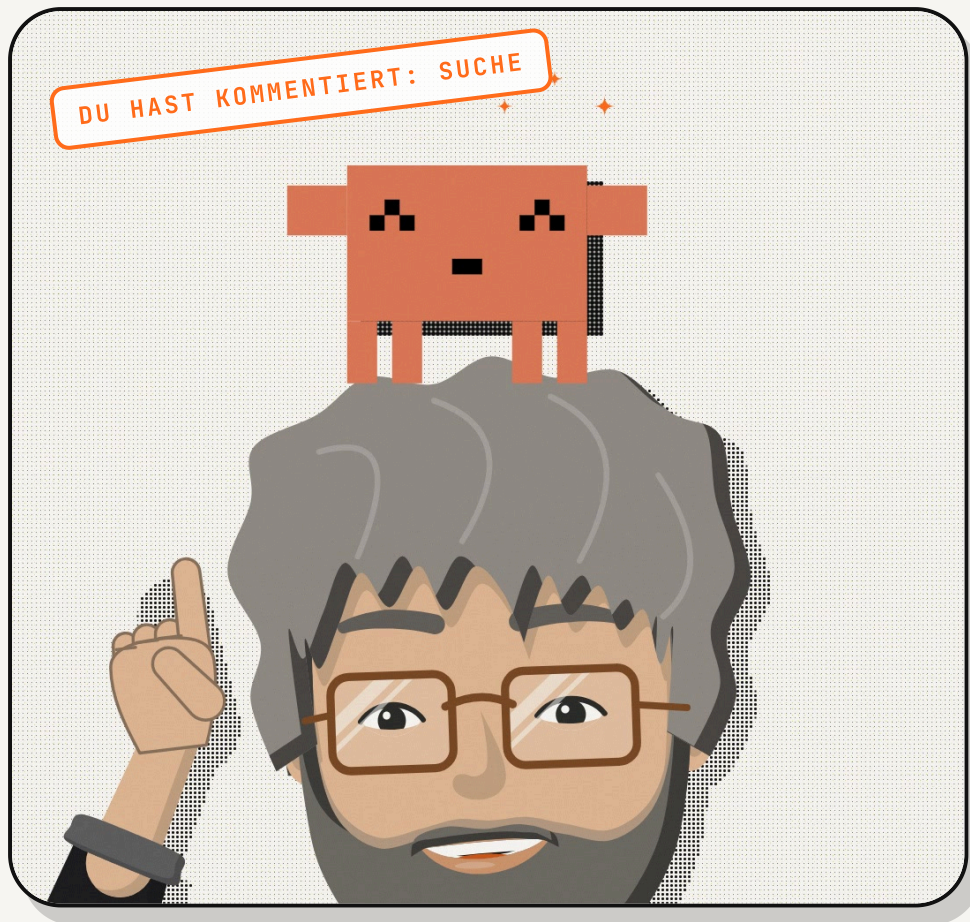




Finde deine *perfekte Skill-Kombi*

Das ganze System hinter dem Video: warum ein vollgestopftes Claude schlechter arbeitet, was skills.sh dir wirklich zeigt, warum dein Business-Projekt der Schlüssel ist, wie Claude mit dir die passende Kombination findet, der Sicherheits-Check vor jeder Installation und wie aus deiner Kombi ein eigener Helfer für deine Kunden wird.

// 01

Auf einen Blick

FRAGE	KURZ GESAGT
Das Problem	Jeder empfiehlt „seinen“ Skill. Wer alles installiert, hat ein Claude voller Werkzeuge, die es nie braucht und die sich gegenseitig in die Quere kommen
Der Überblick	skills.sh von Vercel: Ranglisten über alle Zeit, die letzten 24 Stunden und die letzte Stunde, dazu Sicherheits-Scans
Der Schlüssel	Ein frisches Claude-Projekt, in dem Marke, Business und Ziel komplett beschrieben sind
Die Auswahl	Nichts installieren. Claude recherchiert, welche Kombination genau dir hilft, und begründet jede Wahl
Der Turbo	grill-me: ein Interview-Skill, der dich so lange fragt, bis klar ist, was nur zu dir passt
Das Produkt	Aus der Kombi baut Claude eigene Skills. Sie folgen einem offenen Standard und laufen auch in anderen KI-Tools

