

WAS ICH AUS 9 PRODUKTIONSPROJEKTEN MIT CLAUDE CODE GELERNT HABE



Karsten Silz
6. Mai 2026

KI-generiert

Zusammen- fassung

**Coding Agents
nutzen!**

**Hat die KI
recht?**

Nicht-deterministisches, vergessliches
Junior-Team –
schreiben &
testen lassen

Skills automatisieren
Aufgaben, Subagents
schreiben schneller
besseren Code

**KI klaut "Code
schreiben", aber noch
nicht "KI-Teamleiter"**

**Warum
zuhören?**

9 Projekte

Prinzipien

InfoQ

- Erfahrung von 9 Produktions-Projekten mit Claude Code
- Beschreibe Prinzipien, nicht kurzfristige Tipps & Tricks
- Senior Editor im Java Team von InfoQ

Demo

**Folien,
Skills &
Agenten**



<https://atra.software/jx2>

Was ist ein Coding Agent?

Meine 9 Projekte

Produktiver mit Skills & Subagenten

Klaut die KI meinen Job?

Tipps & Tricks

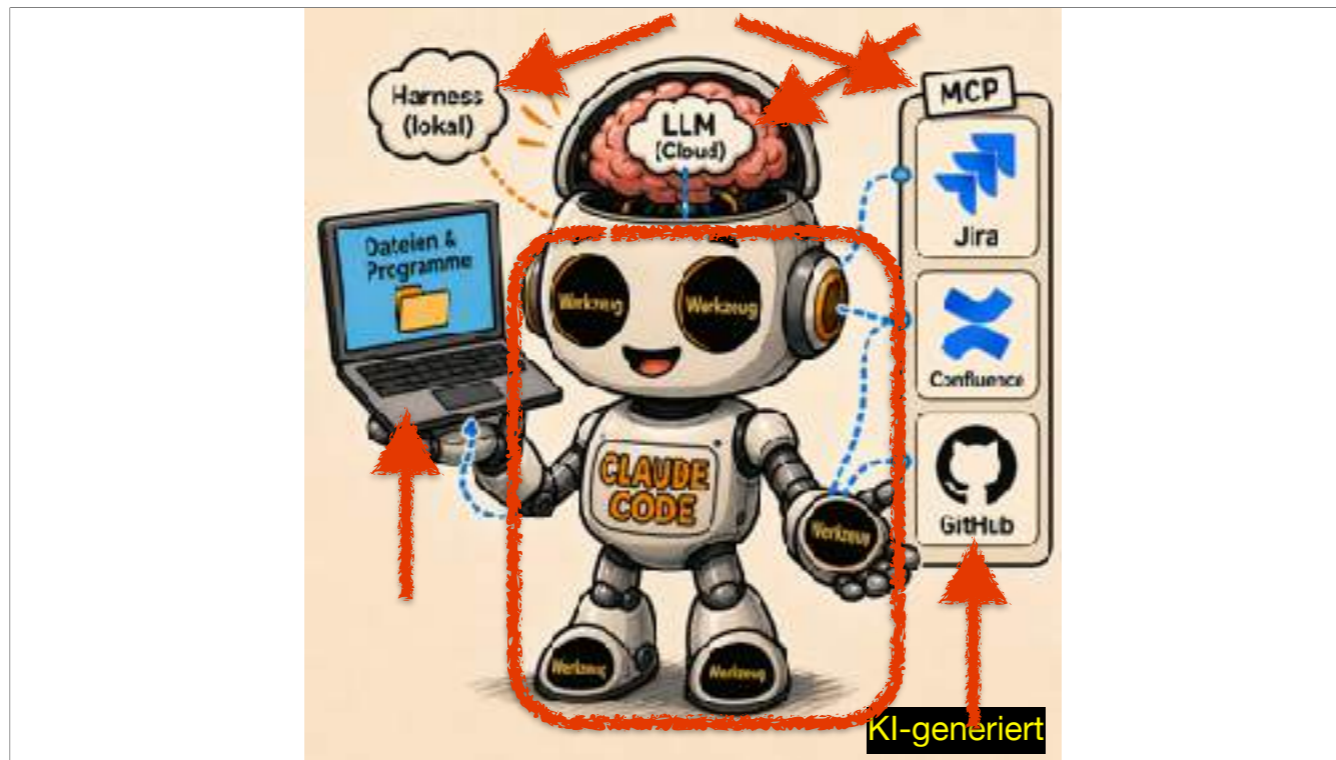
Was ist ein Coding Agent?

