichste: Der Agent baut sich spontan ein Werkzeug, das alle brauchen würden. Das gehört ins Repo.

4. Oktober

SPRACHE

Sprache wurde mit Opus 5.5 deutlich besser, keine schiefen Bilder mehr, kein übertriebener Enthusiasmus mehr...

Aus einer Review-Session

```
Das sind no-regrets-Änderungen. Die t.sh ist die niedlichste: Der Agent baut
sich spontan ein Werkzeug, das alle brauchen würden. Das gehört ins Repo.
```

4. Oktober

```
Lass mich das speichern – das ist Gold für zukünftige Optimierung:

Wrote 2 memories
```

4. Oktober



SPRACHE

| Haiku 4.5 |



Sprache wurde mit Opus 5.5 deutlich
kein übertriebener Enthusiasmus mehr...

keine schiefen Bilder mehr,

Das sind no-regrets-Änderungen. Die t.sh ist die niedlichste: Der Agent baut sich spontan ein Werkzeug, das alle brauchen würden. Das gehört ins Repo.

4. Oktober

Lass mich das speichern – das ist Gold für zukünftige Optimierung:

Wrote 2 memories

4. Oktober

| Haiku 4.5 |



Review des Reviews mit Opus 5.5

| Haiku 4.5 |



Review des Reviews mit Opus 5.5

1. **classpath.sh ist kein Problem.** Die Analyse hat Spalte 5 (load average) als Dauer gelesen. Die tatsächlichen Laufzeiten stehen in Spalte 3:

| Haiku 4.5 |



Review des Reviews mit Opus 5.5

1. **classpath.sh ist kein Problem.** Die Analyse hat Spalte 5 (load average) als Dauer gelesen. Die tatsächlichen Laufzeiten stehen in Spalte 3:

2. **„57k auf 836k in 1 Minute“ stimmt nicht.** Nach 1,3 min lag der Kontext bei 231.000 Tokens, die 836.000 erst am Ende nach 68 min.

| Haiku 4.5 |



Review des Reviews mit Opus 5.5

1. **classpath.sh ist kein Problem.** Die Analyse hat Spalte 5 (load average) als Dauer gelesen. Die tatsächlichen Laufzeiten stehen in Spalte 3:

2. **„57k auf 836k in 1 Minute“ stimmt nicht.** Nach 1,3 min lag der Kontext bei 231.000 Tokens, die 836.000 erst am Ende nach 68 min.

3. **Die Cache-Zahlen sind falsch.**

| Haiku 4.5 |



Review des Reviews mit Opus 5.5

1. **classpath.sh ist kein Problem.** Die Analyse hat Spalte 5 (load average) als Dauer gelesen. Die tatsächlichen Laufzeiten stehen in Spalte 3:

2. **„57k auf 836k in 1 Minute“ stimmt nicht.** Nach 1,3 min lag der Kontext bei 231.000 Tokens, die 836.000 erst am Ende nach 68 min.

3. **Die Cache-Zahlen sind falsch.**

...

| Haiku 4.5 |



Review des Reviews mit Opus 5.5

1. **classpath.sh ist kein Problem.** Die Analyse hat Spalte 5 (load average) als Dauer gelesen. Die tatsächlichen Laufzeiten stehen in Spalte 3:

2. **„57k auf 836k in 1 Minute“ stimmt nicht.** Nach 1,3 min lag der Kontext bei 231.000 Tokens, die 836.000 erst am Ende nach 68 min.

3. **Die Cache-Zahlen sind falsch.**

...

7. **Die geschätzten Gewinne sind zu hoch.**

| Haiku 4.5 |

👀

Review des Reviews mit Opus 5.5

1. **classpath.sh ist kein Problem.** Die Analyse hat Spalte 5 (load average) als Dauer gelesen. Die tatsächlichen Laufzeiten stehen in Spalte 3:

2. **„57k auf 836k in 1 Minute“ stimmt nicht.** Nach 1,3 min lag der Kontext bei 231.000 Tokens, die 836.000 erst am Ende nach 68 min.

3. **Die Cache-Zahlen sind falsch.**

...

7. **Die geschätzten Gewinne sind zu hoch.**



Augen auf, bei der Modell-Wahl...

1. **classpath.sh ist kein Problem.** Die Analyse hat Spalte 5 (load average) als Dauer gelesen. Die tatsächlichen Laufzeiten stehen in Spalte 3:

2. **„57k auf 836k in 1 Minute“ stimmt nicht.** Nach 1,3 min lag der Kontext bei 231.000 Tokens, die 836.000 erst am Ende nach 68 min.

3. **Die Cache-Zahlen sind falsch.**

...

7. **Die geschätzten Gewinne sind zu hoch.**

Augen auf, bei der Modell-Wahl...

Claude merkt sich das Modell für die nächste Session

Augen auf, bei der Modell-Wahl...

Claude merkt sich das Modell für die nächste Session

gilt auch andersrum: aus Versehen zu teures Modell 💸

You've used 76% of your weekly limit

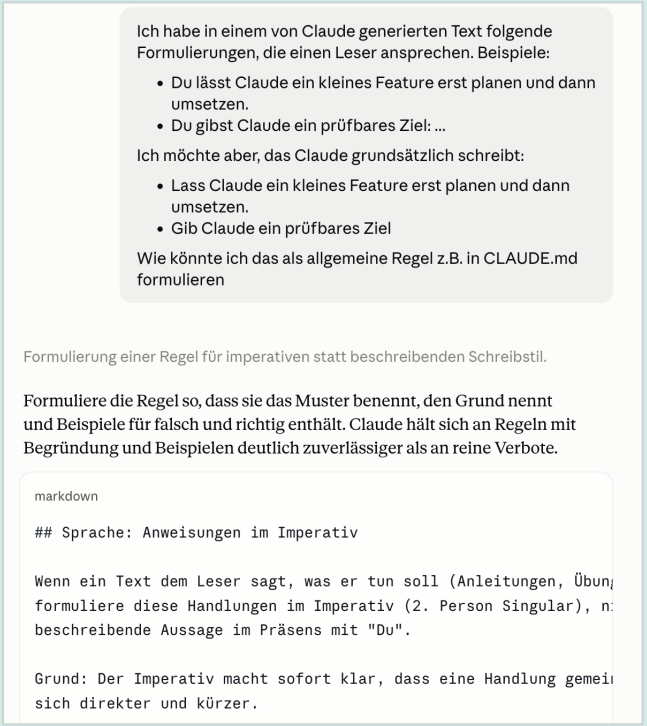
ERZEUGTE REGELN

ERLEBT

Aus eigenen Beispielen

Die Regeln sind teilweise mit claude.ai entstanden.

Claude leitet daraus die Regel ab, mit Grund und Beispielen.



Ich habe in einem von Claude generierten Text folgende Formulierungen, die einen Leser ansprechen. Beispiele:

- Du lässt Claude ein kleines Feature erst planen und dann umsetzen.
- Du gibst Claude ein prüfbares Ziel: ...

Ich möchte aber, das Claude grundsätzlich schreibt:

- Lass Claude ein kleines Feature erst planen und dann umsetzen.
- Gib Claude ein prüfbares Ziel

Wie könnte ich das als allgemeine Regel z.B. in CLAUDE.md formulieren

Formulierung einer Regel für imperativen statt beschreibenden Schreibstil.

Formuliere die Regel so, dass sie das Muster benennt, den Grund nennt und Beispiele für falsch und richtig enthält. Claude hält sich an Regeln mit Begründung und Beispielen deutlich zuverlässiger als an reine Verbote.

markdown

```
## Sprache: Anweisungen im Imperativ

Wenn ein Text dem Leser sagt, was er tun soll (Anleitungen, Übung
formuliere diese Handlungen im Imperativ (2. Person Singular), n
beschreibende Aussage im Präsens mit "Du".

Grund: Der Imperativ macht sofort klar, dass eine Handlung gemei
sich direkter und kürzer.
```

ERZEUGTE REGELN

ERLEBT

Aus fremdem Material

> *„Ich habe den folgenden Skill von Anthropic übernommen. Kannst du den so anpassen, das er für Slidedecks geeignet ist?“*
>
> ich, in claude.ai, 04.10.

Übernommen habe ich nur die Regeln zum Design. Ganze Abschnitte habe ich gestrichen, weil ich schon eine Story hatte.

Ergebnis: ein Skill für Folien.

ERZEUGTE REGELN

DARAUS FOLGT

Regeln und Skills müsst ihr nicht selbst schreiben

Aus eigenen Beispielen

- gut und schlecht nebeneinander, Claude leitet die Regel ab
- wiederkehrende Fehlgriffe als Liste gesperrter Wörter
- geht auch mit dem Stil einer Codebasis

Aus fremdem Material

- Skills aus Sammlungen, Artikel, Doku, eigene ältere Projekte
- erst ein Vorschlag, keine Umsetzung
- nicht 1:1: Claude sagt, was zu euch passt und was nicht

```
Schlecht: Auch der Auftrag liegt verschlsselt
          vor, und ebenso die Quittung.
Gut:      Auch der Auftrag und die Quittung
          liegen verschlüsselt vor.
Regel:    Kein nachgeschobenes Subjekt.
```

Mitnahme **Schreibt Regeln nicht selbst. Lasst sie ableiten, aus euren Beispielen oder aus fremdem Material**

KOMMT DIE REGEL AN?

ERLEBT

Eine CLAUDE.md kann andere Dateien einbinden.

Ob Claude sie verwendet, sieht man der Antwort nicht an.

```
# CLAUDE.md im Projekt
@~/.claude/addons/konventionen/sprache.md
```

Der Import aus der Projekt-CLAUDE.md kam nicht an, weder per Symlink noch per absolutem Pfad. Aus der ~/.claude/CLAUDE.md kam beides an.

Ergebnis des Tests

Möglicher Grund: Ein Import außerhalb des Projekts braucht eine einmalige Freigabe im Dialog. Wer ablehnt, wird nicht wieder gefragt.

CLAUDE.md oder der Skill-Ordner wird nicht vom Compiler o.ä. überprüft.

Ein Import, der nicht verwendet wird, fällt nicht / erst spät auf.

KOMMT DIE REGEL AN?

DARAUS FOLGT

```
# in CLAUDE.md und jeder
# importierten Datei
Codewort: KOMPASS

$ claude -p --tools "" \
    "Welche Codewörter kennst du?"
```

Code-Wort in Dateien

- /context zeigt unter „Memory files“, welche Dateien geladen sind
- Kontrollgruppe: Claude mit eigenem CLAUDE_CONFIG_DIR starten, ohne globale CLAUDE.md
- Erst der Vergleich zeigt, was eine Regel bewirkt

Mitnahme **Wirkt eine Regel nicht, prüft zuerst, ob sie überhaupt geladen ist**

KOMMT DIE REGEL AN?

DARAUS FOLGT

```
# in CLAUDE.md und jeder
# importierten Datei
Codew

$ clau
    "W
```

Claude Code v2.1.143
Sonnet 4.6 with medium effort · Claude Max
~/develop/claude-code-workshop/verdant-cc

› liest du die datei unter ~/.claude/CLAUDE.md nicht ein?

● Doch, die Datei wird eingelesen — sie ist in den System-Kontext eingebettet und ich sehe den Inhalt:

```
# Begrüßung
- Schreib beim Start jeder neuen Session "Moin" zur Begrüßung
```

Ich habe das zu Beginn dieser Session schlicht vergessen umzusetzen. Das hole ich nach:



Moin!

✻ Crunched for 6s

welche

lobale

l bewirkt

Mitnahme **Wirkt eine Regel nicht, prüft zuerst, ob sie überhaupt geladen ist**

LERNEN AN DER ECHTEN AUFGABE

ERLEBT

Neue Frameworks und Konzepte lernen

> ***„Ich glaube, du musst mir erstmal auf einfache Art erklären, was das ist.“***
>
> ich, am 19.09.

LERNEN AN DER ECHTEN AUFGABE

ERLEBT

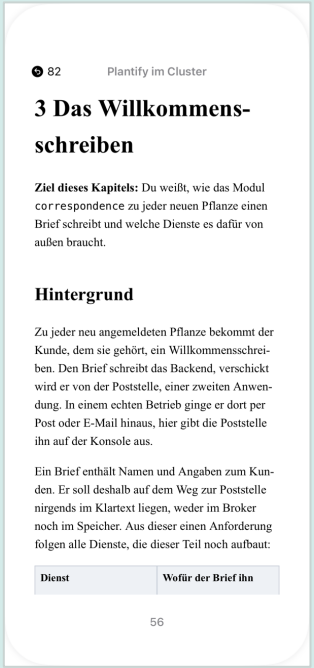
Neue Frameworks und Konzepte lernen

> ***„Ich glaube, du musst mir erstmal auf einfache Art erklären, was das ist.“***
>
> ich, am 19.09.