// AUS DEM VIDEO



Warum mehr Skills nicht mehr bringen

Ein Skill ist ein Ordner mit einer Datei `SKILL.md`: ein Name, eine kurze Beschreibung und darunter die Anleitung, dazu manchmal Skripte, Vorlagen oder Beispiele. Claude liest die Anleitung erst, wenn eine Aufgabe zur Beschreibung passt. Anthropic nennt das „Progressive Disclosure“: Name und Beschreibung sind immer geladen (etwa 100 Tokens pro Skill), die eigentliche Anleitung kommt erst bei Bedarf dazu.

Das klingt, als wäre die Menge egal. Ist sie aber nicht, aus drei Gründen:

1. **Jede Beschreibung kostet Platz.** Die Claude-Code-Doku sagt es klar: Jede gelistete Skill-Beschreibung kostet bei jeder Nachricht Kontext. Ein Skill ist günstig, fünfzig Skills sind es nicht mehr.
2. **Beschreibungen konkurrieren.** Claude entscheidet anhand der Beschreibung, welcher Skill zu deiner Anfrage passt. Drei Skills, die alle „Texte verbessern“, machen die Wahl unsicher, und am Ende greift der falsche.
3. **Doppelt und widersprüchlich.** Gleiche Namen überschreiben sich, und wer ein Skill-Paket einmal als Plugin und einmal über `skills.sh` installiert, hat jeden Skill doppelt. Matt Pocock warnt davor sogar in seinem eigenen README.

Eine feste Obergrenze nennt Anthropic nicht. Die Doku empfiehlt aber, ungenutzte Skills aufzuspüren (in Claude Code mit `/skill-doctor`) und Beschreibungen eng zu halten. Das Ziel ist also nicht „möglichst wenig“, sondern: **nur Skills, die für dein Business einen klaren Job haben.**



DER MENTOR SAGT

Eine Werkstatt wird nicht besser, wenn du mehr Werkzeug an die Wand hängst. Sie wird besser, wenn jedes Werkzeug einen festen Platz und einen Job hat.

Was skills.sh dir wirklich zeigt

skills.sh ist das offene Skill-Verzeichnis von Vercel („The Open Agent Skills Ecosystem“). Die Seite selbst zeigt keine Preise, das Kommandozeilen-Tool dahinter ist Open Source (MIT-Lizenz). Du kannst also kostenlos stöbern.

skills.sh • Vercel

die Rangliste: All Time, Trending (24h) und Hot (letzte Stunde)

skills.sh ➤

skills.sh/audits

Sicherheits-Scans von Gen Agent Trust Hub, Socket und Snyk pro Skill

[skills.sh](#) ➤

skills.sh/official

Skills direkt von den Firmen hinter der Technik, auch von Anthropic

[skills.sh](#) ➤

Transparenz: Ich stehe in keiner Verbindung zu skills.sh oder Vercel. Die Seite gehört mir nicht, ich werde nicht dafür bezahlt, sie zu empfehlen, und ich bekomme kein Geld für Vermittlungen. Alles hier stammt aus meiner eigenen Nutzung und meiner täglichen Arbeit damit. Es ist also meine persönliche Erfahrung als Nutzer.

BEREICH	WAS DU DORT SIEHST
All Time	die meistinstallierten Skills seit Start
Trending (24h)	was in den letzten 24 Stunden am meisten installiert wurde
Hot	Bewegung in der letzten Stunde
Official	Skills von über 100 Herstellern, darunter Anthropic
Audits	drei unabhängige Sicherheits-Scans nebeneinander
Packs, Topics	Sammlungen und Themen zum Stöbern

Warum die Zahlen kein Beweis sind

Die Rangliste entsteht aus anonymen Installationsdaten des Tools `npx skills`. Ein Skill taucht automatisch auf, sobald Leute ihn damit installieren. Geprüft wird dabei nichts. Das hat drei Folgen:

- **Hype zählt mit.** Wenn ein großer Account einen Skill empfiehlt, installieren ihn Tausende an einem Tag, ob er ihnen hilft oder nicht.
- **Nur ein Weg wird gezählt.** Uploads in der Claude-App oder Installationen als Plugin tauchen in der Rangliste nicht auf.
- **Die Spitze ist für Entwickler.** Am 27.10.2026 waren die Top 5 aller Zeiten: find-skills (4,1 Mio.), grill-me (1,4 Mio.), hyperframes-cli (1,3 Mio.), agent-browser (1,3 Mio.) und hyperframes (1,2 Mio.). Viele der Top 10 drehen sich um Code, Tests und Architektur. Für eine Physiopraxis oder einen Online-Shop ist das nicht automatisch die richtige Wahl.

