Kotlin kontra Java

Braucht man 2026 überhaupt noch Kotlin?

Richard Gross (er/ihm)


IT Sanierung


Hypermedia


Programmier-Sprachen

speakerdeck.com/richargh

richargh.de/

richargh.de

JAVAFORUM stuttgart

09.07.26

MAIBORNWOLFF

Erst Code oder erst Analyse?

Was wollt ihr zuerst sehen?

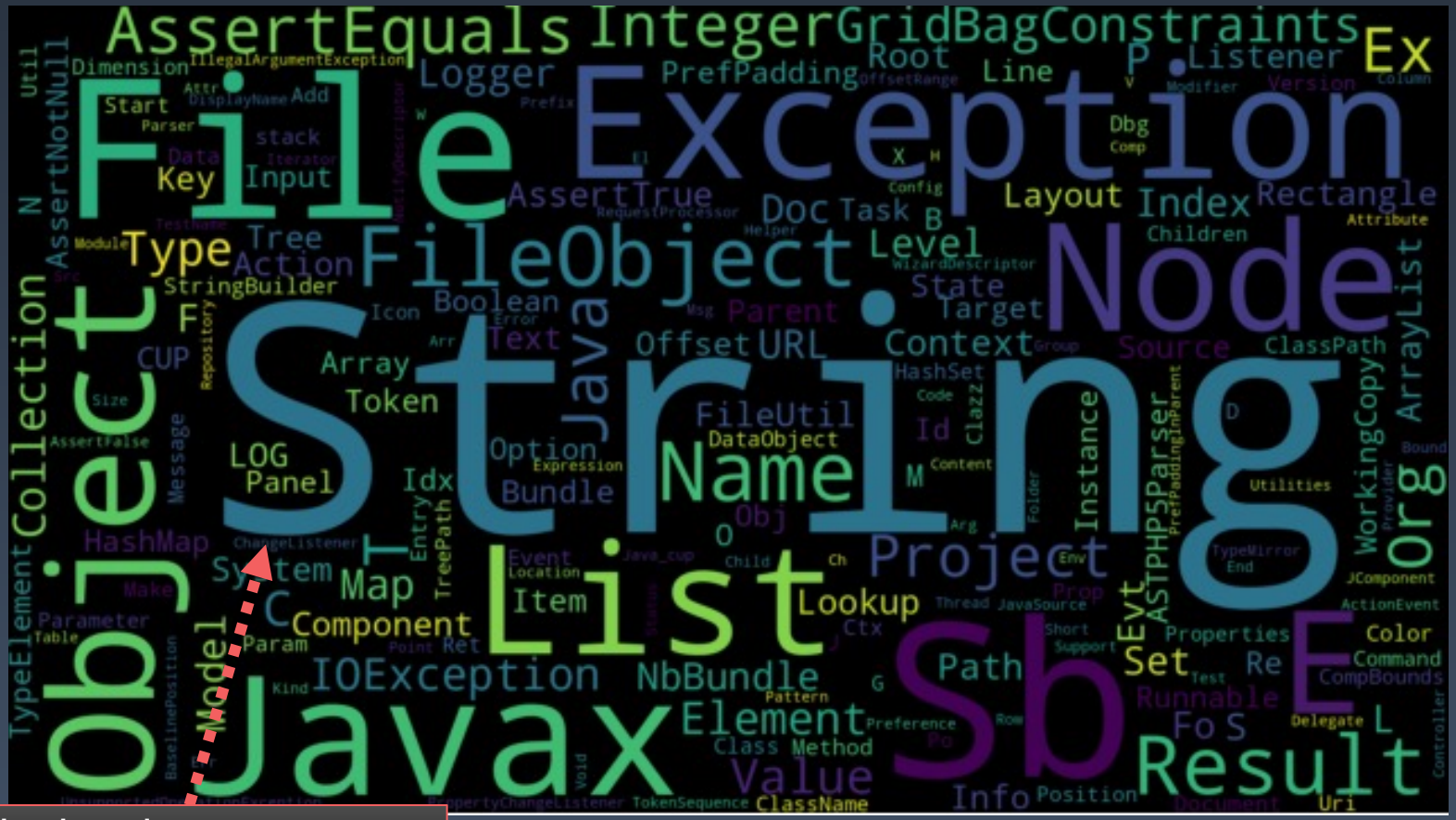
Was sind eigentlich unsere Kriterien für Code?

