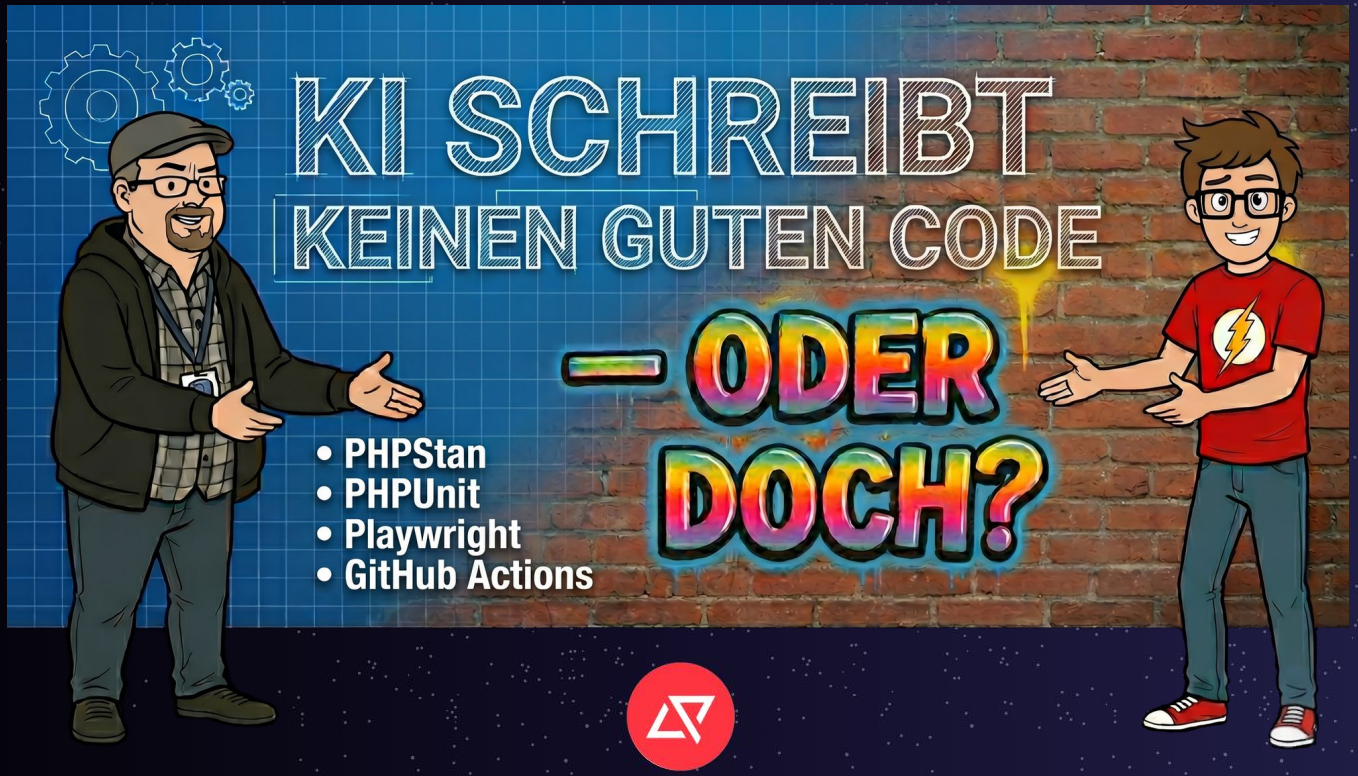

KI SCHREIBT
KEINEN GUTEN CODE
– ODER DOCH?
• PHPStan
• PHPUnit
• Playwright
• GitHub Actions

Über mich

Ich bin ein WordPress and PHP Spezialist

Meine ersten Web Projekte habe ich in den späten 90ern gestartet.

Ich hab an Projekten mit mehreren Millionen Page Impressions gearbeitet. Als Technical Lead habe ich Themes & Plugins entwickelt, Performance optimiert und neue Features geplant.