Als Überblick ist die Seite trotzdem perfekt: Du siehst in zwei Minuten, was es gibt, was gerade neu ist und welche Hersteller offizielle Skills anbieten. Genau dafür brauchst du sie.

Anschauen, ohne zu installieren

Wenn du Claude Code oder ein anderes Agent-Tool nutzt (Node.js 22.20 oder neuer), kannst du Skills ansehen, bevor sie auf deinem Rechner landen:

```
npx skills find [suchbegriff]
npx skills add <owner/repo> -l
npx skills use <source>@<skill>
```

`find` sucht im Verzeichnis, `-l` listet die Skills eines Repos ohne Installation, `use` probiert einen einzelnen Skill aus, ohne ihn dauerhaft zu installieren. Das passt perfekt zu Schritt 3.

// 04

Warum dein Business-Projekt der Schlüssel ist

Ein Claude-Projekt ist ein eigener Arbeitsbereich mit **Projekt-Anweisungen** (gelten für jeden Chat darin) und **Projektwissen** (Dateien und Texte, die Claude als Hintergrund nutzt). Projekte gibt es in jedem Plan, im kostenlosen Plan bis zu fünf. In bezahlten Plänen kann das Projektwissen per RAG bis zu zehnmal mehr Material fassen.

Warum das für die Skill-Wahl so wichtig ist: Ohne Kontext kann Claude nur allgemein antworten („grill-me ist beliebt“). Mit deinem Projekt kann es sagen: „Für deinen Shop mit drei Mitarbeitern und dem Ziel, Produktseiten schneller zu schreiben, bringt dir Skill A mehr als Skill B, weil ...“. Die Empfehlung wird so gut wie die Beschreibung, die du ihm gibst. Darum: Nimm dir hier eine halbe Stunde und füll alles aus.

Vorlage für die Projekt-Anweisungen (kopieren und ausfüllen)

Du bist mein Berater für mein Business und für die Werkzeuge, mit denen ich arbeite. Hier ist alles über mich:

1. Marke: [Name, Website, in einem Satz: wofür stehe ich]
 2. Business: [was ich verkaufe, Preise grob, wie ich Kunden gewinne]
 3. Zielgruppe: [wer kauft, welches Problem sie haben, wo ich sie erreiche]
 4. Ziel für die nächsten 6 Monate: [konkret und messbar]
 5. Was mich am meisten Zeit kostet: [die drei größten Zeitfresser der Woche]
 6. Meine Tools: [z. B. Claude-App, Claude Code, Canva, Shopify, Notion]
 7. Mein Plan und meine Grenzen: [Claude-Plan, Budget, was nicht in eine KI darf]
 8. Ton und Stil: [du/Sie, locker/förmlich, Beispiele für Texte, die ich mag]
- Wenn dir für eine gute Antwort etwas fehlt, frag mich zuerst, statt zu raten.

Lade dazu ins Projektwissen, was Claude über dich wissen sollte: dein Angebot, eine Preisliste, zwei bis drei Texte in deinem Stil, Kundenfragen, die immer wieder kommen. Das Projekt ist ab jetzt die Messlatte, an der jeder Skill sich beweisen muss.

Die Recherche: Claude wählt mit dir

Jetzt kommt die Umkehr. Statt erst zu installieren und dann zu schauen, ob es hilft, gibst du Claude die Kandidaten und lässt es begründen. Erst wenn die Begründung dich überzeugt, wird installiert.