Kriterien für Code

Kotlin	Java	
?	?	Code liest sich wie Fachprosa
?	?	(null-)Sicherheit zur Compile-Zeit
?	?	Kurz & knapp
?	?	Interoperabilität mit Platform
?	?	Erweiterte Nebenläufigkeit

Ein Wort der Warnung

Sowas versuche ich zu vermeiden

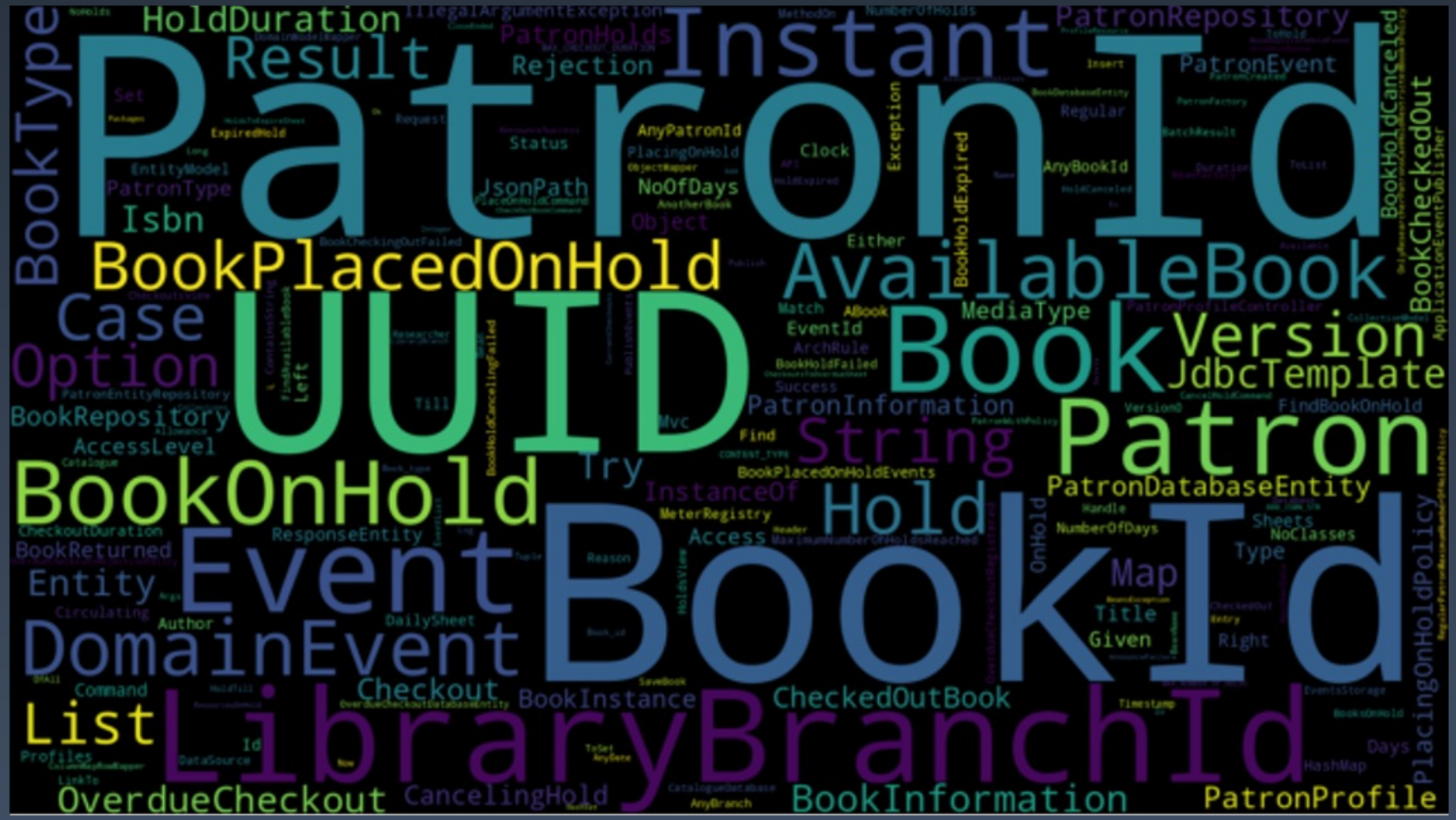


Stringly oder Strongly Typed?

Slide 6 by richargh.de from

Generiert mit: https://github.com/Richargh/code-tagcloud-py-sandbox

Ich programmiere gerne typsicher



Generiert mit: https://github.com/Richargh/code-tagcloud-py-sandbox

Live-Coding

Wie sonst Programmiersprache bewerten.

