ANGULAR ARCHITECTS

JavaScript & Angular Days by entwickler.de

https://LXT.dev

Refactoring verstaubter Angular-Komponenten

mithilfe von KI ☺

Alexander Thalhammer · ANGULARarchitects.io

0.1 Das Problem: Warum viele Angular-Komponenten zu Monstern werden

ANGULAR ARCHITECTS

- Über Jahre kommen Features, Sonderfälle und Workarounds hinzu
- Verantwortlichkeiten verschwimmen zwischen UI, State und Business-Logik
- Code-Lesbarkeit sinkt und Änderungen werden riskanter
- Teams (& KI) gewöhnen sich an schlechten Code statt ihn zu hinterfragen

0.2 Typische Symptome verstaubter Komponenten

ANGULAR ARCHITECTS

- Alte Angular Patterns, die nicht modernisiert wurden (*ngIf, @Input(), ngClass …)
- 400 bis 800 (oder noch mehr) LoC in einer einzelnen Datei
- Viele Lifecycle Hooks (ngOnChanges) und schwer nachvollziehbare Seiteneffekte
- Unklare Symbol-Benennung, AI-generierter Code und fehlende Struktur
- Komplexe View-Templates (.html) mit zu viel Logik
- Unaufgeräumte Stylesheets (.scss) mit viel Redundanz oder obsoleten Styles

0.3 Was kosten solche Komponenten

ANGULAR ARCHITECTS

- Hohe mentale Last bei jeder Anpassung
- Fehleranfällige Änderungen und Regressionen
- Langsame Entwicklung trotz erfahrener Teams
- Schwieriges Onboarding neuer Entwickler:innen

0.4 Was wir in dem Workshop erreichen wollen

ANGULAR ARCHITECTS

- Eine klare Vorstellung für moderne, saubere Angular Komponenten
- inkl. Clean Code-, TypeScript-, Angular-Best-Practices
- Dann modernisieren wir systematisch: APIs, Life Cycles
- Eine wiederholbare Modernisierungs- und Refactoringstrategie → *Blueprint*
- KI nutzen damit Refactoring schneller geht und Spaß macht ☺

