

Meetup  
14.04.2026

## **Agent-SKILLs**

...mit Opencode

Oliver Welz (ScaDS.AI)



# Worum geht's?

*Agent-SKILLS* –

was das ist,  
**was** man damit so anstellen  
kann und **wie** man eigens **Skills**  
erstellt.





**Kurze Abfrage:**


Wer **nutzt** bereits  
*- Agent-SKILLS -*  
im eigenen Setup?



Definition

...***Agent-SKILLS***





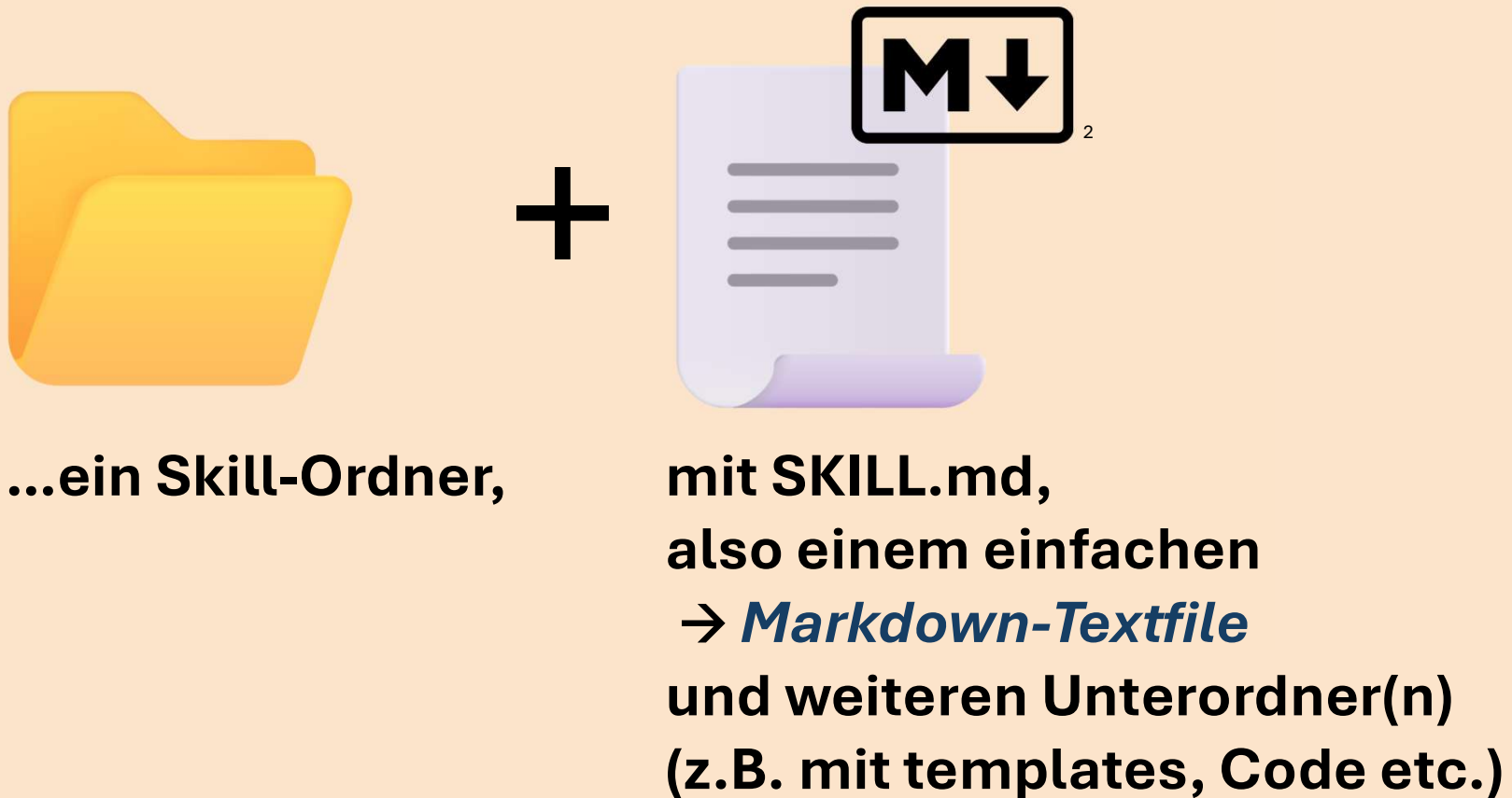
## Was sind Agent- SKILLS?