https://github.com/Richargh/kotlin-kontra-java

Kriterien für Code

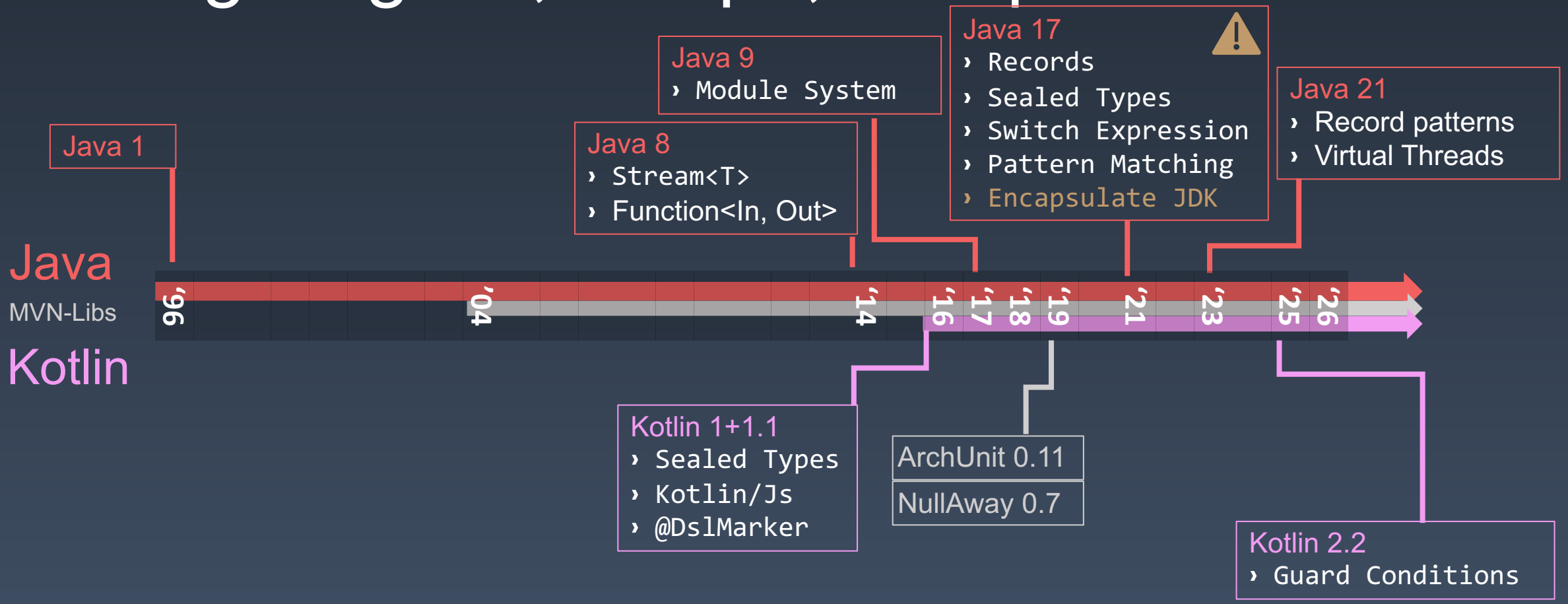
Kotlin	Java	
⇑	⇗	Code liest sich wie Fachprosa
⇑	⇗	(null-)Sicherheit zur Compile-Zeit
⇑	⇗	Kurz & knapp
⇑	⇑	Interoperabilität mit Platform
⇗ Überall suspend	⇗ Nicht so schick; Structured Concurrency nicht final	Erweiterte Nebenläufigkeit

Kriterien für Ökosysteme

	Kotlin	Java
Langlebigkeit, Tempo, Kompatibilität	?	?
Verfügbarkeit von Entwicklern	?	?
IDE-Unterstützung	?	?
Performance	?	?
Ausgereifte Bibliotheken und Frameworks	?	?
Eine Sprache für alles (bonus)	?	?

Fundament: Passt zum Einsatzzweck

Langlebigkeit, Tempo, Kompatibilität



Verfügbarkeit von Entwicklern?

Github PR + SO Activity

Redmonk 01-2025[1]

1. JavaScript
2. Python
3. Java

[…]

14. Kotlin

[…]

19. Rust

Interest (Search, Jobs, …)

IEEE Spectrum 2025[2]

1. Python (1)
2. Java (0,50)

[…]

13. Kotlin (0,13)
14. Rust (0,13)

Survey: done extensive work?

StackOverflow 2025[3]

1. JavaScript (69%)

[…]

8. Java (30%)

[…]

14. Rust (15%)
15. Kotlin (12%)

1 https://redmonk.com/sogrady/2025/06/18/language-rankings-1-25/
2 https://spectrum.ieee.org/top-programming-languages-2025
3 https://survey.stackoverflow.co/2025/technology
plus (Kotlin adopt since 2018) https://www.thoughtworks.com/radar/languages-and-frameworks/kotlin

Kotlin IDE Support ist massiv besser geworden durch offiziellen LSP

IntelliJ

Eclipse

VS Code

(Neo)Vim
(and other editors)

Microsoft/Eclipse/Oracle LSPs

	✓	✓	✓ by Microsoft[1] or Oracle	(✓) Community maintained[2]
	✓	✓ by JetBrains	(✓) JetBrains LSP pre-alpha[3] ~ community (un)maintainted	✗ did not find maintened plugin