1 • Die Kandidaten sammeln

Ich habe auf `skills.sh` diese Skills gefunden: [Liste mit Name, Repo und Link]. Lies zu jedem die Beschreibung und, wenn du kannst, die `SKILL.md` im Repo. Fasse pro Skill in zwei Sätzen zusammen, was er wirklich tut, für wen er gebaut ist und ob er Skripte ausführt oder Daten ins Internet schickt.

2 • Die passende Kombination finden

Vergleiche die Kandidaten mit allem, was du in diesem Projekt über mein Business weißt. Welche zwei bis vier Skills ergänzen sich so, dass sie meinem Ziel [Ziel] am meisten helfen? Gib mir eine Tabelle: Skill, welcher meiner Zeitfresser damit kleiner wird, warum genau dieser Skill, was er nicht kann. Empfiehl mir höchstens vier.

3 • Das Warum und das Warum-nicht

Begründe für jeden Skill, den du NICHT empfiehlst, warum er für mich keinen Sinn ergibt, auch wenn er auf `skills.sh` weit oben steht. Sei ehrlich, wenn ein beliebter Skill für mein Business nutzlos ist.

4 • Der Gegencheck

Spiel jetzt den Skeptiker. Was spricht gegen deine Kombination? Wo überschneiden sich die Skills, wo könnten sie sich widersprechen, und welcher davon wäre der erste, den ich in drei Monaten wieder deinstalliere?

5 • Der Installationsplan

Mach mir einen Plan: in welcher Reihenfolge ich die Skills ausprobiere, wie ich in einer Woche merke, ob sich einer lohnt, und welchen Test-Auftrag ich jedem Skill dafür gebe. Erst danach installiere ich den nächsten.

Geh dabei ruhig hin und her. Widersprich, frag nach, gib neue Infos dazu. Die beste Kombination entsteht im Gespräch, nicht in der ersten Antwort.



DER MENTOR SAGT

Ich installiere keinen Skill, dessen Job ich nicht in einem Satz sagen kann.
Wenn Claude ihn mir nicht erklären kann, brauche ich ihn nicht.

// 06

grill-me: das Interview, das deine Kombi schärft

grill-me ist ein Skill von Matt Pocock (Repo `mattpocock/skills`, MIT-Lizenz) und steht auf `skills.sh` auf Platz 2 aller Zeiten. Er ist ein „unerbittliches Interview, um einen Plan oder ein Design zu schärfen“. In der Praxis: Claude fragt dich in Runden, nummeriert jede Frage und schlägt gleich eine Antwort vor, die du mit „ja“ übernehmen oder korrigieren kannst. Fakten schlägt es selbst nach, Entscheidungen bleiben bei dir. Es hört erst auf, wenn nichts mehr nur angenommen ist, und handelt nicht, bevor du zustimmst.

Das Interview funktioniert nicht nur bei Code, sondern ausdrücklich auch bei Business-Entscheidungen und Texten. Genau deshalb passt es so gut zur Skill-Wahl.

grill-me auf `skills.sh`

Platz, Installationen, Sicherheits-Scans (alle drei: Pass)

[skills.sh](#) ➤

mattpocock/skills • GitHub

Repo mit README und allen Skills

[github.com](#) ➤

So startest du es

In Claude Code oder einem anderen Agent-Tool installierst du es mit dem Befehl, den `skills.sh` anzeigt:

```
npx skills add https://github.com/mattpocock/skills --skill grill-me
```

Dann tippst du in einem frischen Chat `/grill-me` und sagst, worum es geht. Wichtig zu wissen: grill-me startet nie von selbst (es ist so eingestellt, dass nur du es aufrufen kannst), und die eigentliche Logik steckt im Skill `grilling`, den grill-me aufruft. Lass dabei den Plan-Modus aus.

In der Claude-App (Web und Desktop) gibt es keinen dokumentierten Slash-Befehl für Skills, und Uploads mit Zusatzfeldern, die nur Claude Code kennt, schlagen laut Doku fehl. Nimm dort den Skill `grilling` aus demselben Repo oder den Prompt unten [prüfen: Upload von grill-me in `claude.ai` nicht getestet].