Agent

**KI, die
Werkzeuge
nutzt**

Agency

- Agent kommt von "Agency": Handlungsfähigkeit
- Eigene Handlungen und deren Konsequenzen bewusst steuern und beeinflussen



- Zwei Teile des Coding Agents: LLM & Harness
- Lokale Werkzeuge: Dateien, Programme (vor allem im Terminal), Web-Suche
- Werkzeuge im Netz: Model Context Protocol (MCP)

LLM

Text nach Wahrscheinlichkeit

- Autocomplete Engine für Texterzeugung

“Was ist Paris?”

- Beispielfrage

“Die Hauptstadt von Frankreich”

- **Wahrscheinlichste Antwort**

**Kann
programmieren!**

- Viele Fortschritte seit "reine Textvorhersage"

Tokens

- 1 Token = ungefähr 1 Wort

Context Window

- Kurzzeitgedächtnis der LLM
- Alle Werkzeuge, Skills, Agenten, Dateien, Fragen, Antworten, Eingaben, Ausgaben...

Context- Komprimierung

- Context Window voll: Komprimierung von 170k Tokens auf 10
- Möglichst vermeiden, weil die meisten Informationen dann verloren gehen
- Besonders schlimm: Mehrere Context-Komprimierungen hintereinander

CLAUDE .md

GEMINI .md

AGENT .md

- Immer im Context Window
- Wichtige Informationen hier ablegen
- Zu wichtigen Dateien verlinken

Haiku

Sonnet

Opus

LLMs von OpenAI, geordnet nach Fähigkeiten (aufsteigend)
Andere LLMs: siehe Vortrags-Seite

GPT-Codex

Coding-LLM von OpenAPI (ChatGPT)

Gemini Flash

Gemini Pro

Coding-LLM von Google

Harness

Claude Code

Claude Desktop

- Claude Code: Terminal-Tool
- Claude Desktop: Graphisches Tool mit Cowork - "Claude Code für Nicht-Entwickler"

Codex CLI

Codex

- Codex CLI: Terminal-Tool
- Codex: Graphisches Tool

Gemini CLI

- Terminal-Tool

GitHub Copilot Cursor

- Graphische Tools
- Können verschiedene LLMs benutzen

Schwächen

1. Nach dem Mund reden

Sycophancy

2. Machen statt fragen

- Manager würden sagen: "bias for action" 😊

3. Fertig! Nö.

- KI behauptet, fertig zu sein...
- ...ist es aber nicht

4. Halluzinationen

- Falsche Behauptungen
- Beispiel: Claude Code war felsenfest überzeugt, dass Spring Boot 4 Java 21 braucht - ist aber nur Java 17

Coding Agent

- Nachfolgend auch oft nur "KI" genannt

KI-Entwickler- Team

**Analyst,
Programmierer,
Tester, Admin, ...**

Macht Fehler!

**Hat die KI
recht?**

Lösung bekannt:



- Meist bei Features, manchmal bei Bugs
- Dann wissen wir, ob die KI recht hat

Lösung unbekannt



- Meist bei Bugs oder Migrationen
- Manchmal bei Features

***"AI makes people
in the know
more powerful"***

Aus einem Interview mit Benedict Evans - <https://www.ben-evans.com>

**Wie
behandeln?**

Nicht-deterministi-
sches, vergessliches
Junior-Team –
schreiben &
testen lassen

Team



KI-generiert

Produktivität

- Meine Produktivität hängt davon ab, wie viele Agenten ich gleichzeitig wie gut steuern kann

3+

- Mein Ziel: 3 Coding Agents parallel

5

- Der Erfinder von Claude Code hat immer 5 Claude-Code-Instanzen parallel laufen

Spezialisten

- Team hat Spezialisten, die parallel arbeiten
- Front End, Back End, Datenbank, ...