LSP: Language Server Protocol
Kotlin IDEs: https://kotlinlang.org/docs/kotlin-ide.html#eclipse
1 https://marketplace.visualstudio.com/items?itemName=vscjava.vscode-java-pack
2 https://github.com/nvim-java/nvim-java
3 https://kotl.in/lsp

Beide bieten genug Performance[1]

bench.b
Execute Brainf*

1,012s Rust
1,111s C/gcc
1,188s Java
1,227s Kotlin
1,247s Go
1,367s C#/.NET Core

2,777s Scala
3,150s NodeJs
9,575s Python/pypy

Base64
encode/decode

0,663s NodeJs
0,875s Rust
0,918s C/gcc
1,433s Java
1,531s Scala
1,567s Kotlin
1,659s Go

2,494s C#/.NET Core
3,249s Python/pypy

Matmul
Multiply matrixes

3,034s C/gcc
3,064s Rust
3,107s Java
3,153s Go
3,197s Kotlin
3,201s NodeJs
3,263s Python/pypy

3,295s Scala

4,890s C#/.NET Core

1 https://github.com/kostya/benchmarks?tab=readme-ov-file#tests-execution
s=Sekunden Java=JDK23, Kotlin=2.0.21, Scala=3.5.2, C#=Core 4x oder8?
What (Staged) means: https://github.com/kostya/benchmarks/issues/485
Alternative1: https://programming-language-benchmarks.vercel.app/
Alternative2: https://benchmarksgame-team.pages.debian.net/benchmarksgame/index.html
Alternative 3: https://sites.google.com/view/energy-efficiency-languages/results?authuser=0

Multiplatform Java?

Es gibt Projekte:

- iOS/Android: Gluon[1] (third-party)
- iOS/Android: OpenJDK Mobile[2] (experimentell)
- Web: CheerpJ[3] (third-party)
- Web: Graal Web Image Target[3] (experimentell)

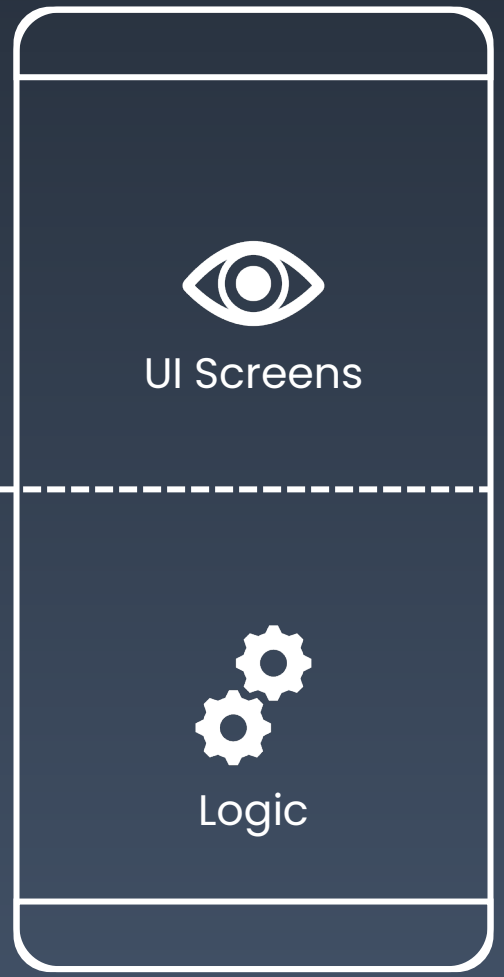
Aber nichts offizielles und oft auf JavaFx aufbauend

1 https://gluonhq.com/products/javafx/
2 https://openjdk-mobile.github.io/openjdk-mobile/ und https://openjdk.org/projects/mobile/
3 https://cheerpj.com/
4 https://www.graalvm.org/latest/reference-manual/web-image/