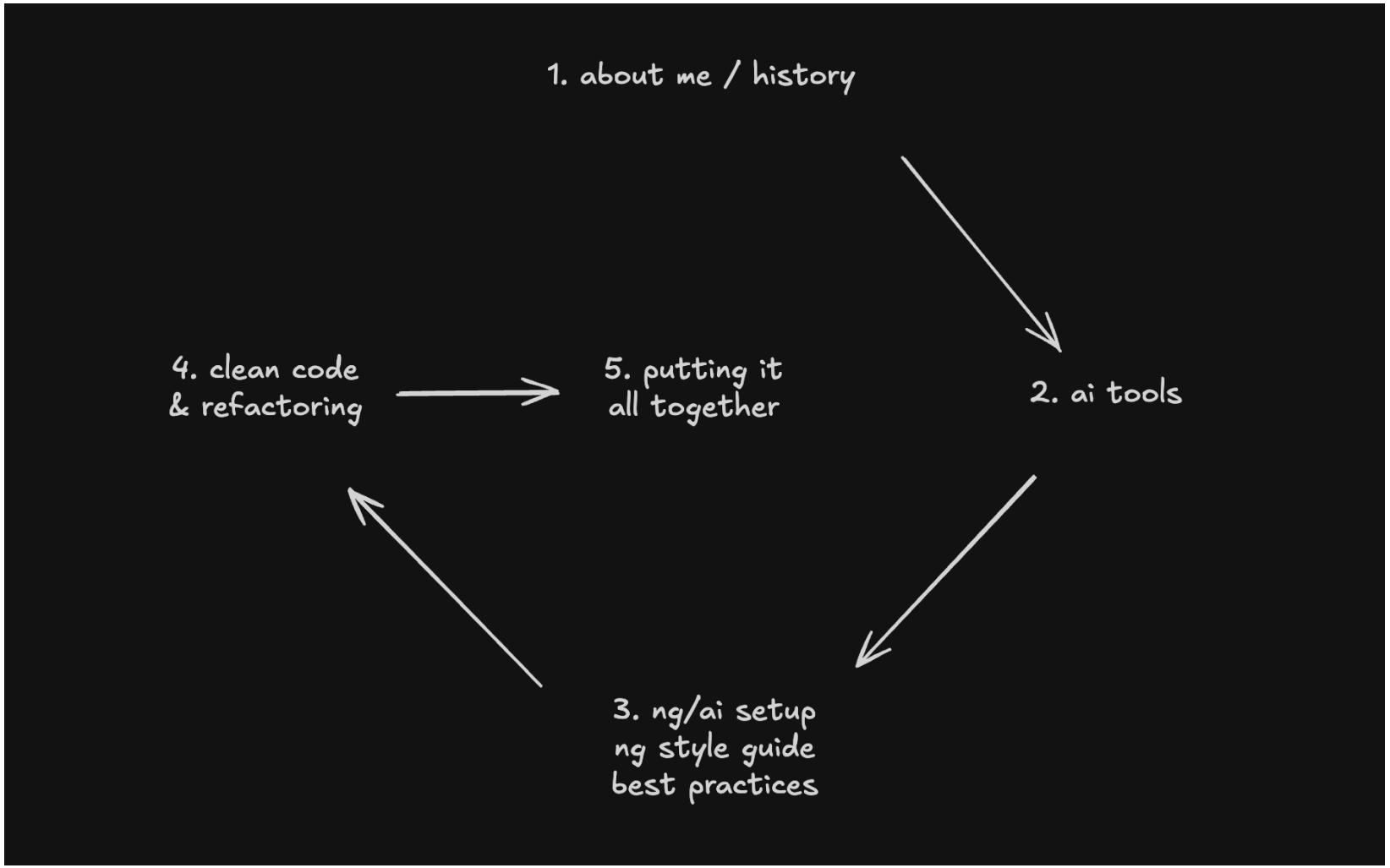
0.5 Wie gute Angular Komponenten aussehen

ANGULAR ARCHITECTS

- Kleine, klar fokussierte Komponenten mit einer erkennbaren Hauptverantwortung
- Signale, ***computed*** state und schlanke Templates ersetzen implizite Seiteneffekte
- Regeln aus Style Guide, ESLint und Teamkonventionen machen Qualität reproduzierbar
- KI beschleunigt Umbauten, aber die Architekturentscheidungen bleiben menschlich

Agenda

ANGULAR ARCHITECTS



Ziele

ANGULAR ARCHITECTS

Wie behandeln hier nicht, wie man KI nutzt, sondern ich möchte euch zeigen

- welche AI-Tools ich gerade am liebsten verwende
- *wie man ein Angular Projekt mit KI aufsetzt*
- *wie saubere Angular-Komponenten aussehen*
- ihr bekommt einen Blueprint zur *Modernisierung* & zum *Refactoring*

Hi, it's me → @LX_T

ANGULAR ARCHITECTS

- Alexander Thalhammer aus Graz, AT, EU (est. 1983)
  - **Angular Software Tree GmbH** (est. 2019)
- Web Dev seit 25 Jahren
  - 2002 – 2009 Web Developer
  - 2010 – 2017 WordPress Dev
  - 2018 – 2025 Angular Dev, Coach & Consultant
  - Seit 2026 **Agentic Angular Dev & AI Consultant**

