

Kontinuierlich und automatisiert dokumentieren Docs-as-Code in der Praxis

FALK SIPPACH // EMBARC

OOP 2023, Online

Donnerstag, 09.02.2023



1

Kontinuierlich und automatisiert dokumentieren

Dokumentation wird häufig vernachlässigt. Mit dem Docs-as-Code-Ansatz wird die in Softwareprojekten relevante Dokumentation genau wie Quellcode behandelt, in der Versionsverwaltung abgelegt, mit leichtgewichtigen Entwicklerwerkzeugen (IDE/Texteditor, Build-Tools, CI/CD-Pipelines) bearbeitet und in die Softwareentwicklungsprozesse integriert. Die Inhalte können redundanzfrei verwaltet und manche Informationen aus Modellen oder dem Quellcode generiert werden. Durch die Verwendung leichtgewichtiger Text- und Grafikformate lassen sich die Ergebnisse einfach zielgruppenorientiert zusammenstellen. Die Verarbeitung erfolgt automatisiert über die schon vorhandenen Build-Prozesse.

Jedwede Art von Dokumentation gewinnt somit an Sichtbarkeit, durch die Eingliederung in die Entwicklungsprozesse und die damit verbundene kontinuierliche Weiterentwicklung steigt die Qualität und damit die Akzeptanz bei den Lesern. Dokumentation kann sogar ausgeführt werden, um zum Beispiel eingebettete Architekturregeln regelmäßig automatisiert zu testen. Die Zuhörer erfahren in diesem Vortrag an konkreten Beispielen, wie sie mit Documentation as Code starten können, welche typischen Fallstricke sie umschiffen und mit welchen konkreten Tools sie am besten arbeiten sollten.



2

Falk Sippach

- Softwarearchitekt, Berater, Trainer bei embarc
- früher bei Orientation in Objects (OIO), Trivadis

Schwerpunkte:

- Architekturberatung und -bewertung
- Cloud- und Java-Technologien



✉ fs@embarc.de

🐦 [@sippsack](https://twitter.com/sippsack)

➔ [xing.to/fsi](https://www.xing.to/fsi)



Docs-as-Code in der Praxis

embarc.de

3

3

Wer seid Ihr?



Docs-as-Code in der Praxis

embarc.de

6

6



sw_architecture_handson
@HandsonSw

...

Antwort an @HandsonSw @grafandreas und 3 weitere Personen

I think this "docs-as-code", "architecture-as-code" thing is currently at the beginning. People just start to find out how much sense it makes to embed the architecture documents into your git repo in a branch/mergable way. Sure we will see some great solutions in the future.

[Tweet übersetzen](#)

11:56 vorm. · 2. Okt. 2021 · Twitter Web App

4 Retweets 1 Tweet zitieren 24 „Gefällt mir“-Angaben



Docs-as-Code in der Praxis

embarc.de

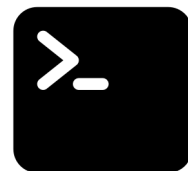
7

7

Inhalte



Docs-as-Code Reifegrad-Modell



Werkzeuge



Docs-as-Code in der Praxis

embarc.de

8

8

Agenda



- 1 Einstieg und Motivation
- 2 Docs-as-Code Maturity Level
- 3 Werkzeuge
- 4 Fazit und Ausblick



Agenda

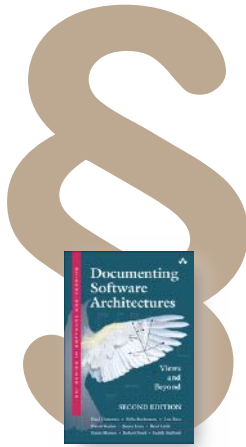


- 1 Einstieg und Motivation**
- 2 Docs-as-Code Maturity Level
- 3 Werkzeuge
- 4 Fazit und Ausblick

1



7 Regeln für gute Dokumentation



„Documenting Software Architectures:
Views and Beyond“
Clements, et.al, 2. Auflage 2010

1. Schreibe aus Sicht des Lesers
2. Vermeide unnötige Wiederholungen
3. Vermeide Mehrdeutigkeiten
 3. a) Erkläre Deine Notation
4. Verwende eine Standardstrukturierung
5. Halte Begründungen für Entscheidungen fest
6. Halte die Dokumentation aktuell, aber auch nicht zu aktuell
7. Überprüfe Dokumentation auf ihre Gebrauchstauglichkeit



Docs-as-Code in der Praxis

embarc.de

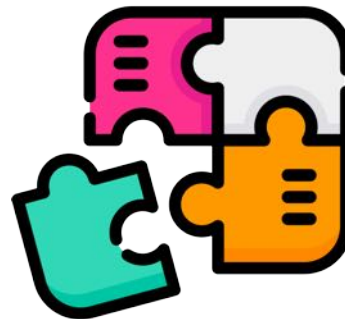
12

12

Architekturentwurf



Anforderungen/
Architekturziele



Lösungsansätze/
Entscheidungen



Docs-as-Code in der Praxis

embarc.de

13

13

Zutaten für einen Architekturüberblick



Qualitätsziele



Vorgaben



Systemkontext



Metapher
"Produktkarton"



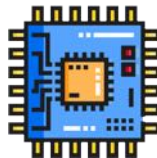
Informeller
Überblick



Architektur-Stil/-
Muster/...



Entscheidungen



Technologien

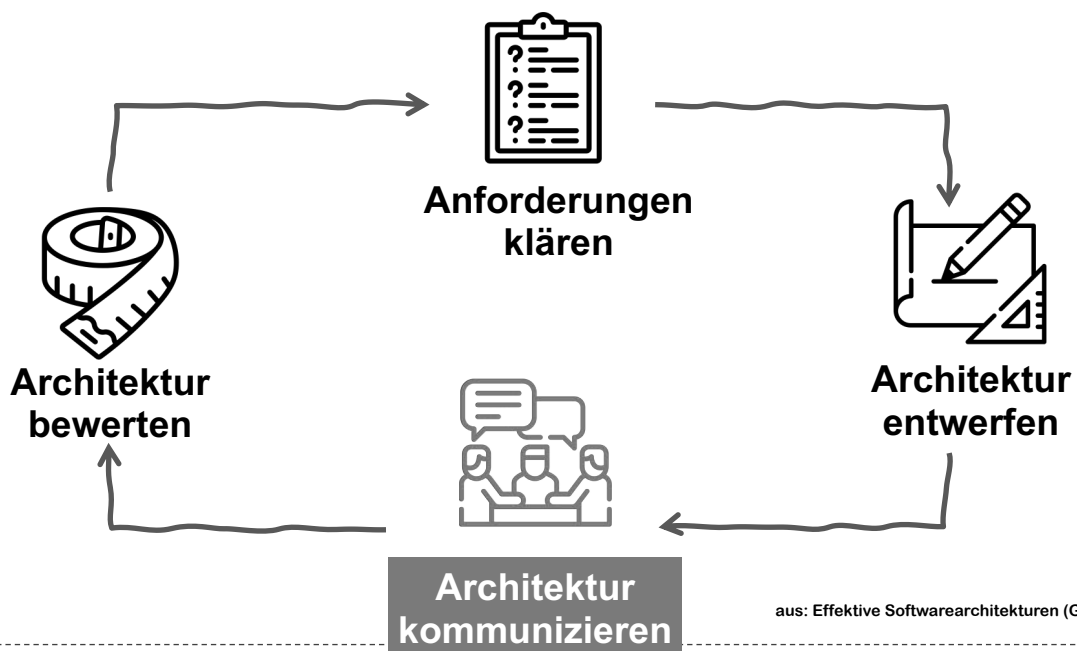


Docs-as-Code in der Praxis

embarc.de

15

15



Docs-as-Code in der Praxis

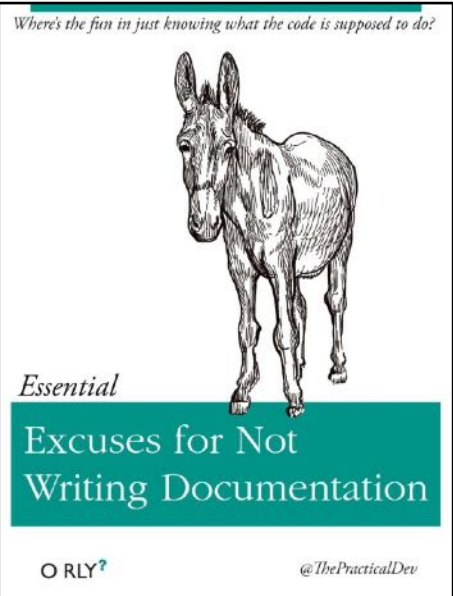
embarc.de

35

35

35

Where's the fun in just knowing what the code is supposed to do?

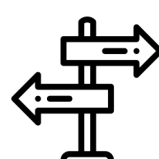


Essential
Excuses for Not Writing Documentation
O RLY? @ThePracticalDev

Grafik von [The Practical Dev](#) (CC0 Public Domain Lizenz)



Entwurfs-
unterstützung



Frage nach
dem Warum



Bewertbarkeit



Neue Mitarbeiter



Kommunikation



Docs-as-Code in der Praxis

embarc.de

36 36

36



Liest sowieso keiner!