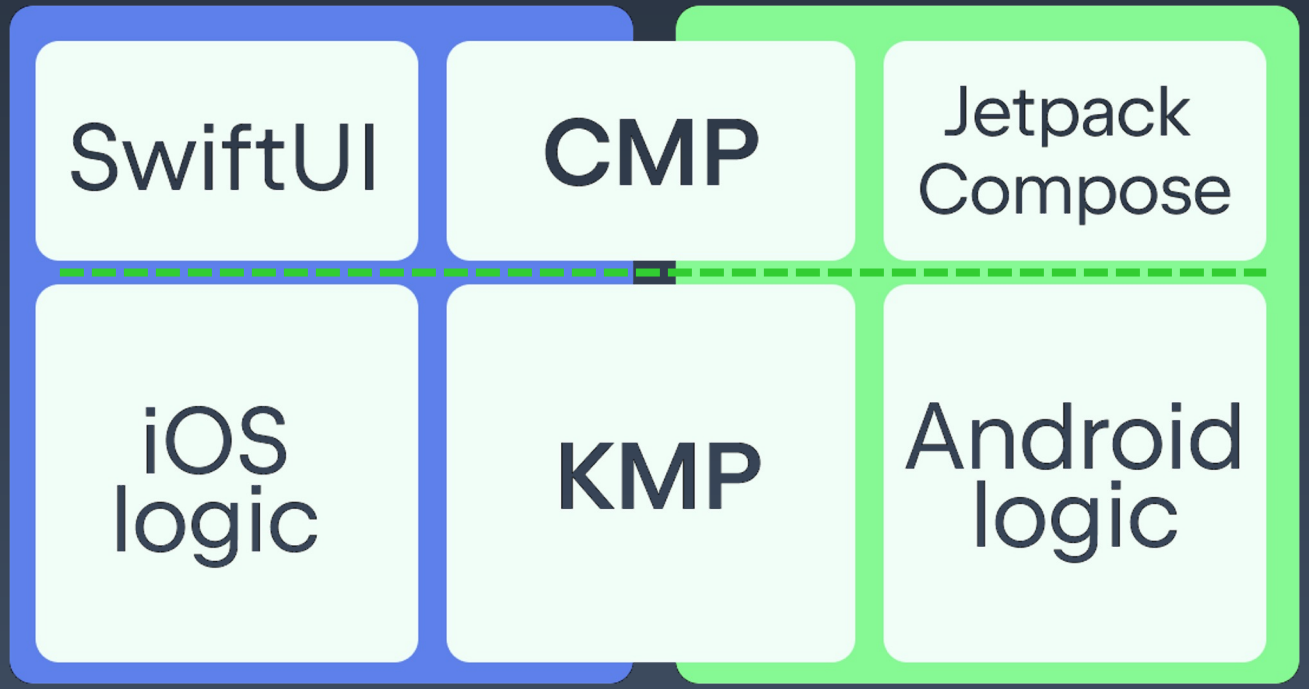
Multiplatform mit Kotlin

Compose
Multi
Platform
(CMP)

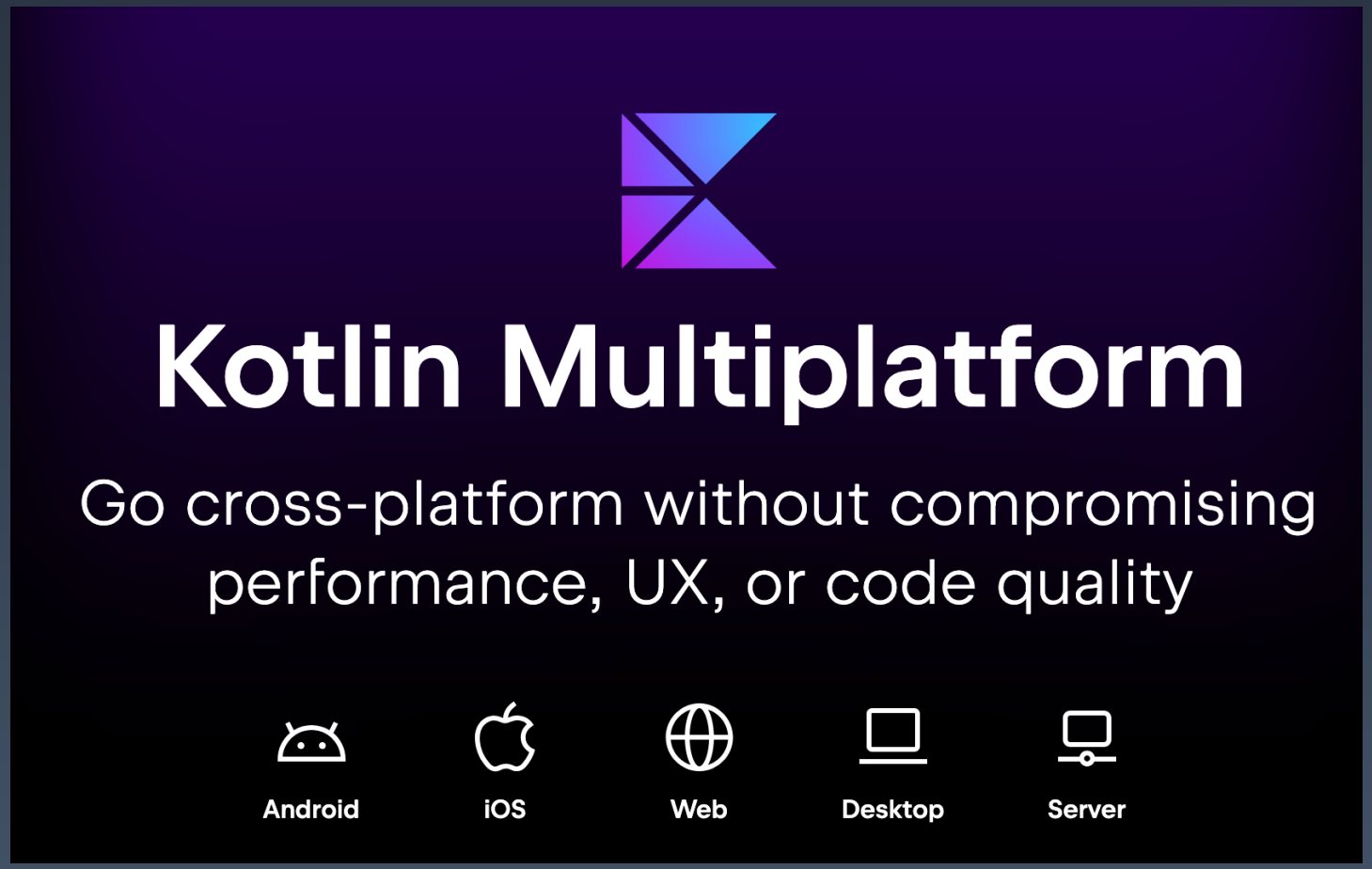
Kotlin
Multi
Platform
(KMP)