Der Interview-Prompt für deine Skill-Wahl

Interview mich wie ein strenger Berater, bevor du mir irgendeinen Skill empfiehlst. Stell mir deine Fragen in Runden, nummeriere jede Frage und schlag zu jeder eine Antwort vor, die ich mit "ja" annehmen oder ändern kann. Frag nach meinem Ziel, meinen Zeitfressern, meinen Kunden, meinen Tools und meinen Grenzen. Hör erst auf, wenn du nichts mehr annehmen musst. Dann empfiehl mir die zwei bis vier Skills aus meiner Liste, die nur zu mir passen, und sag bei jedem, warum.

Ein Tipp aus der grill-me-Doku selbst: Wenn du vierzigmal nur „ja“ sagst, schreibt am Ende die KI deinen Plan und nicht du. Antworte ehrlich, auch mit „weiß ich nicht“.

// 07

Warum die Kombination so wertvoll ist

Ein einzelner Skill löst eine Aufgabe. Eine Kombination löst einen ganzen Ablauf. Erst wenn Recherche, Erstellung und Prüfung ineinandergreifen, spart dir Claude wirklich Stunden. Und weil deine Kombination aus deinem Projekt kommt, hat sie niemand sonst genau so.

WER	TYPISCHE KOMBI	WAS SIE BRINGT
Creator	Interview-Skill + Design-Skill + eigener Marken-Skill	Ideen schärfen, im eigenen Look umsetzen, immer im eigenen Ton
Agentur	Recherche-Skill + Berichts-Skill + eigener Kunden-Skill	jeder Kunde bekommt Analysen im eigenen Design, ohne Copy-Paste
Shop	Produkttext-Skill + Bild-Skill + eigener Shop-Regeln-Skill	Produktseiten in Minuten, mit deinen Pflichtangaben
Dienstleister	Interview-Skill + Angebots-Skill + eigener Preis-Skill	aus dem Erstgespräch wird ein sauberes Angebot

Die Beispiele zeigen Skill-Arten, keine festen Empfehlungen. Welche konkreten Skills deine Kombi bilden, findet Claude mit dir in deinem Projekt heraus.

Der eigentliche Wert liegt in der Spalte „eigener Skill“: Das ist dein Wissen, das Claude bei jeder Aufgabe mitnimmt. Wie du ihn baust, steht weiter unten.

Der Sicherheits-Check vor jeder Installation

Ein Skill ist mehr als Text. Er kann Skripte enthalten, Werkzeuge aufrufen und Daten verschicken. Anthropic sagt dazu deutlich: Nutze Skills nur aus vertrauenswürdigen Quellen, also eigene oder solche von Anthropic, und behandle jede Installation wie die Installation von Software.

- ☐ **Quelle prüfen:** Wer steht hinter dem Repo? Offizieller Hersteller, bekannte Person oder ein Account ohne Geschichte?
- ☐ **Audits ansehen:** Auf skills.sh/audits die drei Scans vergleichen. Die Urteile weichen oft voneinander ab; ein „High Risk“ heißt: erst genau hinsehen.
- ☐ **Inhalt anschauen:** das Repo auf GitHub öffnen oder den Inhalt mit dem Befehl `-l` (siehe oben) auflisten. Steckt nur Text drin oder auch Skripte?
- ☐ **SKILL.md lesen:** Tut die Anleitung das, was die Beschreibung verspricht? Nichts Verstecktes, keine Anweisung, Daten irgendwohin zu schicken?
- ☐ **Skripte lesen:** Netzwerkzugriffe, Dateizugriffe außerhalb des Projekts, Befehle, die nicht zum Zweck passen.
- ☐ **Externe Links:** Skills, die Inhalte von fremden Adressen nachladen, sind riskanter, weil sich diese Inhalte ändern können.
- ☐ **Lizenz:** Darfst du den Skill nutzen, verändern und weitergeben? MIT ist großzügig, andere Lizenzen nicht unbedingt.
- ☐ **Doppelte meiden:** Ein Paket entweder als Plugin oder über skills.sh installieren, nicht beides.

Lass Claude dabei helfen:

```
Prüfe diesen Skill, bevor ich ihn installiere: [Link zum Repo]. Lies die SKILL.md und alle Skripte. Sag mir, was er wirklich tut, ob er Daten ins Internet schickt oder Dateien außerhalb des Projekts anfasst, und ob etwas nicht zur Beschreibung passt. Am Ende: installieren, mit Vorsicht installieren oder lassen.
```

Kleiner Bonus: Wer keine anonymen Nutzungsdaten an skills.sh senden will, setzt vor dem Befehl `DISABLE_TELEMETRY=1` (oder `DO_NOT_TRACK=1`). Das schaltet laut README auch die Abfrage der Sicherheits-Scans ab.