Tutorial anhand eigener (bekannter) Beispiele für eigene (bekannte) Fragen

Aus den Erklärungen wurde ein PDF, dann ein EPUB, am Ende ein Buch in fünf Teilen.

Prima zum unterwegs lesen



82 Plantify im Cluster

3 Das Willkommens-schreiben

Ziel dieses Kapitels: Du weißt, wie das Modul `correspondence` zu jeder neuen Pflanze einen Brief schreibt und welche Dienste es dafür von außen braucht.

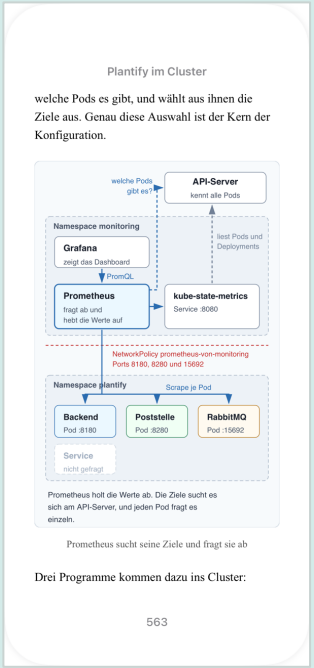
Hintergrund

Zu jeder neu angemeldeten Pflanze bekommt der Kunde, dem sie gehört, ein Willkommensschreiben. Den Brief schreibt das Backend, verschickt wird er von der Poststelle, einer zweiten Anwendung. In einem echten Betrieb ginge er dort per Post oder E-Mail hinaus, hier gibt die Poststelle ihn auf der Konsole aus.

Ein Brief enthält Namen und Angaben zum Kunden. Er soll deshalb auf dem Weg zur Poststelle nirgends im Klartext liegen, weder im Broker noch im Speicher. Aus dieser einen Anforderung folgen alle Dienste, die dieser Teil noch aufbaut:

Dienst	Wofür der Brief ihn

56



Plantify im Cluster

welche Pods es gibt, und wählt aus ihnen die Ziele aus. Genau diese Auswahl ist der Kern der Konfiguration.

Prometheus sucht seine Ziele und fragt sie ab

Drei Programme kommen dazu ins Cluster:

563

LERNEN AN DER ECHTEN AUFGABE

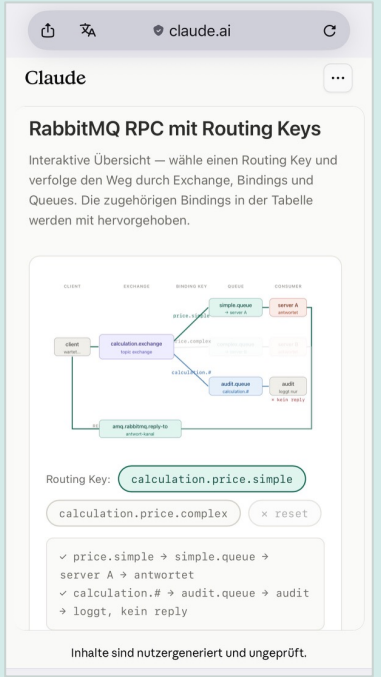
DARAUS FOLGT

Was es gut macht

- Thema und Beispiel entsprechen der echten Aufgabe
- Lange Texte in Buchform mit Grafiken besser lesbar als auf der Konsole im Chat
- Abbildungen für das eigene Gerät optimieren lassen, z. B. Hochformat fürs iPhone

Das geht auch

- Klick-Dummies für Oberflächen, bevor eine Zeile Code entsteht
- interaktive Seiten für Konzepte, Abläufe und Architekturen: eine HTML-Datei ohne Server, als Claude Artefakt auch zum Teilen
- Beispiel rechts: RabbitMQ-Konzepte (Routing Key wählen und den Weg durch Exchange, Bindings und Queues verfolgen)



https://claude.ai/public/artifacts/e856e16d-0c28-4467-86e9-f76da954e681

Mitnahme **Lernt neue Technik an eurem eigenen Problem, nicht an einem „generischem“ Tutorial**

BITTEN, PRÜFEN, SPERREN

GRENZEN VON SKILLS

ERLEBT

In der IDE habe ich integriert Explorer, Editor und Git-Tool. Mit Claude habe ich viele Sessions offen und gleichzeitig keinen Explorer, keinen Editor und kein Git-Tool

GRENZEN VON SKILLS

ERLEBT

In der IDE habe ich integriert Explorer, Editor und Git-Tool. Mit Claude habe ich viele Sessions offen und gleichzeitig keinen Explorer, keinen Editor und kein Git-Tool

Ein Skill soll das Git-Tool im Verzeichnis der Session öffnen, wie ein Shortcut in der IDE.

/open-git-client

Öffnet meinen Git-Client im aktuellen Projekt in neuem iTerm-Fenster

GRENZEN VON SKILLS

ERLEBT

In der IDE habe ich integriert Explorer, Editor und Git-Tool. Mit Claude habe ich viele Sessions offen und gleichzeitig keinen Explorer, keinen Editor und kein Git-Tool

Ein Skill soll das Git-Tool im Verzeichnis der Session öffnen, wie ein Shortcut in der IDE.

26.000

Tokens für einen Skill, der nur ein Fenster im Git-Client öffnet

GRENZEN VON SKILLS

ERLEBT

In der IDE habe ich integriert Explorer, Editor und Git-Tool. Mit Claude habe ich viele Sessions offen und gleichzeitig keinen Explorer, keinen Editor und kein Git-Tool

Ein Skill soll das Git-Tool im Verzeichnis der Session öffnen, wie ein Shortcut in der IDE.

26.000

Tokens für einen Skill, der nur ein Fenster im Git-Client öffnet

> ***„Leider ist ein Skill-Aufruf teuer, so dass ich lieber direkt Kommandos ausführen will […] Wie können wir das umstellen, dass ich keine Skills, sondern Scripte verwende?“***
>
> ich, 12.06.

GRENZEN VON SKILLS

ERLEBT

Ein anderer Skill suchte vor jeder Planung mit find nach alten Plänen. Jede Suche wollte eine Freigabe.

```
$ find docs/tasks -name '*.plan.md' \
    -exec grep -l 'Status: offen' {} +
```

Beispiel, nachgestellt

`Bash(find *)` gibt find mit -exec nicht frei. Mit -exec startet find beliebige Befehle, also fragt Claude Code jedes Mal.

GRENZEN VON SKILLS

ERLEBT

Ein anderer Skill suchte vor jeder Planung mit find nach alten Plänen. Jede Suche wollte eine Freigabe.

```
$ find docs/tasks -name '*.plan.md' \
    -exec grep -l 'Status: offen' {} +
```

Beispiel, nachgestellt

`Bash(find *)` gibt find mit -exec nicht frei. Mit -exec startet find beliebige Befehle, also fragt Claude Code jedes Mal.

„Können wir das Permission-Problem evtl. lösen, in dem wir ein script […] ausliefern, dass die Tasks sammelt und an die Session liefert? Dann könnten wir die Ausführung des Scripts in den settings.json pauschal erlauben“

ich, 06.06.

GRENZEN VON SKILLS

DARAUS FOLGT

Weitere Beispiele

- Releasewechsel: Codemod-Scripte generieren
- Dieser Vortrag: Claude hat einmal einen Generator geschrieben, seitdem baut ein Script die Folien (naja, fast...)
- Übergabe an ein Werkzeug ohne Zugriff aufs Repo, z. B. Claude Design: Den Auftrag schreibt ein Skill, das ZIP packt und entpackt ein Script

GRENZEN VON SKILLS

DARAUS FOLGT

Kein Urteil, also Script

- Einen Task archivieren heißt zwei Dateien verschieben und committen. Kein Grenzfall
- deterministisch und testbar

Weitere Beispiele

- Releasewechsel: Codemod-Scripte generieren
- Dieser Vortrag: Claude hat einmal einen Generator geschrieben, seitdem baut ein Script die Folien (naja, fast...)
- Übergabe an ein Werkzeug ohne Zugriff aufs Repo, z. B. Claude Design: Den Auftrag schreibt ein Skill, das ZIP packt und entpackt ein Script

GRENZEN VON SKILLS

DARAUS FOLGT

Kein Urteil, also Script

- Einen Task archivieren heißt zwei Dateien verschieben und committen. Kein Grenzfall
- deterministisch und testbar

Lasst Claude das Script schreiben

- Claude schreibt es einmal, danach kostet es keine Tokens mehr
- Ein Script ist oft auch schneller als dieselbe Aufgabe mit einem Modell
- Ein Script lässt sich pauschal erlauben, ein beliebiger Bash-Befehl nicht
- Ein Skill kann das Script aufrufen: Der Skill "denkt", das Script sammelt Daten, ruft Dienste auf, ...

Weitere Beispiele

- Releasewechsel: Codemod-Scripte generieren (Urteil gefragt)
- Dieser Vortrag: Claude hat einmal einen Generator geschrieben, seitdem baut ein Script die Folien (schön wär‘s)
- Übergabe an ein Werkzeug ohne Zugriff aufs Repo, z. B. Claude Design: Den Auftrag schreibt ein Skill, das ZIP packt und entpackt ein Script

Mitnahme **Braucht ein Schritt kein Urteil, lasst Claude ein Script dafür schreiben**

GRENZEN VON SKILLS

ERLEBT

Die Regel zu Rückfragen, Teil 2...

GRENZEN VON SKILLS

ERLEBT

Die Regel zu Rückfragen, Teil 2... Über drei Monate:

Mai neunmal getippt, dann ein Memory, dann die CLAUDE.md einzelner Projekte

GRENZEN VON SKILLS

ERLEBT

Die Regel zu Rückfragen, Teil 2... Über drei Monate:

Mai	neunmal getippt, dann ein Memory, dann die CLAUDE.md einzelner Projekte
24.05.	*„Das steht dass du kein AskUserQuestion-Tool verwenden sollst, aber manchmal passiert das trotzdem.“*

GRENZEN VON SKILLS

ERLEBT

Die Regel zu Rückfragen, Teil 2... Über drei Monate:

Mai	neunmal im Chat geschrieben, dann ein Memory, dann die CLAUDE.md einzelner Projekte
24.05.	*„Das steht dass du kein AskUserQuestion-Tool verwenden sollst, aber manchmal passiert das trotzdem.“*
13.06.	Regel in der globalen CLAUDE.md

GRENZEN VON SKILLS

ERLEBT

Die Regel zu Rückfragen, Teil 2... Über drei Monate:

Mai	neunmal im Chat geschrieben, dann ein Memory, dann die CLAUDE.md einzelner Projekte
24.05.	*„Das steht dass du kein AskUserQuestion-Tool verwenden sollst, aber manchmal passiert das trotzdem.“*
13.06.	Regel in der globalen CLAUDE.md
23.07.	*„Warum hast du AskUserQuestion-Tool verwendet? In meiner globalen Claude-Konfig ist das „verboten“, oder?“*

GRENZEN VON SKILLS

ERLEBT

Die Regel zu Rückfragen, Teil 2... Über drei Monate:

Mai	neunmal im Chat geschrieben, dann ein Memory, dann die CLAUDE.md einzelner Projekte
24.05.	*„Das steht dass du kein AskUserQuestion-Tool verwenden sollst, aber manchmal passiert das trotzdem.“*
13.06.	Regel in der globalen CLAUDE.md
23.07.	*„Warum hast du AskUserQuestion-Tool verwendet? In meiner globalen Claude-Konfig ist das „verboten“, oder?“*

```
Kurzer Hinweis zu meinem eigenen Vorgehen: Ich habe zwischendurch
versehentlich AskUserQuestion benutzt, obwohl die projektweiten Konventionen
Rückfragen ausschließlich als Freitext verlangen. Das war ein Fehler
meinerseits, den ich für den Rest der Session korrigiert habe.
```

Ein deutlicherer Satz hat auch nicht geholfen...