Man muss nicht alles mit Kotlin bauen



Kotlin ist ein bisschen umfangreicher



Kriterien für Ökosysteme

	 Kotlin	 Java
Langlebigkeit, Tempo, Kompatibilität	⇑	⇑
Verfügbarkeit von Entwicklern	⇑	⇑
IDE-Unterstützung	⇑	⇑
Performance	⇑	⇑
Ausgereifte Bibliotheken und Frameworks	⇑	⇑
Eine Sprache für alles (bonus)	⇗ CMP iOS noch jung	⇒ Nur logic, nur WASM

Fundament: Passt zum Einsatzzweck

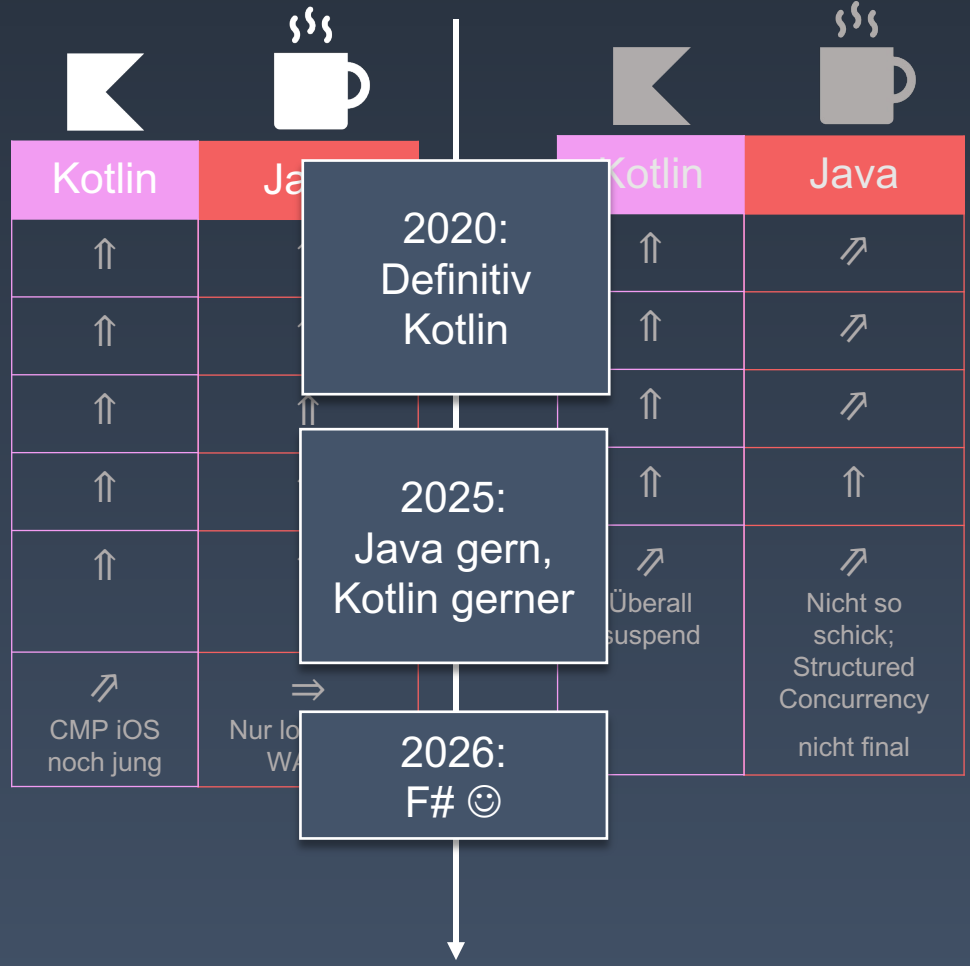
Kriterien für Programmiersprachen

	Kotlin	Java
Langlebigkeit, Tempo, Kompatibilität	⇑	⇑
Verfügbarkeit von Entwicklern	⇑	⇑
IDE-Unterstützung	⇑	⇑
Performance	⇑	⇑
Ausgereifte Bibliotheken und Frameworks	⇑	⇑
Eine Sprache für alles (bonus)	⇗ CMP iOS noch jung	⇒ Nur logic, nur WASM