- Agent Skills<sup>1</sup> sind (**kleine**), **wiederverwendbare Fähigkeiten** für KI-Agenten.
- Sie beschreiben, **wie ein Agent eine bestimmte Aufgabe lösen soll**.
- Typisch ist: Ein Skill enthält **Anweisungen, Regeln, Beispiele** und bei Bedarf auch **Templates oder Skripte**.
- So lassen sich Aufgaben **konsistenter, verständlicher und wiederholbar** bearbeiten.

Stellt es euch wie eine Art Drehbuch für einzelne Aufgaben oder Arbeitsabläufe vor!

*...kurz & knapp: „**Agent Skills sind wiederverwendbare Anleitungs- und Arbeitsbausteine, mit denen KI-Agenten gezielt für bestimmte Aufgaben erweitert werden.**“*

... ***vereinfacht also:***





## Was sind die *Vorteile, Nutzen*

...und gibt es **Nachteile?**

### **Vorteile:**

#### **Effizienz & Zeitersparnis**

Wiederkehrende Aufgaben müssen nicht jedes Mal neu erklärt werden. Setup ist grundlegend ziemlich einfach.

#### **Konsistenz & Qualität**

Ergebnisse folgen klaren Regeln und bleiben nachvollziehbar.

#### **Wiederverwendbarkeit**

Einmal erstellte Skills können flexibel in vielen Kontexten genutzt werden.

#### **Zugänglichkeit von Wissen**

Komplexe Prozesse werden einfacher nutzbar – auch für Nicht-Expert:innen.



## Was sind die *Vorteile, Nutzen*

...und gibt es **Nachteile**?

### **Nachteile:**

#### **Initialer Aufwand**

Gute Skills zu erstellen erfordert Zeit und Struktur. (Eigene) Arbeitsabläufe müssen klar und in einem Ablauf fassbar sein

#### **Begrenzte Flexibilität**

Sehr spezifische Skills können bei neuen oder unerwarteten Aufgaben an Grenzen stoßen.

#### **Sicherheitsrisiken**

Prompt Injection, Datenabfluss, zu viele Berechtigungen

#### **Governance nötig**

Insb. bei externen Skills, eingebundenen Tools; Quellen müssen geprüft werden





Wie sieht so  
ein SKILL im  
Detail aus,

...und gibt es **Template(s)**?

## Ordnerstruktur:

```
mein-skill/  
├─ SKILL.md      ← Pflicht: Anweisungen + Metadaten  
├─ references/   ← Optional: Zusätzliche Dokumente  
└─ assets/       ← Optional: Vorlagen, Beispieldateien
```

## SKILL.md:

```
---  
name: your-skill-name  
description: Beschreibe hier, was der Skill tut und wann er verwendet werden soll. Nenne am  
---  
  
# Titel des Skills  
  
## Wann dieser Skill verwendet werden soll  
- Auslöser 1  
- Auslöser 2  
  
## Voraussetzungen  
- Anforderung 1  
- Anforderung 2  
  
## Schritt-für-Schritt-Ablauf  
1. Erster Schritt  
2. Zweiter Schritt  
3. Dritter Schritt
```



Wie sieht so  
ein SKILL im  
Detail aus,

...und gibt es **Template(s)**?

### SKILL.md:

...oft hilft noch sowas

```
## Checkliste zur Validierung
- Die Ausgabe enthält alle erforderlichen Abschnitte
- Benennung und Struktur entsprechen den Projektregeln

## Fehlerbehebung
- Wenn X passiert, dann tue Y
- Wenn Eingaben fehlen, frage nach Z
```

Und man kann Code, Templates, Beispiel, Abläufe  
uvm. **referenzieren**.



# Best Practice...

...wie **baut** man einen *guten SKILL*?

## Step-by-step: Wie baut man einen Skill?

### 1. Aufgabe auswählen

Wähle einen klaren, wiederkehrenden Ablauf mit erkennbarem Ziel.

#### ➤ Scope klein halten

Ein guter Skill macht **eine Sache gut**, statt alles auf einmal zu wollen.

### 2. Ablauf erst manuell durchspielen

Notiere die einzelnen Schritte so, wie du sie selbst ausführen würdest bzw. chatte Schritt-für-Schritt.

### 3. Ziel und Ergebnis festlegen

Beschreibe, **was der Skill tun soll** und wie ein gutes Ergebnis aussieht.

### 4. Schritte in Anweisungen übersetzen

Formuliere daraus eine klare, wiederholbare Anleitung mit und für den Agenten.

### 5. Inputs, Beispiele und Grenzen definieren

Was braucht der Skill? Was darf er tun – und was nicht?