Eigene Helfer bauen und an Kunden verkaufen

Jetzt wird aus deiner Kombi ein Produkt. Lass Claude die Teile, die für dich funktionieren, zu eigenen Skills zusammenbauen: deine Abläufe, deine Regeln, deine Vorlagen, dein Ton. Das ist Schritt 4 aus dem Video.

Der Aufbau eines Skills

Ein Skill ist ein Ordner, dessen Name zum Skill-Namen passt, mit einer `SKILL.md` und optional den Unterordnern `scripts/`, `references/` und `assets/`. Oben in der Datei stehen Name und Beschreibung, darunter die Anleitung:

```
---
name: anbot-aus-erstgesprach
description: Macht aus den Notizen eines Erstgesprächs ein Angebot im Stil von [Marke]. Nutze ihn, wenn der Nutzer Gesprächsnotizen schickt und ein Angebot möchte.
---

# Angebot aus Erstgespräch
1. Lies die Notizen und frag nach, was fehlt: Umfang, Termin, Budget.
2. Nimm die Preise aus references/preise.md.
3. Schreib das Angebot nach assets/vorlage.md, im Ton aus references/ton.md.
```

Regeln aus der Doku: Der Name hat höchstens 64 Zeichen, nur Kleinbuchstaben, Zahlen und Bindestriche, und darf weder „anthropic“ noch „claude“ enthalten. Die Beschreibung hat höchstens 1.024 Zeichen und sagt, was der Skill tut und wann er gebraucht wird. Sie ist das, woran Claude entscheidet, ob er passt, also: eng und konkret.

Der Bau-Prompt

```
Bau mir aus unserer Skill-Kombi und allem, was du über mein Business weißt, einen eigenen Skill nach dem offenen Agent-Skills-Standard. Er soll [Aufgabe] erledigen. Gib mir den Ordner mit SKILL.md (name, description, Anleitung in Schritten) und den Dateien in references/ und assets/. Nutze nur die Standard-Felder name, description und license, damit er auch in anderen KI-Tools läuft. Schreib die Beschreibung so eng, dass er sich nicht mit meinen anderen Skills überschneidet.
```

Warum das auch in anderen KIs läuft

Das Skill-Format hat Anthropic entwickelt und als offenen Standard veröffentlicht (agentskills.io). OpenAI schreibt, dass seine Skills auf diesem Standard aufbauen (ChatGPT-Desktop-App, Codex CLI und Codex-IDE-Erweiterung). GitHub Copilot liest Skills unter anderem aus `.claude/skills` und `.agents/skills` [prüfen: Copilot-Doku nicht direkt geöffnet]. Das Tool hinter skills.sh nennt rund 80 unterstützte Agents. Wichtig: Felder, die nur Claude Code kennt, sind nicht übertragbar. Bleib bei den Standard-Feldern, wenn der Skill überall laufen soll.

So lieferst du ihn an Kunden aus

WEG	SO GEHT'S	GUT FÜR
ZIP	Ordner zippen, Kunde lädt ihn in Claude unter Customize > Skills hoch	einzelne Nutzer in der Claude-App
Team	Inhaber von Team- oder Enterprise-Konten stellen Skills für alle bereit	Firmenkunden mit Claude Team
Repo	Kunde installiert mit <code>npx skills add <owner/repo></code>	Kunden mit Claude Code, Codex, Cursor
API	Upload über <code>/v1/skills</code> , gilt im ganzen Workspace	Kunden, die Claude in eigene Software einbauen

Für die Claude-App braucht dein Kunde aktivierte Code-Ausführung. Skills gibt es dort laut Hilfe-Artikel in allen Plänen, auch Free [prüfen: die ältere Plattform-Doku nennt erst Pro und einen anderen Menüweg]. Skills werden nicht zwischen Claude-App, API und Claude Code synchronisiert; jeder Ort braucht seinen eigenen Upload. Ein öffentliches Repo taucht übrigens irgendwann selbst auf skills.sh auf, sobald es über das Tool installiert wird. Für zahlende Kunden ist ein privates Repo oder ein ZIP meist die bessere Wahl.