Ungeeignete Werkzeuge

Veraltet

Medienbruch

Handarbeit

Dateiablage

Textwüsten

Redundanzen



Docs-as-Code in der Praxis

embarc.de

37 37

37

Agenda



- 1 Einstieg und Motivation
- 2 Docs-as-Code Maturity Level**
- 3 Werkzeuge
- 4 Fazit und Ausblick

2



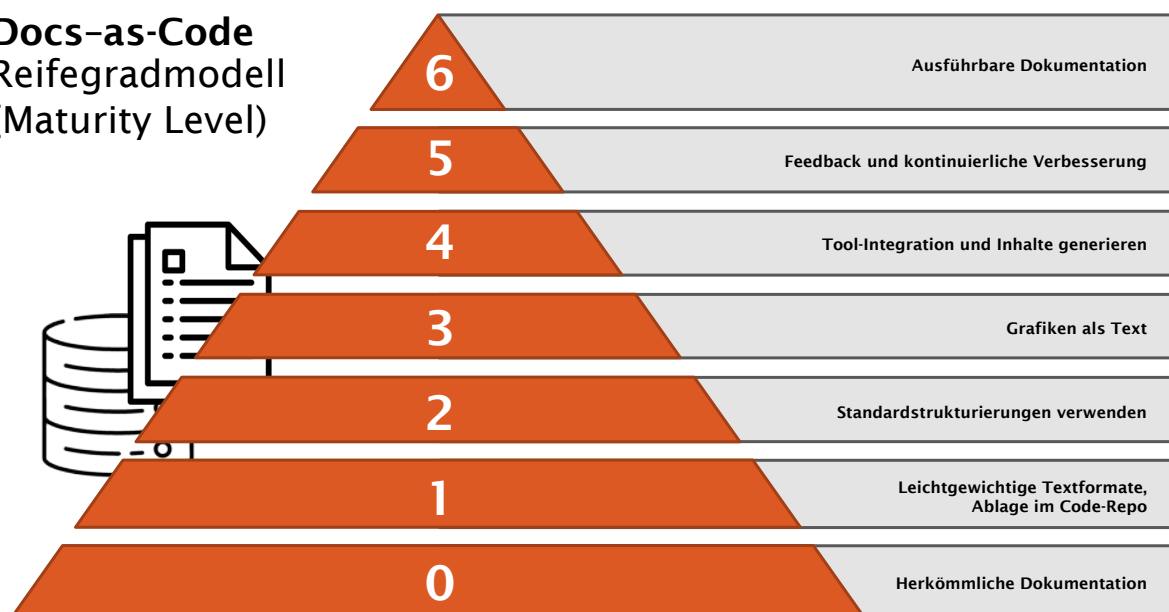
Docs-as-Code in der Praxis

embarc.de

38

38

Docs-as-Code Reifegradmodell (Maturity Level)



Docs-as-Code in der Praxis

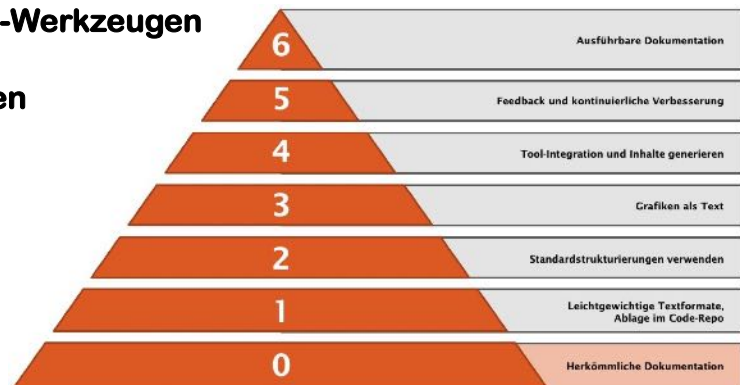
embarc.de

39

39

Herkömmliche Dokumentation

- Dokumentation wird getrennt vom Code verwaltet
- Bearbeitung in WYSIWYG-Werkzeugen
- Ablage auf Netzlaufwerken
- Verwendung von Wikis



... der Ort, wo die Dokumente zum Sterben hingehen



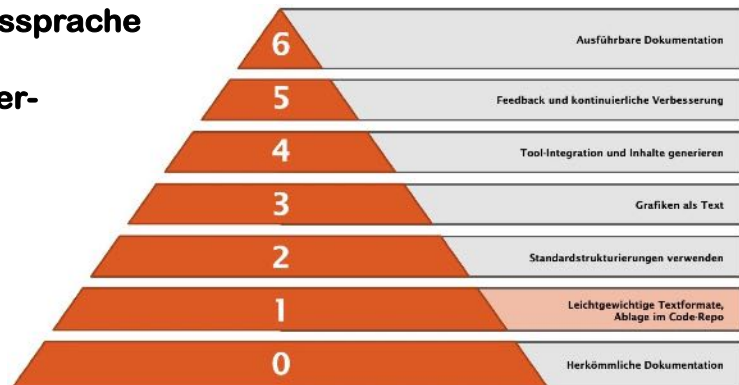
Wiki = Where Information Kills Itself

CC BY-SA 3.0, https://de.wikipedia.org/wiki/Ward_Cunningham#/media/File:Ward_Cunningham_-_Commons-1.jpg



Leichtgewichtige Textformate, Ablage im Repo

- Dokumentation ist gemeinsam mit dem Quelltext versionierbar
- Wahl einer Auszeichnungssprache
- Bearbeitung mit Entwickler-Werkzeugen



Hauptsache,
du machst es
nicht mit
Word!



Probleme mit Word und Co.

Binärformat

Mehrbenutzer schwierig

Dokument per Mail verschicken und Änderungen wieder zusammenbringen

Historisierung, Versionen vergleichen

Word ist seitenorientiert (Seitenzahl, Header, Footer)

Subdokumente und Bild-Referenzen (möglich, aber wird kaum verwendet)



Docs-as-Code in der Praxis

embarc.de

44

44

Unser täglich Entwickler-Brot

- Plain-Text
- Entwicklungsumgebung
- Kommandozeilenwerkzeuge
- **Versionsverwaltung**

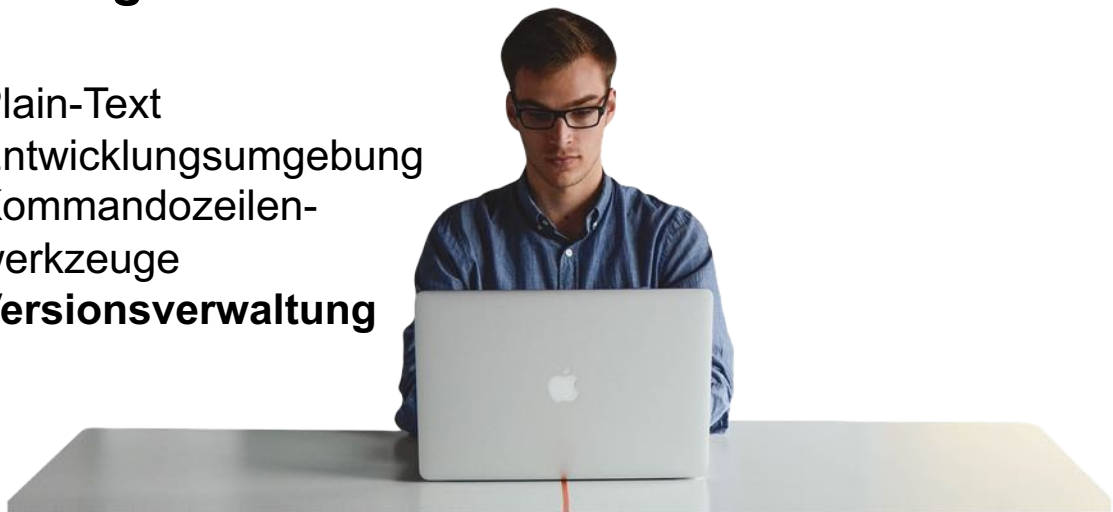


Foto von geralt: <https://pixabay.com/de/unternehmer-start-start-up-karriere-696976/> (CC0 Public Domain Lizenz)



Docs-as-Code in der Praxis

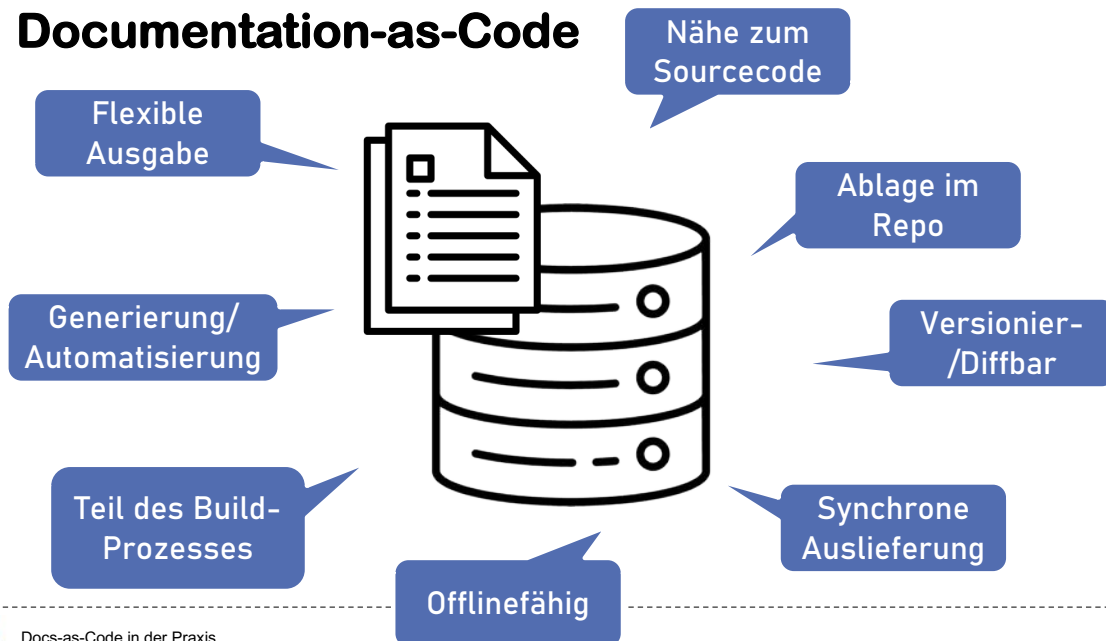
embarc.de

45

45

45

Documentation-as-Code



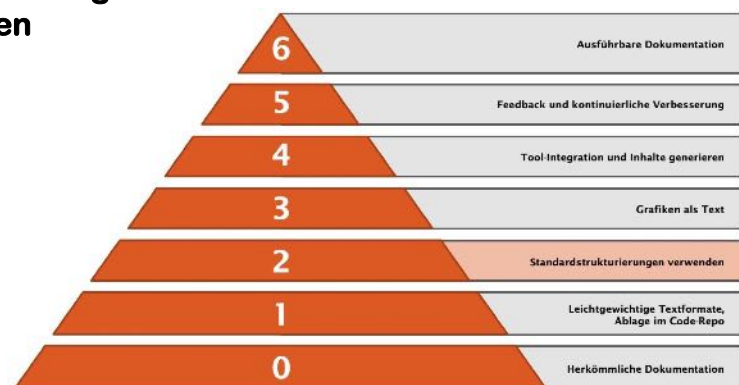
Docs-as-Code in der Praxis

46

46

Standardstrukturierungen verwenden

- Leser finden sich leichter zurecht
- Schreiber erhalten Orientierung bei der Erstellung von Inhalten