https://LXT.dev

ANGULAR ARCHITECTS

Demo: WebStorm mit Junie

ANGULAR ARCHITECTS

Demo: VS Code mit Claude Code Plugin

@LX_T → meine Lieblingsthemen

ANGULAR ARCHITECTS

Performance 🚀

Accessibility ♿

Best Practices 📈

Refactoring 🤖

ANGULAR ARCHITECTS

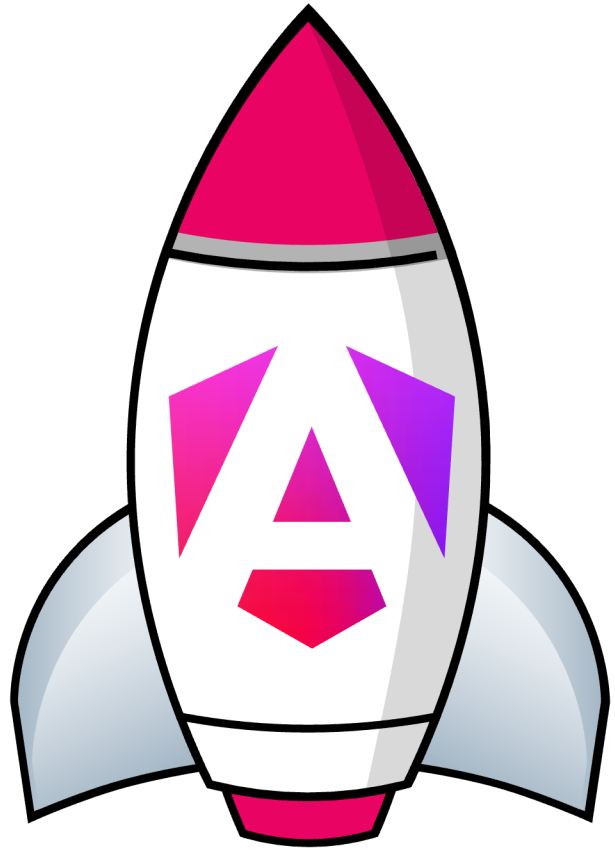
Teil 2 – Setup IDE & KI 🚀

Agenda Teil 2

ANGULAR ARCHITECTS

- Welche KI-Tools

- Deine Lieblings-IDE



ANGULAR ARCHITECTS

Demo: AI Tools

Lieblings-IDE noch state of the art?

ANGULAR ARCHITECTS



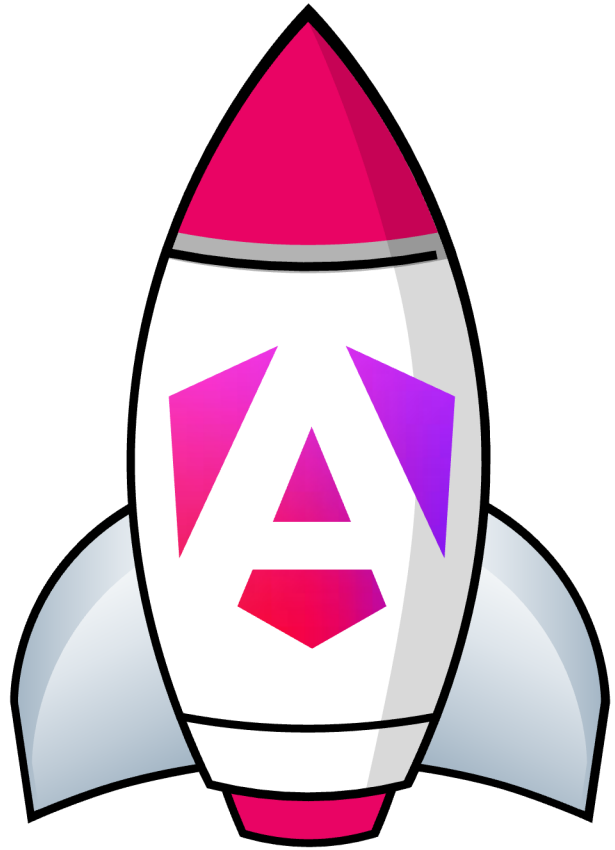
ANGULAR ARCHITECTS

Teil 2 – Guardrails für gutes KI-Refactoring

Agenda Teil 3

ANGULAR ARCHITECTS

- Modernes Angular
  - Angular Migrations
- **Clean Code** Principles
- Style Guide Präsentation



Angular Migrations (opt-in is a must!)