Subagents

- Spezialisten in Cluade Code

Junior

1. Onboarding

- Domäne: Was ist das Fachgebiet?
- Spezifikation: Was macht Anwendung wie?
- Wie bauen, testen, deployen und überwachen?
- Was immer tun - und was nie?
- Zugriff auf Werkzeuge (Tickets, Wiki, Repos, ...)

2. Plan prüfen

- Für kleine & mittlere Aufgaben: Plan
- Für große Aufgaben: erst Spezifikation, dann detaillierter Plan

3. Erfolg?

- Wie testen – und was?
- Wie Produktion überwachen?

**Hat die KI
recht?**

**Nicht
deterministisch**

Software

- Software ist deterministisch
- Gleiche Eingaben = gleiche Ausgaben

LLMs

- LLMs nicht-deterministisch: gleiche Eingabe, andere Ausgabe

Problem

- Wie kriege ich wiederholbare Abläufe hin?

Chance

- Testet unvoreingenommen

Subagenten finden mehr Fehler

- Subagenten reviewen auch Fehler im Ansatz

Vergesslich

Menschen

- Menschen erinnern sich an und lernen von ihrer Arbeit

~~Coding Agents*~~

- Coding Agents lernen nicht von ihrer Arbeit, sehen Code jeden Tag zum ersten Mal
- Claude Code Subagents erinnern sich - Memory.md

Modelle

- Neue Modelle machen Agenten allgemein besser

Aufschreiben!

- Ich muss (fast) alles aufschreiben!

**Schreiben
lassen**

Wer schreibt?

- Wer soll all die Spezifikationen, Pläne und Dokumente schreiben?

Agent!

- Agent schreibt, Mensch prüft

**Testen
lassen**

Wer testet?

- Wer soll all die Back-End- und Front-End-Tests machen?

Agent!

- Front-End-Tests gern mit Playwright & Browser-Steuerung

Zusammen- fassung

LLM & Harness, die Werkzeuge nutzt

**Nicht-deterministi-
sches, vergessliches
Junior-Team –
schreiben &
testen lassen**

Was ist ein Coding Agent?

~~Was ist ein Coding Agent?~~

Meine 9 Projekte

Produktiver mit Skills & Subagenten

Klaut die KI meinen Job?

Tipps & Tricks

Meine 9 Projekte

1. Cloud Microservices in Kotlin

Bug Fixes + Features

**Geteilte
Custom Skills &
Subagents**



2-4. Katzenhüter- SAAS

- SAAS = **S**oftware **a**s **a** **S**ervice
- Mitgründer eines Startups in England, wo ich 5 Jahre lebte

- 2. Spring Boot, Java,
PostgresSQL**
- 3. Angular Web**
- 4. Flutter nativ
iOS & Android**

- Web-App für Manager
- Mobile App für Teammitglieder (in englischen App Stores)

Spring Boot 2 zu 4

Angular 13 zu 20

Flutter 3.29 zu 3.41

Observability

- Fehler-Reports mit Sentry (Angular, Java, Flutter)
- Server-Alarme mit New Relic
- Logs: Spring Boot, Java Flight Recorder, Java Garbage Collection, Docker, DB, System

Bug Fixes & Features



5. Slide Searcher (neu)

Sucht nach Vortrags- Seiten ohne Folien

- Für eine Webseite

**Spring Boot, Java,
PostgreSQL
Thymeleaf
Lokale LLM**



- Overkill - wäre schneller gewesen, Namen aus Bilder zu lesen, statt das per LLM zu machen