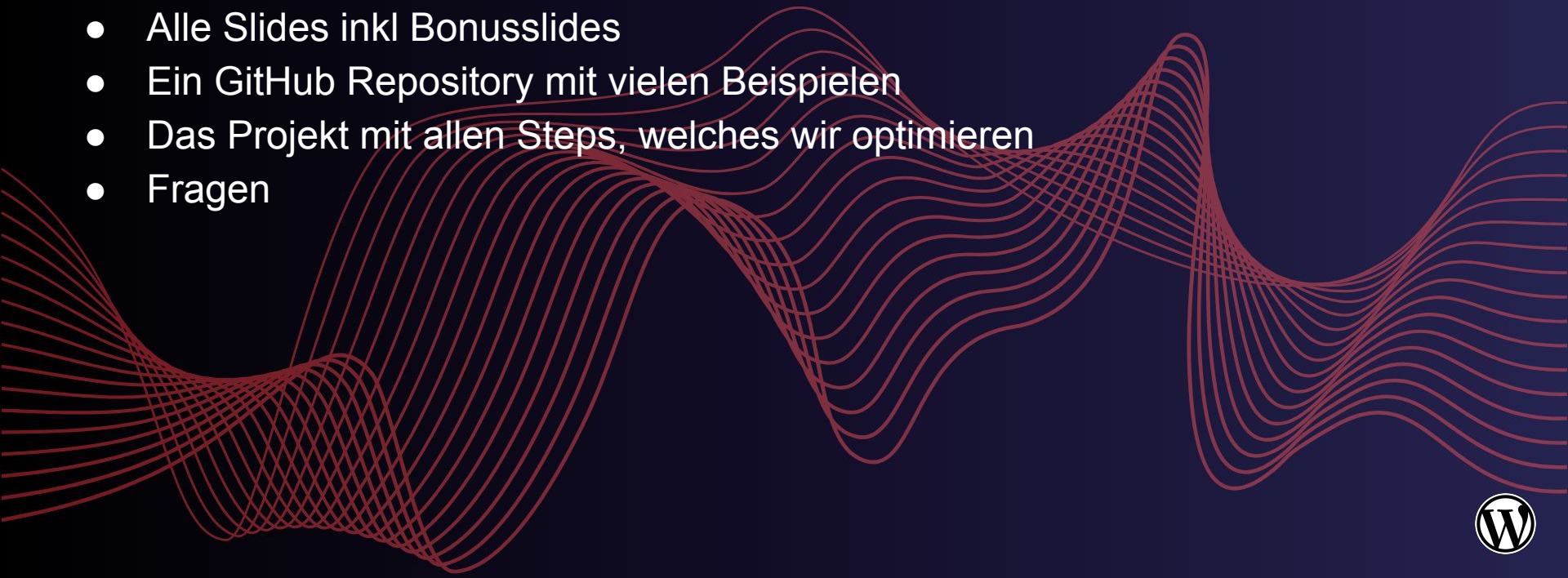
Ich spreche regelmäßig auf WordPress Events, und bin Gründer und Co-Host des Dortmunder Meetups.

Aktuell arbeite ich als Senior Web Developer, WordPress Expert und DX Engineer für die Coding Pioneers in Iserlohn.

https://www.linkedin.com/in/christoph-daum/

Material am Ende des Talks

- Alle Slides inkl Bonusslides
- Ein GitHub Repository mit vielen Beispielen
- Das Projekt mit allen Steps, welches wir optimieren
- Fragen



Wer braucht schon Code Qualität?

A WordPress Perspective

Das ist Sheldon

- Er hat meist allein gearbeitet
- Er ist sehr überzeugt von sich
- Er mag wie WordPress programmiert ist

Q: Coding Standards

Q: Coding Standards

Und denk dran:

"Code is Poetry"!

