

Create cloud happy applications with Quarkus

Martin Östmark
Chief Architect AppDev, Nordics

Magnus Eklund
Specialist Solution Architect

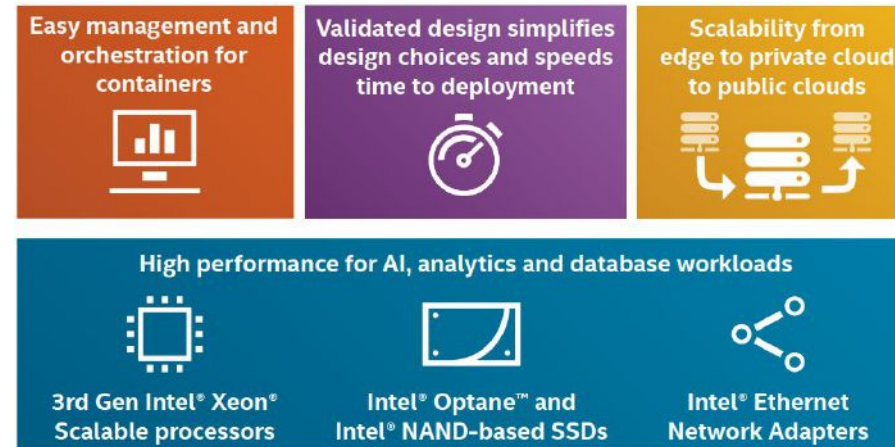
Red Hat OpenShift Reference Architecture

Joint Red Hat and Intel OpenShift Reference Architecture

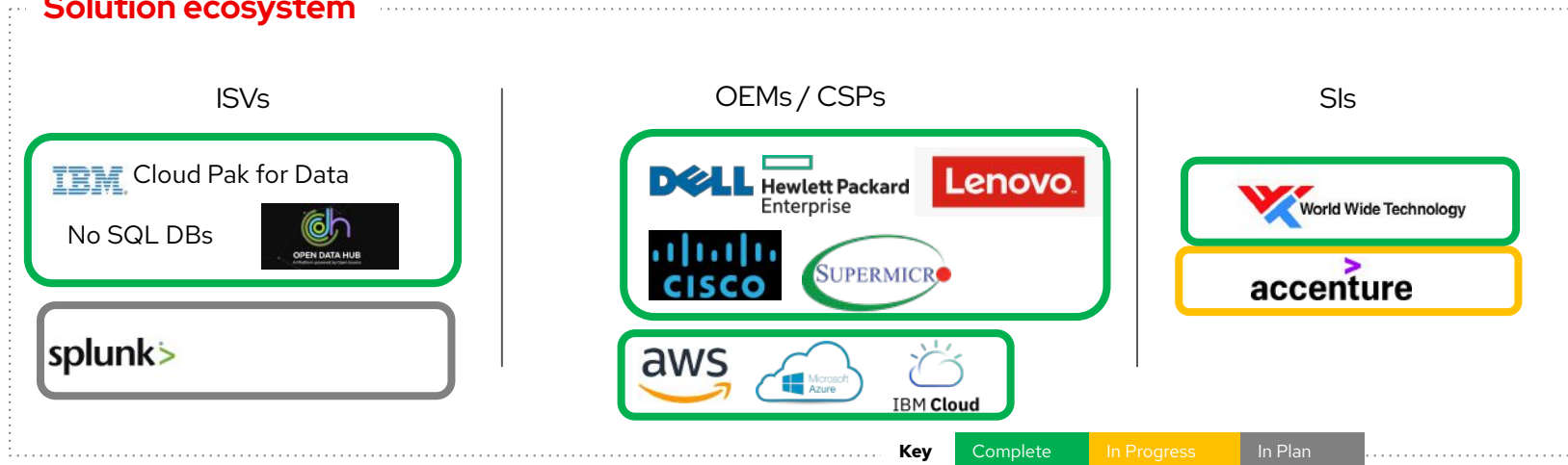
Solution overview

Summary: The RA enables deployment of performant and low-latency container-based workloads onto different footprints, such as bare metal, virtual, private cloud, public cloud, or a combination of these, in either a centralized data center or at the edge

