

Size ⚡ does ⚡ matter!

The battle of the

# MICROFRAMEWORK

Karlsruher Entwicklertag 2019

Christian Schwörer

Novatec Consulting GmbH





# JVM-Microframeworks on the rise



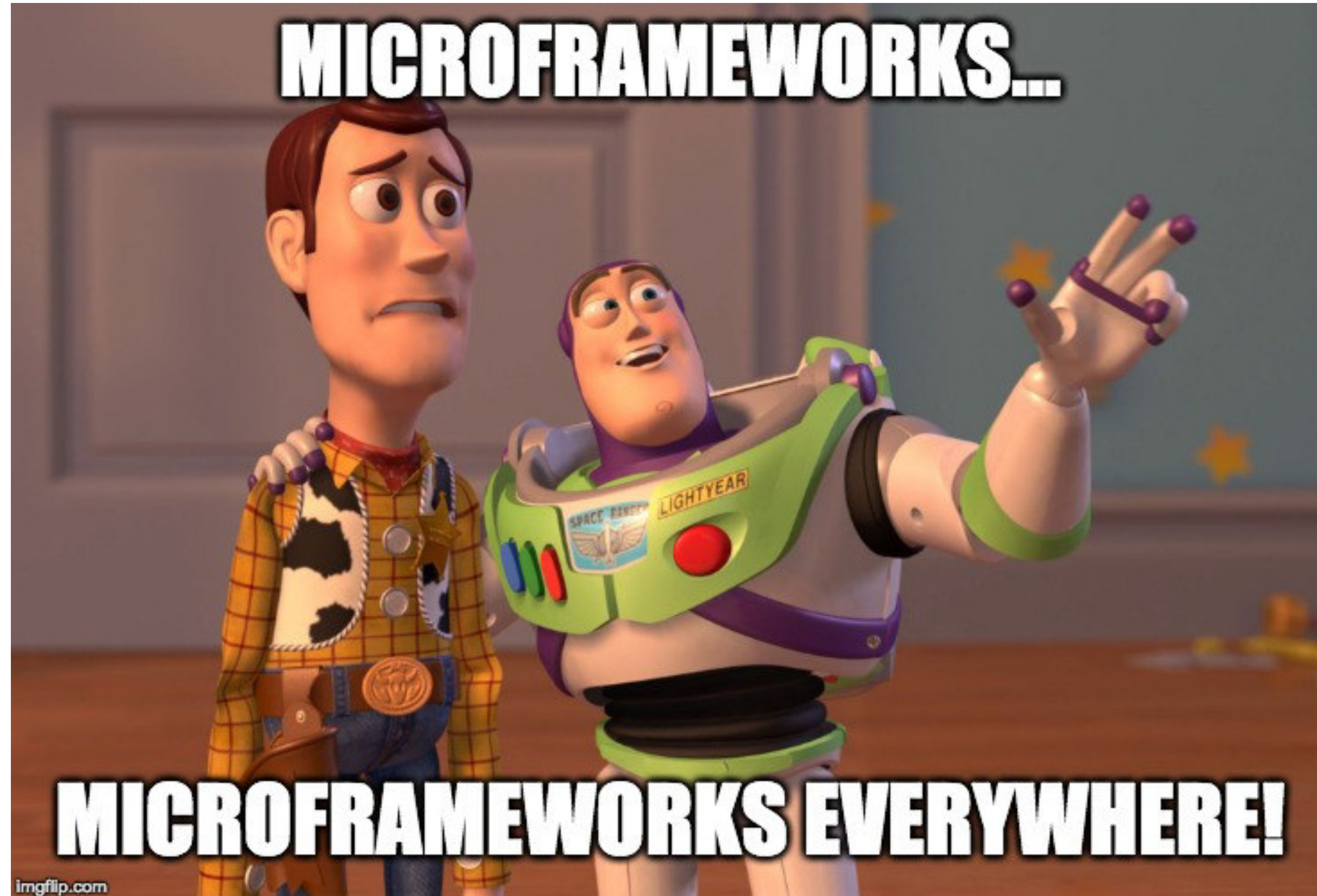


# JVM-Microframeworks on the rise

„Maturity“

Spark

PIPPO



ERT.X

ICRONAUT™

QUARKUS

ngFu

res“



# What is this session about?

---

- (Subjective) overview on current JVM-Microframeworks
- Practical introduction
- Possible fields of application for microframeworks – and where not to use them (yet)
- **And what you won't get:**
  - Deep dive in a single framework
  - Silver Bullet



