Agentic Coding

Der nächste Schritt in der KI-gestützten Softwareentwicklung

Die Ausgangssituation: Viele Entwickler nutzen bereits Large Language Models, um Code zu generieren – meist durch Copy-Paste zwischen Browser und IDE. Doch dieser Workflow hat fundamentale Grenzen.

> **Das Problem mit klassischem Prompting:**
>
> Das LLM sieht nur isolierte Code-Snippets, nicht Ihr Projekt. Es kennt weder die Architektur noch die Dependencies. Es kann nicht testen, ob der generierte Code funktioniert. Jede Integration erfordert manuelle Arbeit – und bei Fehlern beginnt der Zyklus von vorne.

Agentic Coding löst diese Probleme grundlegend: Statt passiv auf einzelne Prompts zu reagieren, arbeiten KI-Agents **aktiv und autonom** an Ihrem Projekt. Sie haben Zugriff auf alle Dateien, können Code ausführen, testen und iterieren selbständig bis zur funktionierenden Lösung.

Der direkte Vergleich

Klassisches Prompting	Agentic Coding
✗ Nur isolierte Code-Snippets	✓ Vollständiger Projektkontext
✗ Manuelle Integration	✓ Autonome Code-Organisation
✗ Kein Testing möglich	✓ Automatische Ausführung & Tests
✗ Copy-Paste-Workflow	✓ Nahtlose Integration
✗ Keine Iteration bei Fehlern	✓ Selbstständige Fehlerkorrektur

Die fünf Kernvorteile

	**Vollständiger Projektkontext** – Kennt alle Dateien, Dependencies und Architektur
	**Autonome Organisation** – Verteilt Code sinnvoll auf mehrere Dateien
	**Automatisches Testing** – Führt Tests aus und erkennt Fehler selbstständig
	**Iterative Verbesserung** – Korrigiert Fehler bis zur funktionierenden Lösung
	**Git-Integration** – Erstellt Commits, Branches und dokumentiert automatisch

Praxisbeispiel: Neue API-Route hinzufügen

Aufgabe: Eine neue REST-API-Route mit Datenbankanbindung implementieren.

Was der Agent automatisch erledigt:

1. Analysiert bestehende Routen und deren Struktur
2. Erstellt src/routes/new_endpoint.py nach bestehendem Muster
3. Schreibt Tests in tests/test_new_endpoint.py
4. Führt Tests aus → erkennt Fehler in der Datenbank-Query
5. Korrigiert den Fehler selbstständig
6. Verifiziert: Alle Tests grün ✓
7. Aktualisiert Imports und Dokumentation

Ergebnis: Review-fertige Implementierung ohne manuellen Copy-Paste-Zyklus.

*Wie Sie bessere Ergebnisse durch effektives Kontextmanagement erzielen, erläutern wir in unserem Blogpost KI-Kontext und Kontextmanagement in Claude Code

Bewertung der Vorteile

Vorteil	Bewertung	Impact
Vollständiger Projektkontext	★★★	Reduziert Integrationsfehler drastisch
Autonome Code-Organisation	★★★	Spart Zeit, saubere Struktur
Code-Ausführung & Testing	★★★	Erhöht Code-Qualität des Modells
Automatisches Test-Schreiben	★★☆	Guter Startpunkt, erfordert Review
Iterative Verbesserung	★★★	Reduziert Trial-and-Error-Zyklen

Wann eignet sich Agentic Coding?

✓ Ideal geeignet	⚠ Mit Vorsicht
• Feature-Entwicklung mit klaren Anforderungen • Bug-Fixes mit reproduzierbaren Test-Cases • Refactoring und Code-Cleanup • Hinzufügen von Tests zu bestehendem Code • Boilerplate-Code für neue Module	• Initiales Architektur-Design • Komplexe domänenspezifische Algorithmen • Security-kritische Komponenten • Strategische Technologie-Entscheidungen *→ Hier bleibt menschliche Expertise unverzichtbar*

Grenzen und Empfehlungen

Agentic Coding ist ein mächtiges Werkzeug, aber **kein** Ersatz für Entwickler-Expertise.
Architektur-Entscheidungen, komplexe Geschäftslogik und sicherheitskritische Komponenten erfordern weiterhin menschliches Urteilsvermögen und sorgfältiges Code-Review.

Unsere Empfehlungen für den erfolgreichen Einsatz:

✓ Klare Anforderungen formulieren – Je präziser die Aufgabe, desto besser das Ergebnis
✓ Aktiv reviewen, nicht blind vertrauen – Der Agent ist ein Werkzeug, keine Blackbox
✓ Tests kritisch prüfen – Auch generierte Tests müssen das Richtige testen
✓ Iterativ vorgehen – Komplexe Features in kleinere Aufgaben aufteilen

Fazit

Agentic Coding stellt einen Paradigmenwechsel dar: Vom passiven Assistenten zum aktiven Mitarbeiter, der Ihr Projekt kennt, selbständig arbeitet und iteriert. Bei korrekter Nutzung sind Produktivitätsgewinne spürbar – bei gleichzeitiger Chance auf verbesserte Codequalität durch automatisiertes Testing.

Haben wir Ihr Interesse geweckt?

Wir unterstützen Sie bei der Einführung von Agentic Coding in Ihrem Unternehmen:

• Evaluierung geeigneter Tools & Modelle
• Pilotprojekte mit begleitendem Coaching
• Best Practices und Workflow-Integration

Kontaktieren Sie uns für ein unverbindliches Beratungsgespräch.
Nutzen Sie unser Kontaktformular unter https://42ways.de/contact/
oder direkt eine E-Mail an info@42ways.de

Wir freuen uns auf ein Gespräch!