GRENZEN VON SKILLS

ERLEBT

Die Regel zu Rückfragen, Teil 2... Über drei Monate:

Mai	neunmal im Chat geschrieben, dann ein Memory, dann die CLAUDE.md einzelner Projekte
24.05.	*„Das steht dass du kein AskUserQuestion-Tool verwenden sollst, aber manchmal passiert das trotzdem.“*
13.06.	Regel in der globalen CLAUDE.md
23.07.	*„Warum hast du AskUserQuestion-Tool verwendet? In meiner globalen Claude-Konfig ist das „verboten“, oder?“*

```
Kurzer Hinweis zu meinem eigenen Vorgehen: Ich habe zwischendurch
versehentlich AskUserQuestion benutzt, obwohl die projektweiten Konventionen
Rückfragen ausschließlich als Freitext verlangen. Das war ein Fehler
meinerseits, den ich für den Rest der Session korrigiert habe.
```

Ein deutlicherer Satz hat auch nicht geholfen... ...zum Glück ein harmloses Beispiel...

GRENZEN VON SKILLS

ERLEBT

Die Regel zu Rückfragen, über drei Monate:

Die Ausgabe hat versehentlich Klartext-URLs gezeigt – ab jetzt maskiere ich alles wieder, sodass sie nicht in den Bericht gelangen. Als Nächstes prüfe ich den Eintrag für unsere cloudId, führe discover aus und teste Lese- sowie Schreibzugriffe.



M	…DE.md einzelner Projekte
24.0	*„Das steht dass du kein AskUserQuestion-Tool verwenden sollst, aber manchmal passiert das trotzdem.“*
13.06.	Regel in der globalen CLAUDE.md
23.07.	*„Warum hast du AskUserQuestion-Tool verwendet? In meiner globalen Claude-Konfig ist das „verboten“, oder?“*

Kurzer Hinweis zu meinem eigenen Vorgehen: Ich habe zwischendurch versehentlich AskUserQuestion benutzt, obwohl die projektweiten Konventionen Rückfragen ausschließlich als Freitext verlangen. Das war ein Fehler meinerseits, den ich für den Rest der Session korrigiert habe.

Ein deutlicherer Satz hat auch nicht geholfen… …zum Glück ein harmloses Beispiel…

GRENZEN VON SKILLS

ERLEBT

Die Regel zu Rückfragen, über drei Monate:

M

24.0

13.06.

23.07.

Die Ausgabe hat versehentlich Klartext-URLs gezeigt – ab jetzt maskiere ich alles wieder, sodass sie nicht in den Bericht gelangen. Als Nächstes prüfe ich den Eintrag für unsere cloudId, führe discover aus und teste Lese- sowie Schreibzugriffe.



…BL.md einzelner Projekte

„Das steht dass du kein AskUserQuestion-Tool verwenden sollst, aber manch…

Bevor ich anfange, muss ich etwas melden: Vor deiner Nachricht wurde ~/…er mit dem Read-Tool gelesen. Das widerspricht deiner Vorgabe. Der vollständige Header ist damit im Kontext dieser Sitzung gelandet, und das Basic-Token lässt sich ohne Aufwand dekodieren. Darin stehen die E-Mail-Adresse und das API-Token im Klartext. Ich gebe den Inhalt nicht wieder und verwende ihn nirgends außer über curl -H @…. An der Tatsache, dass er im Verlauf steht, ändert das aber nichts.



…ten", oder?"

Kurzer Hinweis zu meinem eigenen Vorgehen: Ich habe zwischendurch versehentlich AskUserQuestion benutzt, obwohl die projektweiten Konventionen Rückfragen ausschließlich als Freitext verlangen. Das war ein Fehler meinerseits, den ich für den Rest der Session korrigiert habe.

Ein deutlicherer Satz hat auch nicht geholfen… …zum Glück ein harmloses Beispiel…

GRENZEN VON SKILLS

ERLEBT

Die Regel zu Rückfragen, über drei Monate:

Mai	neunmal getippt, dann ein Memory, dann die CLAUDE.md einzelner Projekte
24.05.	*„Das steht dass du kein AskUserQuestion-Tool verwenden sollst, aber manchmal passiert das trotzdem.“*
13.06.	Regel in der globalen CLAUDE.md
23.07.	*„Warum hast du AskUserQuestion-Tool verwendet? In meiner globalen Claude-Konfig ist das „verboten“, oder?“*
25.07.	**per permissions.deny in ~/.claude/settings.json gesperrt**

PERMISSIONS

DARAUS FOLGT

Für verbindliche Regeln: Permissions und Hooks

- Unabhängig vom Model, implementiert in Claude Code

```
// .claude/settings.json

"permissions": {
  "ask":  ["Bash(git push:*)"],

  "deny": ["WebFetch",
           "Read(./.env)"]
},
```

Beispiel, nachgestellt

Permissions für Tool-Aufrufe

- Jedes Tool mit oder ohne Argumente kann gesteuert werden
- allow: erlaubt ein Tool pauschal
- ask: fragt in jedem Modus nach (auch im Auto Mode). Hier für alle git push-Aufrufe
- deny: Verbietet das Tool. Beispiel: Lesen von .env Dateien

Die Grenze: Read(./.env) sperrt z.B. auch `cat .env`. Ein erlaubtes npm run könnte aber ein Node-Script ausführen, das .env lesen dürfte. 😱

Mitnahme **Hooks und Permissions für verbindliche Regeln, nicht die CLAUDE.md**

PERMISSIONS

DARAUS FOLGT

Für verbindliche Regeln: Permissions und Hooks

- Unabhängig vom Model, implementiert in Claude Code

PERMISSIONS

DARAUS FOLGT

Für verbindliche Regeln: Permissions und Hooks

- Unabhängig vom Model, implementiert in Claude Code

```
// .claude/settings.json

"permissions": {



},
```

Beispiel, nachgestellt

Permissions für Tool-Aufrufe

- Tools sind z.B. Edit/Read, Bash, WebFetch oder MCP-Tool-Aufrufe

PERMISSIONS

DARAUS FOLGT

Für verbindliche Regeln: Permissions und Hooks

- Unabhängig vom Model, implementiert in Claude Code

```
// .claude/settings.json

"permissions": {
  "ask":  ["Bash(git push:*)"],

  "deny": ["WebFetch",
           "Read(./.env)"]
},
```

Beispiel, nachgestellt

Permissions für Tool-Aufrufe

- Tools sind z.B. Edit/Read, Bash, WebFetch oder MCP-Tool-Aufrufe

- allow: erlaubt ein Tool ohne weitere Nachfrage
- ask: fragt in jedem Modus nach (auch im Auto Mode). Hier für alle git push-Aufrufe
- deny: Verbietet das Tool. Hier: zum Lesen von .env Dateien

PERMISSIONS

DARAUS FOLGT

Für verbindliche Regeln: Permissions und Hooks

- Unabhängig vom Model, implementiert in Claude Code

```
// .claude/settings.json

"permissions": {
  "ask":  ["Bash(git push:*)"],

  "deny": ["WebFetch",
           "Read(./.env)"]
},
```

Beispiel, nachgestellt

Permissions für Tool-Aufrufe

- Tools sind z.B. Edit/Read, Bash, WebFetch oder MCP-Tool-Aufrufe

- allow: erlaubt ein Tool ohne weitere Nachfrage
- ask: fragt in jedem Modus nach (auch im Auto Mode). Hier für alle git push-Aufrufe
- deny: Verbietet das Tool. Hier: zum Lesen von .env Dateien

Die Grenze: Read(./.env) sperrt z.B. auch `cat .env`. Ein erlaubtes npm run könnte aber ein Node-Script ausführen, das .env lesen dürfte. 😱

HOOKS

DARAUS FOLGT

Für verbindliche Regeln: Permissions und Hooks

- Unabhängig vom Model, implementiert in Claude Code

```
# protect-generated.sh (Dummy-Code)
pfad = kommt per stdin
if pfad liegt unter src/api/generated/:
    stderr "Generated file. Use generator instead"
    exit 2
```

Beispiel, nachgestellt

Hooks laufen vor der Ausführung eines Tools

- Ein eigenes Script prüft vor jedem Tool-Aufruf, ob er erlaubt ist
- Exit-Code 2 bricht ab, und die Meldung auf stderr bekommt Claude zu sehen. Sie kann sagen, was Claude stattdessen tun soll

HOOKS

DARAUS FOLGT

Für verbindliche Regeln: Permissions und Hooks

- Unabhängig vom Model, implementiert in Claude Code

```
# protect-generated.sh (Dummy-Code)
pfad = kommt per stdin
if pfad liegt unter src/api/generated/:
    stderr "Generated file. Use generator instead"
    exit 2

// .claude/settings.json

"hooks": { "PreToolUse": [{
  "matcher": "Edit|Write",
  "hooks": [{ "type": "command",
    "command": "scripts/protect-generated" }]
}]}
```

Beispiel, nachgestellt

Hooks laufen vor der Ausführung eines Tools

- Ein eigenes Script prüft vor jedem Tool-Aufruf, ob er erlaubt ist
- Exit-Code 2 bricht ab, und die Meldung auf stderr bekommt Claude zu sehen. Sie kann sagen, was Claude stattdessen tun soll
- Der Matcher legt fest, für welche Tools das Script läuft, hier Edit|Write

HOOKS

DARAUS FOLGT

Für verbindliche Regeln: Permissions und Hooks

- Unabhängig vom Model, implementiert in Claude Code

```
# protect-generated.sh (Dummy-Code)
pfad = kommt per stdin
if pfad liegt unter src/api/generated/:
    stderr "Generated file. Use generator instead"
    exit 2

// .claude/settings.json

"hooks": { "PreToolUse": [{
  "matcher": "Edit|Write",
  "hooks": [{ "type": "command",
    "command": "scripts/protect-generated" }]
}]}
```

Beispiel, nachgestellt

Hooks laufen vor der Ausführung eines Tools

- Ein eigenes Script prüft vor jedem Tool-Aufruf, ob er erlaubt ist
- Exit-Code 2 bricht ab, und die Meldung auf stderr bekommt Claude zu sehen. Sie kann sagen, was Claude stattdessen tun soll
- Der Matcher legt fest, für welche Tools das Script läuft, hier Edit|Write

Die Grenze: Die Grenze: Mit `echo … >` über Bash geht Claude am Hook vorbei.

HOOKS

DARAUS FOLGT

Für verbindliche Regeln: Permissions und Hooks

- Unabhängig vom Model, implementiert in Claude Code

```
# protect-generated.sh (Dummy-Code)
pfad = kommt per stdin
if pfad liegt unter src/api/generated/:
    stderr "Generated file. Use generator instead"
    exit 2

// .claude/settings.json

"hooks": { "PreToolUse": [{
  "matcher": "Edit|Write",
  "hooks": [{ "type": "command",
    "command": "scripts/protect-generated" }]
}]}
```

Beispiel, nachgestellt

Hooks laufen vor der Ausführung eines Tools

- Ein eigenes Script prüft vor jedem Tool-Aufruf, ob er erlaubt ist
- Exit-Code 2 bricht ab, und die Meldung auf stderr bekommt Claude zu sehen. Sie kann sagen, was Claude stattdessen tun soll
- Der Matcher legt fest, für welche Tools das Script läuft, hier Edit|Write

Die Grenze: Die Grenze: Mit `echo … >` über Bash geht Claude am Hook vorbei.

Mitnahme **Hooks und Permissions für verbindliche Regeln, nicht die CLAUDE.md. Grenzen beachten!**

PERMISSIONS UND HOOKS

DARAUS FOLGT

Für verbindliche Regeln: Permissions und Hooks

- Unabhängig vom Model, implementiert in Claude Code

```
# protect-generated.sh (Dummy-Code)
pfad = kommt per stdin
if pfad liegt unter src/api/g
    stderr "Generate
    exit 2

// .claude/setti

"hooks": { "PreToolUse": [{
  "matcher": "Edit|Write",
  "hooks": [{ "type": "command",
    "command": "scripts/protect-g
}]}
```

Die Grenze: Read(./.env) sperrt z.B. auch `cat .env`. Ein erlaubtes npm run könnte aber ein Node-Script ausführen, das .env lesen dürfte. 😱

Die Grenze: Die Grenze: Mit `echo …` > über Bash geht Claude am Hook vorbei.