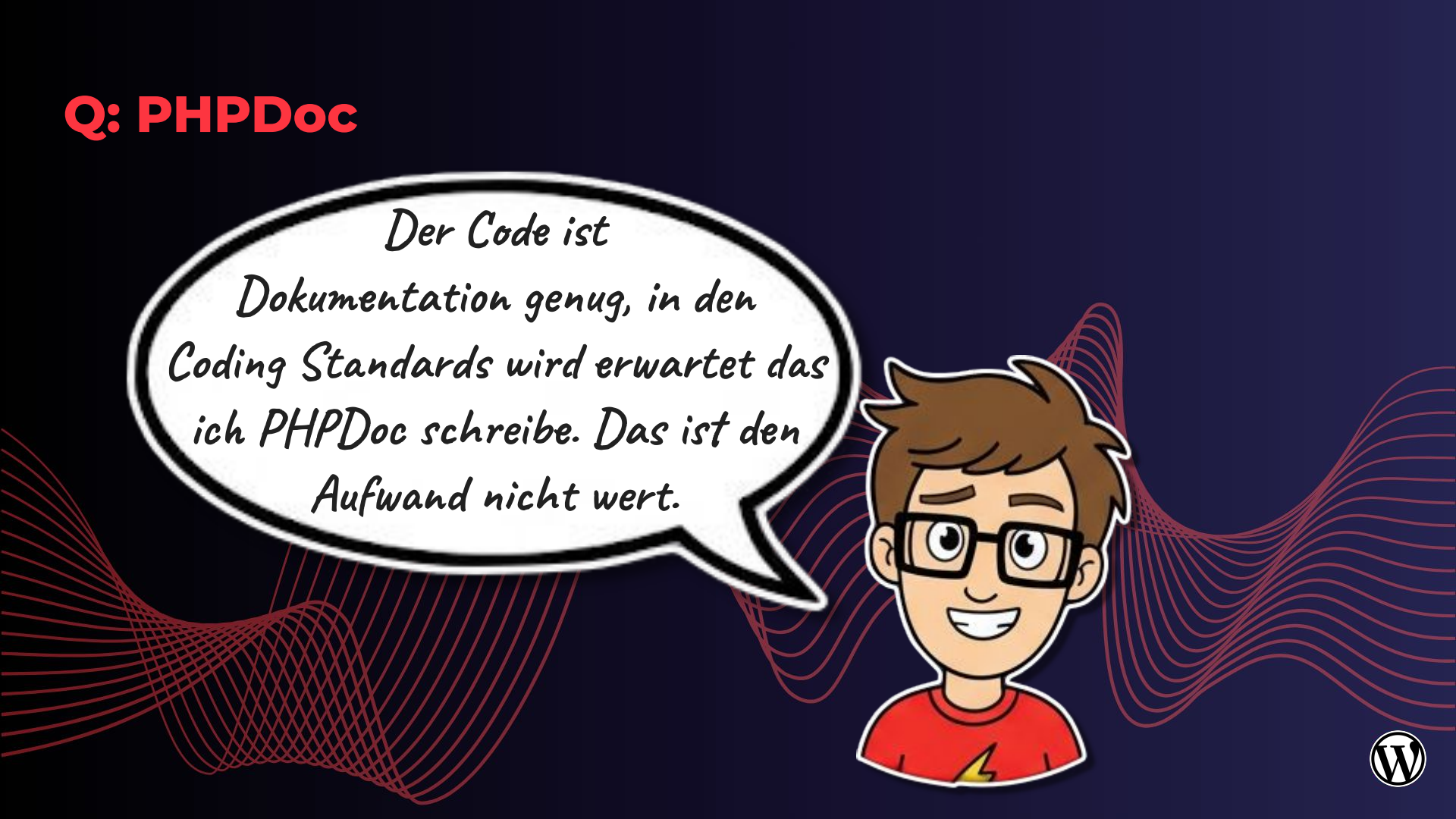
A: Coding Standards

Coding ist keine Kunst - Coding ist ein Handwerk

- Coding Standards helfen uns zusammenzuarbeiten
- Einheitliches Styling (Leerzeichen, Naming Conventions etc.)
- Durch den Code zu navigieren wird leichter
- Technische Sniffs
  - Escaping
  - Saubere Vorbereitung für i18n
  - Warnungen das Meta Queries und Co. langsam sind

