CLAUDE CODE · FABLE 5.1

Deine Fable-Limits halten länger

Die zwei Zeilen, mit denen deine Subagents auf Opus laufen statt auf Fable

2 Zeilen
in deiner settings.json, mehr brauchst du nicht

–50 %
kostet Opus 5 pro Token im Vergleich zu Fable 5.1

2.1.257
ist die Mindestversion von Claude Code für die zweite Zeile

Der Tipp kommt direkt aus dem Claude-Code-Team.
Lydia Hallie von Anthropic hat am 13. September geschrieben: „Setting my subagent model to Opus has helped me save so much usage on Fable 5.1." Also: Die Subagents auf Opus zu stellen, hat ihr mit Fable 5.1 viel Limit gespart. Geteilt hat sie dafür eine Zeile. Warum du eine zweite brauchst, steht auf Seite 3.

01 **Der Block zum Kopieren**
Auf Seite 2 steht die komplette settings.json mit beiden Zeilen, und was jede Zeile genau macht.

02 **Wer bei den Modellen gewinnt**
Auf Seite 3 steht die Rangfolge, nach der Claude Code das Modell für einen Subagent auswählt. Genau daran scheitert die eine Zeile allein.

03 **Der ehrliche Teil**
Auf Seite 4 steht, wann dich die Einstellung mehr kostet statt weniger, und wie du das vermeidest.

DER BLOCK

Zwei Zeilen in die settings.json

Einmal eintragen, danach gilt es für deine Sessions

~/.CLAUDE/SETTINGS.JSON

```
{
  "env": {
    "CLAUDE_CODE_SUBAGENT_MODEL": "opus",
    "CLAUDE_CODE_SUBAGENT_MODEL_FORCE": "1"
  }
}
```

01 **Die erste Zeile**

Sie stellt das Standardmodell für Subagents auf Opus. Das gilt auch für Agent-Teams und Workflow-Agents, solange ihnen nichts anderes ein Modell zuweist.

02 **Die zweite Zeile**

Sie zwingt dieses Modell auf alle Subagents, also auch auf den eingebauten Plan-Agent und auf Subagents, denen Claude beim Start selbst ein Modell mitgibt.

03 **Wenn schon etwas in der Datei steht**

Dann kommt der env-Block einfach mit in die äußeren geschweiften Klammern, mit einem Komma nach dem Eintrag davor. Gibt es env schon, trägst du nur die zwei Zeilen darin ein.

Vorher kurz die Version prüfen.
Mit claude --version siehst du, ob du mindestens Version 2.1.257 hast. Sonst aktualisierst du Claude Code zuerst. Und starte nach dem Eintragen am besten eine neue Session, damit die Einstellung sicher geladen ist.

WER GEWINNT

Warum eine Zeile nicht reicht

Die Rangfolge aus der offiziellen Doku von Claude Code

RANG	WOHER DAS MODELL FÜR EINEN SUBAGENT KOMMT
1	Das Modell, das Claude beim Start eines Subagents selbst mitgibt
2	Das model-Feld in der Definition des Agents, auch der Wert inherit
3	Deine erste Zeile, also CLAUDE_CODE_SUBAGENT_MODEL
4	Das Modell deines Hauptchats

Mit der zweiten Zeile springt deine Einstellung ganz nach oben.
Ohne sie gewinnt Claude, sobald er einem Subagent selbst ein Modell mitgibt. Und der eingebaute Plan-Agent übernimmt ohne sie einfach das Modell deines Hauptchats, bei dir also Fable.

EINGEBAUTER AGENT	WELCHES MODELL ER NIMMT
Explore	Das Modell deines Hauptchats, aber auf Opus gedeckelt. Mit Fable im Hauptchat läuft er also schon auf Opus
Plan	Das Modell deines Hauptchats, außer beide Zeilen sind gesetzt
General-purpose	Deine erste Zeile, wenn nichts anderes ein Modell vorgibt, sonst das Modell deines Hauptchats

DER EHRLICHE TEIL

Wann es dich mehr kostet

Drei Stellen, an denen du kurz nachdenken solltest

01 **Eigene Subagents auf Haiku**

Die zweite Zeile zwingt Opus auf alle Subagents, also auch auf eigene Agents, die du bewusst auf Haiku gestellt hast. Die laufen dann teurer als vorher. Hast du solche Agents, lässt du die zweite Zeile weg. Dann bleibt allerdings der Plan-Agent auf Fable.

02 **Aufgaben, die wirklich Fable brauchen**

Opus reicht für die meisten Zwischenschritte, aber nicht für alle. Merkst du, dass ein Subagent bei einer schweren Aufgabe schwächer wird, nimmst du die zwei Zeilen einfach wieder raus.

03 **Die 50 Prozent sind der API-Preis**

Opus 5 kostet 5 $ Input und 25 $ Output pro Million Tokens, Fable 5.1 kostet 10 $ und 50 $. Wie viel länger dein Abo-Limit damit hält, hängt davon ab, wie viel deine Subagents bei dir überhaupt arbeiten.

Nachsehen statt raten.
/usage zeigt dir, was deine Limits verbraucht, und /skill-doctor zeigt dir, was jeder deiner Skills im Kontext kostet und wie oft du ihn benutzt. Damit siehst du ziemlich schnell, ob das Problem bei den Subagents liegt oder ganz woanders.

NÄCHSTER SCHRITT

Direkt umsetzen

Du brauchst dafür kein Plugin, nur deine settings.json

01 **Version prüfen**

Tipp im Terminal claude --version ein. Steht dort weniger als 2.1.257, aktualisierst du Claude Code zuerst.

02 **Block eintragen**

Öffne ~/.claude/settings.json, trag den Block von Seite 2 ein und starte eine neue Session.

03 **Nach ein paar Tagen vergleichen**

Schau mit /usage, ob dein Fable-Limit jetzt länger hält. Wenn nicht, liegt der Verbrauch eher im Hauptchat als bei den Subagents.

Du willst solche Sachen mitbekommen, ohne danach suchen zu müssen?

Ich zeige auf Instagram und TikTok, was mit Claude, ChatGPT und den anderen Tools gerade geht, und die Anleitung dazu gibt es jedes Mal als PDF.

→ **Instagram: @timhuck.ai**
→ **TikTok: @tim_huck_ai**
→ **www.shortmedia.agency**

Angaben geprüft am 16. September 2026. Nach der Doku von Claude Code zu Environment variables, Subagents und Model configuration, der Preisliste auf platform.claude.com, dem Wochen-Digest von Claude Code aus Woche 36 und dem Post von Lydia Hallie auf X vom 13. September 2026. Claude Code ändert sich jede Woche, prüf die Einstellungen deshalb bei Gelegenheit selbst nach.