Purpose: A general purpose OpenShift reference architecture to showcase the best of Intel and Red Hat products with key workloads



Solution ecosystem



Intel enabling status

- Intel® Xeon (2nd Gen – Cascade Lake, 3rd Gen – Ice Lake)
- Intel Optane (PMEM, SSD); Columbiaville

Collateral

- [Intel OpenShift RA for 4.6](#)
- [Intel OpenShift Solution Brief for 4.6](#)
- [Red Hat: OpenShift Ref Arch – Multiple OEMs](#)
- [Dell: OpenShift Offering](#)
- HPE: [OpenShift Offering](#)
- Cisco: [OpenShift Offering](#)
- Lenovo: [OpenShift Offering](#)
- Supermicro: [OpenShift Offering](#)
- Penguin Computing: [OpenShift Offering](#)

Agenda

- ▶ Introduction
- ▶ New architectures drives new technology needs
- ▶ Approach to meet new needs
- ▶ Summary





• **9M+**

Java developers
worldwide

#1

**Availability of
developers**



• **90%**

of the Fortune 500
are using Java

#2

Specifications



• **40%**

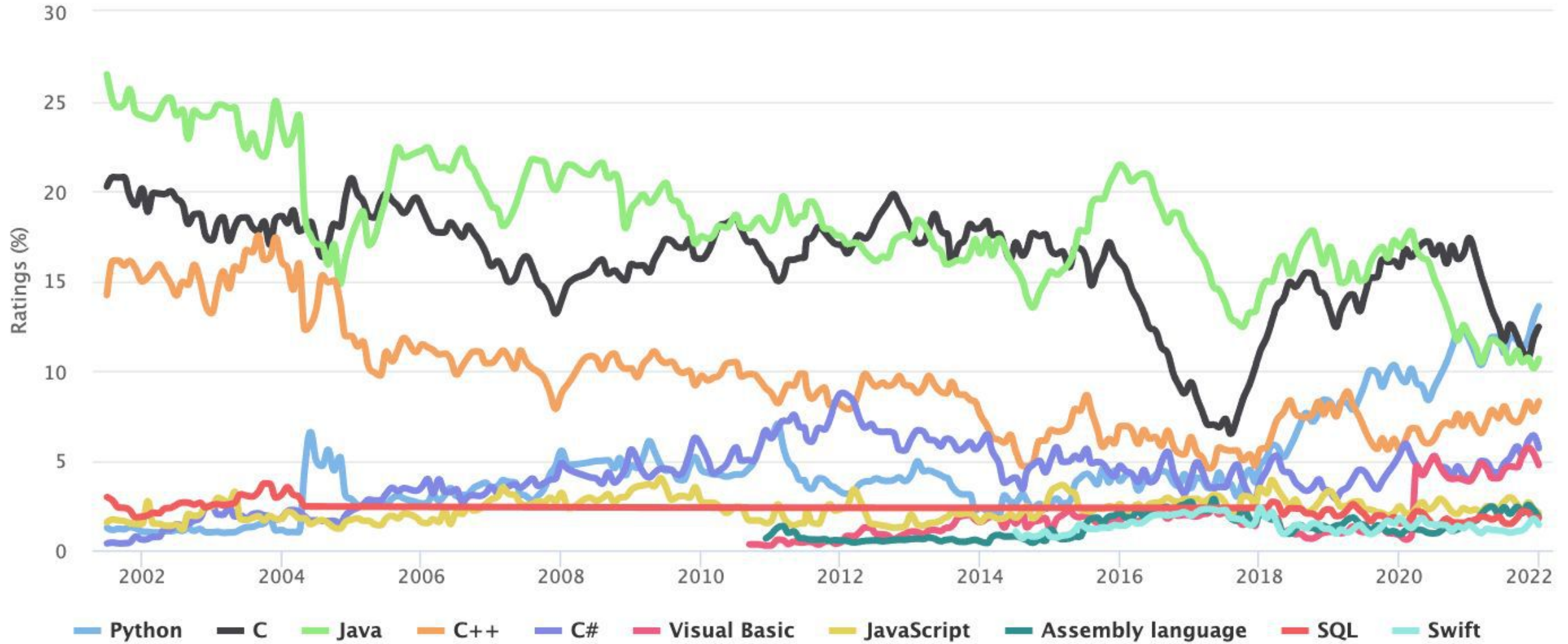
of companies use Java to build
over 80% of their applications