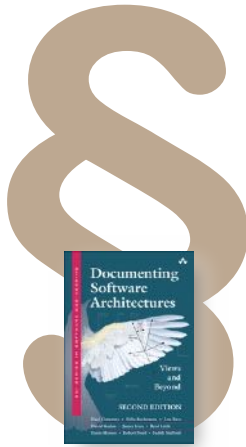
Docs-as-Code in der Praxis

embarc.de

47

47

7 Regeln für gute Dokumentation



„Documenting Software Architectures:
Views and Beyond“
Clements, et.al, 2. Auflage 2010

1. Schreibe aus Sicht des Lesers
2. Vermeide unnötige Wiederholungen
3. Vermeide Mehrdeutigkeiten
 3. a) Erkläre Deine Notation
- 4. Verwende eine Standardstrukturierung**
5. Halte Begründungen für Entscheidungen fest
6. Halte die Dokumentation aktuell, aber auch nicht zu aktuell
7. Überprüfe Dokumentation auf ihre Gebrauchstauglichkeit



Docs-as-Code in der Praxis

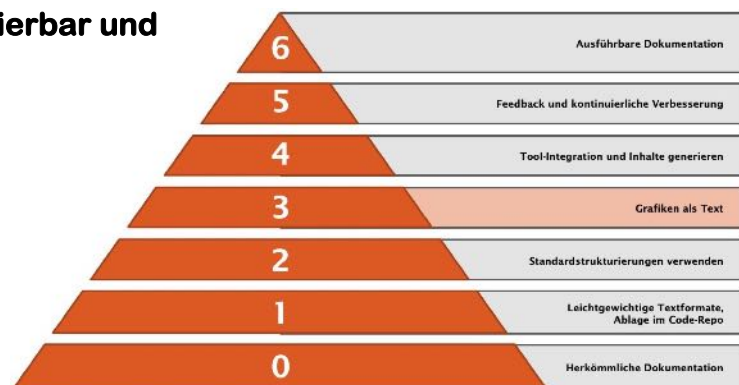
embarc.de

48

48

Grafiken als Text

- **Mindert Risiko des Vendor-Lock-In bei Diagrammen**
- **macht Graphiken versionierbar und diffbar**



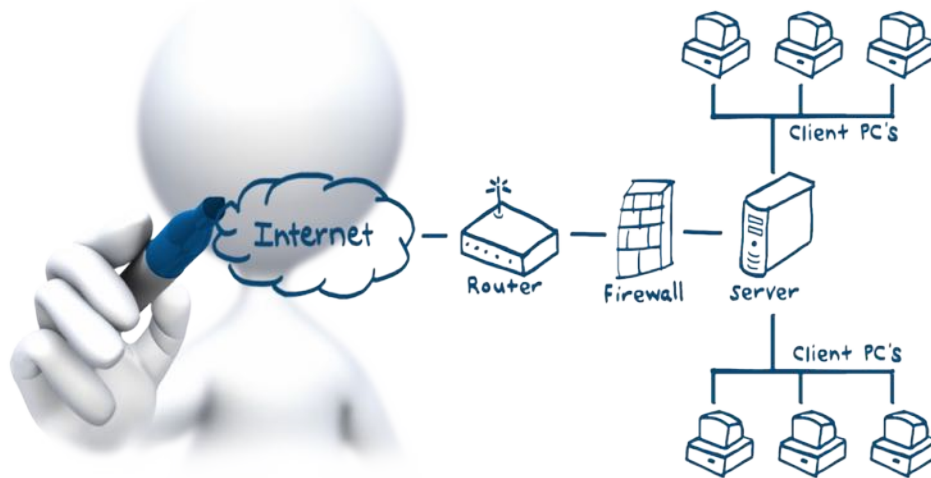
Docs-as-Code in der Praxis

embarc.de

49

49

Diagramme



Docs-as-Code in der Praxis

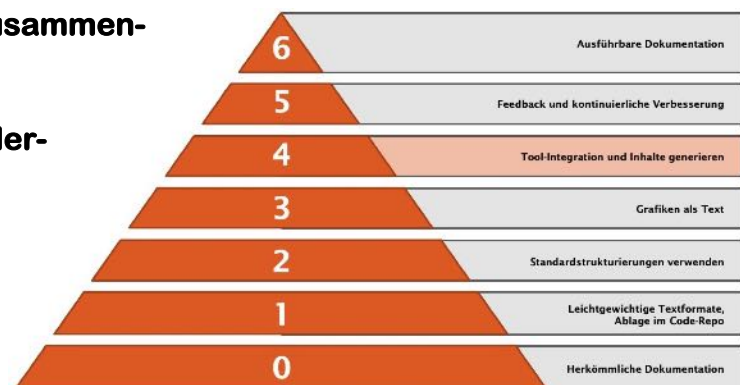
embarc.de

50

50

Tool-Integration und Inhalte generieren

- automatisierte Erstellung im Build-Prozess
- zielgruppenorientierte Zusammenstellung der Ergebnisse
- Verwendung der Entwicklerwerkzeuge ohne Kontextwechsel
- einfache Im- und Exporte