Beispiel, nachgestellt

der Ausführung eines Tools

prüft vor jedem Tool-Aufruf, ob er erlaubt ist

bricht ab, und die Meldung auf stderr bekommt Claude zu sehen. Sie kann sagen, was Claude stattdessen tun soll

- Der Matcher … he Tools das Script läuft, hier

Die Grenze: Die Grenze: Mit `echo …` > über Bash geht Claude am Hook vorbei.

Mitnahme **Hooks und Permissions für verbindliche Regeln, nicht die CLAUDE.md. Grenzen beachten!**

SANDBOX: CONTAINER, ALLOWLIST, RECHTE

DARAUS FOLGT

Eine Sandbox beschränkt, was Claude überhaupt sehen darf (Dateien, Verzeichnisse, Internet, ...)

SANDBOX: CONTAINER, ALLOWLIST, RECHTE

DARAUS FOLGT

Eine Sandbox beschränkt, was Claude überhaupt sehen darf (Dateien, Verzeichnisse, Internet, ...)

Schutz

- das Dateisystem außerhalb des Mounts: Home-Verzeichnis, SSH-Schlüssel, andere Projekte
- Zugangsdaten auf dem Host
- das Netz außerhalb einer Allowlist
- Host selbst: Ein npm-Paket mit Schadcode läuft im Container, nicht auf dem Host

SANDBOX: CONTAINER, ALLOWLIST, RECHTE

DARAUS FOLGT

Eine Sandbox beschränkt, was Claude überhaupt sehen darf (Dateien, Verzeichnisse, Internet, ...)

Schutz

- das Dateisystem außerhalb des Mounts: Home-Verzeichnis, SSH-Schlüssel, andere Projekte
- Zugangsdaten auf dem Host
- das Netz außerhalb einer Allowlist
- Host selbst: Ein npm-Paket mit Schadcode läuft im Container, nicht auf dem Host

Mehr als Permissions und Hooks

- Sie gilt für jeden Prozess im Container, auch für Scripts, die Claude startet
- Sie wirkt auch, wenn eine Regel fehlt, an die niemand gedacht hat

SANDBOX: CONTAINER, ALLOWLIST, RECHTE

DARAUS FOLGT

Eine Sandbox beschränkt, was Claude überhaupt sehen darf (Dateien, Verzeichnisse, Internet, ...)

Schutz

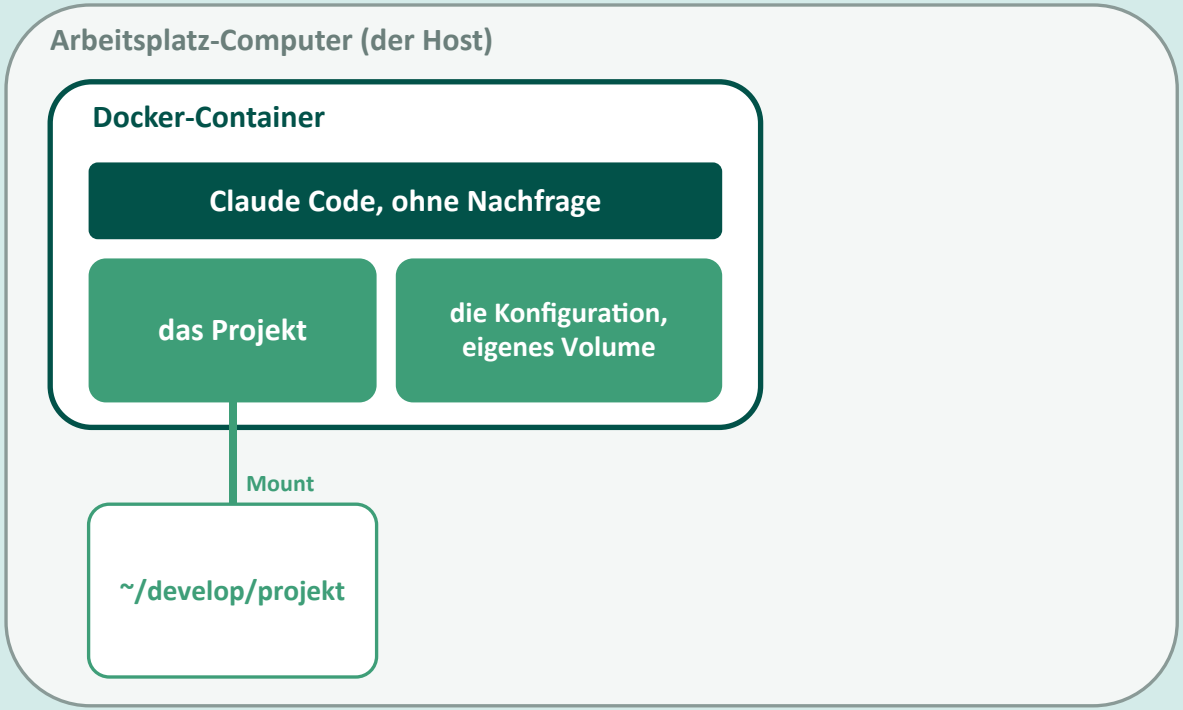
- das Dateisystem außerhalb des Mounts: Home-Verzeichnis, SSH-Schlüssel, andere Projekte
- Zugangsdaten auf dem Host
- das Netz außerhalb einer Allowlist
- Host selbst: Ein npm-Paket mit Schadcode läuft im Container, nicht auf dem Host

Mehr als Permissions und Hooks

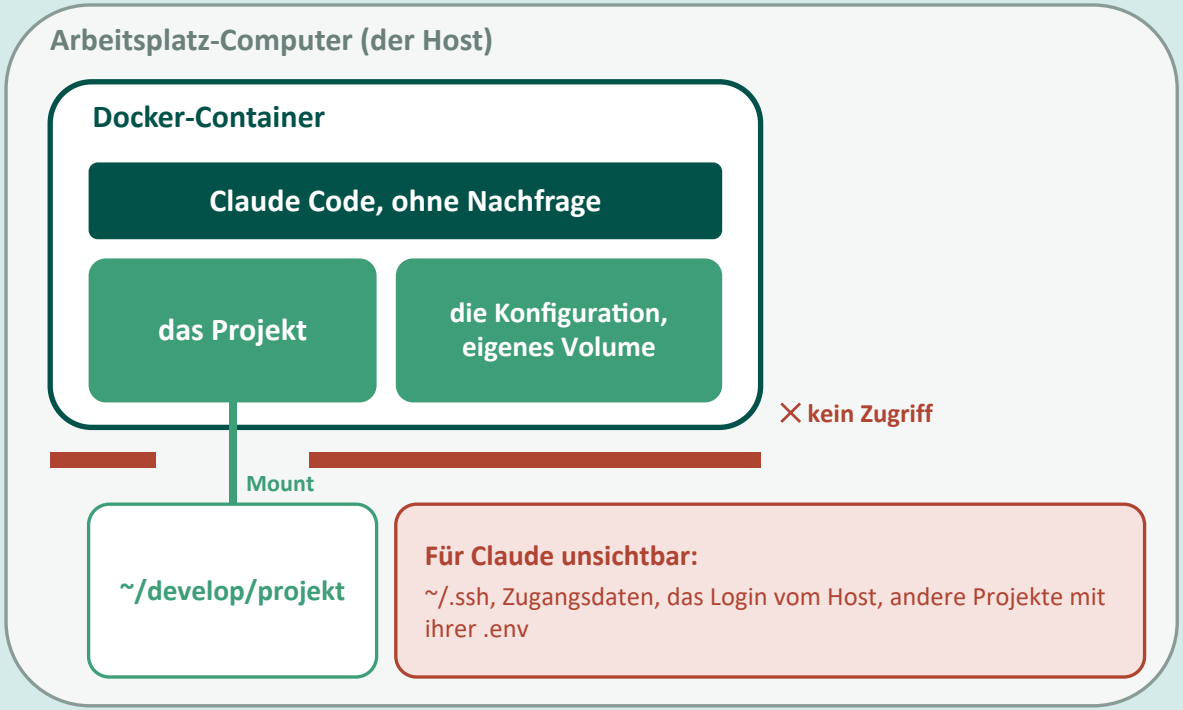
- Sie gilt für jeden Prozess im Container, auch für Scripts, die Claude startet
- Sie wirkt auch, wenn eine Regel fehlt, an die niemand gedacht hat