6. Spesen-Rechner (neu)

**Spring Boot, Java,
PostgreSQL
Thymeleaf
Lokale LLM**



7. Webseiten- Konsolidierung

**Freelancer-Seite +
Lern-Seite =
GmbH-Seite**

**Lern-Seiten-Links
funktionieren noch**

Hugo Netlify

- Hugo: Statischer Seiten-Generator
- Netlify: Webseiten-Hosting



8-9. Marketing- Webseiten (neu)

Ferienwohnungen Nachhilfe

Astro Cloudflare

- Astro: Statischer Seiten-Generator
- Cloudflare: Webseiten-Hosting



**10-11. Noch 'nen
SAAS
(neu, nicht in PROD)**

Marketing-Seiten

Kundenportal

- Back End erzeugt Kundenseiten aus Template-Projekten in Git und veröffentlicht sie auf Hosting Platform Cloudflare

**Spring Boot, Kotlin,
PostgreSQL
Angular
Cloudflare**



Zusammen- fassung

**Java, Kotlin, Spring
Boot, PostgreSQL,
Angular, Thymeleaf,
Flutter, Hugo, Astro,
Netlify, Cloudflare**

Meine 9 Projekte

~~Was ist ein Coding Agent?~~

~~Meine 9 Projekte~~

Produktiver mit Skills & Subagenten

Klaut die KI meinen Job?

Tipps & Tricks

**Produktiver
mit Skills &
Subagenten**

Custom Skills

Automatisierte Workflows

**Aufruf wie
Claude-Skills
/clear**

Harness

Teil der Harness (Claude Code / Claude Enterprise), nicht der LLM

Prompts

Skripts/Programme

Prompts: nicht-deterministisch
Skripts/Programme: deterministisch
Im .claude/skills-Ordner

Global Projekt

Global: in ~/.claude/skills

Projekt: in .claude/skills

`/review`

Reviewet Pull Requests oder lokalen Code

/plan-and-do

Vom Ticket/freier Aufgabe zum PR

Erzeugt Branch, schreibt optional Anforderungen & immer einen Plan, implementiert & testet
Außerdem Code Review & PR

/merge-branch

Fetcht main branch und simuliert Merging
Hilft bei Konflikten
Macht push & pull

`/list-my-prs`

Zeigt Status meiner eigenen PRs und den von den PRs, die ich kommentiert habe

Projekt- Konfiguration: CLAUDE.md

Zum Beispiel: Test-Kommando und Liste der Subagenten

Skill Engineering

/plan-and-do
Write me a skill that
does...

Warum /plan-and-do? Nutzt Subagenten, hat Reviews, macht Git Branch & PR

skill-writer
skill-reviewer
skill-creator

/plan-and-do nutzt automatisch Agents, die

Tribe-Projekt hat Subagents für Skills: skill-writer & skill-reviewer

Außerdem empfohlen: skill-creator Skill von Anthropic: <https://claude.com/plugins/skill-creator>

/plan-and-do
Change/fix skill..

Installation

Global: Per Skript

Projekt: Sind automatisch in `.claude/skills`

Fehler/Änderung?

**Ich editiere Skills
nicht direkt**

Claude Code macht das für mich

Nicht- deterministisch?

LLMs sind so



Skills sind ein gutes Gerüst für Nondeterminismus



Nur selten machen sie Mist: /p1an-and-do ignoriert dann Prompts und läuft einfach durch

**Auch nur
Software**

DRY

Don't repeat yourself
Reuse software

/plan-and-do

- /plan-and-do ruft /review auf

Prompts für /plan-and-do

- Zum Beispiel Installations von Subagenten

Subagents

KI-Spezialisten

Normale LLM

Eher Personas

ba-writer

admin

db-coder

ui-designer

be-coder

be-test-coder

fe-coder

fe-test-coder

- ba: Business Analyst
- be: Back End
- fe: Front End

**Schneller
besseren Code**

1. Eigenes Context Window

2. Laufen parallel

3. Regeln anpassbar

Mehr Regeln als in CLAUDE.md
Höhere Wahrscheinlichkeit, dass Regeln beachtet werde