→ Teste den SKILL anschließend

# Woher bekommt man SKILLS...

...und **was** gibt's zu  
**BEACHTEN?**

## Woher bekommt man Skills?

### „No-Brainer“: Selbst erstellen


Am sichersten sind Skills, die man selbst schreibt, versteht und testet.

### Aus vertrauenswürdigen Sammlungen

Z. B. aus eurer kuratierten Quelle wie **skills.sh** oder aus (eigens od. extern) geprüften Team-Repositories.

### Aus der Community / von Dritten

Nur sinnvoll, wenn Herkunft, Inhalt und Rechte sauber geprüft wurden. **Risiken** in der **KI-Lieferkette** sind real.



## Woher bekommt man SKILLS...

...und **was** gibt's zu  
**BEACHTEN?**

### Worauf sollte man bei fremden Skills achten?

#### **Skill wirklich lesen und verstehen**

Nicht blind übernehmen: prüfen, was der Skill tut, wann er greift und welche Tools er nutzt.

#### **Berechtigungen klein halten**

Web, Bash, Edit, externe Verzeichnisse oder Subagents nur mit Kontrolle freigeben.

#### **Auf Prompt Injection achten**

Fremde Inhalte, Webseiten oder Dateien können Anweisungen enthalten, die den Agenten manipulieren.

#### **Supply-Chain-Risiken bedenken**

Externe Skills, Vorlagen und Abhängigkeiten können unsicher, veraltet oder manipuliert sein.

#### **Keine Secrets oder sensiblen Daten blind weitergeben**

Ein Skill sollte keine Zugangsdaten, internen Dateien oder vertraulichen Inhalte unnötig verarbeiten.

*...bis hierhin  
klar?*

**Gibt es Fragen  
bis hier her?**





# Kleine Demo

**...repeating is *learnding***

**„...nun aber mal  
eigens testen!“**

...was wir für **heute** vorbereitet  
haben







# Open Code

## Was ist OpenCode?

**Open-Source AI Coding Agent:** OpenCode ist ein KI-gestütztes Coding-Tool für Terminal und Editor, das Entwickler direkt beim Schreiben, Verstehen und Refaktorisieren von Code unterstützt.

**Offene Alternative zu proprietären Tools:** Es versteht sich als transparente, freie Alternative zu kommerziellen Lösungen wie Cursor oder Copilot.

**Flexible LLM-Integration:** Unterstützt verschiedene Modelle – lokal (z. B. über Ollama/vLLM) oder über Cloud-Provider (OpenAI, Claude, Gemini usw.).

**State-of-the-Art Agent Harness:** profitiert von den neuesten Funktionalitäten rund um (coding) Agenten mit: MCP, Skills, etc.

Link für weitere Infos: <https://opencode.ai/>



# Repo- sitory

## GitHub Repository für heute:

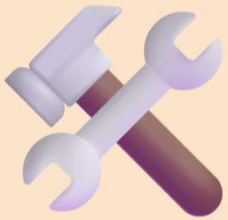
***meetup-agent-skills:***

→ <https://github.com/ScaDSAILLLE/meetup-agent-skills>



## Hier findet ihr alles:

- Vom **Setup und Nutzung von Opencode**, falls ihr es noch nie benutzt habt.
- Die **Einführung zu Agent-Skills** sowie alle nötigen Grundlagen.
- Einen **3-teiligen Workshop** je nach Vorkenntnissen:
  - /basics
  - /midlevel (skills nutzen für *office-tasks*)
  - /highlevel (wenn ihr z.B. skills schon nutzt oder die erweiterten Möglichkeiten sowie den „neuesten Schrei“ mal testen wollt)
- Teilt das Repository gerne und nehmt euch die Inhalte mit!



Wo finde  
ich **was**?

- <https://shorturl.at/EiR9i> ...gelangt ihr zum **GitHub Repository** mit weiteren Infos zu den Basics, fortgeschrittenen Skills, Beispielen, Setup etc.


*(Disclaimer: Installation und Nutzung der hier genannten Software natürlich wie immer auf eigene Gefahr.)*



QR zum GitHub Repo



...nochmal  
zum **Ablauf**

- 1 ... Findet euch in alleine oder in Teams zusammen
  - 2 ... Macht euch mit dem Content vertraut
  - 3 ... Probiert es aus, arbeitet an euren Ideen
  - 4 ... Skills, SKILLs, Skills
  - 5 ... teilt gerne nebenbei oder am Ende Eure Erfahrungen
-  ... Spaß haben, was lernen, Skills mitnehmen

# Legen wir los!



*Lets get practical, practical...*



<https://github.com/ScaDSAILLLE/meetup-agent-skills>