Automatische Prüfung via Code Sniffer

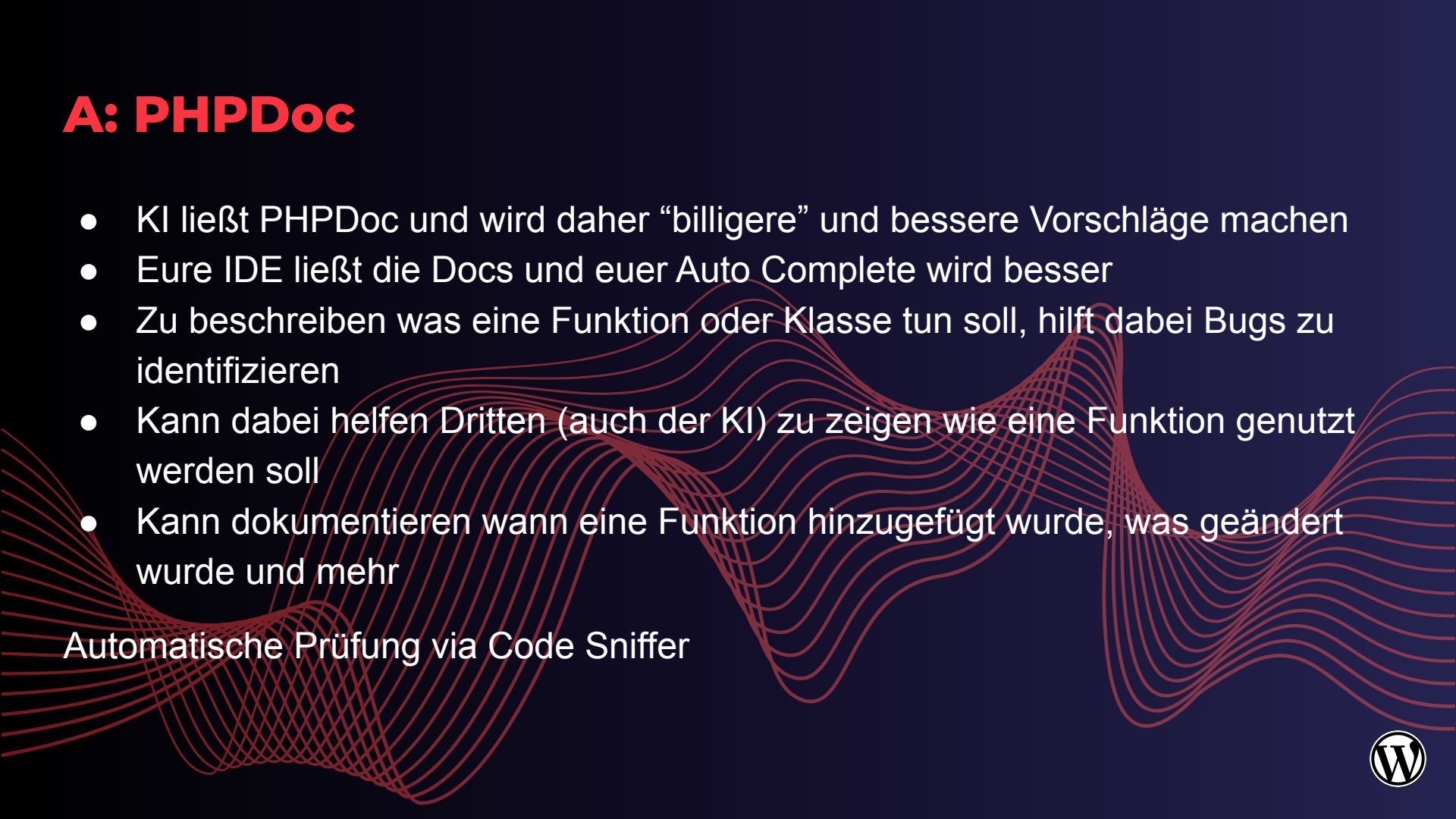
Q: PHPDoc


Der Code ist Dokumentation genug, in den Coding Standards wird erwartet das ich PHPDoc schreibe. Das ist den Aufwand nicht wert.

A: PHPDoc

- KI ließt PHPDoc und wird daher “billigere” und bessere Vorschläge machen
- Eure IDE ließt die Docs und euer Auto Complete wird besser
- Zu beschreiben was eine Funktion oder Klasse tun soll, hilft dabei Bugs zu identifizieren
- Kann dabei helfen Dritten (auch der KI) zu zeigen wie eine Funktion genutzt werden soll
- Kann dokumentieren wann eine Funktion hinzugefügt wurde, was geändert wurde und mehr

Automatische Prüfung via Code Sniffer



Q: Professionelle Kommentare

Q: Professional Comments

A: Professional Comments

- Gewohnheit professionelle Kommentare zu schreiben
- Kommentare können durch Kollegen, Kunden oder künftige Arbeitgeber gelesen werden, gerade in öffentlichen Repos
- Rants, Beleidigungen oder andere aggressive Kommentare sind können der Reputation schaden
- Kommentare sind nicht um Luft abzulassen

Fundstück aus einem echten Projekt

```
// Es tut mir leid. Es musste sein. Schnell, hart und
// dreckig. Genau wie du es magst...
```

Q: Auskommentierter Code

A: Auskommentierter Code

- Schreibt immer mindestens warum Code auskommentiert wurde
- Wenn ihr vergesst den Code zu entfernen, bleibt er für immer
- Meist reicht die Git History aus
- Am besten nur dokumentieren wann es entfernt wurde

Automatische Prüfung via Code Sniffer (mit Custom Sniff)

Q: Commits & Commit Messages

Q: Commits & Commit Messages

A: Commits & Commit Messages

- Eure Commit Messages sollten auch Regeln haben
- Zum Beispiel “Conventional Commits”
- Commited regelmäßig
  - Ein Commit pro Task, Feature oder Fix
  - Kein Commit sollte das Projekt kaputt machen
- Dokumentiert ordentlich
  - Haltet die erste Zeile kurz (unter 50 Zeichen)
  - Brecht den Body nach 72 Zeichen (zur Lesbarkeit im Terminal) **50/72 Rule**
  - Erklärt warum und nicht nur was
  - Verlinkt das Ticket wenn vorhanden
  - Jeder sollte eine Idee haben was im Commit passiert ist, nur durchs lesen der Commit Message
- Lasst euch von KI helfen!

Automatische Prüfung via commitLint

https://www.conventionalcommits.org/en/v1.0.0/

A: Commits & Commit Messages

readme	Sheldon	0d9361d	7. Dec 2016 at 07:42
css changes	Sheldon	feee1ec	7. Dec 2016 at 07:32
readme	Sheldon	c786942	7. Dec 2016 at 07:16
css tweaks	Sheldon	46d5f35	7. Dec 2016 at 07:14
v1.1.1 Version 1.1.1 * fixed keyboard changed	Sheldon	327578f	7. Dec 2016 at 07:10
v1.1.0 new version 1.1.0, watermark, statusbox, keyboard shortcuts	Sheldon	d21f12a	6. Dec 2016 at 22:49
v1.0.0 missing version bump fixed	Sheldon	dea29e0	29. Oct 2016 at 22:46
1.0.0	Sheldon	4d564b3	28. Oct 2016 at 22:52
v0.9.11 update to 0.9.11	Sheldon	051d0fe	28. Oct 2016 at 07:50
v0.9.10 Update README.md	Sheldon	aaf014b	26. Oct 2016 at 15:27
Update README.md ↩	Sheldon	2d4665c	26. Oct 2016 at 15:25
Update README.md	Sheldon	b2495d7	26. Oct 2016 at 15:15
Update apermo-adminbar.php ↩	Sheldon	f5174bd	26. Oct 2016 at 15:14
v0.9.9 Create script.js	Sheldon	f216bd0	22. Oct 2016 at 09:07
Add files via upload	Sheldon	8c73c10	22. Oct 2016 at 09:06
Update apermo-adminbar.php ↩	Sheldon	e3dec79	7. Jul 2016 at 22:19
compatibility with domain mapping for multisite	Sheldon	54356b4	1. Jul 2016 at 08:38
v0.9.5 fixed bug on subfolder installations	Sheldon	d77fbc6	30. Jun 2016 at 17:57
v0.9.4 export & import	Sheldon	de61a0d	30. Jun 2016 at 08:08
typos	Sheldon	3cab6d1	29. Jun 2016 at 22:52
v0.9.3 added new filter, removed scheme url from options	Sheldon	625f541	29. Jun 2016 at 22:39
v0.9.2 Merge pull request #1 from 2ndkauboy/master ↩	Sheldon	c2678cf	29. Jun 2016 at 08:04
some minor code improvements and removing the readme.txt file which is no longer need...	2ndkauboy	0eeb464	28. Jun 2016 at 23:42
v0.9.1 Bug fixes	Sheldon	6d44122	28. Jun 2016 at 21:02
v0.9.0 First release	Sheldon	00f89cc	28. Jun 2016 at 19:20
Initial commit	Sheldon	316ef5d	28. Jun 2016 at 18:46