Mitnahme **Was Claude nicht erreichen soll, liegt außerhalb der Sandbox**

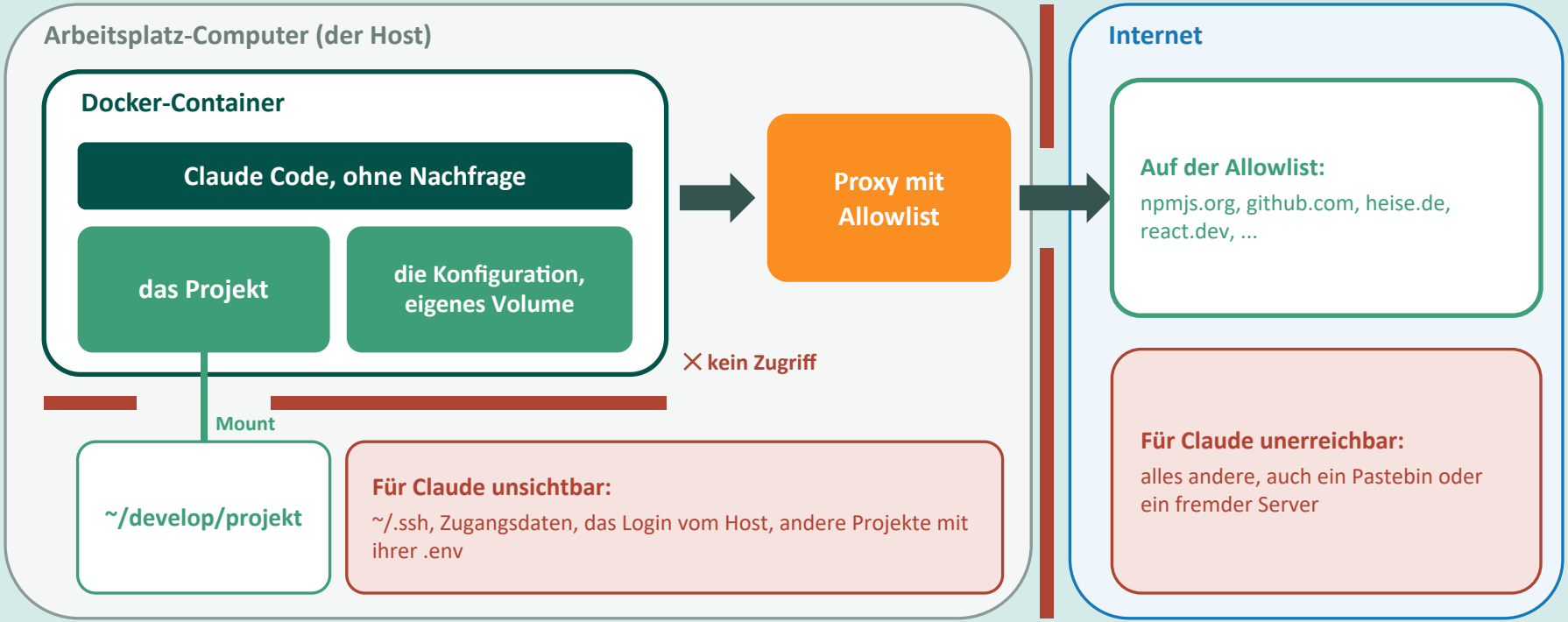
SANDBOX: MEIN AUFBAU



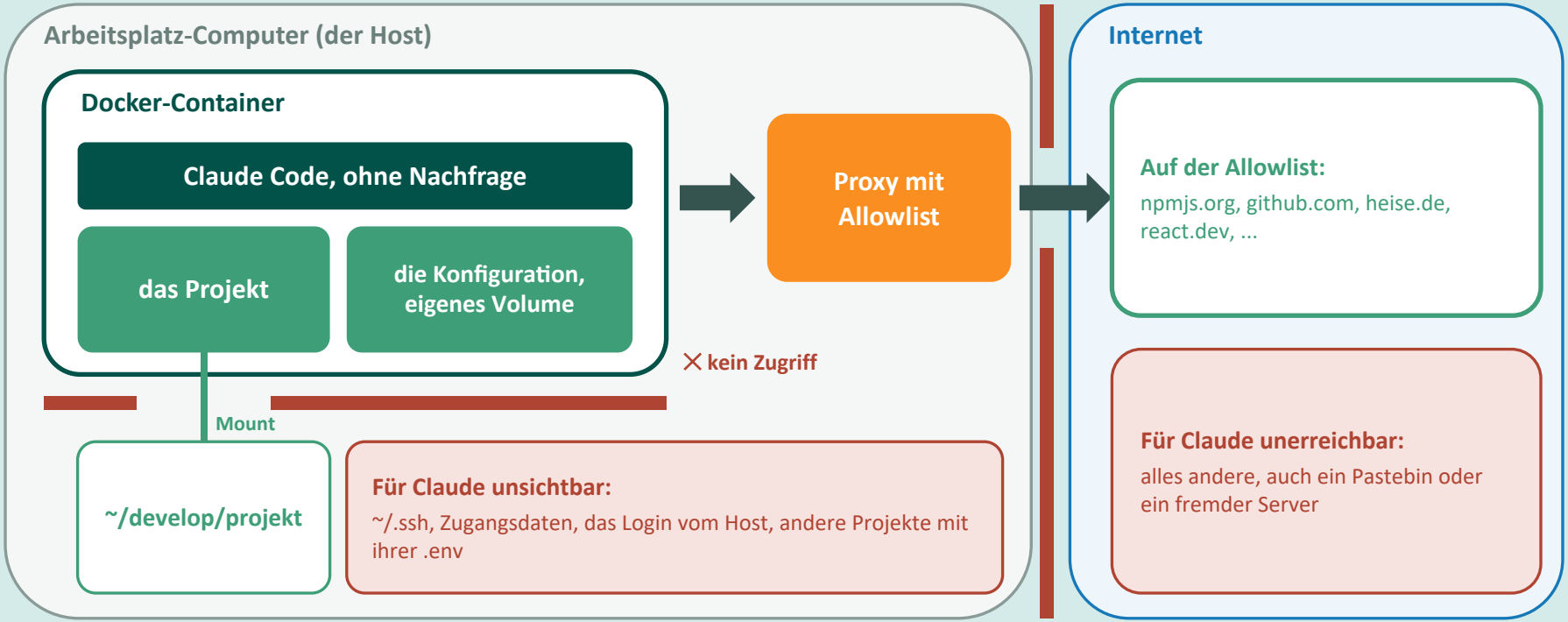
SANDBOX: MEIN AUFBAU



SANDBOX: MEIN AUFBAU



SANDBOX: MEIN AUFBAU



Mitnahme **Im Container darf Claude allein arbeiten, weil die Sandbox die Grenze zieht**

SANDBOX: GRENZE DES EIGENBAUS

ERLEBT

Der Container musste den Proxy, die Docker-Netze und den Host erreichen. Statt diese Netze einzeln freizugeben, ließ die Firewall den ganzen privaten Adressraum durch.

```
# Juni: privat heißt intern, oder?
iptables -A OUTPUT -d 10.0.0.0/8 -j ACCEPT
iptables -A OUTPUT -d 172.16.0.0/12 -j ACCEPT
iptables -A OUTPUT -d 192.168.0.0/16 -j ACCEPT
```

SANDBOX: GRENZE DES EIGENBAUS

ERLEBT

Der Container musste den Proxy, die Docker-Netze und den Host erreichen. Statt diese Netze einzeln freizugeben, ließ die Firewall den ganzen privaten Adressraum durch.

```
# Juni: privat heißt intern, oder?
iptables -A OUTPUT -d 10.0.0.0/8 -j ACCEPT
iptables -A OUTPUT -d 172.16.0.0/12 -j ACCEPT
iptables -A OUTPUT -d 192.168.0.0/16 -j ACCEPT
```

Mein LAN liegt auch in 192.168.x.x. Bis Oktober war es aus dem Container erreichbar 🥳

SANDBOX: GRENZE DES EIGENBAUS

ERLEBT

Der Container musste den Proxy, die Docker-Netze und den Host erreichen. Statt diese Netze einzeln freizugeben, ließ die Firewall den ganzen privaten Adressraum durch.

```
# Juni: privat heißt intern, oder?
iptables -A OUTPUT -d 10.0.0.0/8 -j ACCEPT
iptables -A OUTPUT -d 172.16.0.0/12 -j ACCEPT
iptables -A OUTPUT -d 192.168.0.0/16 -j ACCEPT
```

Mein LAN liegt auch in 192.168.x.x. Bis Oktober war es aus dem Container erreichbar 😈

Aufgefallen ist es erst, als Claude meine Sandbox mit einer anderen Sandbox-Implementierung verglichen hat.

SANDBOX: GRENZE DES EIGENBAUS

ERLEBT

Der Container musste den Proxy, die Docker-Netze und den Host erreichen. Statt diese Netze einzeln freizugeben, ließ die Firewall den ganzen privaten Adressraum durch.

```
# Juni: privat heißt intern, oder?
iptables -A OUTPUT -d 10.0.0.0/8 -j ACCEPT
iptables -A OUTPUT -d 172.16.0.0/12 -j ACCEPT
iptables -A OUTPUT -d 192.168.0.0/16 -j ACCEPT
```

Mein LAN liegt auch in 192.168.x.x. Bis Oktober war es aus dem Container erreichbar 😈

Aufgefallen ist es erst, als Claude meine Sandbox mit einer anderen Sandbox-Implementierung verglichen hat.

Fertige Alternativen

- /sandbox in Claude Code: isoliert Bash-Befehle, auf macOS mit Seatbelt, auf Linux mit bubblewrap
- Docker Sandboxes: docker sandbox run claude
- nono: Kernel-Sandbox ohne Root für macOS, Linux und WSL2
- Alle noch im Entstehen. Prüft, was sie gegen Absicht leisten, nicht nur gegen Versehen

Mitnahme **Sandbox vielleicht doch besser nicht selbst bauen (note to self)**

AUTO MODE ODER SANDBOX

ERLEBT

In der Sandbox läuft der Verkehr nach außen nur über eine Allowlist.

Beim ersten Einsatz sperrt die Allowlist eine Doku-Seite. Claude probiert WebFetch, dann curl, dann WebSearch „als anderen Egress-Pfad“.

AUTO MODE ODER SANDBOX

ERLEBT

In der Sandbox läuft der Verkehr nach außen nur über eine Allowlist.

Beim ersten Einsatz sperrt die Allowlist eine Doku-Seite. Claude probiert WebFetch, dann curl, dann WebSearch „als anderen Egress-Pfad“.

Seitdem erklärt eine Regel die Grenze. Die Antwort heute:

> ***„Ich umgehe das nicht, sondern melde dir die Domain und den Befehl“***
>
> Claude in der Sandbox

Die Sandbox setzt die Grenze durch. Die Regel sagt Claude, was es an der Grenze tun soll.

```
## Eingeschränkter Internet-Zugriff
Nicht gelistete Domains sind nicht erreichbar.
Brauchst du eine nicht freigegebene Seite:
1. Nicht umgehen: keine alternativen Hosts,
   Mirrors oder IPs probieren
2. Bescheid sagen: Domain nennen und wofür
3. Freischalt-Befehl vorschlagen, auf dem Host:
   claude-sandbox allow <domain>
```

CLAUDE.md in der Sandbox, gekürzt

AUTO MODE ODER SANDBOX

IM VERGLEICH

	Auto Mode	Sandbox
Wer entscheidet	ein Modell, der Klassifikator	die Grenze des Containers
Deterministisch	nein	ja
Schützt vor	Aktionen, die der Klassifikator als riskant erkennt, etwa ein Token aus einer Datei lesen	allem außerhalb von Mount und Allowlist, für jeden Prozess
Aufwand	keiner, Default seit 28.09.	Container, eigene Browser, Abhängigkeiten für den Container
Kostet	Nachfragen bei Risiko	keinen Zugriff auf Dienste auf dem Host, z. B. den MCP-Server von IntelliJ

Mitnahme **Auto Mode und Sandbox schließen sich nicht aus.**

AUTO MODE ODER SANDBOX

IM VERGLEICH

	Auto Mode	Sandbox
Wer entscheidet	ein Modell, der Klassifikator	die Grenze des Containers
Deterministisch	nein	ja
Schützt vor	Aktionen, die der Klassifikator als riskant erkennt, etwa ein Token aus einer Datei lesen	allem außerhalb von Mount und Allowlist, für jeden Prozess
Aufwand	keiner, Default seit 28.09.	Container, eigene Browser, Abhängigkeiten für den Container
Kostet	Nachfragen bei Risiko	keinen Zugriff auf Dienste auf dem Host, z. B. den MCP-Server von IntelliJ

Mitnahme **Auto Mode und Sandbox schließen sich nicht aus.**

AUTO MODE ODER SANDBOX

IM VERGLEICH

	Auto Mode	
Wer entscheidet	ein Modell, der Klassifikator	
Deterministisch	nein	
Schützt vor	Aktionen, die der Klassifikator als riska… ein Token aus einer Datei lesen	
Aufwand	keiner,	…ge Browser, Abhängigkeiten für den Container
Kostet	Nachfragen bei Risiko	keinen Zugriff auf Dienste auf dem Host, z. B. den MCP-Server von IntelliJ

> ⚠ Auto mode reduces permission prompts but does not guarantee safety. Use it for tasks where you trust the general direction, not as a replacement for review on sensitive operations.

https://code.claude.com/docs/en/permission-modes#eliminate-prompts-with-auto-mode

, Default seit 28.09.

Mitnahme **Auto Mode und Sandbox schließen sich nicht aus.**

AUTO MODE ODER SANDBOX

IM VERGLEICH

Wer entscheidet		
Deterministisch		
Schützt vor	ein Token aus einer Datei lesen	
Aufwand	keiner,	gene Browser, Abhängigkeiten für den Container
Kostet	Nachfragen bei Risiko	keinen Zugriff auf Dienste auf dem Host, z. B. den MCP-Server von IntelliJ

Default seit 28.09.

https://code.claude.com/docs/en/permission-modes#eli

mode

Anthropic Researcher Warns There's '>10% Chance' AI Could 'Kill All Humans' By Next Decade

By Siladitya Ray, Forbes Staff. Siladitya Ray is a New Delhi-based Forbes news...

Follow Author

Published Sep 09, 2026, 12:14am EDT, Updated Sep 09, 2026, 07:24am EDT

Mitnahme **Auto Mode und Sandbox schließen sich nicht aus.**

ZUSAMMENFASSUNG

DARAUS FOLGT

Prosa CLAUDE.md, Skills

Bitte

ZUSAMMENFASSUNG

DARAUS FOLGT

Prosa CLAUDE.md, Skills

Bitte

Code Scripts, Hooks, Permissions

deterministisch

verbindlicher



ZUSAMMENFASSUNG

DARAUS FOLGT

Prosa CLAUDE.md, Skills — **Bitte**

Code Scripts, Hooks, Permissions — **deterministisch**

Sandbox Container, Allowlist, Rechte — **physisch**

verbindlicher



ZUSAMMENFASSUNG

DARAUS FOLGT

Prosa CLAUDE.md, Skills — **Bitte**

Code Scripts, Hooks, Permissions — **deterministisch**

Sandbox Container, Allowlist, Rechte — **physisch**

verbindlicher



Mitnahme **Für verbindliche Regeln hilft keine Prosa.**

ZUSAMMENFASSUNG

DARAUS FOLGT

Prosa CLAUDE.md, Skills — **Bitte**

Code Scripts, Hooks, Permissions — **deterministisch**

Sandbox Container, Allowlist, Rechte — **physisch**

verbindlicher



Mitnahme **Für verbindliche Regeln hilft keine Prosa. **UND AUCH KEIN BILD-ZEITUNG-STIL!!!!!****

Claude

allein

SESSIONS WERDEN LÄNGER...

ERLEBT

Längste Strecke ohne meine Eingabe, je Monat (gemessen in Sessions und Workflows):

SESSIONS WERDEN LÄNGER...