4. Reviewer- Subagents

Finden mehr Fehler

ba-writer

admin

db-coder

ui-designer

be-coder

be-test-coder

fe-coder

be-test-coder

ba-reviewer

db-reviewer

be-reviewer

fe-reviewer

ui-reviewer

be-test-reviewer

be-test-reviewer

```
* Phase 4 requires SQL safety guards and Java changes, let us launch db-coder and spring-coder in parallel.
* 4 agents finished (still to expand)
├─ db-coder (Phase 4 SQL safety guards) - 7 tool uses + 29.8k tokens
│   └─ Done
├─ spring-coder (Phase 4 Java safety guards) - 34 tool uses + 63.8k tokens
│   └─ Done
```

1. db-coder und spring-coder arbeiten parallel

```
* Phase 4 requires SQL safety guards and Java changes, let us launch db-coder and spring-coder in parallel.

* 4 agents finished (ctrl-c to expand)
├─ db-coder (Phase 4 SQL safety guards) - 7 tool uses + 29.8k tokens
│   └─ Done
├─ spring-coder (Phase 4 Java safety guards) - 34 tool uses + 43.8k tokens
│   └─ Done
└─ Done

* total agents completed, let us verify compilation.

* Executing gradle compileJava compileTestJava D-8E | tail -20
├─ Task :compileJava UP-TO-DATE
├─ Task :bootBuildInfo UP-TO-DATE
├─ Task :generateGitProperties
└─ * 8 lines (ctrl-c to expand)

* Compiler successfully, let us run the reviewers in parallel. 2
```

2. Reviewer-Subagents werden gestartet

```
* Phase 4 requires SQL safety guards and Java changes, let us launch db-coder and spring-coder in parallel.
* 4 agents finished (ctrl-c to expand)
├─ db-coder (Phase 4 SQL safety guards) - 7 tool uses - 29.8k tokens
│  └─ Done
├─ spring-coder (Phase 4 Java safety guards) - 34 tool uses - 63.8k tokens
│  └─ Done
└─ Done

* total agents completed, let me verify compilation.

* head(gradle compileJava compileTestJava D-4E | tail -20)
├─ Task :compileJava UP-TO-DATE
├─ Task :bootBuildInfo UP-TO-DATE
├─ Task :generateGitProperties
├─ * 8 lines (ctrl-c to expand)
└─ Done

* Compiler successfully, let me run the reviewers in parallel.

* 2 agents finished (ctrl-c to expand)
├─ db-reviewer (Review Phase 4 SQL changes) - 2 tool uses - 17.4k tokens
│  └─ Done
├─ spring-reviewer (Review Phase 4 Java changes) - 19 tool uses - 34.1k tokens
│  └─ Done
└─ Done

* Phase 4 Review Results

CRITICAL Issues (Must fix)



| # | Issue                                                                        | Location               |
|---|------------------------------------------------------------------------------|------------------------|
| 1 | SQL error: last.modified.date referenced in WHERE but not in subquery SELECT | All 3 @jection methods |



The SQL will fail with:
ERROR: column "last.modified.date" does not exist

Fix: Add s.last.modified.date to subquery SELECT list.
```

3. db-reviewer und spring-reviewer haben Fehler gefunden

Konfiguration

Direkt im Agent

Agent Engineering

Direkt editieren

Produktivität

Effizienz

- Eine Form der Produktivitäts-Steigerung
- Prozess-Optimierung
- "Dinge richtig machen"

**/plan-and-do
für Bug-Fixing**

Einfach

Manuelle Schritte automatisieren

3-4x
produktiver

Produktivitäts-Gewinn beschränkt - geschätzt!

Nicht-optimalen Prozess zu automatisieren ändert nicht "nicht-optimal"

Effektivität

- Weitere Form der Produktivitäts-Steigerung
- Verändert Prozesse
- "Die richtigen Dinge machen"

Schwerer

Prozesse ändern

10x
produktiver

Mehr Produktivitäts-Gewinn - geschätzt!