A: Commits & Commit Messages

release/1.4.0 **fix: remove indie-action web actions** ↩	**Christoph Daum**	**75a513a**	**26. Mar 2026 at 13:35**
test: update tests for microdata removal ↩	Christoph Daum	c445a71	26. Mar 2026 at 12:41
refactor(images): remove microdata ↩	Christoph Daum	254a3b6	26. Mar 2026 at 09:57
refactor(templates): remove microdata ↩	Christoph Daum	e475a51	26. Mar 2026 at 09:55
refactor(comment): remove microdata ↩	Christoph Daum	e85b8c1	26. Mar 2026 at 09:48
refactor(tags): remove microdata ↩	Christoph Daum	c2b13c8	26. Mar 2026 at 09:47
refactor(semantics): remove microdata ↩	Christoph Daum	1403470	26. Mar 2026 at 09:38
test(schema): add unit tests ↩	Christoph Daum	f163f00	26. Mar 2026 at 09:34
feat(schema): add Yoast SEO integration ↩	Christoph Daum	4a78852	26. Mar 2026 at 09:11
feat(schema): add JSON-LD structured data ↩	Christoph Daum	f238099	26. Mar 2026 at 09:09
fix: correct 500.php template description ↩	Christoph Daum	10aa996	26. Mar 2026 at 08:15
fix: remove dead wp_localize_script call ↩	Christoph Daum	b908e4c	26. Mar 2026 at 08:14
fix: replace bloginfo() with escaped output ↩	Christoph Daum	14f4d35	26. Mar 2026 at 08:14
fix: remove type=\"text/css\" from style tag ↩	Christoph Daum	90846e3	26. Mar 2026 at 08:13
fix: remove wp_body_open() compat shim ↩	Christoph Daum	42faec3	26. Mar 2026 at 08:12
fix: remove get_self_link() polyfill ↩	Christoph Daum	168a8ce	26. Mar 2026 at 08:10
origin/main v1.3.0 Merge pull request #46 from apermo/release/1.3.0 ↩	Christoph Daum	0e6d197	22. Mar 2026 at 22:14
release/1.3.0 **fix**: clean up the_content method ↩	Christoph Daum	4a63b25	22. Mar 2026 at 22:06
fix: use @see for WP core hook docblocks ↩	Christoph Daum	0f7b836	22. Mar 2026 at 21:42
fix: keep named parameters for get_the_content ↩	Christoph Daum	f180e9f	22. Mar 2026 at 21:27
fix: address PR review feedback (round 4) ↩	Christoph Daum	5a278d9	22. Mar 2026 at 21:18
fix: address PR review feedback (round 3) ↩	Christoph Daum	9f02aae	22. Mar 2026 at 21:05
docs: update README, CHANGELOG, CLAUDE.md for 1.3.0 ↩	Christoph Daum	f807919	22. Mar 2026 at 20:06
chore: remove unused entry-nav template part ↩	Christoph Daum	075df37	22. Mar 2026 at 19:35
test: expand E2E suite to 31 tests (#49) ↩	Christoph Daum	fee01c8	22. Mar 2026 at 19:17
docs: add 1.3.0 changelog entry	Christoph Daum	946a257	22. Mar 2026 at 18:01

Q: Main Branch vs feature branches

Direkt im Main Branch ist schneller und einfacher. Feature Branches sind overhead und ich hab dafür auch keine Zeit.