#3

Stability

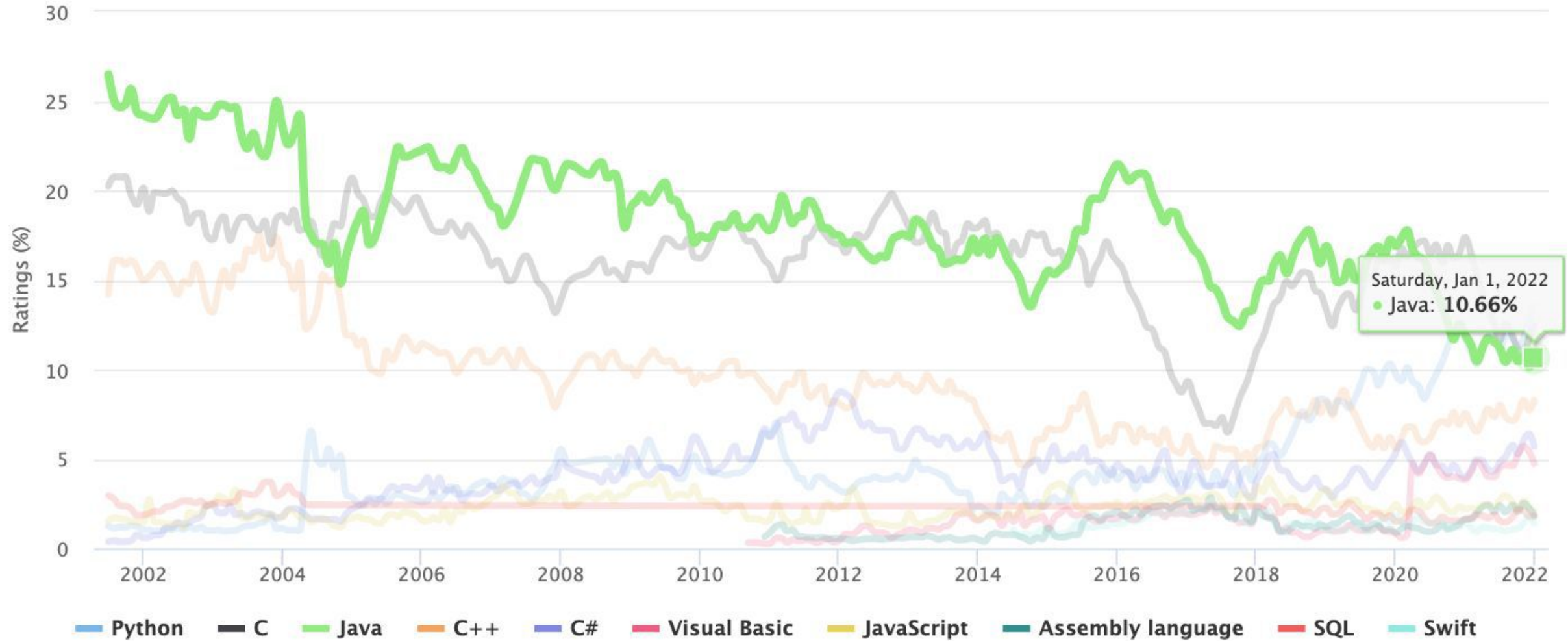
TIOBE Programming Community Index

Source: www.tiobe.com



TIOBE Programming Community Index

Source: www.tiobe.com



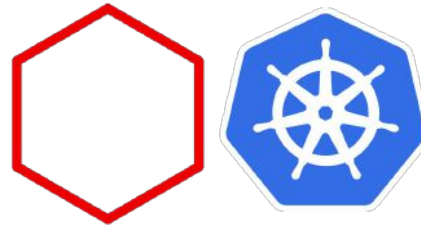


New architectures drives new technology needs

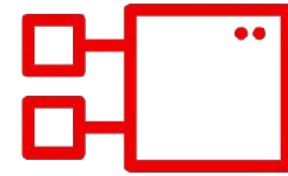
Technology trends



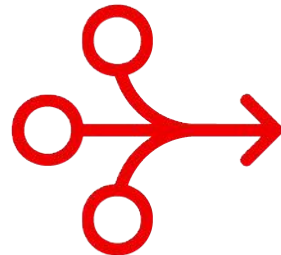
Cloud and Edge Computing



Containers and Kubernetes



Microservices Architecture



Event-driven Architectures and
reactive systems



Serverless and FaaS

"Historical" Enterprise Java Stack

Architecture: **Monoliths**

Deployment: **multi-app, appserver**

App Lifecycle: **Months**

Memory: **1GB+ RAM**

Startup Time: **10s of sec**

App

App

App

App

App

Dynamic Application Frameworks

Application Server

Java Virtual Machine (Hotspot)