So verkaufst du es: nicht als „Skill“, sondern als Feature. „Ab jetzt schreibt Ihr Claude jedes Angebot in Ihrem Stil“ versteht jeder Kunde. Gut verkaufen lässt sich ein Einrichtungspaket (Projekt anlegen, Kombi finden, eigenen Skill bauen) oder ein Pflege-Abo, in dem du den Skill an neue Preise und Abläufe anpasst.

Wenn du fremde Skills in dein Paket einbaust, prüfe vorher deren Lizenz. Und wenn du Logos, Marken oder Texte deiner Kunden in einen Skill packst, bist du dafür verantwortlich, die Rechte daran zu haben.

// 10

Grenzen und Risiken

- **Kein Skill ersetzt dein Urteil.** Claude kann begründen, aber nur so gut wie die Infos in deinem Projekt. Prüf die Empfehlung an einer echten Aufgabe.
- **Ranglisten ändern sich täglich.** Die Zahlen in diesem Guide sind vom 27.10.2026 und morgen schon anders.
- **Fremder Code bleibt fremder Code.** Die Scans auf skills.sh sind ein Hinweis, keine Garantie. skills.sh selbst prüft die Skills nicht.
- **Unterschiedliche Doku.** Anthropic's Hilfe-Artikel und die Plattform-Doku beschreiben Pläne und Menüs in der Claude-App unterschiedlich. Im Zweifel gilt, was du in deiner App siehst.
- **Daten im Skill.** Was in `references/` steht, liest Claude bei jeder Nutzung. Keine Passwörter, keine Kundendaten, die dort nicht hingehören.

Agent Skills • Anthropic-Doku

wie Skills funktionieren, Regeln und Sicherheit

platform.claude.com ↗

Skills in Claude nutzen • Hilfe-Center

Upload, Pläne, Team-Freigabe

support.claude.com ↗

Skills in Claude Code

Ordner, Slash-Befehle, Kontext und /skill-doctor

code.claude.com ↗

agentskills.io

der offene Standard

agentskills.io ↗

Deine erste Stunde

- ☐ Auf skills.sh 20 Minuten stöbern: All Time, Trending (24h), Official. Fünf bis acht Kandidaten notieren.
- ☐ Ein neues Claude-Projekt anlegen und die Projekt-Anweisungen aus diesem Guide komplett ausfüllen.
- ☐ Angebot, Preisliste und zwei Texte in deinem Stil ins Projektwissen laden.
- ☐ Prompt 1 und 2 laufen lassen: Kandidaten zusammenfassen, Kombi vorschlagen lassen.
- ☐ Den Interview-Prompt (oder grill-me in Claude Code) nutzen und ehrlich antworten.
- ☐ Prompt 4: den Skeptiker spielen lassen.
- ☐ Für die zwei bis vier Gewinner den Sicherheits-Check machen.
- ☐ Erst jetzt den ersten Skill installieren und eine Woche mit einem echten Auftrag testen.
- ☐ Ungenutzte Skills wieder entfernen (mit `npx skills remove` oder in Claude unter Customize > Skills ausschalten).

Mein Tipp

Fang mit dem Projekt an, nicht mit der Rangliste. Wer seine Marke, sein Business und sein Ziel einmal sauber aufgeschrieben hat, merkt bei jedem Skill in Sekunden, ob er passt. Und genau dieses Projekt ist später auch der Bauplan für deinen eigenen Skill, den du Kunden als Feature anbietest.

Alle Links, Befehle und Zahlen habe ich am 27. Oktober 2026 auf den offiziellen Seiten geprüft.



DER MENTOR SAGT

Nicht mehr Skills machen dein Claude stark, sondern die richtigen. Und die richtigen findest nur du, mit deinem Projekt.

— *REPOSPRITES*

Du willst auch so animierte Videos machen? Dann besuch meine Webseite und hol dir die Reel-Engine.

reposprites.com ↗

REPOSPRITES · reposprites.com

SUCHE