Beispiel: Bug Fixing

Team/Agent:
analysieren,
schreiben, fixen

Team bespricht Bugs im Refinement

- 1. Jeder soll Bugs
fixen können**
- 2. Bugs fixen ist teuer
=> priorisieren**

Gründe für aktuellen Prozess

1.
Nur Coding Agents
arbeiten an Bugs

Neuer Ansatz

Agent:
analysieren,
schreiben, fixen

~~Refinement~~

Nicht mehr nötig...

**1. Jeder soll Bugs
fixen können**

**2. Bugs fixen ist teuer
=> priorisieren**

- 1. Jeder soll Bugs fixen können**
- 2. Bugs fixen ist teuer
=> priorisieren**

...weil Gründe entfallen

Mensch wählt Bugs aus

Mensch wählt nur noch Bugs zum Fixen aus

Effektiver!

Neuer, besserer Prozess!

10x
produktiver?

Weiß nicht, ob man das so rechnen kann

2. Mehr Quellen

Noch ein neuer Ansatz

Automatisch Alarme, Logs für DB/Garbage Collection/JFR, ...

- Datenbanken-Logs: vermehren langsame Abfragen & ineffiziente Strukturen
- Garbage Collection Logs: Protokolliert Speicherverbrauch von unseren Back-End-Services
- JFR: Java Flight Recorder - die "24-Stunden Black Box" der JVM

Coding Agents: Analyse 👍

Kann Zusammenhänge über alle Logs & Alarme hinweg herstellen

Coding Agents finden mehr Bugs

Kunden melden nicht immer alle Fehler

20x
produktiver?

Weiß nicht, ob man das so rechnen kann

Zusammen- fassung

**Skills automatisieren
Aufgaben, Subagents
schreiben schneller
beseren Code**

**Effizienz gut,
Effektivität besser**

**Produktiver
mit Skills &
Subagenten**

~~Was ist ein Coding Agent?~~

~~Meine 9 Projekte~~

~~Produktiver mit Skills & Subagenten~~

Klaut die KI meinen Job?

Tipps & Tricks

**Klaut die KI
meinen Job?**

Automati- sierung

**KI = Produktivitäts-
Erhöhung**

Automatisie- rungs-Frage

Wieder von Benedict Evans – <https://www.ben-evans.com>

Entwickler 

Software =

Entwickler 

Software 

Entwickler 

Software 

**Bisher:
Letzteres**

Für die Software-Entwicklung

Jevons- Paradoxon

"Wenn technischer Fortschritt die effizientere Nutzung eines Rohstoffes erlaubt, steigt der Rohstoff-Verbrauch an."

Mein Job?

1. Business- Sicht

Die unser Gehalt bezahlen

Business Value

Business Sicht
Bugs fixen oder Features hinzufügen

Code-Schreiben Nebensache!

Dem Business ist es egal, wie wir Entwickler zum Ziel kommen



Ob da jetzt ein Entwickler den Business Value erzeugt...



KI-generiert

...oder die KI, ist dem Business vollkommen schnurzpiepegal. Zu Recht!



Da könnte auch ein Affe sitzen! Deswegen heißen Entwickler im Englischen oft auch "Coding Monkeys".

2. Entwickler- Sicht

Siehe <https://blog.lmorchard.com/2026/03/11/grief-and-the-ai-split/>

2.1 Code schreiben

Programmieren als Puzzle lösen, als Kampf mit dem Compiler und dem Linter

Klaut die KI! *(für die meisten)*

Für Enterprise-Entwickler - Vorhersage mit 75% Wahrscheinlichkeit

Für Nischen vielleicht nicht - wie Röntgengeräte bei Siemens oder Auto-Software beim Daimler

2.2 Computer Dinge tun lassen

Programmieren als Weg zum Ziel

**Klaut die KI
noch nicht...**

KI macht aktuell noch zu viele Fehler

**...aber wie
lange?**



- Zum Beispiel 2029: KI macht meist alles allein
- Nur noch "Entwickler auf Abruf", wie Monteure in vollautomatisierter Fabrik

Trends

Coding Agents



Umsatzstärkste KI-Anwendung

Software, die Software schreibt

Umsatzstärkste KI-Anwendung
Coding Agents with Claude Code

"Annual Run Rate" für Anthropic (Milliarden US\$)

Aktueller Umsatz unverändert fortgeschrieben auf das nächste Jahr
Zum Beispiel aktueller Monatsumsatz x 12
Anthropic: Claude Code, Claude, LLMs

März 2026: 19 Mrd.