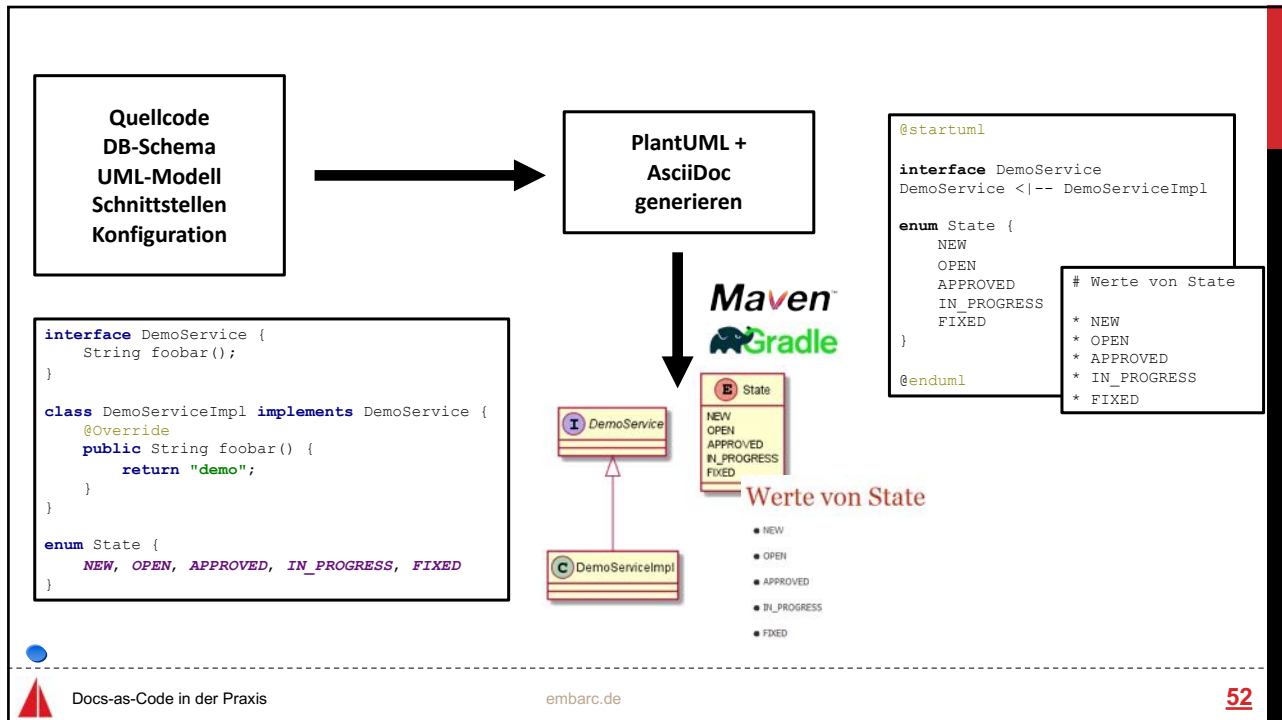


Docs-as-Code in der Praxis

embarc.de

51

51

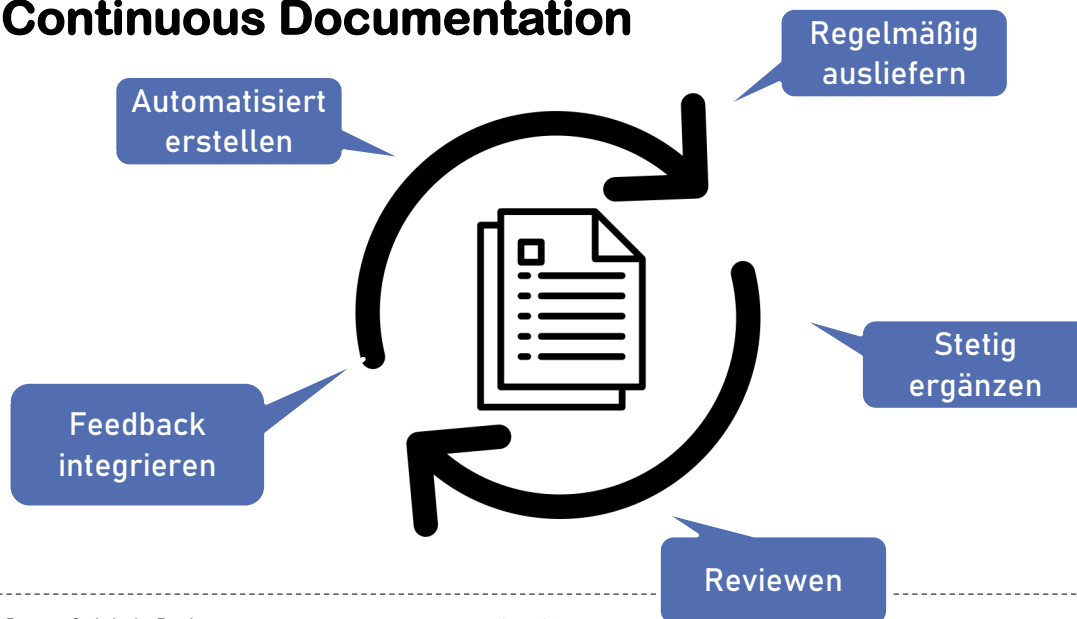


52



53

Continuous Documentation



Docs-as-Code in der Praxis

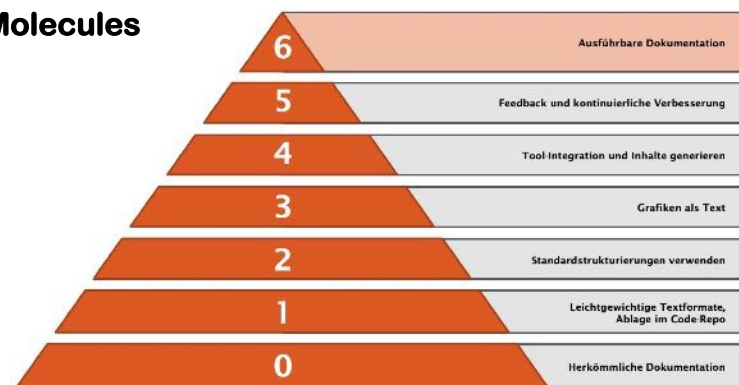
embarc.de

54

54

Ausführbare Dokumentation

- **Strukturvorgaben dokumentieren und automatisiert testen**
- **jQAssistant, ArchUnit, xMolecules**



Docs-as-Code in der Praxis

embarc.de

61

61

Agenda



- 1 Einstieg und Motivation
- 2 Docs-as-Code Maturity Level
- 3 Werkzeuge**
- 4 Fazit und Ausblick

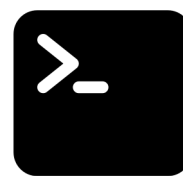
3



Werkzeuge



IDEs, Texteditoren



Build-Management



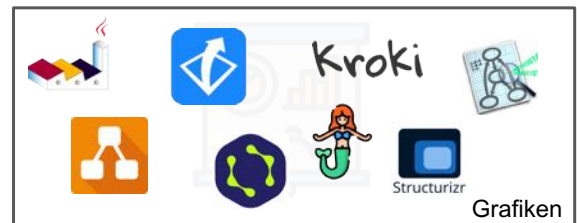
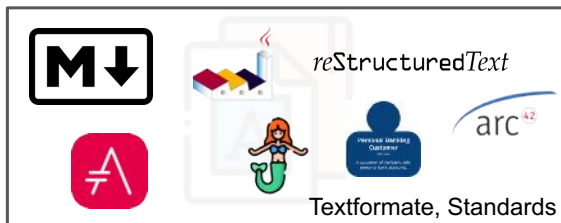
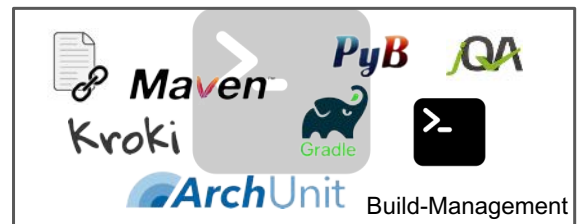
Textformate, Standards



Grafiken



Werkzeuge



Docs-as-Code in der Praxis

embarc.de

65

65

Werkzeuge



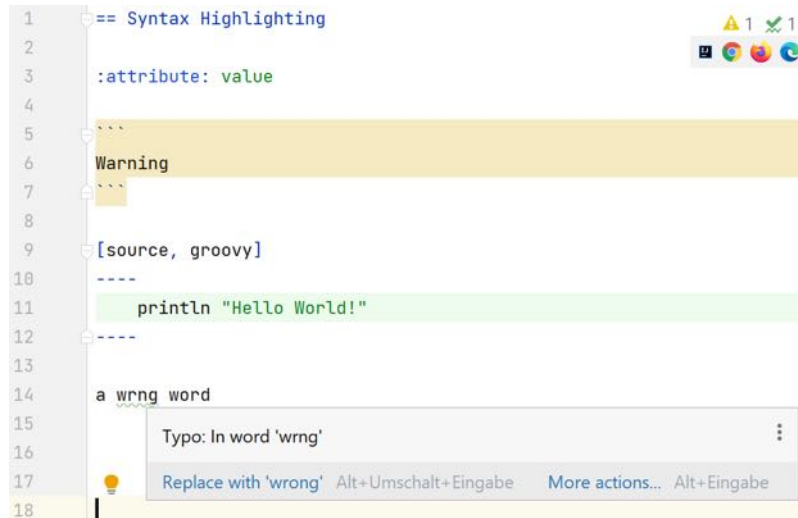
Docs-as-Code in der Praxis

embarc.de

67

67

Docs-as-Code in der IDE



Docs-as-Code in der IDE



Docs-as-Code in der IDE

```

22 == Autovervollständigung
23
24 include::p
25
26 part2.adoc
27 pictures.png (400x289)
28 auto-attribute.png (845x330)
29 autovervollstaendigung.png (590x508)
30 autovervollständigung.png (884x502)
31 grammar_check.png (881x166)
32 chapter_1.adoc
33 live template.png (558x373)
34
35
36
37

```

Press Strg+. to choose the selected (or first) suggestion and insert a dot afterwards [Next Tip](#)



Docs-as-Code in der IDE

```

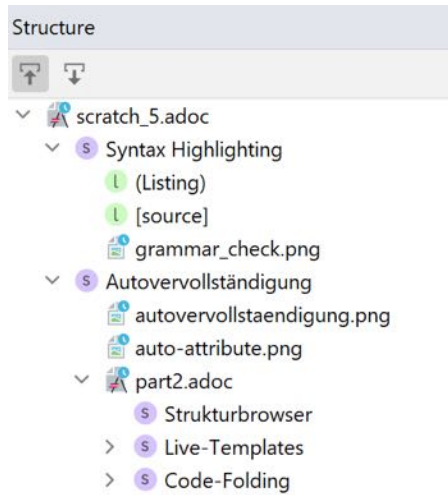
22 == Autovervollständigung
23
24 :app-id: 4711
25
26 {app}
27 app-id (4711)
28 app-name (not set)
29 appendix-caption (Appendix)
30 appendix-number (A)
31 appendix-refsig (Appendix)
32
33
34
35
36
37

```

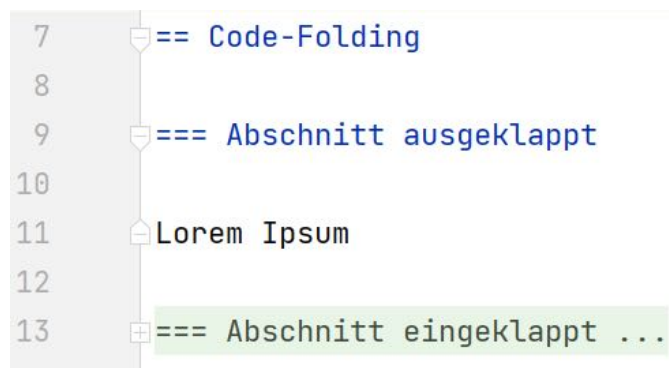
Strg+Unten and Strg+Oben will move caret down and up in the editor [Next Tip](#)



Docs-as-Code in der IDE



Docs-as-Code in der IDE



Docs-as-Code in der IDE

```

3  == Live-Templates
4
5  ad-t
6  ad-table2      Table with two columns
7  ad-tag-include  AsciiDoc Tags to be used with include macro
8  ad-title1      Title 1
9  ad-title2      Title 2
10 ad-title3      Title 3
11 ad-title4      Title 4
12 ad-title5      Title 5
13 ad-title6      Title 6
14 ad-config-toc   config-attributes for table of contents and s...
15 Press Eingabe to insert, Tabulator to replace Next Tip

```



Code einbetten

```

1  :source-highlighter: pygments
2
3  [source,java]
4  ----
5  include::MyTest.java[]
6  ----
7
8
9
10
11
12 [source,java,indent=10]
13 ----
14 include::MyTest.java[lines=2..4]
15 ----
16 <1> method declaration
17 <2> print statement
18
19
20
21
22
23
24 :source-dir: ../../src/main/java
25
26 [source,java]
27 ----
28 include::[source-dir]/de/oio/vavz/functions/CheckedFunction.java[lines=12..14]
29 ----
30

```



```

public class MyTest {
    public static void main(String[] args) { (1)
        System.out.println("Hello world!"); (2)
    }
}

```

```

public static void main(String[] args) { (1)
    System.out.println("Hello world!"); (2)
}