A: Main Branch vs feature branches

Das führt zu Problemen, sobald…

- Ihr zu Mehreren arbeitet
- Ihr an mehreren Features parallel arbeitet
- Ihr größere Änderungen (ein Refactoring) vorhabt
- Ihr mehrere Stages eure Websites pflegt
- Oder mehrer Versionen der Plugin/Themes wartet

Q: Pull Request und Code Review

A: Pull Request und Code Review

- Code Review ist ein wichtiges Standbein in der Software Entwicklung
- Teile Wissen und lerne von Anderen
- 4 Augen Prinzip für Code
- Verringert die Gefahr von Logik Fehlern
- Verhindert Debug Code im Prod System
- Korrekturen für Tippfehler
- Mehr Vertrauen in den Code vorm Deployment

Q: Pair Programming

A: Pair Programming

- Pair Programming kann das Code Review ersetzen - je nachdem wie es im Team geregelt ist
- Zum teilen von Wissen
- Zum Debugging
- Zum Brainstormen
- Für Software Architektur

Q: AI Review

Ok, also kann ich Claude, Gemini oder CoPilot meine Änderungen prüfen lassen.

A: AI Review

Prinzipiell eine gute Idee… **ABER**

- Kennt vielleicht nicht den kompletten Kontext
- Wird Sachen übersehen
- Wird Sachen finden die es nicht gibt
- Wird nicht die Verantwortung tragen

In unserem Code Review Prozess ist es ein wichtiger Bestandteil seit einigen Monaten.
Zuerst das Review durch mehrere KI Modelle
und zum Abschluss das Review durch einen Menschen

Q: PHP7&8 Type Hints

A: PHP7&8 Type Hints

- Strikte Typisierung hilft dabei den Code Robust und Vorhersehbarer zu machen
- Fehler werden sichtbar, statt dass sie seltsame Seiteneffekte sind
- Kann Sicherheit erhöhen
- Eindeutiger Absprache was empfangen und gesendet wird
- Verbessert die Lesbarkeit
- Kann Validation und Sanitization vereinfachen und teilweise ersetzen
- Vereinfacht Test-Szenarien

Automatische Prüfung via Code Sniffer und PHPStan

Q: Strikte Vergleiche

Zu prüfen ob ein Wert truthy oder falsy ist, reicht völlig. Strikte Vergleiche sind kleinkariert.

A: Strikte Vergleiche

Strikte Vergleiche können unvorhersehbares Verhalten verhindern.

Statt zu prüfen ob etwas truthy ist, ist es besser zu prüfen ob es den richtigen type beinhaltet.

```
if ( $post ) {
    echo $post->post_title;
}
```

Kann einen fatal error werfen, falls $post ein WP_Error ist

```
if ( $post instanceof \WP_Post ) {
    echo $post->post_title;
}
```

Q: DRY - Don’t repeat yourself

A: DRY - Don’t repeat yourself

Copy-pasting produziert die Wartungshölle.

- ✅ Extrahiert wiederholte Logik
- ✅ Erstellt wiederverwendbare Funktionen
- ✅ Einheitliche Validation
- ✅ Verwendet Vererbung weise

- ❌ Kein copy-paste Code
- ❌ Keine wiederholte Validation
- ❌ Keine doppelten Queries
- ❌ Keine Über Abstraktion
- ❌ Keine unnötigen Abhängigkeiten

Q: KISS - Keep it simple, stupid

Ich möchte meine Skills zeigen,
Es tut schon nicht weh wenn ich ein bisschen übertreibe.
Der Kunde wird es LIEBEN.