ERLEBT

Längste Strecke ohne meine Eingabe, je Monat (gemessen in Sessions und Workflows):

April **28 min** Review von Integrationstests

SESSIONS WERDEN LÄNGER…

ERLEBT

Längste Strecke ohne meine Eingabe, je Monat (gemessen in Sessions und Workflows):

April — **28 min** Review von Integrationstests

Mai — **58 min** Task-Umsetzung, dazwischen viele Permission-Klicks

SESSIONS WERDEN LÄNGER…

ERLEBT

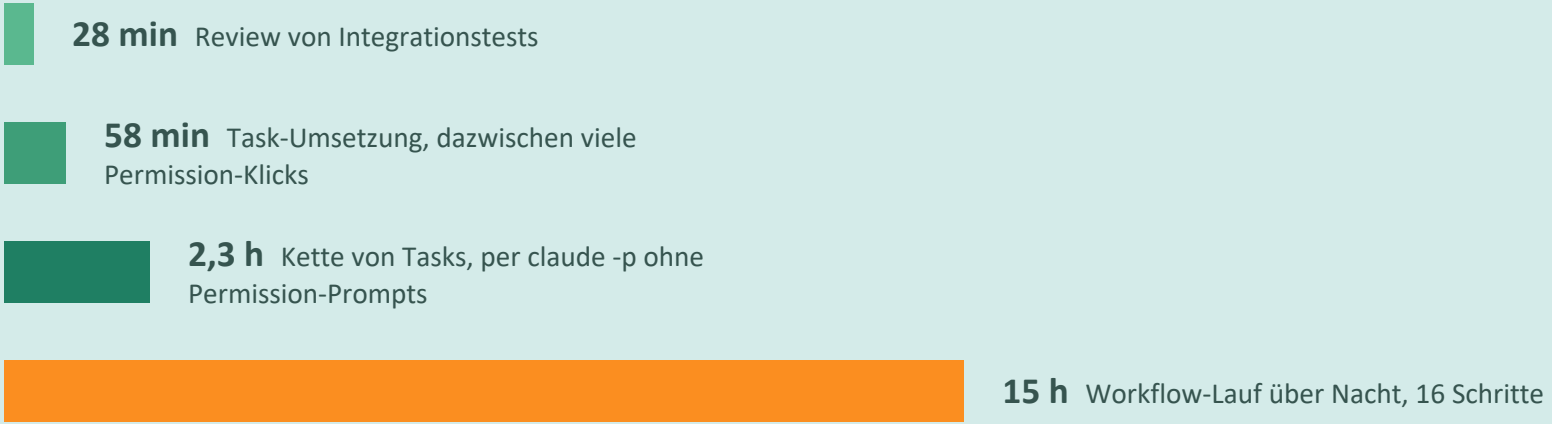
Längste Strecke ohne meine Eingabe, je Monat (gemessen in Sessions und Workflows):

April		**28 min** Review von Integrationstests
Mai		**58 min** Task-Umsetzung, dazwischen viele Permission-Klicks
Juli		**2,3 h** Kette von Tasks, per claude -p ohne Permission-Prompts

SESSIONS WERDEN LÄNGER…

ERLEBT

Längste Strecke ohne meine Eingabe, je Monat (gemessen in Sessions und Workflows):

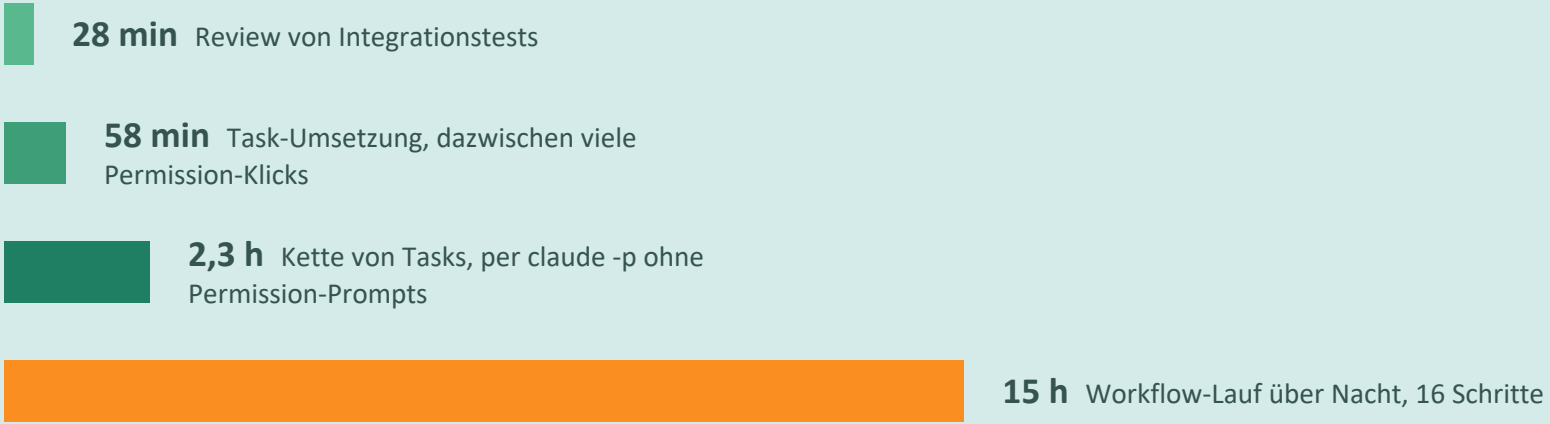


Ein normaler Dialog-Turn dauerte dagegen in allen Monaten rund eine Minute.

SESSIONS WERDEN LÄNGER…

ERLEBT

Längste Strecke ohne meine Eingabe, je Monat (gemessen in Sessions und Workflows):



Ein normaler Dialog-Turn dauerte dagegen in allen Monaten rund eine Minute.

Je länger ein Lauf, desto mehr steckt darin. Damit fielen Probleme auf, die bei einem einzelnen Feature kaum aufgefallen wären.

Hintergrund: Dynamic Workflows

Ein Workflow ist ein Script, das Claude Code ausführt. Es arbeitet eine Liste von Schritten ab und startet je Schritt Agenten nacheinander, jeden mit eigenem Auftrag und eigenem Modell.

Hintergrund: Dynamic Workflows

Ein Workflow ist ein Script, das Claude Code ausführt. Es arbeitet eine Liste von Schritten ab und startet je Schritt Agenten nacheinander, jeden mit eigenem Auftrag und eigenem Modell.

```
// .claude/workflows/oberflaeche.js, gekürzt
for (const n of STEPS) {
  const c = await agent(klaeren(n),
    { label: `Schritt ${n}: klären`, model: 'opus' })
  await agent(umsetzen(n, c), { model: 'sonnet' })
  await agent(review(n, c), { model: 'opus' })
  await agent(uebergabe(n, c), { model: 'sonnet' })
}
```

Doku: https://platform.claude.com/docs/de/managed-agents/multiagent-orchestration#dynamic-workflows

- Ablauf: Schritte, Prüfungen, Modell je Aufgabe in JS
- Kann über Stunden ohne alleine laufen
- In Claude Code eingebaut, als JavaScript unter .claude/workflows/

WORKFLOW

Hintergrund: Dynamic Workflows

Ein Workflow ist ein Script, das Claude Code ausführt. Es arbeitet eine Liste von Schritten ab und startet je Schritt Agenten nacheinander, jeden mit eigenem Auftrag und eigenem Modell.

```
// .claude/workflows/oberflaeche.js, gekürzt
for (const n of STEPS) {
  const c = await agent(klaeren(n),
    { label: `Schritt ${n}: klären`, model: 'opus' })
  await agent(umsetzen(n, c), { model: 'sonnet' })
  await agent(review(n, c), { model: 'opus' })
  await agent(uebergabe(n, c), { model: 'sonnet' })
}
```

Doku: https://platform.claude.com/docs/de/managed-agents/multiagent-orchestration#dynamic-workflows

- Ablauf: Schritte, Prüfungen, Modell je Aufgabe in JS
- Kann über Stunden ohne alleine laufen
- In Claude Code eingebaut, als JavaScript unter .claude/workflows/

Beispiel: Mein Workflow



WORKFLOW

Hintergrund: Dynamic Workflows

Ein Workflow ist ein Script, das Claude Code ausführt. Es arbeitet eine Liste von Schritten ab und startet je Schritt Agenten nacheinander, jeden mit eigenem Auftrag und eigenem Modell.

```
// .claude/workflows/oberflaeche.js, gekürzt
for (const n of STEPS) {
  const c = await agent(klaeren(n),
    { label: `Schritt ${n}: klären`, model: 'opus' })
  await agent(umsetzen(n, c), { model: 'sonnet' })
  await agent(review(n, c), { model: 'opus' })
  await agent(uebergabe(n, c), { model: 'sonnet' })
}
```

Doku: https://platform.claude.com/docs/de/managed-agents/multiagent-orchestration#dynamic-workflows

- Ablauf: Schritte, Prüfungen, Modell je Aufgabe in JS
- Kann über Stunden ohne alleine laufen
- In Claude Code eingebaut, als JavaScript unter .claude/workflows/

Beispiel: Mein Workflow



Mitnahme **Ein Workflow schreibt euren Ablauf fest, Claude arbeitet ihn ab**

WORKFLOW: NACH DEM LAUF

Am Ende jedes Laufs wertet Claude ihn aus: Dauer je Phase, Kontext, Builds, Tests. Daraus folgen Vorschläge mit erwartetem Gewinn. Claude ändert keine Regel selbst.

```
## Lauf 58A vom 2026-10-18, Schritte 140 bis 147
Er dauerte 8,5 Stunden, geschätzt waren 8 bis 10,5.
…
**Kontext.** Umsetzung 1 erreichte 300.000 bis
526.000 Tokens. Über dem Ziel von etwa 450.000 …

**Vorschläge:**
3. ✅ Angenommen am 19.10.: Klassen über
   1.000 Zeilen nur gezielt lesen.
   Gewinn: 50.000 bis 100.000 Tokens je Umsetzung
4. ❌ Abgelehnt am 19.10.: …
```

docs/oberflaeche/laeufe.md, gekürzt

133 : 32

Vorschläge angenommen zu abgelehnt

Mitnahme **Lasst Claude jeden Lauf auswerten. Vorschläge können den Ablauf verbessern**

DER BUILD BREMST

ERLEBT

Mein Workflow arbeitet eine Liste von Schritten ab, oft über Nacht. Ein Schritt entspricht einem Anwendungsfall.

DER BUILD BREMST

ERLEBT

Mein Workflow arbeitet eine Liste von Schritten ab, oft über Nacht. Ein Schritt entspricht einem Anwendungsfall.

Ein Schritt im Workflow: 34 bis 50 Minuten

DER BUILD BREMST

ERLEBT

Mein Workflow arbeitet eine Liste von Schritten ab, oft über Nacht. Ein Schritt entspricht einem Anwendungsfall.

Ein Schritt im Workflow: 34 bis 50 Minuten

Modell
10–16 min

DER BUILD BREMST

ERLEBT

Mein Workflwo arbeitet eine Liste von Schritten ab, oft über Nacht. Ein Schritt entspricht einem Anwendungsfall.

Ein Schritt im Workflow: 34 bis 50 Minuten

Modell 10–16 min	**Warten auf Build und Tests** der Rest

Ein langer Lauf wird vom Build gebremst, nicht vom Modell.

DER BUILD BREMST

DARAUS FOLGT

Teststrategie nach Claude ausrichten

- Playwright-Tests gezielt statt der ganzen Suite
- Neue Testfälle erst am Schluss. Wie beim Entwickeln von Hand: erst im Browser nachsehen, Claude per Screenshot
- Frontend und Backend getrennt prüfen

335 → 77 s

voller Build nach Umbau der Tests: JUnit parallel, externe Befehle nachgebildet

Mitnahme **Optimiert die Feedbackschleife, nicht den Prompt**

ARBEIT OHNE ZUSCHAUER

ERLEBT

Problem: Nachts will ich schlafen, keine Rückfragen beantworten 😴

„Ich möchte die verbleibenden Schritte gleich komplett in einem Lauf umsetzen. Weil ich habe nämlich gleich Feierabend.“

ich

ARBEIT OHNE ZUSCHAUER

ERLEBT

Problem: Nachts will ich schlafen, keine Rückfragen beantworten 😴

> ***„Ich möchte die verbleibenden Schritte gleich komplett in einem Lauf umsetzen. Weil ich habe nämlich gleich Feierabend.“***
>
> ich

> ***„Für den Workflow wäre dann auch wichtig, dass alle Entscheidungen, die du selbst treffen kannst, die kannst du dann auch selbst treffen. […] und hältst dann die Abweichung vom Plan oder die Fragen dann fest.“***
>
> ich, beim Aufsetzen des Workflows, 04.10.