```

1. method declaration
2. print statement

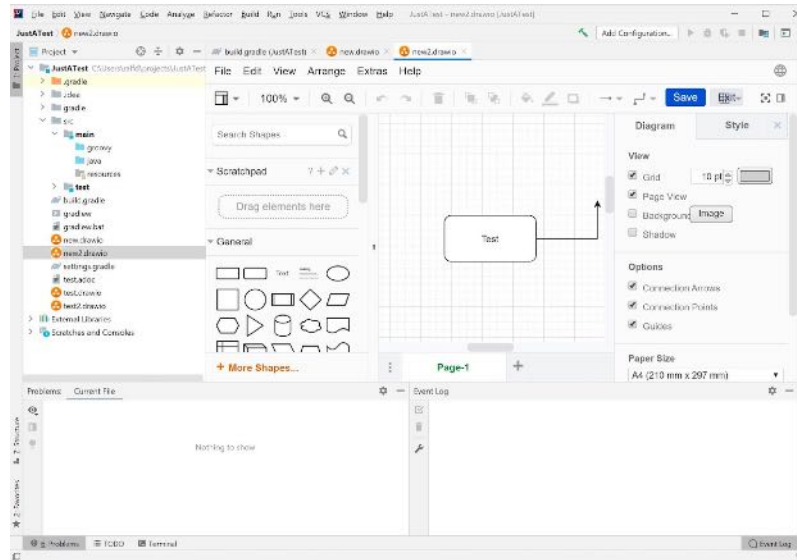
```

CheckedFunction1<Integer, Integer> readFunction = i -> readFromFile(i);
integers.stream().map(readFunction.unchecked());

```



Erweiterung über Plugins, z. B. draw.io



Docs-as-Code in der Praxis

embarc.de

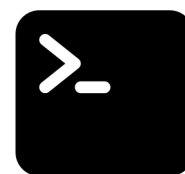
76

76

Werkzeuge



IDEs, Texteditoren



Build-Management



Textformate, Standards



Grafiken



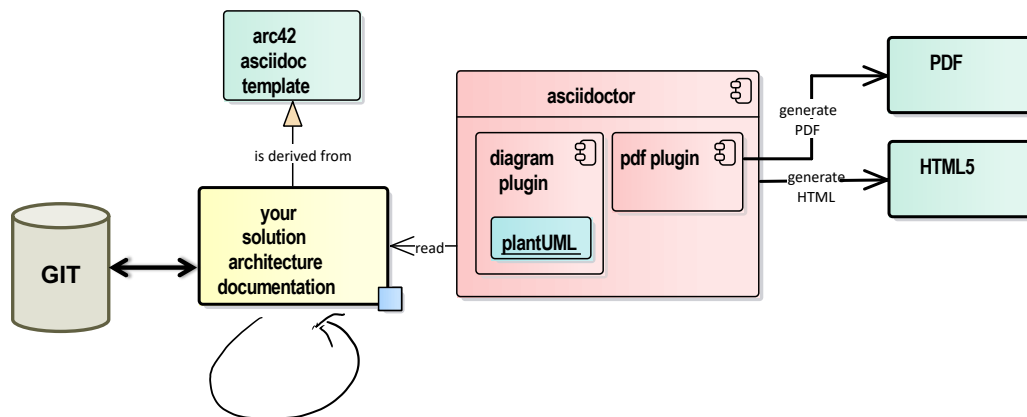
Docs-as-Code in der Praxis

embarc.de

77

77

Einfacher Workflow mit AsciiDoctor



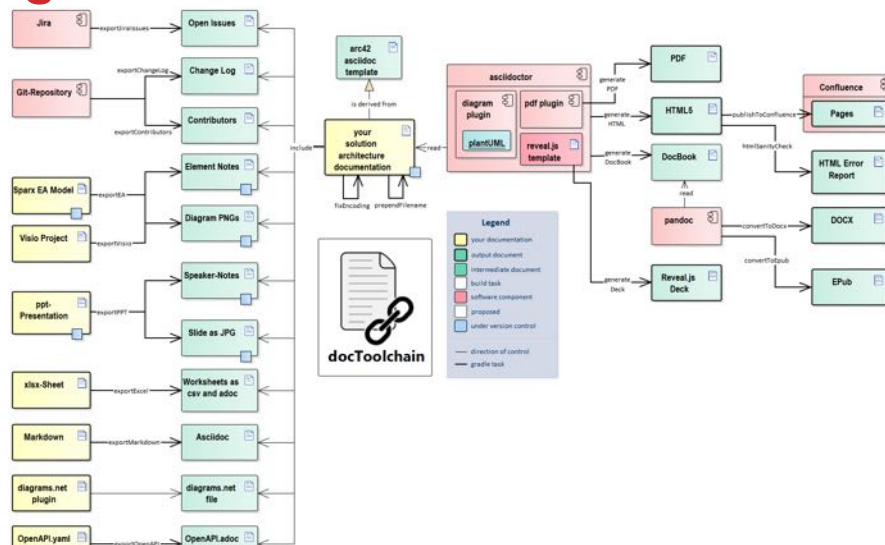
Docs-as-Code in der Praxis

embarc.de

78

78

Werkzeugkette: docToolchain



Docs-as-Code in der Praxis

embarc.de

79

79



docToolchain @docToolchain · 26. Sep.

...

A full blown docu toolchain with @arc42Tipps as template installed in just one tweet:

```
wget doctoolchain.github.io/dtcw
chmod +x dtcw
./dtcw downloadTemplate
./dtcw generateSite
```

=> doctoolchain.org



Docs-as-Code in der Praxis

embarc.de

80

80

```
fish /Users/falk/doc-project
~/doc-project tree
├── docToolchainConfig.groovy
├── dtcw
├── src
│   └── docs
│       ├── arc42
│       │   ├── arc42.adoc
│       │   └── chapters
│       │       ├── 01_introduction_and_goals.adoc
│       │       ├── 02_architecture_constraints.adoc
│       │       ├── 03_system_scope_and_context.adoc
│       │       ├── 04_solution_strategy.adoc
│       │       ├── 05_building_block_view.adoc
│       │       ├── 06_runtime_view.adoc
│       │       ├── 07_deployment_view.adoc
│       │       ├── 08_concepts.adoc
│       │       ├── 09_design_decisions.adoc
│       │       ├── 10_quality_scenarios.adoc
│       │       ├── 11_technical_risks.adoc
│       │       ├── 12_glossary.adoc
│       │       ├── about-arc42.adoc
│       │       └── config.adoc
│       └── images
│           ├── 05_building_blocks-DE.png
│           ├── 08-Crosscutting-Concepts-Structure-DE.png
│           └── arc42-logo.png
5 directories, 20 files
~/doc-project
```



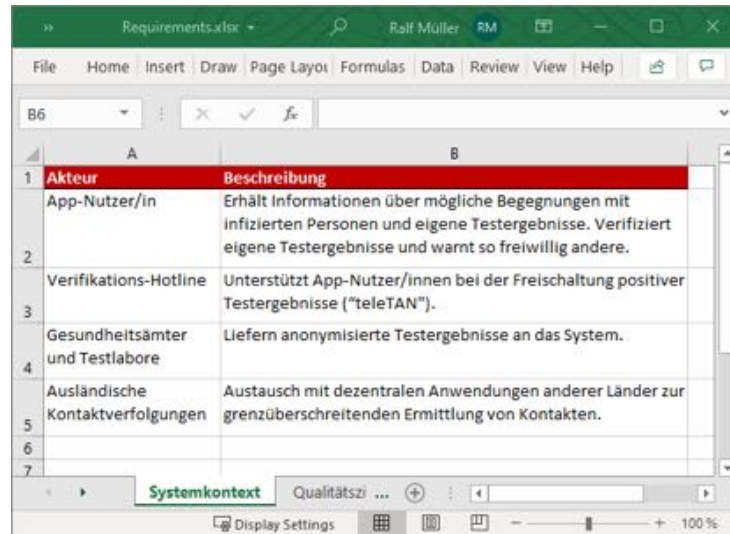
Docs-as-Code in der Praxis

embarc.de

81

81

Tabellen!?



Akteur	Beschreibung
App-Nutzer/in	Erhält Informationen über mögliche Begegnungen mit infizierten Personen und eigene Testergebnisse. Verifiziert eigene Testergebnisse und warnt so freiwillig andere.
Verifikations-Hotline	Unterstützt App-Nutzer/innen bei der Freischaltung positiver Testergebnisse ("teleTAN").
Gesundheitsämter und Testlabore	Liefern anonymisierte Testergebnisse an das System.
Ausländische Kontaktverfolgungen	Austausch mit dezentralen Anwendungen anderer Länder zur grenzüberschreitenden Ermittlung von Kontakten.



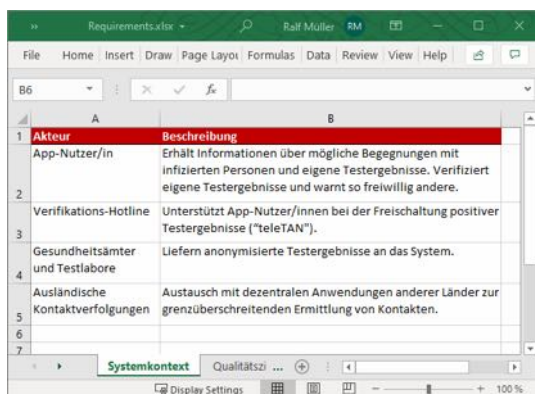
Docs-as-Code in der Praxis

embarc.de

82

82

Tabellen!?



Akteur	Beschreibung
App-Nutzer/in	Erhält Informationen über mögliche Begegnungen mit infizierten Personen und eigene Testergebnisse. Verifiziert eigene Testergebnisse und warnt so freiwillig andere.
Verifikations-Hotline	Unterstützt App-Nutzer/innen bei der Freischaltung positiver Testergebnisse ("teleTAN").
Gesundheitsämter und Testlabore	Liefern anonymisierte Testergebnisse an das System.
Ausländische Kontaktverfolgungen	Austausch mit dezentralen Anwendungen anderer Länder zur grenzüberschreitenden Ermittlung von Kontakten.



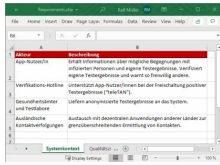
Docs-as-Code in der Praxis

embarc.de

83

83

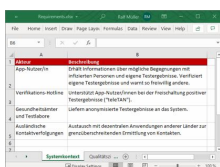
Tabellen!?



