

**HOCHSCHULE
HANNOVER**
UNIVERSITY OF
APPLIED SCIENCES
AND ARTS
–
*Fakultät IV
Wirtschaft und
Informatik*

Performance-Vergleich von Nebenläufigkeits-Konzepten in objektorientierten Programmiersprachen

Philipp Ermer

Master-Arbeit im Studiengang „Angewandte Informatik“

22. Januar 2024



Autor Philipp Ermer
Matrikelnummer: 1313395
philipp.ermer@web.de

Erstprüferin: Prof. Dr. Holger Peine
Abteilung Informatik, Fakultät IV
Hochschule Hannover
holger.peine@hs-hannover.de

Zweitprüfer: Prof. Dr. Robert Garmann
Abteilung Informatik, Fakultät IV
Hochschule Hannover
robert.garmann@hs-hannover.de

Soweit nicht anders gekennzeichnet, ist dieses Werk unter einem Creative-Commons-Lizenzvertrag Namensnennung 4.0 lizenziert. Dies gilt nicht für Zitate und Werke, die aufgrund einer anderen Erlaubnis genutzt werden. Um die Bedingungen der Lizenz einzusehen, folgen Sie bitte dem Hyperlink:

<https://creativecommons.org/licenses/by/4.0/deed.de>

Selbständigkeitserklärung

Hiermit erkläre ich, dass ich die eingereichte Master-Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die von mir angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Werken wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Hannover, den 22. Januar 2024

Unterschrift

Inhaltsverzeichnis

1	Einführung	9
1.1	Motivation	9
1.2	Ziel der Arbeit	9
1.3	Verwandte Arbeiten	9
1.4	Methodik	9
1.5	Gliederung und Aufbau	9
2	Nebenläufigkeitskonzepte	10
2.1	Java: Virtual Threads	10
2.2	Java: Plattform Threads	10
2.3	Java: Completable Futures	10
2.4	Kotlin: CO-Routinen	10
2.5	Kotlin: Threads	10
2.6	Go: Go-Routinen	10
2.7	C#: async/await	10
2.8	Vergleich der Konzepte	10
3	Benchmarks	11
3.1	Benchmark 1	11
3.2	Benchmark 2	11
3.3	Benchmark 3	11
3.4	Benchmark 4	11
4	Testaufbau	12
4.1	Testumgebung	12
4.2	Performancefaktoren	12
4.3	Werkzeuge zur Datenerhebung	12
5	Messungen	13
5.1	Messung: Benchmark 1	13
5.2	Messung: Benchmark 2	13
5.3	Messung: Benchmark 3	13
5.4	Messung: Benchmark 4	13

6	Auswertung	14
6.1	Ergebnisse und Beobachtungen	14
6.2	Diskussion und Bewertung	14
7	Zusammenfassung und Ausblick	15
7.1	Zusammenfassung	15
7.2	Ausblick	15
8	Anhang	16

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einführung

1.1 Motivation

1.2 Ziel der Arbeit

1.3 Verwandte Arbeiten

1.4 Methodik

1.5 Gliederung und Aufbau

2 Nebenläufigkeitskonzepte

2.1 Java: Virtual Threads

Virtual Threads [1]

2.2 Java: Plattform Threads

2.3 Java: Completable Futures

2.4 Kotlin: CO-Routinen

2.5 Kotlin: Threads

2.6 Go: Go-Routinen

2.7 C#: async/await

2.8 Vergleich der Konzepte

3 Benchmarks

3.1 Benchmark 1

3.2 Benchmark 2

3.3 Benchmark 3

3.4 Benchmark 4

4 Testaufbau

4.1 Testumgebung

4.2 Performancefaktoren

4.3 Werkzeuge zur Datenerhebung

5 Messungen

5.1 Messung: Benchmark 1

5.2 Messung: Benchmark 2

5.3 Messung: Benchmark 3

5.4 Messung: Benchmark 4

6 Auswertung

6.1 Ergebnisse und Beobachtungen

6.2 Diskussion und Bewertung

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung

7.2 Ausblick

Literatur

- [1] Martin Karsten und Saman Barghi. “User-level Threading: Have Your Cake and Eat It Too”. In: *Proc. ACM Meas. Anal. Comput. Syst.* 4.1 (Mai 2020). DOI: 10.1145/3379483. URL: <https://doi.org/10.1145/3379483>.