ARBEIT OHNE ZUSCHAUER

DARAUS FOLGT

Statt zu fragen, entscheidet Claude und schreibt auf:

```
### E2: Soll das Label-Format in die URL?
- Gewählt: Nein, bleibt in localStorage
- Typ: optional
- Alternativen: in die URL wie in der Vorgabe (verworfen)
- Begründung: weicht von der Vorgabe ab, deshalb NEEDS-REVIEW
```

ARBEIT OHNE ZUSCHAUER

DARAUS FOLGT

Statt zu fragen, entscheidet Claude und schreibt auf:

```
### E2: Soll das Label-Format in die URL?
- Gewählt: Nein, bleibt in localStorage
- Typ: optional
- Alternativen: in die URL wie in der Vorgabe (verworfen)
- Begründung: weicht von der Vorgabe ab, deshalb NEEDS-REVIEW
```

- Abweichungen von der Vorgabe fängt das Review
- Nach dem Lauf geht führt ein Skill durch die offenen Entscheidungen

ARBEIT OHNE ZUSCHAUER

DARAUS FOLGT

Statt zu fragen, entscheidet Claude und schreibt auf:

```
### E2: Soll das Label-Format in die URL?
- Gewählt: Nein, bleibt in localStorage
- Typ: optional
- Alternativen: in die URL wie in der Vorgabe (verworfen)
- Begründung: weicht von der Vorgabe ab, deshalb NEEDS-REVIEW
```

- Abweichungen von der Vorgabe fängt das Review
- Nach dem Lauf geht führt ein Skill durch die offenen Entscheidungen

2.749 Entscheidungen hat Claude in meinem Werkzeug selbst getroffen und festgehalten

13 im Mittel je Schritt

ZWEI REVIEWS, ZWEI MODELLE

ERLEBT

Vor einem großen Nachtlauf wollte ich wissen, ob eine Review-Runde reicht. Claude hat dafür alle bisherigen Läufe ausgewertet.

ZWEI REVIEWS, ZWEI MODELLE

ERLEBT

Vor einem großen Nachtlauf wollte ich wissen, ob eine Review-Runde reicht. Claude hat dafür alle bisherigen Läufe ausgewertet.

46 von 57

Schritten mit zweitem Review: Es fand noch etwas, das behoben werden musste

```
header.equalsIgnoreCase("Authorization")

"Authorızation" (punktloses ı) → true
```

Java hält Authorızation für Authorization, der Proxy nicht. Zwei Teile lesen denselben Header verschieden.

Eine der Lücken, die erst das zweite Review fand

ZWEI REVIEWS, ZWEI MODELLE

DARAUS FOLGT

- Ein Modell, das den Code geschrieben hat, übersieht beim Prüfen leicht dieselben Fehler
- In einer App von mir prüft deshalb Opus, umgesetzt wird mit Sonnet
- Beide Reviews prüfen die ganze Änderung: Review 2 nur auf das Diff seit Umsetzung 1 habe ich messen lassen und verworfen
- Ohne eigenen Workflow: ein Subagent mit eigenem Modell im Frontmatter

Mitnahme **Ein zweites Review ist günstig und findet fast immer noch etwas**

SESSIONS ALS DATENQUELLE

ERLEBT

Dieser Vortrag ist so entstanden: Claude hat meine Prompt-Historie und Sessions über alle Projekte ausgewertet.

SESSIONS ALS DATENQUELLE

ERLEBT

Dieser Vortrag ist so entstanden: Claude hat meine Prompt-Historie und Sessions über alle Projekte ausgewertet.

Sessions sind eine gute Datenquelle für Optimierungen!

Mitnahme **Lass Claude in euren Sessions analysieren und nach Optimierungen suchen**

SESSIONS ALS DATENQUELLE

ERLEBT

Dieser Vortrag ist so entstanden: Claude hat meine Prompt-Historie und Sessions über alle Projekte ausgewertet.

Sessions sind eine gute Datenquelle für Optimierungen!

```
Lass mich das speichern – das ist Gold für zukünftige Optimierung:

Wrote 2 memories
```

Mitnahme **Lass Claude in euren Sessions analysieren und nach Optimierungen suchen**

SESSIONS ALS DATENQUELLE

ERLEBT

Dieser Vortrag ist so entstanden: Claude hat meine Prompt-Historie und Sessions über alle Projekte ausgewertet.

Sessions sind eine gute Datenquelle für Optimierungen!

```
Lass mich das speichern – das ist Gold für zukünftige Optimierung:

Wrote 2 memories
```

Achtung: Claude Code löscht Sessions standardmäßig nach 30 Tagen. Während meiner Auswertung hat es drei gelöscht. Mehr mit cleanupPeriodDays in der settings.json.

Mitnahme **Lass Claude in euren Sessions analysieren und nach Optimierungen suchen**

MESSEN STATT VERMUTEN

MESSEN STATT VERMUTEN

ERLEBT

„Ich finde die Buildzeit von über fünf Minuten lang.“

ich, 17.10., nach einem Lauf über Nacht

MESSEN STATT VERMUTEN

ERLEBT

> *„Ich finde die Buildzeit von über fünf Minuten lang.“*
>
> ich, 17.10., nach einem Lauf über Nacht

Mit Claude gemessen und Annahmen aufgestellt, jede in einem Lauf geprüft:

MESSEN STATT VERMUTEN

ERLEBT

> *„Ich finde die Buildzeit von über fünf Minuten lang.“*
>
> ich, 17.10., nach einem Lauf über Nacht

Mit Claude gemessen und Annahmen aufgestellt, jede in einem Lauf geprüft:

Annahme	Ergebnis
Prozessstarts bremsen die Tests	**bestätigt: 95.082 Starts**
mehr Parallelität hilft	**kein Effekt**
andere git-Einstellungen helfen	**kein Effekt**
Vorlagen statt neuer Prozesse sparen etwa 61 s	**gemessen etwa 60 s**
doppelte git-Aufrufe streichen spart bis 20 s	**kein messbarer Effekt**

309 → 245 s

voller Build, nachgemessen in Lauf 52

„Damit hält der erwartete Gewinn von etwa einer Minute je vollem Build.“

MESSEN STATT VERMUTEN

ERLEBT

> *„Ich finde die Buildzeit von über fünf Minuten lang.“*
>
> ich, 17.10., nach einem Lauf über Nacht

Mit Claude gemessen und Annahmen aufgestellt, jede in einem Lauf geprüft:

Annahme	Ergebnis
Prozessstarts bremsen die Tests	**bestätigt: 95.082 Starts**
mehr Parallelität hilft	**kein Effekt**
andere git-Einstellungen helfen	**kein Effekt**
Vorlagen statt neuer Prozesse sparen etwa 61 s	**gemessen etwa 60 s**
doppelte git-Aufrufe streichen spart bis 20 s	**kein messbarer Effekt**

309 → 245 s

voller Build, nachgemessen in Lauf 52

„Damit hält der erwartete Gewinn von etwa einer Minute je vollem Build.“

Mitnahme **Messt, bevor ihr optimiert. (Auch ohne KI eine gute Idee...)**

Jeder Build kennt seinen Agenten

Ein Wrapper um pnpm schreibt je Aufruf eine Zeile: Agent, Dauer, Exit-Code, Last des Rechners.

Damit sieht die Auswertung, wofür ein Schritt seine Zeit braucht. Aus den Zeiten schätzt Claude jeden neuen Lauf und schlägt vor, die Schätzung nachzuziehen: 22-mal angenommen, 4-mal abgelehnt.

11 von 29

Läufen lagen seit dem Wechsel zu React in der Schätzung, davor 14 von 23. Die alten Messungen passten nicht mehr zum neuen Aufbau.

```
# scripts/pnpm.sh
t0=$(date +%s)
last=$(uptime | sed 's/.*averages*: //; s/[, ].*//')
pnpm "$@"; code=$?
dauer=$(( $(date +%s) - t0 ))
echo "$LAB_KENNUNG ${dauer}s exit=$code last=$last" >>
pnpm.log
exit $code

LAB_KENNUNG="Schritt 112: Umsetzung 1" scripts/pnpm.sh test
```

Jede Runde hat eine Git-Referenz

Nach jeder Umsetzung landet der Stand unter einer eigenen Referenz, nicht im Branch. So lassen sich Runden vergleichen.

Claude schlug vor, Review 2 nur auf die Änderungen seit Umsetzung 1 anzusetzen. Das spare „15 bis 20 Minuten Opus je Lauf“. Der Vergleich der Runden ergab: „V1 allein wäre schlecht.“

45 %

der 237 Befunde aus Review 2 steckten schon im Code von Umsetzung 1. Review 1 hatte sie übersehen, ein Review nur der Änderungen hätte sie nie gesehen.

```
# nach jeder Runde: Stand sichern, ohne Commit im Branch
idx=$(mktemp)
cp "$(git rev-parse --git-path index)" "$idx"
GIT_INDEX_FILE=$idx git add -A
tree=$(GIT_INDEX_FILE=$idx git write-tree)
c=$(git commit-tree "$tree" -p HEAD -m "Runde $1")
git update-ref "refs/agent/runde-$1" "$c"

git diff refs/agent/runde-1 refs/agent/runde-2
```

💝 Die direkteste der Messfolien (Nils). Seit 21.10. mit dem widerlegten Sparvorschlag von Claude.

MESSEN

Die Auswertung gehört in den Skill

Nach jedem Lauf rechnet ein Script, Claude schreibt eine Bewertung mit Vorschlägen und erwartetem Gewinn.

Ich entscheide. Kleinere Bilder wurden viermal vorgeschlagen und waren viermal hinfällig. Jetzt verbietet der Skill den Vorschlag.

133 : 32

Vorschläge angenommen und abgelehnt, mindestens

```
## Nach dem Lauf
1. Führe node scripts/auswertung.mjs aus.
2. Ergänze docs/laeufe.md um eine Bewertung
   mit Vorschlägen.
3. Jeder Vorschlag nennt seinen erwarteten
   Gewinn. Ändere keine Regel selbst.
4. Kein Vorschlag zu kleineren Bildern
   (1.700 Tokens je Bild, viermal nachgerechnet).
```

🪛🪛🪛 Doppelt mit „Workflow: nach dem Lauf“.

GANZE APP

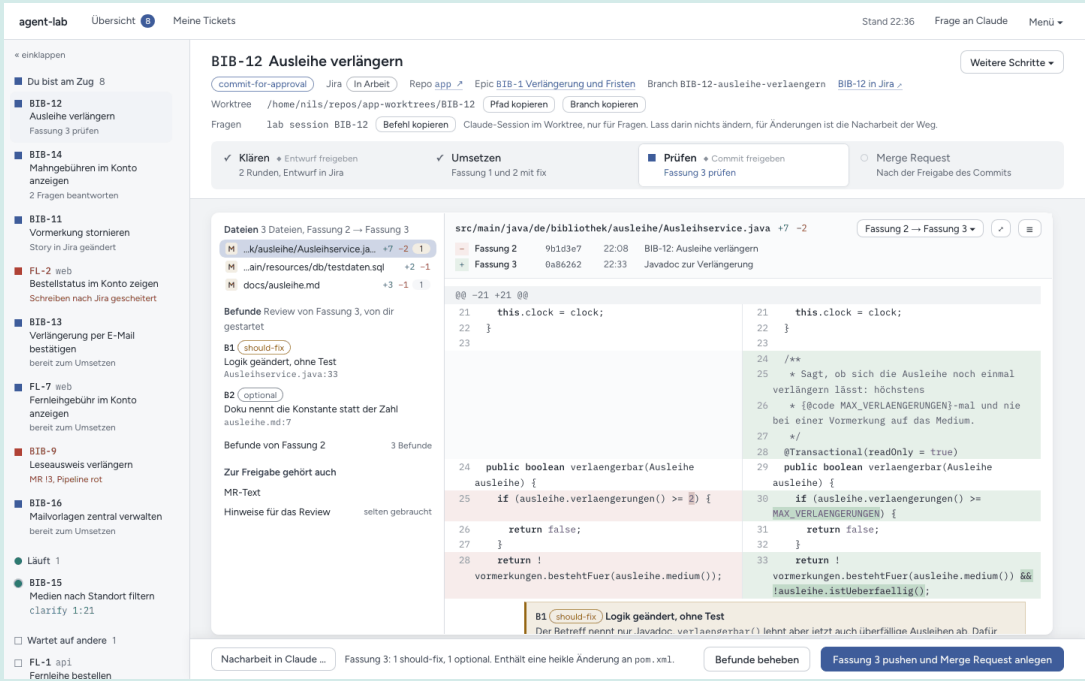
ERLEBT

Tool für Claude-basierten Workflow mit Jira und GitLab-Integration gebaut. Gebaut mit Claude Code, über Workflows, oft über Nacht.