# Characteristics of Microframeworks (1/2)

---

- **Designed for cloud-native microservice architectures**
- **Lightweight web framework**
  - Start and configure a Server Engine (Tomcat, Jetty, Netty, etc.)
  - Provide REST endpoints
  - Deliver web content
- **Resource efficiency**
  - Fast startup time
  - Low memory consumption



# Characteristics of Microframeworks (2/2)

---

- Focus on simplicity and speed of development
- Usually lack some advanced features  
(e.g. extended security, monitoring, multiple database abstractions)
- GraalVM friendly



# Example scenario





# Micronaut

„A modern, JVM-based, full-stack framework for building modular, easily testable microservice and serverless applications“



- Numerous modules (Kafka, SQL, NoSQL, Micrometer, AWS, etc.)
- Cloud-native modules for Service Discovery, Circuit Breakers, Distributed Tracing etc.
- Fast startup and low memory consumption
  - Uses Ahead of Time (AOT) compilation
  - Reflection free, runtime proxy free and no dynamic classloading
- Current version: 1.1.2



# Demo

---

**“Talk is cheap. Show me the code.”**

Linus Torvalds



# Javalin

“Javalin is a lightweight webframework which supports WebSockets, HTTP2 and async requests. Javalin offers first class interoperability between Kotlin and Java.”



- Runs on top of Jetty
- Very simple: only few concepts that need to be learned
- Primarily blocking – as this is the easiest programming model (but may be switched into asynchronous mode)
- Current version: 2.8.0



# Ktor

“Ktor is a framework for building asynchronous servers and clients in connected systems using the powerful Kotlin programming language.”



- Created by JetBrains: Takes full advantage of Kotlin
- Asynchronous: uses Kotlin coroutines to provide an easy-to-use asynchronous programming model
- Provides a DSL for configuring the application
- Supports multiple Server Engines (Jetty, Netty, Tomcat, CIO)
- Current version: 1.2.1



# Spring Fu

“Spring Fu is an incubator for Kofu (Ko for Kotlin, fu for functional), which provides a Kotlin API to configure Spring Boot applications programmatically.”



- Explicit, functional configuration via Kotlin DSL instead of annotations
- Minimal set of features enabled by default
- Faster startup and lower memory consumption
  - No classpath scanning, Minimal reflection and annotation usage
  - Pure lambdas, no CGLIB proxy
- Current version: 0.1



# Recap

Framework	Characteristics	Startup time*	Memory**	JAR
Micronaut	<ul style="list-style-type: none"><li>- Numerous modules</li><li>- AoT compilation</li></ul>	~ 1.1 s	~ 280 MB	~ 19 MB
Javalin	<ul style="list-style-type: none"><li>- Lightweight</li><li>- Focus on simplicity</li></ul>	~ 0.2 s	~ 210 MB	~ 11 MB
Ktor	<ul style="list-style-type: none"><li>- Asynchronous</li><li>- Kotlin coroutines</li></ul>	~ 0.6 s	~ 240 MB	~ 18 MB
SpringFu	<ul style="list-style-type: none"><li>- Explicit configuration</li><li>- Spring Boot's future?</li></ul>	~ 0.7s	~ 250 MB	~ 29 MB

\* Starting the example application as (Java 8-)JAR on a MacBook Pro

\*\* Allocated heap size after start (not constrained with `-Xmx` or `-Xms`)




# Possible fields of application

- **Where microframeworks make sense:**
  - (Dockerized) Microservices with dynamic load scenarios
  - Serverless Functions (but obviously not the webserver part)
  - API Gateways (non-blocking frameworks)
  - Mocking services
- **And where they might not be appropriate (yet):**
  - CloudFoundry as PaaS (because of its outstanding Spring Boot integration)
  - Standardization: choose one framework to rule them all



# Further information

---

-  @csh\_0711
- <https://github.com/csh0711/jvm-microframeworks-kotlin-samples>
- <https://blog.novatec-gmbh.de>
- Frameworks
  - <https://micronaut.io>
  - <https://javalin.io>
  - <https://ktor.io>
  - <https://github.com/spring-projects/spring-fu>



# Interested in more sessions?

“Sichere reaktive Webanwendungen mit  
Spring Security 5.1”  
(Hands-on-Workshop)

Andreas Falk

05.06.2019 | 09:15 – 15:00 Uhr





# Please provide your feedback!

---



Thanks a lot! 😊