A: KISS - Keep it simple stupid

✅ Wählt einfache Lösungen

✅ Vermeidet unnötige Komplexität

✅ Schreibt lessbaren Code

✅ Verwendet übliche Muster

✅ Hinterfragt jede Abstraktion

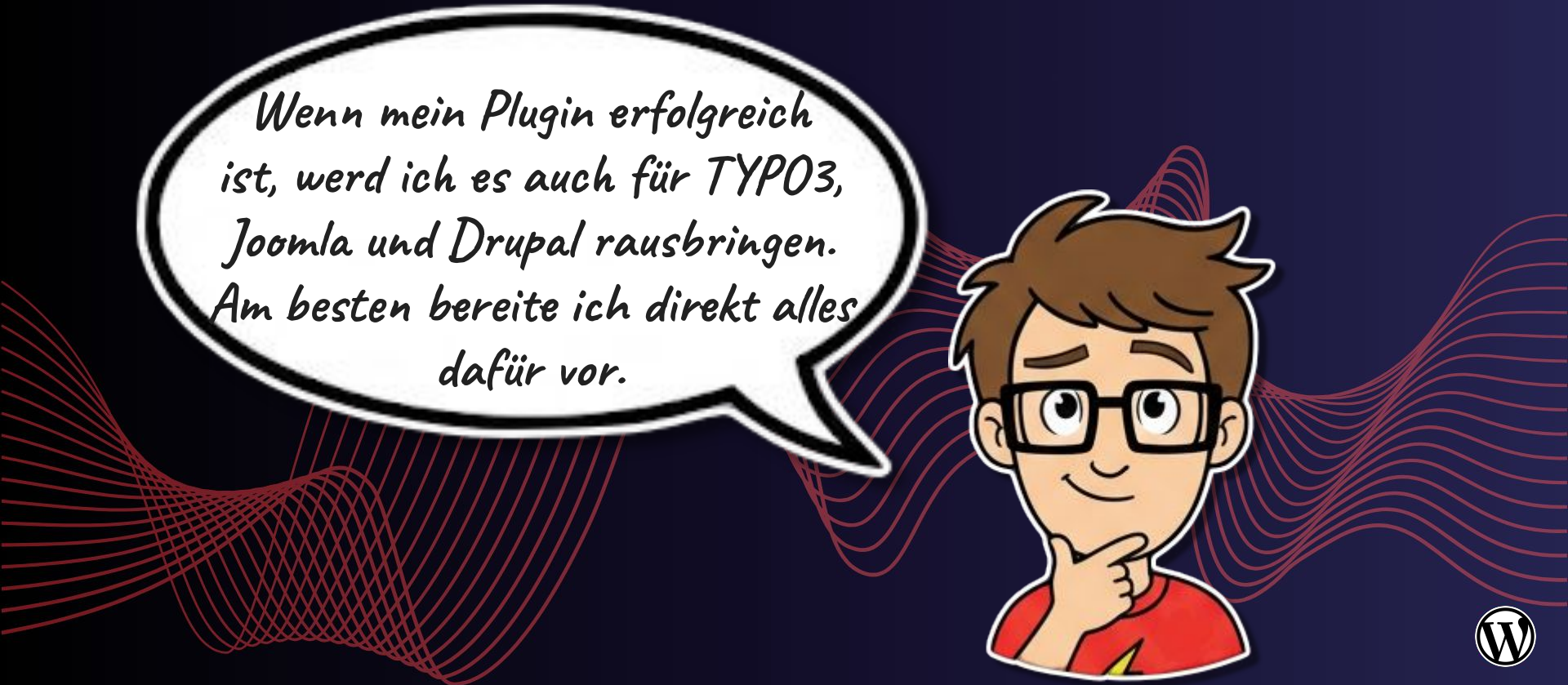
❌ Vermeidet design patterns für simple Aufgaben

❌ Fügt keine Layer ohne Grund hinzu

❌ Macht keine simplen Dinge kompliziert

❌ "Clever" ist nicht besser als "Klar”

Q: YAGNI - You ain’t gonna need it



A: YAGNI - You ain’t gonna need it

✅ Entwickelt was ihr JETZT braucht

✅ Fügt Features hinzu SOBALD sie angefordert werden

✅ Haltet es einfach

✅ Refactored sobald Muster absehbar sind

❌ Nicht "Nur für den Fall"

❌ Kein over-engineering

❌ Keine “Vielleicht” Features

❌ Keine ungenutzten Abstraktionen

❌ Aber überspringt das Planning auch nicht vollständig

Q: Software Design Principles (SOLID)

A: Software Design Principles (SOLID)

- **S**ingle Responsibility
- **O**pen-closed principle
- **L**iskov substitution principle
- **I**nterface segregation principle
- **D**ependency inversion principle

https://de.wikipedia.org/wiki/Prinzipien_objektorientierten_Designs#SOLID-Prinzipien