Operating System + Hardware/VM



"Modern" Enterprise Java Stack

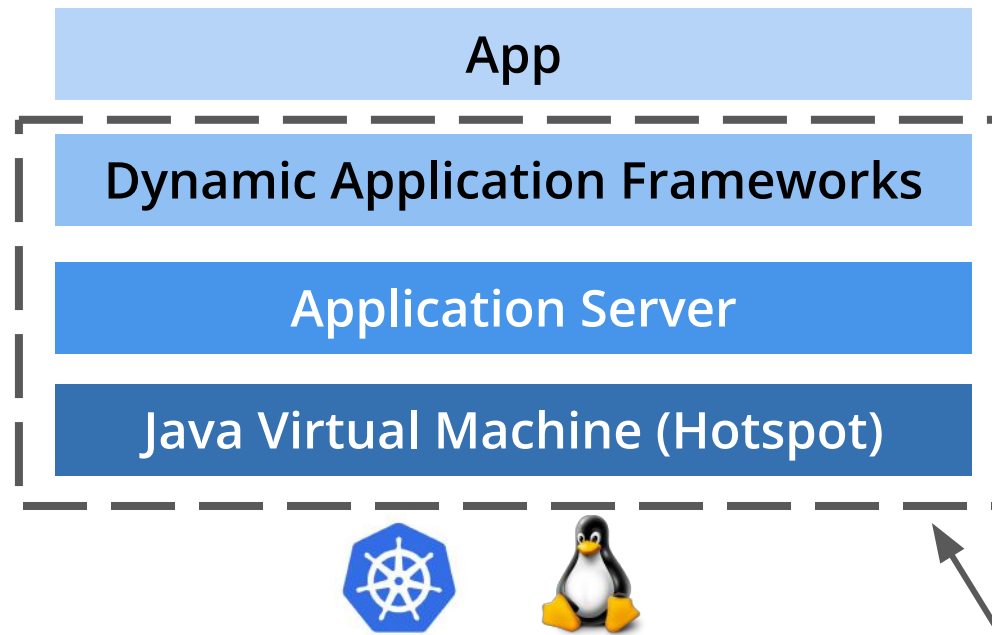
Architecture: **Microservices**

Deployment: **Single App**

App Lifecycle: **Days**

Memory: **100MBs+**
RAM