Akteur	Beschreibung
App-Nutzer/in	Erhält Informationen über mögliche Begegnungen mit infizierten Personen und eigene Testergebnisse. Verifiziert eigene Testergebnisse und warnt so freiwillig andere.
Verifikations-Hotline	Unterstützt App-Nutzer/innen bei der Freischaltung positiver Testergebnisse ("teleTAN").
Gesundheitsämter und Testlabore	Liefern anonymisierte Testergebnisse an das System.
Robert Koch-Institut (RKI)	Stellt Inhalte ("Content") für die App zur Verfügung und bestimmt Parameter für die Messung der Kontakte ("Risiko-Ermittlung"). Empfängt Auswertungen, etwa aus der Datenspende.
Ausländische Kontaktverfolgungen	Austausch mit dezentralen Anwendungen anderer Länder zur grenzüberschreitenden Ermittlung von Kontakten.



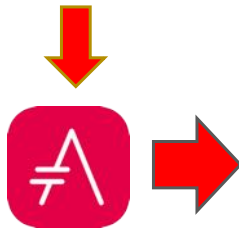
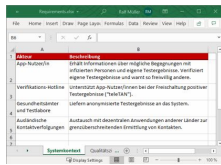
Tabellen!?



Akteur	Beschreibung
App-Nutzer/in	Erhält Informationen über mögliche Begegnungen mit infizierten Personen und eigene Testergebnisse. Verifiziert eigene Testergebnisse und warnt so freiwillig andere.
Verifikations-Hotline	Unterstützt App-Nutzer/innen bei der Freischaltung positiver Testergebnisse ("teleTAN").
Gesundheitsämter und Testlabore	Liefern anonymisierte Testergebnisse an das System.
Robert Koch-Institut (RKI)	Stellt Inhalte ("Content") für die App zur Verfügung und bestimmt Parameter für die Messung der Kontakte ("Risiko-Ermittlung"). Empfängt Auswertungen, etwa aus der Datenspende.
Ausländische Kontaktverfolgungen	Austausch mit dezentralen Anwendungen anderer Länder zur grenzüberschreitenden Ermittlung von Kontakten.



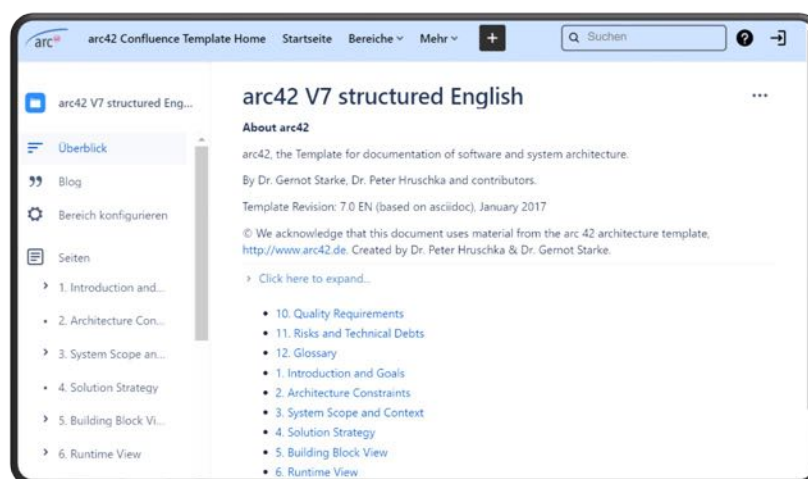
Tabellen!?



Akteur	Beschreibung
App-Nutzer/in	Erhält Informationen über mögliche Begegnungen mit infizierten Personen und eigene Testergebnisse. Verifiziert eigene Testergebnisse und warnt so freiwillig andere.
Verifikations-Hotline	Unterstützt App-Nutzer/innen bei der Freischaltung positiver Testergebnisse ("teleTAN").
Gesundheitsämter und Testlabore	Liefern anonymisierte Testergebnisse an das System.
Robert Koch-Institut (RKI)	Stellt Inhalte ("Content") für die App zur Verfügung und bestimmt Parameter für die Messung der Kontakte ("Risiko-Ermittlung"). Empfängt Auswertungen, etwa aus der Datenspende.
Ausländische Kontaktverfolgungen	Austausch mit dezentralen Anwendungen anderer Länder zur grenzüberschreitenden Ermittlung von Kontakten.



publishToConfluence



Schnittstellenbeschreibung generieren


[Swagger2Markup / swagger2markup](#)
Watch 15
Star 294
Fork 54

[Code](#)
[Issues 7](#)
[Pull requests 0](#)
[Pulse](#)
[Graphs](#)

A Swagger to AsciiDoc or Markdown converter to simplify the generation of an up-to-date RESTful API documentation by combining documentation that's been hand-written with auto-generated API documentation.

Spring REST Docs

SUCCESS

Overview

The primary goal of this project is to make it easy to document RESTful services by combining content that's been hand-written using [Asciidoctor](#) with auto-generated examples produced with the [Spring MVC Test](#) framework. The result is intended to be an easy-to-read user guide, akin to [GitHub's API documentation](#) for example, rather than the fully automated, dense API documentation produced by tools like [Swagger](#).

JAX-RS Analyzer

Generates an overview of all JAX-RS resources in a project by bytecode analysis.



Werkzeuge



IDEs, Texteditoren



Build-Management



Textformate, Standards



Grafiken





Michael Simons
@rothroll666



Folge ich

Think I'm more a Markdown person than AsciiDoc. The results are great, but Markdown needs less concentration to write.



Markdown

Markdown

- Toller Standard für einfache Auszeichnungen
- Features

Span Elements

Links
Emphasis
Code
Images

Block Elements

Paragraphs and Line Breaks
Headers
Blockquotes
Lists
Code Blocks
Horizontal Rules



TOC

Tables (Feature Rich)
Includes (Level-Offset)
PlantUML
Admonitions
Attributes
Anchors
Footnotes, Index, Glossary
Videos
Syntax Highlighting
Callouts
Math Rendering
Outputformats



Markdown

Wir brauchen eine Erweiterung!

Welche wählen wir?

- [CommonMark](#)
- [CriticMarkup](#)
- [Discount](#)
- [DocFX](#)
- [ExtraMark](#)
- [Ghost's Markdown/Haunted Markdown](#)
- [GitHub Flavored Markdown](#)
- [GitLab Flavored Markdown \(with login\)](#)
- [Haroopad Flavored Markdown](#)
- [iA Writer's Markdown](#)
- [Kramdown](#)
- [Leanpub Flavored Markdown](#)
- [Litedown](#)
- [Lunamark](#)
- [Madoko](#)
- [Markdown](#)
- [Markdown 2](#)
- [Markdown Extra](#)
- [Markdown-it](#)
- [Markua](#)
- [Maruku](#)
- [MultiMarkdown](#)
- [Pandoc's Markdown](#)
- [PHP Markdown Extra Extended](#)
- [Python Markdown](#)
- [R Markdown](#)
- [Redcarpet](#)
- [Remarkable](#)
- [Rhythmus](#)
- [Scholarly Markdown](#)
- [Showdown](#)
- [StackOverflow's Markdown](#)
- [Taiga Markdown](#)
- [Trello's Markdown](#)
- [vfm](#)
- [Xcode/Swift Playgrounds Markup](#)

<https://github.com/commonmark/commonmark-spec/wiki/markdown-flavors>



Markdown

einen Dialekt

Wir brauchen eine ~~Erweiterung!~~

Welche wählen wir?

Funktioniert dann unsere Toolchain noch?

Gibt es einen Editor-Preview?



AsciiDoc / AsciiDoctor

leistungsfähige Syntax für technische Dokumentation
in Ruby geschrieben
mit Opal nach JavaScript transpiliert
mit jRuby auf der JVM gewrapped

=> Keine Dialekte!



Michael Simons
@rotnroll666



Folge ich

Think I'm more a Markdown person than AsciiDoc. The results are great, but Markdown needs less concentration to write.



Ralf D. Müller
@RalfDMueller

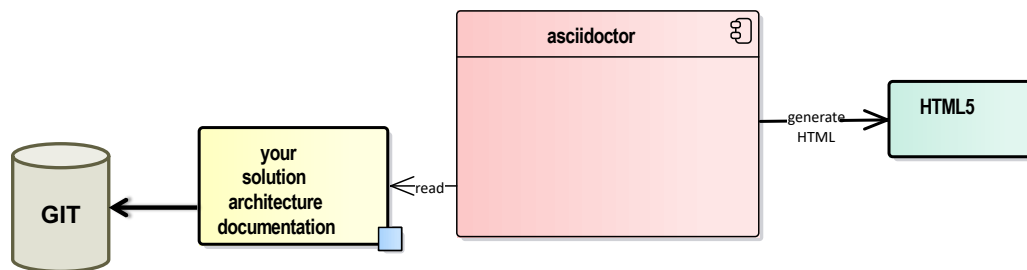


Folgen

@rotnroll666 but the possibilities of AsciiDoc are better! Include code directly from the repository, render plantUML, subdocuments etc...