April 2026: 30 Mrd.

+58%

In einem Monat Umsatz um 58% gesteigert

Software, die Software schreibt

Umsatzstärkste KI-Anwendung

Software, die Software schreibt, die Software schreibt

Coding Agents schreiben sich selbst
Tempo des Fortschritts direkt proportional
Einfacher zu skalieren als "Neue Entwickler einstellen"

Beispiellose Geschwindigkeit!

Hoffnungschimmer für "Team Human"?

Für Menschen

Börsengang

Bei Anthropic & OpenAI geplant für dieses Jahr

Quartals- berichte

Mit Gewinnen und Verlusten

Werden Coding Agents teurer?

Bei börsennotierten Unternehmen wächst Druck, profitabel zu werden
Außerdem: Preiserhöhung durch Oligopol

Zusammen- fassung

**KI klaut "Code
schreiben", aber noch
nicht "KI-Teamleiter"**

**Klaut die KI
meinen Job?**

~~Was ist ein Coding Agent?~~

~~Meine 9 Projekte~~

~~Produktiver mit Skills & Subagenten~~

~~Klaut die KI meinen Job?~~

Tipps und Tricks

Tipps und Tricks

Wie gut ist KI?

- Applikationen ohne Programmierer bauen?

**KI oft
beeindruckend...**

Kann ganze Features auf Anhieb richtig bauen

**...und bleibt dann
hängen**

Bleibt oft bei kleinen Sachen hängen

**Macht kaputt, was
schon mal lief**

Das ist das frustrierendste für mich

**Hat die KI
recht?**

Test schreiben lassen

Vom Coding Agents

Mensch prüft

Zum Beispiel PR Review

Regression Tests

Coding Agents

Umgang

Offene Fragen

KI will Menschen bestätigen - nicht eigene Meinung bestätigen lassen

Nachfragen

"Erklär mir..."

"Warum hast Du..."

Spezifikationen

Siehe <https://martinfowler.com/articles/exploring-gen-ai/sdd-3-tools.html>

Spec-first

Spec-anchored

Spec-as-source

Spec-first: A well thought-out spec is written first, and then used in the AI-assisted development workflow for the task at hand.

Spec-anchored: The spec is kept even after the task is complete, to continue using it for evolution and maintenance of the respective feature.

Spec-as-source: The spec is the main source file over time, and only the spec is edited by the human, the human never touches the code.

**Advantage:
Back End**

Back-End Devs bauen Front End



Einfacher, weil Front End leichter zu prüfen: Funktioniert es?
Fehler haben geringere Auswirkung

Front-End Devs bauen Back End



Schwere, weil Back End schwerer zu prüfen: Funktioniert es?
Fehler haben schwere Auswirkungen

Business programmiert?

- Applikationen ohne Programmierer bauen?

Nicht allein...

KI macht zu viele Fehler

Kann technische Entscheidungen nicht beurteilen

...aber vieles

Viele Änderungen sind klein genug, dass sie keine Programmierer-Beurteilung brauchen

Tipps und Tricks

~~Was ist ein Coding Agent?~~

~~Meine 9 Projekte~~

~~Produktiver mit Skills & Subagenten~~

~~Klaut die KI meinen Job?~~

~~Tipps und Tricks~~

Zusammen- fassung

**Coding Agents
nutzen!**

**Hat die KI
recht?**

Skills automatisieren
Aufgaben, Subagents
schreiben schneller
besseren Code

**Nicht-deterministi-
sches, vergessliches
Junior-Team –
schreiben &
testen lassen**

**KI klaut "Code
schreiben", aber noch
nicht "KI-Teamleiter"**



**Folien,
Skills &
Agenten**



<https://atra.software/jx2>