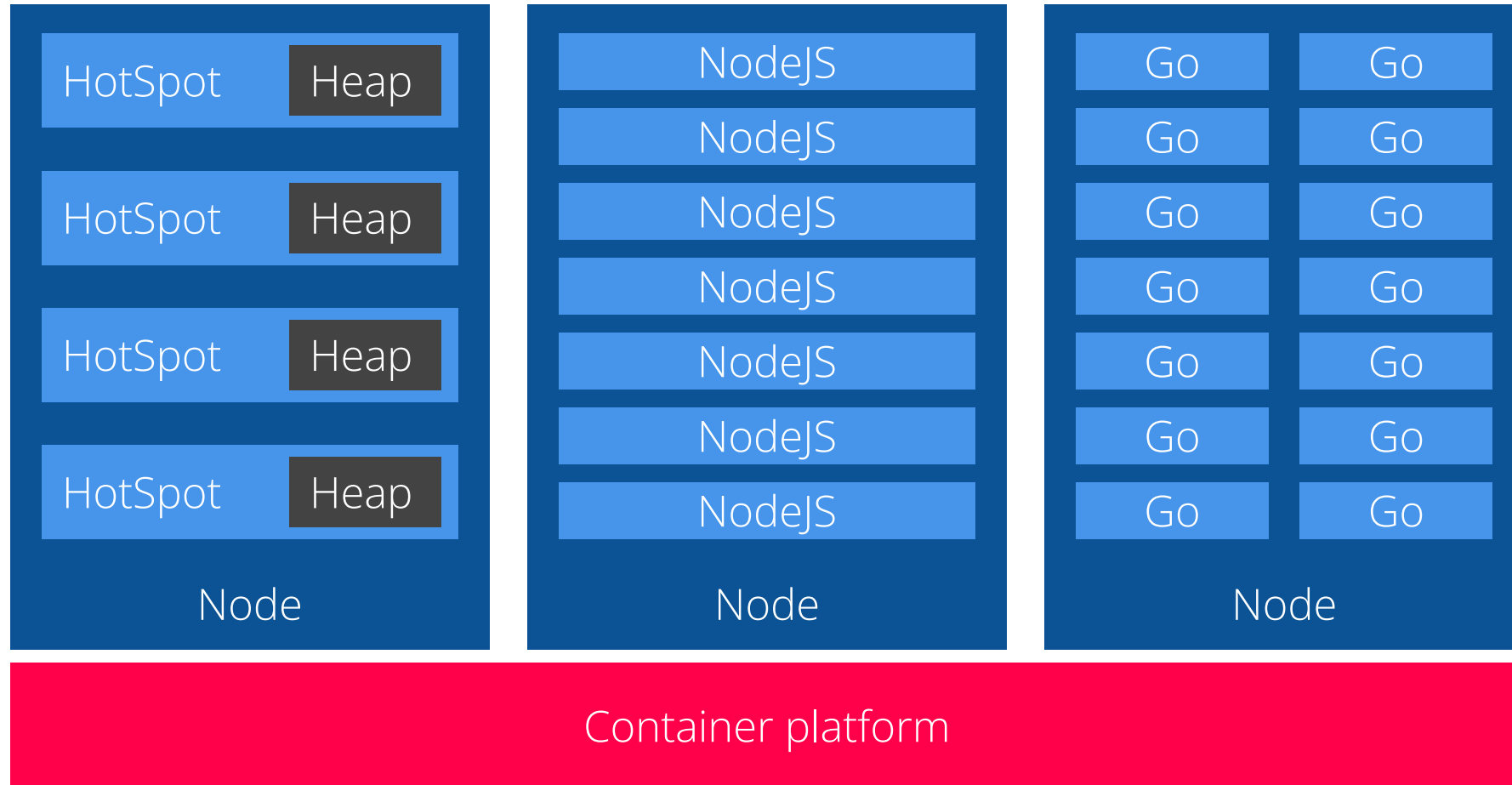
Startup Time: **Seconds**



No
Change



Hidden Truth About Java + Containers



**THERE IS A NEED FOR A
NEW JAVA STACK FOR
CLOUD-NATIVE AND
SERVERLESS**





QUARKUS

Supersonic. Subatomic. Java.