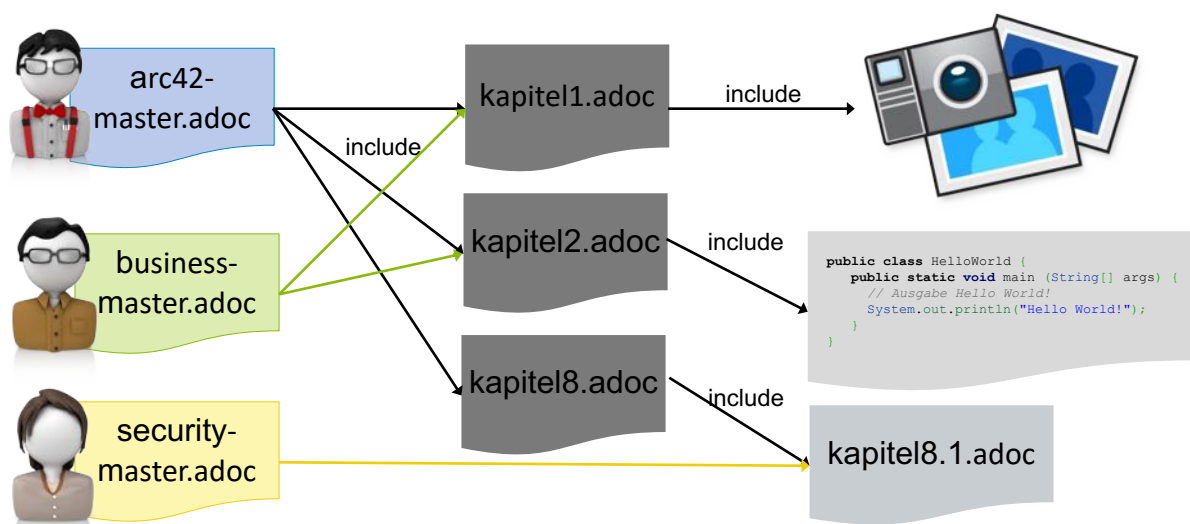


Basic Docs-as-Code with AsciiDoc



97

Modulare Dokumentation



98

Out-of-the-Box Features

„ablenkungsfrei“ – Dokumentation wie E-Mails schreiben

Gliederung in Unterdokumente

Neugliederung je nach Stakeholder

Bilder werden referenziert, nicht eingebettet

leichte Versionierung „handle Docs-as-Code“

Formatierung von Source-Code

Reviews, Pull-Requests, Versionierung durch Git

Konvertierung nach HTML5 und DocBook



Docs-as-Code in der Praxis

embarc.de

99

99

99

7 Regeln für gute Dokumentation



1. Schreibe aus Sicht des Lesers

2. Vermeide unnötige Wiederholungen

3. Vermeide Mehrdeutigkeiten

3. a) Erkläre Deine Notation

4. Verwende eine Standardstrukturierung

5. Halte Begründungen für Entscheidungen fest

6. Halte die Dokumentation aktuell, aber auch nicht zu aktuell

7. Überprüfe Dokumentation auf ihre Gebrauchstauglichkeit

„Documenting Software Architectures:
Views and Beyond“
Clements, et.al, 2. Auflage 2010



Docs-as-Code in der Praxis

embarc.de

100

100

„Abheften“ in arc42

1. Einführung und Ziele 1.1 Aufgabenstellung 1.2 Qualitätsziele 1.3 Stakeholder	7. Verteilungssicht 7.1 Infrastruktur Ebene 1 7.2 Infrastruktur Ebene 2 ...
2. Randbedingungen 2.1 Technische Randbedingungen 2.2 Organisatorische Randbedingungen 2.3 Konventionen	8. Konzepte 8.1 Fachliche Strukturen und Modelle 8.2 Typische Muster und Strukturen 8.3 Persistenz 8.4 Benutzungsoberfläche ...
3. Kontextabgrenzung 3.1 Fachlicher Kontext 3.2 Technischer- oder Verteilungskontext	9. Entwurfsentscheidungen 9.1 Entwurfsentscheidung 1 9.2 Entwurfsentscheidung 2 ...
4. Lösungsstrategie	10. Qualitätsszenarien 10.1 Qualitätsbaum 10.2 Bewertungsszenarien
5. Bausteinsicht 5.1 Ebene 1 5.2 Ebene 2 ...	11. Risiken
6. Laufzeitsicht 6.1 Laufzeitszenario 1 6.2 Laufzeitszenario 2 ...	12. Glossar



Dr. Peter Hruschka

<http://www.peterhruschka.eu/>



Dr. Gernot Starke

<http://gernotstarke.de/>



<https://arc42.de/>



Docs-as-Code in der Praxis

embarc.de

101

101

1. Requirements & Goals

2. Constraints

3. Scope & Context

4. Solution Strategy

5. Building Block View

6. Runtime View



7. Deployment View

8. Crosscutting Concepts

9. Decisions

10. Quality Scenarios

11. Risks & Tech Debt

12. Glossary



Docs-as-Code in der Praxis

embarc.de

102

102

Ein anderer Blick: Simon Brown



Simon Brown:
Software Architecture for Developers

Technical leadership by coding, coaching, collaboration, architecture sketching and just enough up front design

→ <http://leanpub.com/>



Docs-as-Code in der Praxis

embarc.de

103

103

Ein anderer Blick: Simon Brown



Simon Brown:
Software Architecture for Developers, Volume 2

Visualize, document and explore your software architecture

→ <http://leanpub.com/>



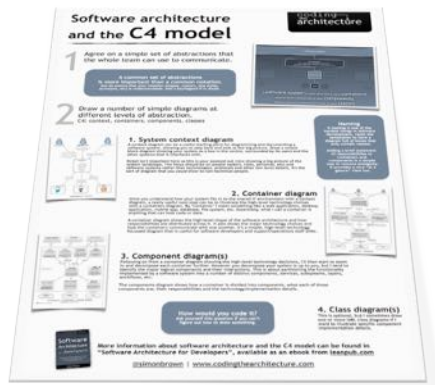
Docs-as-Code in der Praxis

embarc.de

104

104

Simon Brown: C4 Model



- Agree on a simple set of abstractions that the whole team can use to communicate.
- Draw a number of simple diagrams at different levels of abstraction.
- C4: context, containers, components, classes (code)

→ <http://static.codingthearchitecture.com/c4.pdf>



Werkzeuge



IDEs, Texteditoren



Build-Management



Textformate, Standards

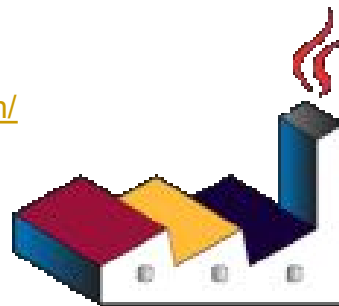


Grafiken



Diagramme

- <http://plantuml.com/>
- <http://asciidoctor.org/docs/asciidoctor-diagram/>
- Komplexe Diagramme als einfachen Text verwalten



PlantUML

```

1 @startuml
2
3 User -> Server: via Browser
4   Server -> Database: fetch profile
5   Server <-- Database
6 User <-- Server
7
8 @enduml

```

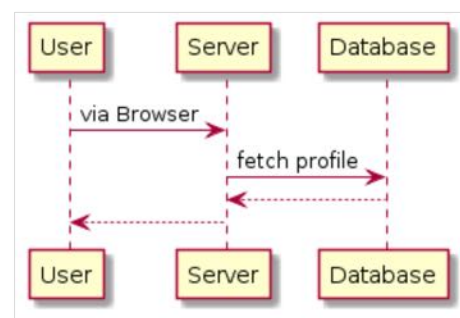


Diagramme als textuelle Beschreibung

```
@startuml
skinparam handwritten true
```

```
actor :Conference Participant:
```

```
cloud "DukeCon Infra" #lightgray {
[DukeCon] <<Service>>
}
```

```
cloud "DOAG" {
[Backoffice] <<Service>>
[Vortragsplaner] <<Service>>
}
```

```
:Conference Participant: -right-> (DukeCon) : Read/Compile
conference program (via mobile app)
:Conference Participant: -right-> (Vortragsplaner) :
Read/Compile conference program (old style)
```

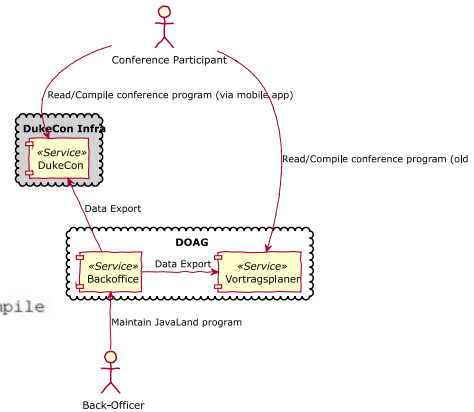
```
(Backoffice) -left-> (DukeCon) : Data Export
(Backoffice) -left->
```

```
actor :Back-Officer:
```

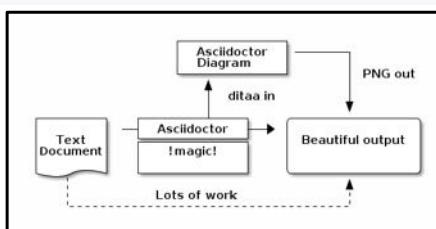
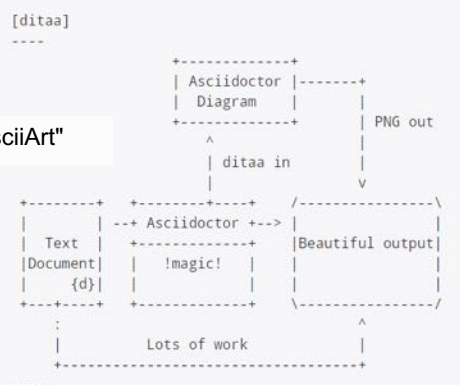
```
:Back-Officer: -left-
program
@enduml
```

```
["plantuml", "dukecon-systemcontext", "svg"]
```

```
include:../systemcontext.puml[]
```

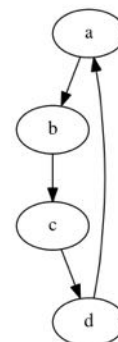


"AsciiArt"



```
[graphviz, dot-example, svg]
```

```
digraph g {
a -> b
b -> c
c -> d
d -> a
}
```



Kroki

Creates diagrams from textual descriptions!

Kroki provides a unified API with support for BlockDiag (BlockDiag, SeqDiag, ActDiag, NwDiag, PacketDiag, RackDiag), BPMN, Bytefield, C4 (with PlantUML), Dita, Erd, Excalidraw, GraphViz, Mermaid, Nomnoml, PlantUML, SvgBob, UMLet, Vega, Vega-Lite, WaveDrom... and more to come!