Kotlin	Java	
⇑	⇗	Code liest sich wie Fachprosa
⇑	⇗	(null-)Sicherheit zur Compile-Zeit
⇑	⇗	Kurz & knapp
⇑	⇑	Interoperabilität mit Platform
⇗ Überall suspend	⇗ Nicht so schick; Structured Concurrency nicht final	Erweiterte Nebenläufigkeit

Fundament: Passt zum Einsatzzweck

Braucht man 2025 überhaupt noch Kotlin?



Langlebigkeit, Tempo, Kompatibilität

Verfügbarkeit von Entwicklern

IDE-Unterstützung

Performance

Ausgereifte Bibliotheken und Frameworks

Eine Sprache für alles (bonus)

Code liest sich wie Fachprosa

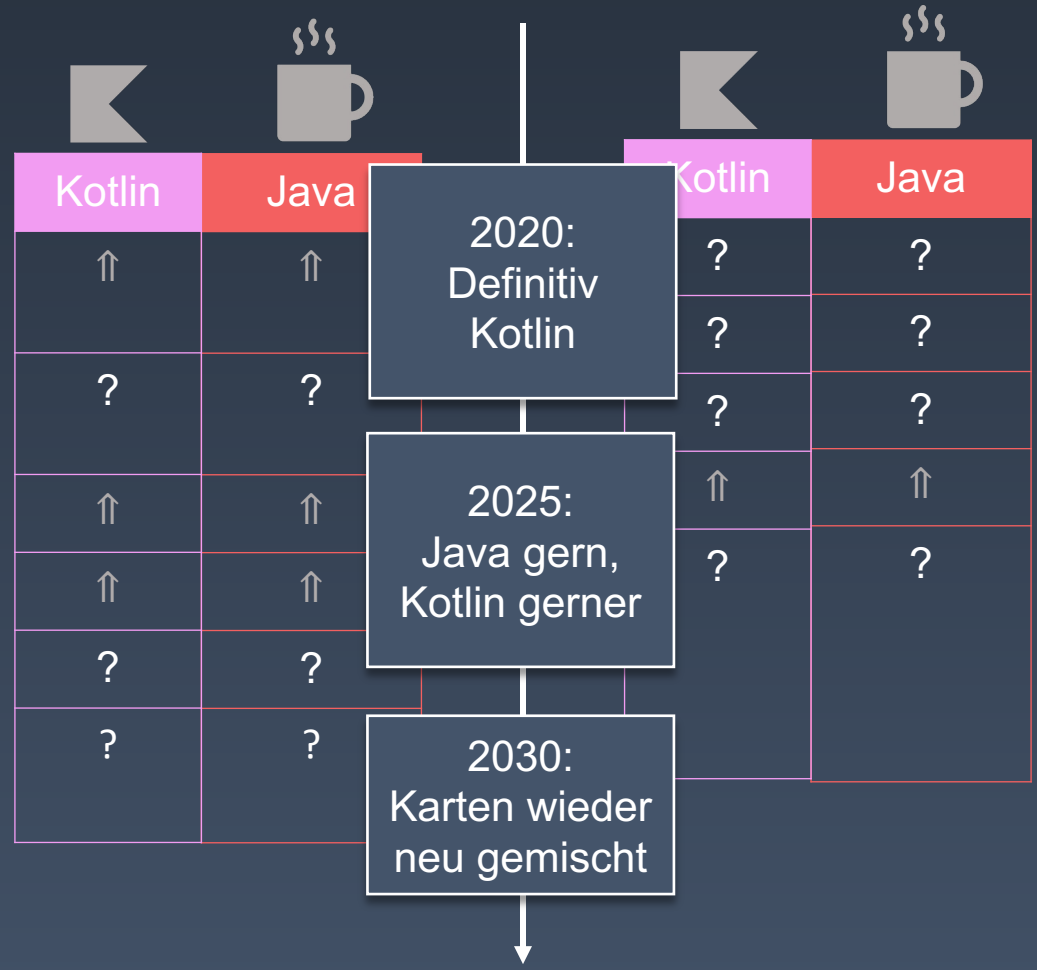
(null-)Sicherheit zur Compile-Zeit

Kurz & knapp

Interoperabilität mit Platform

Erweiterte Nebenläufigkeit

Braucht man 2030 überhaupt noch Kotlin?