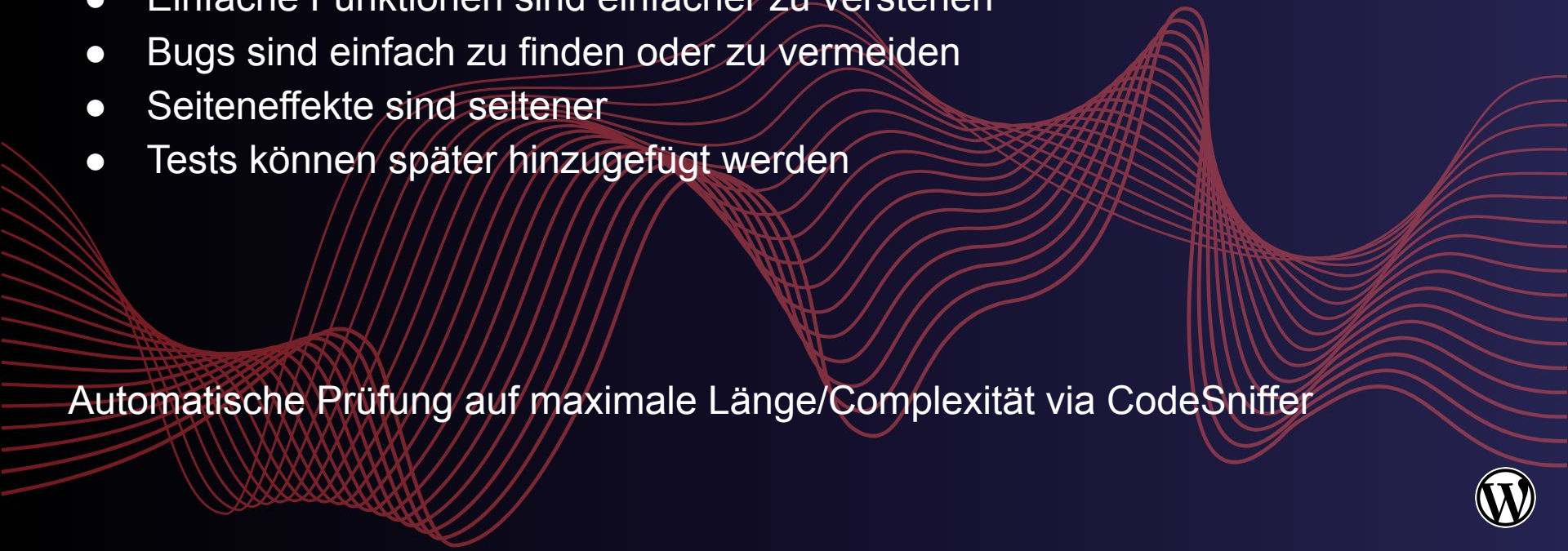
Q: Testbarer Code

A: Testbarer Code

Testbarer Code oder Code nach SOLID, ist auch auch ohne Tests gut (besser)

- Einfache Funktionen sind einfacher zu verstehen
- Bugs sind einfach zu finden oder zu vermeiden
- Seiteneffekte sind seltener
- Tests können später hinzugefügt werden

Automatische Prüfung auf maximale Länge/Complexität via CodeSniffer



Q: A11y, I18n, SEO und mehr

A: A11y, I18n, SEO und mehr

In kurzen Worten

Aber das ist ein Thema
für ein andermal

Teilt euer Wissen und lernt von anderen


So...
WHO NEEDS
CODE QUALITY
ANYWAY?!

DEMO

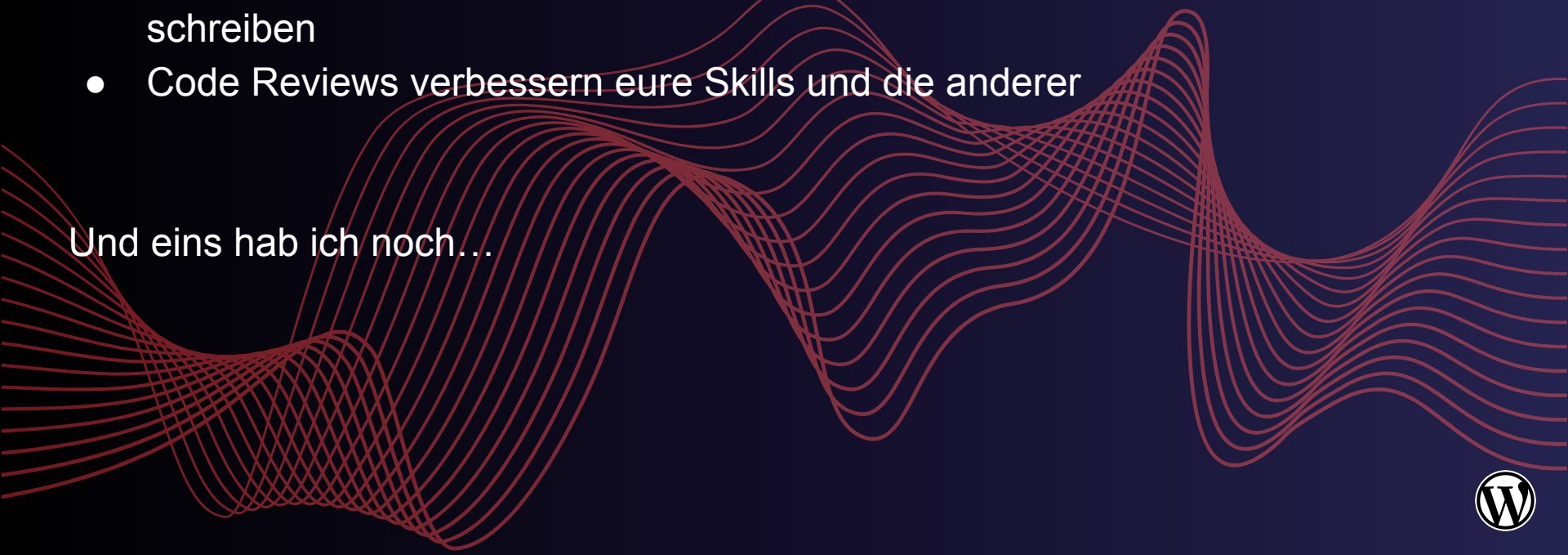
Video zum Vortrag: youtu.be/WovPT2JsPN8

Projekt Fortschritt: github.com/apermo/apermo-adminbar/pull/39

Teilt euer Wissen und lernt von anderen

- Zu teilen und zu lernen ist wichtig
- Guten Code zu schreiben, hilft anderen (auch der KI) besseren Code zu schreiben
- Code Reviews verbessern eure Skills und die anderer

Und eins hab ich noch…



Ich bin Sheldon… wir alle sind Sheldon.

Fragen?
Danke für eure Aufmerksamkeit!

Folien auf:
c13s.com/wordcamp

Weitere Beispiele:
github.com/apermo

linkedin.com/in/christoph-daum/