Stand heute

Frage am Anfang: wer nutzt es schon?

Was ist ein Coding Assistant?

Warum gibt es davon 2 Varianten (lokal/zentral)?

Was ist diese Inferenz?

Warum brauchen wir mehrere Inferenz-Services?

Wie installiere ich den Coding Assistant?

Wo finde ich die Dokumentation?

Konzepte: Prompt + Context

Context extrem wichtig! 3 Möglichkeiten den zu setzen: a) implizit b) command-L c) Context-Provider

Anwendung

Wir nähern uns dem Code:

Dateien verstehen

@README.MD Was macht dieses Projekt fachlich? @README.MD Wie ist das Projekt technisch aufgebaut?

Codebase verstehen

@Codebase Welcher Service verfwaltet im Frontend die todos?

Codeänderungen verstehen

@order by Was wurde durch diesen Commit verändert? @fehler behoben Welchen Fehler hat der Kollege mit dem Commit behoben?

Coder verbessern

Code Completion: sort-Logik ändern

Edit: Dokumentation einfügen

dokumentiere die sortTodosAlphabetically Methode

kleine Verbesserungen

@todo.service.ts Ich habe nur 5 Minuten Zeit. Was wäre die wichtigsten Optimierung an diesem Code?

größere Verbesserungen über mehrere Dateien

Context: alle Files von todo-item.component Beim Verlassen des Eingabe-Felds für neue Todos wird das Todo nicht noch gespeichert. Woran liegt das? Schlage mir eine Implementierung vor.

Refactoring

@todo.service.ts Prüfe auf ausdrucksstarke Namen und mache ggf Verbesserungsvorschläge

Qualität durch Tests verbessern

todo.service service und spec im Context Schreibe einen test für die sortier-logik

Routineaufgabe: Commit Message definieren

@Git Diff Fasse mir bitte zusammen, welche Änderungen ich seit dem letzten Commit gemacht habe. Erstelle eine Tabelle mit den namen der geänderten Dateien und ihren Änderungen. Generiere eine Commit-Message aus einem einzigen Satz und befolge dabei den Standard für Conventional Commits.

Fehler-Analyse:

@Terminal wie groß ist die main.js @Terminal analysiere den comoile fehler @todo.service.ts Meine Todos werden nicht mehr richtig sortiert dargestellt. Woran kann das liegen?

Konzept: Context is king

Chat: Refactor eines Service ohne Context

Chat: Refactor eines Service mit gesetztem Context

Ausblick 2026 ff

Wechsel auf Privat-Rechner

Konzept: ToolUse + MCP

agenticAi erst 2026

Aufhänger Context automatisch füllen

Beispiel: richtige Dateien finden - erkläre mir, warum ich mal positive und mal negativer Ids sehe

Lösung für das technische Brownfield-Problem

Aufhänger Aktionen

Beispiel: Atlassian MCP Server

https://github.com/sooperset/mcp-atlassian

Konzept: agentic loop

claude code - sort feature naiv umsetzen

Konzept: Orchestrator + SubAgents

Problem: Context bei sehr langen Läufen zu klein

Konzept: Rollen

Problem: SWE ist i.d.r. technisch Brownfield + inkkrementell und methodisch brownfield

Lösung: agentic ai + commands + rollen

Konzept: Dev-by-spec statt vibe coding im bisherigen Sinn

Gemeinsam Lernen

Community