#Features

Ready to use

Diagrams libraries are written in a variety of languages: Haskell, Python, JavaScript, Go, PHP, Java... some also have C bindings. Trust us, you have better things to do than install all the requirements to use them. Get started in no time!

Simple

Kroki provides a unified API for all the diagram libraries. Learn once use diagrams anywhere!

Free & Open source

All the code is available on GitHub and our goal is to provide Kroki as a free service.

Fast

Built using a modern architecture, Kroki offers great performance.

<https://kroki.io/>

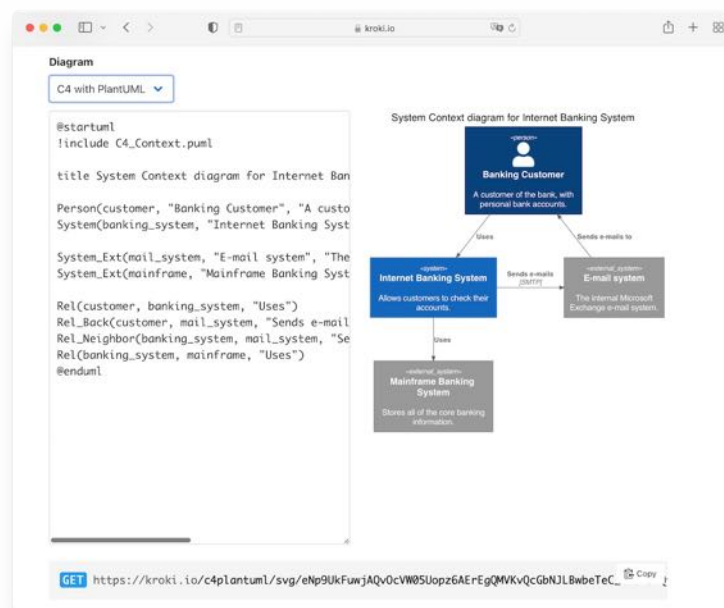


Docs-as-Code in der Praxis

embarc.de

111

111



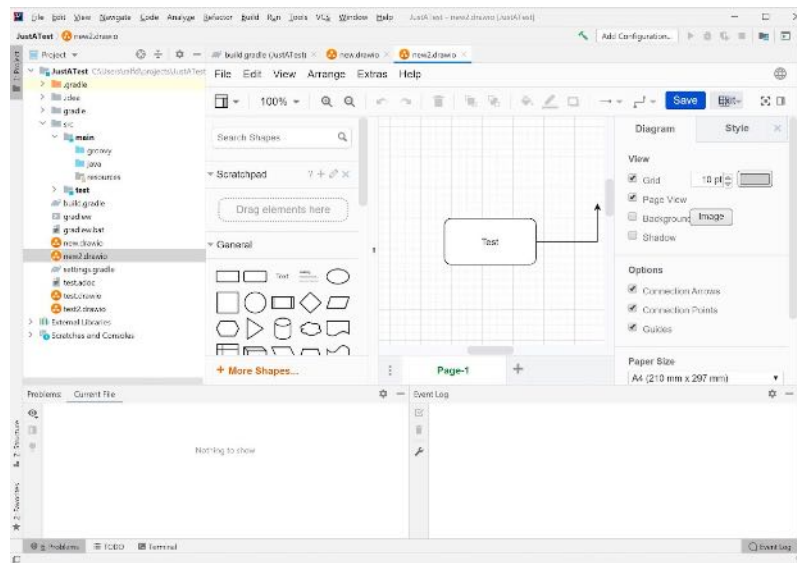
Docs-as-Code in der Praxis

embarc.de

112

112

Draw.io/Diagrams.net und Co.



Agenda



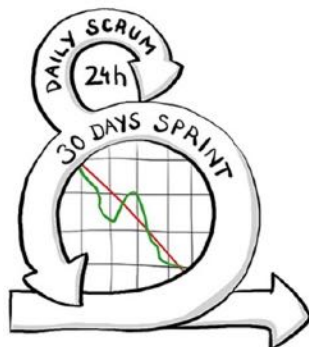
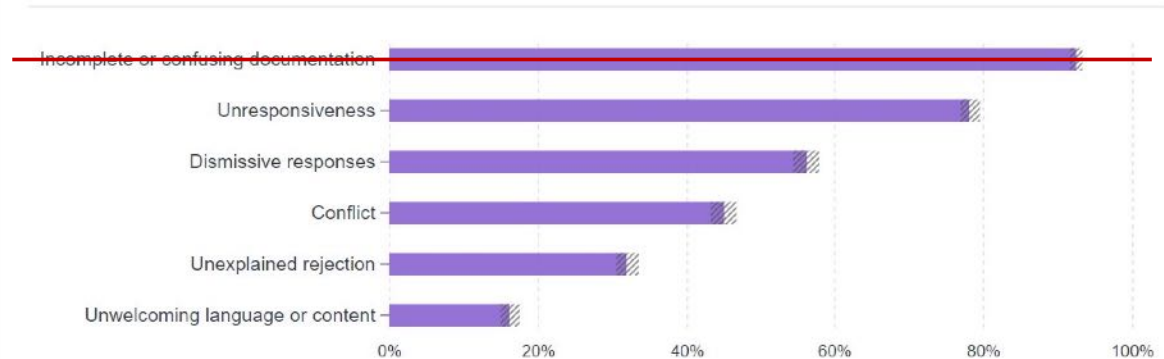
- 1 Einstieg und Motivation
- 2 Docs-as-Code Maturity Level
- 3 Werkzeuge
- 4 Fazit und Ausblick**

4

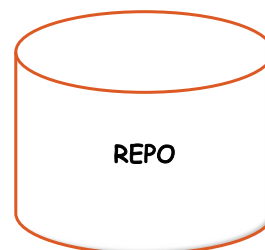


Fig1. - Problems encountered in open source

Source: opensourcesurvey.org



**Continuous
Documentation**



**Documentation
as Code**



Moderne (Architektur-)Dokumentation



Textformate



Single Source
of Truth



Inhalte
generieren



Bilder/Grafiken



Docs-as-Code



Continuous
Documentation



Ausführen/
Validieren



Kümmerer



Docs-as-Code in der Praxis

embarc.de

118

118

Home > Trainings > Documentation-as-Code

Documentation-as-Code

Training Softwarearchitekturdokumentation praktisch mit Entwicklerwerkzeugen umsetzen - 2 Tage

embarc INNOQ

-- BESCHREIBUNG

Architektur-Dokumentation wird sehr oft stiefmütterlich behandelt. Dabei unterstützt das Dokumentieren bei der Entwurfsarbeit, schafft Transparenz bzw. Leitplanken für die Umsetzung, Wartung und Bewertung der Softwarearchitektur. Einerseits ist es aber nicht einfach, die wichtigen Informationen aus dem Entwurf der Softwarearchitektur strukturiert und leicht verständlich festzuhalten. Andererseits enden die meisten Versuche, auf der Suche nach einer praktikablen Handhabung zur Erstellung und Pflege, in der WYSIWYG-Hölle einer Textverarbeitung oder im tiefen Schlund eines Wikis. Dieses Seminar zeigt aufbauend auf leichtgewichtigen Tools und Textformaten die Erstellung einer möglichst redundanzfreien Dokumentation, die für verschiedene

Online-Termine

TERMINE AUF ANFRAGE

Vor-Ort-Termine

Doc-as-Code (Darmstadt)
10.-11. Juli 2023
Best Western Darmstadt
[jetzt buchen](#)

SOCREATORY



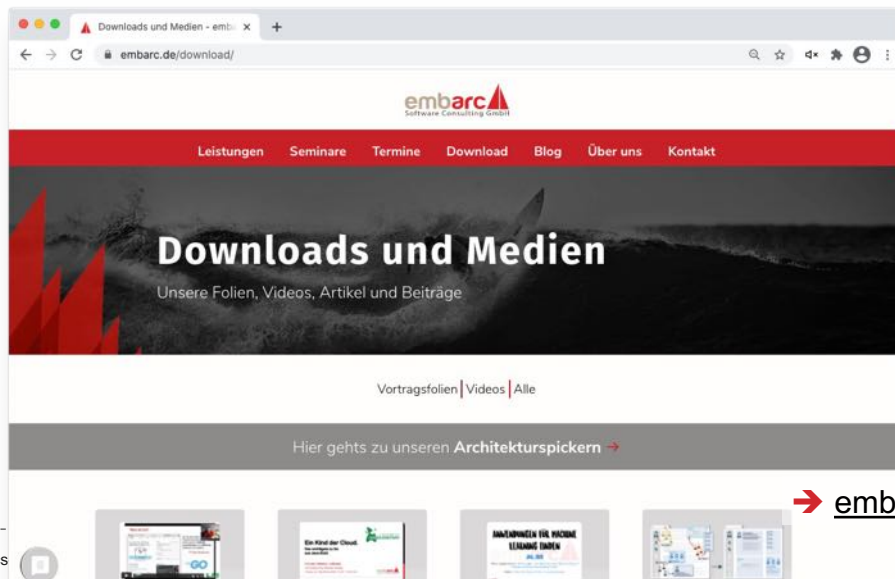
Docs-as-Code in der Praxis

embarc.de

119

119

Folien von heute als PDF zum Download



→ embarc.de/download/

120

120

Flyer zum CWA Architekturüberblick



Wir schicken Ihnen gerne ein gedrucktes Exemplar dieses Überblicks als Flyer Einfach eine E-Mail senden an

info@embarc.de

mit Betreff „CWA-Flyer“ und Eurer Postadresse im Text, dann geht es los ...

<https://www.embarc.de/architektur-ueberblicke/#cwa>



Docs-as-Code in der Praxis

embarc.de

121

121

Vielen Dank.

Ich freue mich auf Eure Fragen!



Falk Sippach



fs@embarc.de



@sippsack



→ [xing.to/fsi](https://www.xing.to/fsi)