Erwartete Langlebigkeit, mit guter Kompatibilität

Genug Entwickler, die sie auch lernen wollen, wenig footguns

Guter IDE-Support

Reifes Ökosystem

Schnell und Schnelles Feedback

Eine Sprache für alles (bonus)

Code liest sich wie Fachprosa

(null-)Sicherheit zur Compile-Zeit

Kurz & knapp

Interoperabilität mit Platform

Erweiterte Nebenläufigkeit

Alles nochmal detailliert als Artikel-Serie

Teil 1/5 auf Deutsch

https://javapro.io/de/kotlin-kontra-java-teil-1-oekosystem/

Part 1/5 in English

https://javapro.io/2026/04/16/kotlin-kontra-java-part-1-ecosystem/

Die Serie beinhaltet 5 Teile. Die englischen Teile erscheinen typischerweise vor den deutschen.

Dankeschön

Gerne auf diese Themen ansprechen ☺

Kontakt

Richard Gross (er/ihm)

IT Sanierung

Hypermedia

Programmier-Sprachen

richargh.de

richargh.de

richargh

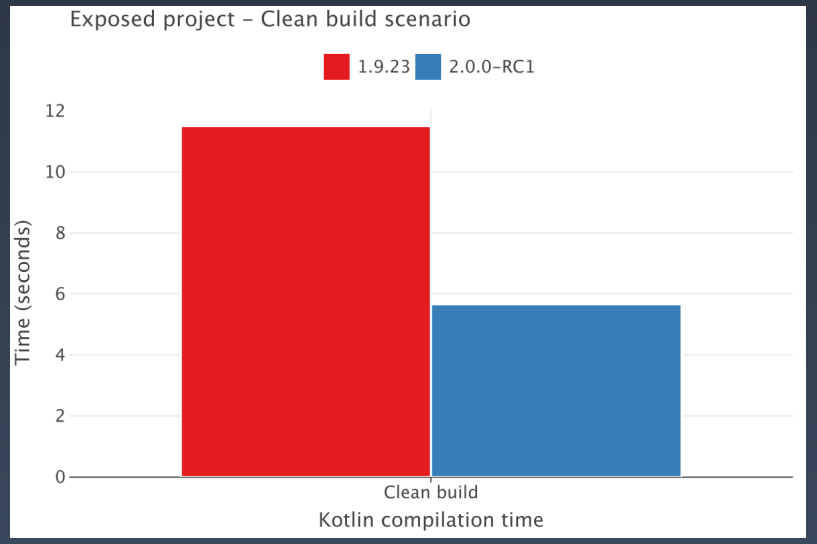
Arbeitet für maibornwolff.de/

Kontakt

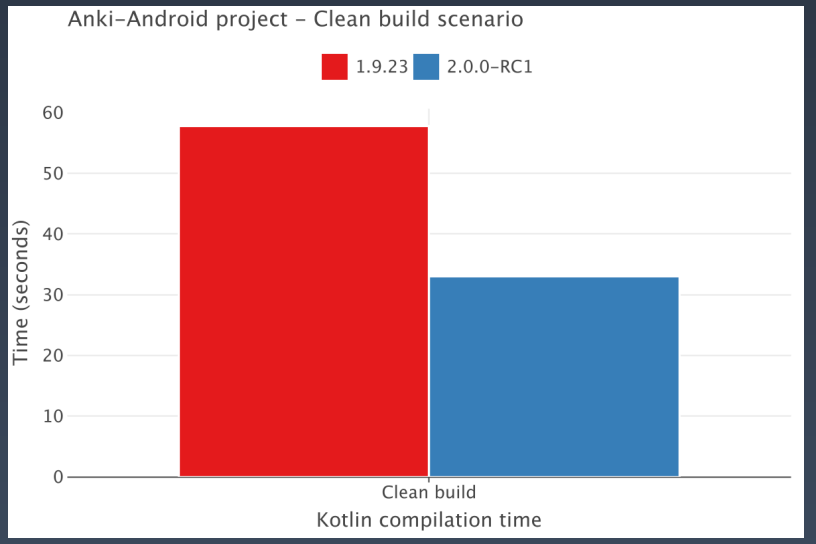
Code https://github.com/Richargh/kotlin-kontra-java

Appendix

Performance bedeutet auch wie schnell es kompiliert



[JetBrains/Exposed]
54k RLoC*



[Anki-Droid]
99k RLoC*

In der Praxis hat sich das bei bis zu 500k RLoC auch mit 1.5 schon gut angefühlt.

Mehr performance is aber immer super ☺

https://blog.jetbrains.com/kotlin/2024/04/k2-compiler-performance-benchmarks-and-how-to-measure-them-on-your-projects/
Hier ist aber nicht ganz klar welcher Task gemessen wurde. Annahme wäre:

```
./gradlew clean compileKotlin --profile --no-build-cache --no-parallel
```

* Real Lines of Code = reine Codezeilen, keine Kommentare oder Leerzeichen