ANGULAR ARCHITECTS

standalone

build w/ esbuild & vite

signal inputs, outputs & queries

control flow

lazy-loaded routes mit loadComponent

inject() & cleanup imports & self-closing tags & ngClass & ngStyle router & common

ANGULAR ARCHITECTS

Showcase: Angular Migrations

https://angular.dev/reference/migrations

Clean Code in Angular

ANGULAR ARCHITECTS

- Clean Code reduziert mentale Last für Menschen und KI
- Klare Namen und präzise Typen machen Refactorings sicherer
- Kleine Komponenten und Funktionen machen Änderungen beherrschbar
- Gute Typisierung und Datenbindung hilft ebenfalls

Jetzt, wo Struktur, Typen & Lesefluss stimmen, modernisieren wir Angular

ANGULAR
ARCHITECTS

Showcase: Angular v20 Style Guide

https://angular.dev/style-guide

ANGULAR ARCHITECTS

Showcase: My Coding Style Guide

https://github.com/L-X-T/ng-b357/blob/main/style-guide/style-guide.md

Teil 3 Zusammenfassung

ANGULAR ARCHITECTS

- Angular unbedingt up2date halten um Sicherheit und moderne Features zu gewährleisten → auch die Migrations mitnehmen → KI hilft ☺
- Angular Coding Style Guide befolgen → KI braucht Angular MCP
- Angular ESLint Regeln scharf schalten → KI braucht extra Hinweis

ANGULAR ARCHITECTS

Teil 4 – Angular AI Skills

https://github.com/L-X-T/ng-ai-setup

ANGULAR ARCHITECTS

Teil 5 – Blueprint & Praxisbeispiel

https://github.com/L-X-T/ng-days-refactoring-ai

Die 7 Schritte-Vorgehensweise (Blueprint)

ANGULAR ARCHITECTS

1. Analyse
2. Entschlacken
3. NG Update
4. Modernisieren
5. Typisierung
6. Refactoring
7. Review

ANGULAR ARCHITECTS

Teil 5 – Praxis: Refactoring einer (sehr) verstaubten Angular-Komponente

https://github.com/L-X-T/ng-ai-setup

Branch: refactoring

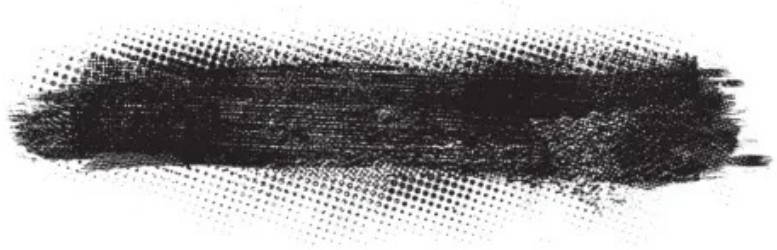
ANGULAR
ARCHITECTS

Let AI do the hard work to improve & modernize our Angular workspaces!

Alexander Thalhammer

ANGULAR ARCHITECTS

The *FUTURE* is



agentic refactoring

ANGULAR
ARCHITECTS

What will you refactor?

Alexander Thalhammer

ANGULAR
ARCHITECTS

Questions & Feedback

Thank you for the attention

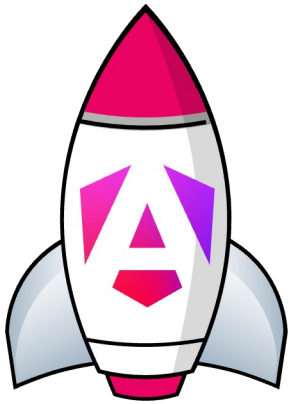
ANGULAR ARCHITECTS

[X] @LX_T

[web] https://LXT.dev

Angular Updates

Slides & GitHub & Style Guide



Workshops
and Consulting

https://angulararchitects.io