GANZE APP

ERLEBT

Tool für Claude-basierten Workflow mit Jira und GitLab-Integration gebaut. Gebaut mit Claude Code, über Workflows, oft über Nacht.



GANZE APP

ERLEBT

Tool für Claude-basierten Workflow mit Jira und GitLab-Integration gebaut. Gebaut mit Claude Code, über Workflows, oft über Nacht.

79

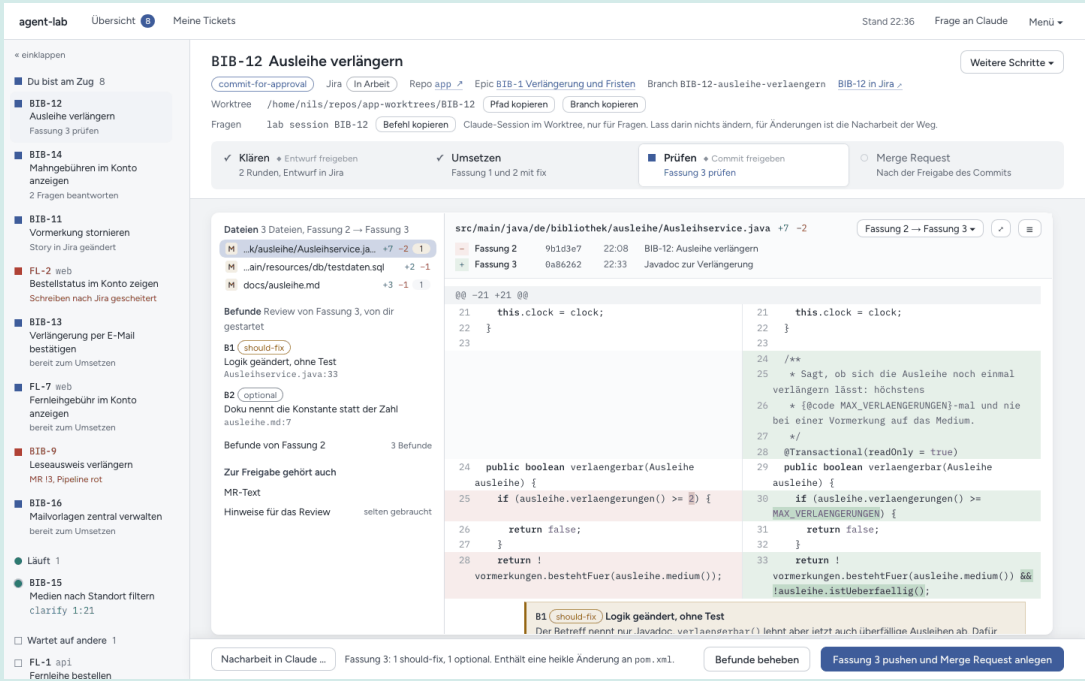
Workflow-Läufe seit 04.10.

217 h

Laufzeit der Workflows zusammen

340.000

Zeilen Code, vor allem Java+React, davon zwei Drittel Tests



GANZE APP

ERLEBT

Tool für Claude-basierten Workflow… …ut. Gebaut mit Claude Code, über Workflows, oft über Nac…

~~Beeindruckende~~ Große Zahlen…

79

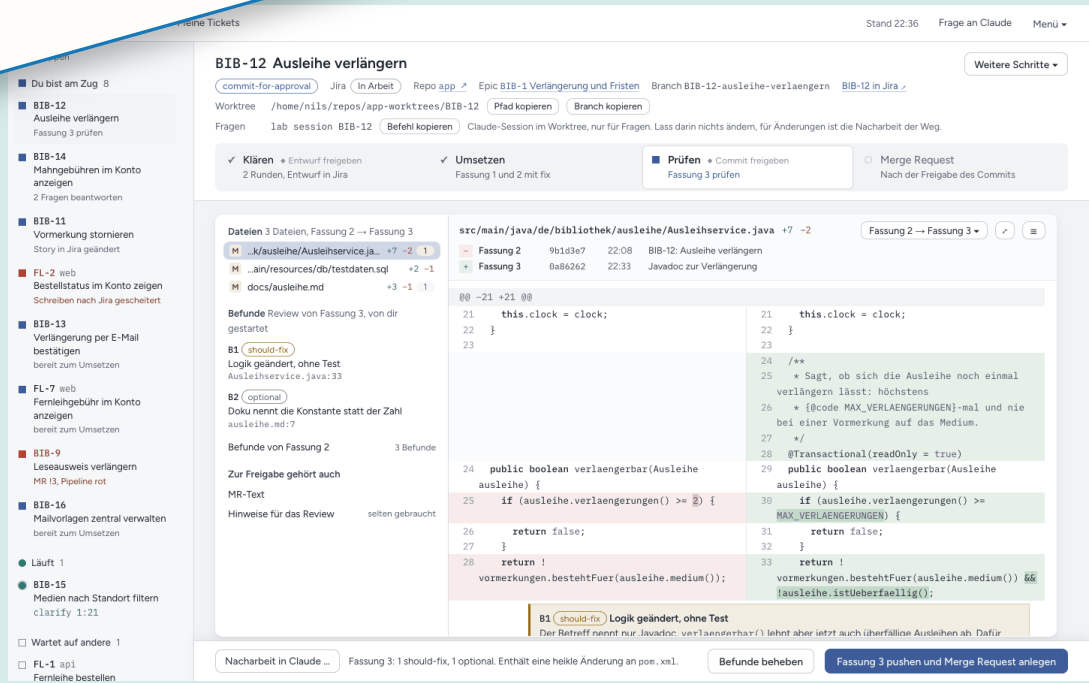
Workflow-Läufe seit 04.10.

217 h

Laufzeit der Workflows zusammen

340.000

Zeilen Code, vor allem Java+React, davon zwei Drittel Tests



GANZE APP

ERLEBT

Tool für Claude-basierten Workflow… …ut. Gebaut mit Claude Code, über Workflows, oft über Nac…

~~Beeindruckende~~ Große Zahlen…

Im echten Projekt: ein einziger Einsatz.

79

Workflow-Läufe seit 04.10.

217 h

Laufzeit der Workflows zusammen

340.000

Zeilen Code, vor allem Java+React, davon zwei Drittel Tests



GANZE APP

ERLEBT

Tool für Claude-basierten Workflow… …ut. Gebaut mit Claude Code, über Workflows, oft über Nac…

79

Workflow-Läufe seit 04.10.

217 h

Laufzeit der Workflows zusammen

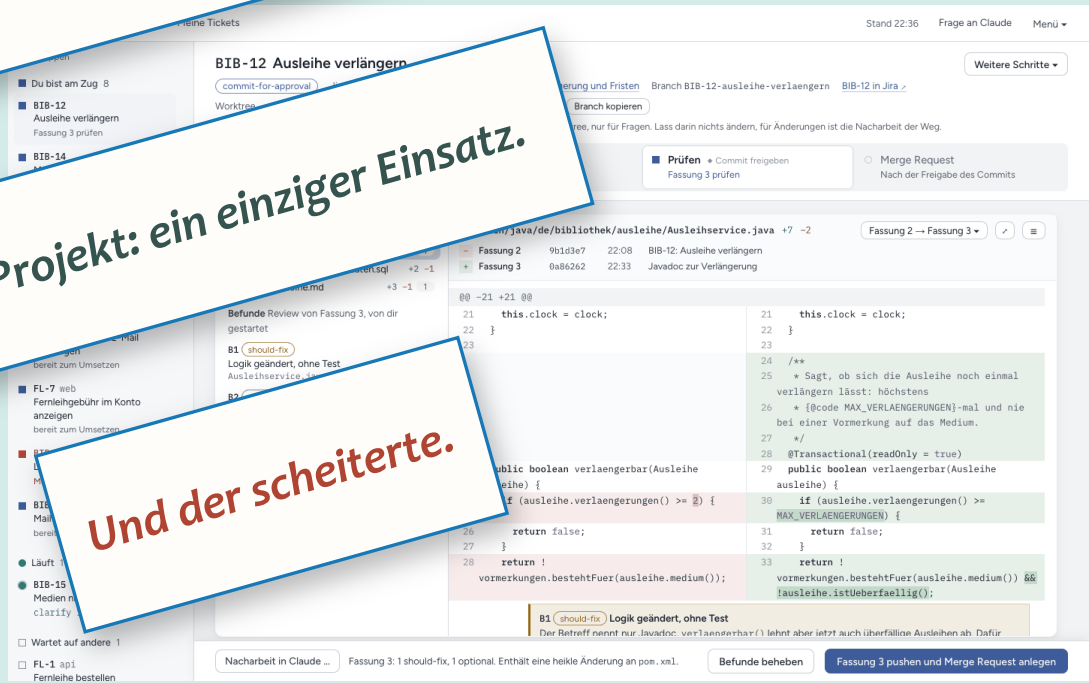
340.000

Zeilen Code, vor allem Java+React, davon zwei Drittel Tests

~~Beeindruckende~~ Große Zahlen…

Im echten Projekt: ein einziger Einsatz.

Und der scheiterte.



FEHLERFREIE TEILE, NUTZLOSES GANZES

DARAUS FOLGT

Was stimmte

- Jedes Teil gut gebaut und getestet
- Entwürfe aus Claude Design pixelgenau umgesetzt
- Asynchroner Code an der Oberfläche funktionierte

FEHLERFREIE TEILE, NUTZLOSES GANZES

DARAUS FOLGT

Was stimmte

- Jedes Teil gut gebaut und getestet
- Entwürfe aus Claude Design pixelgenau umgesetzt
- Asynchroner Code an der Oberfläche funktionierte

Was fehlte

- Eine Oberfläche, die sich selbst erklärt: Sie war überladen, die Navigation chaotisch, Buttons nicht dort, wo man sie erwartet
- Umwege an den Lücken des MVP
- Ich habe es während des Baus praktisch nicht benutzt

Dazu Kleinigkeiten: Cmd+Enter schickt die Seite ab und die Antwort ist weg, Uhrzeiten in UTC, Antworten auf Englisch. Alle Tests waren grün.

FEHLERFREIE TEILE, NUTZLOSES GANZES

DARAUS FOLGT

Was stimmte

- Jedes Teil gut gebaut und getestet
- Entwürfe aus Claude Design pixelgenau umgesetzt
- Asynchroner Code an der Oberfläche funktionierte

Was fehlte

- Eine Oberfläche, die sich selbst erklärt: Sie war überladen, die Navigation chaotisch, Buttons nicht dort, wo man sie erwartet
- Umwege an den Lücken des MVP
- Ich habe es während des Baus praktisch nicht benutzt

Dazu Kleinigkeiten: Cmd+Enter schickt die Seite ab und die Antwort ist weg, Uhrzeiten in UTC, Antworten auf Englisch. Alle Tests waren grün.

Mitnahme **Claude baut, was beschrieben ist. Benutzt es früh selbst, sonst prüft niemand das Ganze**

DARAUS FOLGT

Wenn Bauen fast nichts kostet, baut man zu viel.

Auch Ideen außerhalb des MVP habe ich bauen lassen, weil sie ja fast nichts kosten.

Bezahlt habe ich doppelt: erst mit dem Bau, dann mit dem späteren echten Einsatz und mit langsameren Läufen durch mehr Code, Tests und Kontext.

Dieselbe Lektion wie bei meinen Slash-Commands im Mai: 14 an einem Abend gebaut, vier Tage später 6 übrig.

ZUM MITNEHMEN

1 **Was gelten muss, gehört nicht in die CLAUDE.md**

Sondern in Script, Hook oder Sandbox. Prosa scheitert unbemerkt: der Import, der nicht ankommt, die Regel, die Claude überliest

2 **Prüft nach, statt zu glauben**

Kommt die Regel an? Prüft der Test etwas? Lohnt das zweite Review? Die Daten liegen schon in euren Sessions

3 **Fragt Claude, wie es besser geht**

Wie etwas in Claude Code geht, was sich an eurem Ablauf verbessern lässt. Und probiert neue Wege: ein Buch zum Lernen, eine Seite zum Anklicken



Vielen Dank!