Experts from cloud-native Java OS projects

VERT.x



REST
Easy



WildFly



OpenJDK™

Eclipse Vert.x

Hibernate

RESTEasy

Eclipse MicroProfile

WildFly

Undertow

OpenJDK



QUARKUS



Benefits



Container First

Tailors your app for HotSpot & GraalVM

Fast boot time and low RSS memory

Serverless fit



Developer Joy

Live coding

Unified configuration

Frictionless local dev with dev services



Unifies Imperative & Reactive

Combines blocking and non-blocking

Built-in event bus



Best of Breed Libraries & Standards

500+ extensions

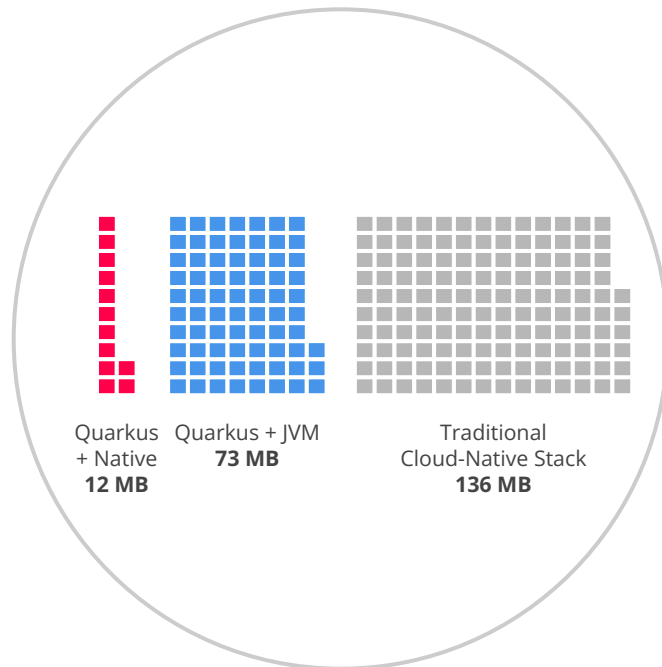
"Powered by Quarkus" applications



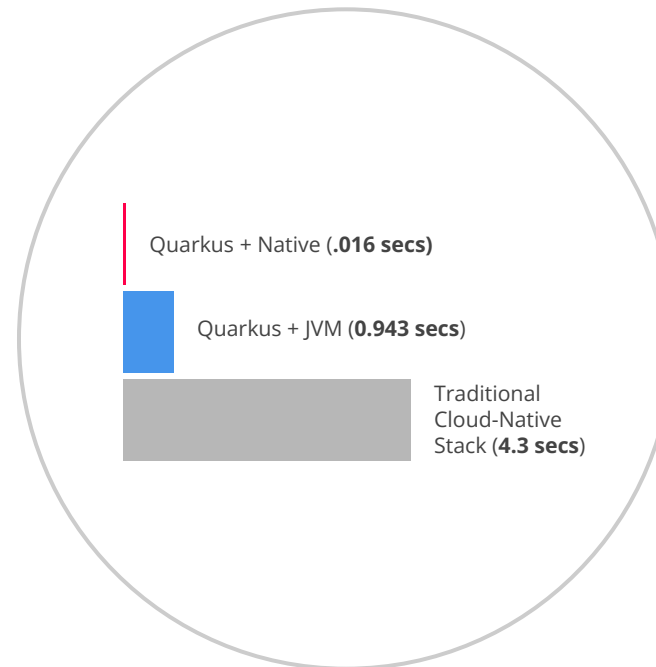
Benefit No. 1: Container First

*“We went from **1-min** startup times to **400 milliseconds**”*

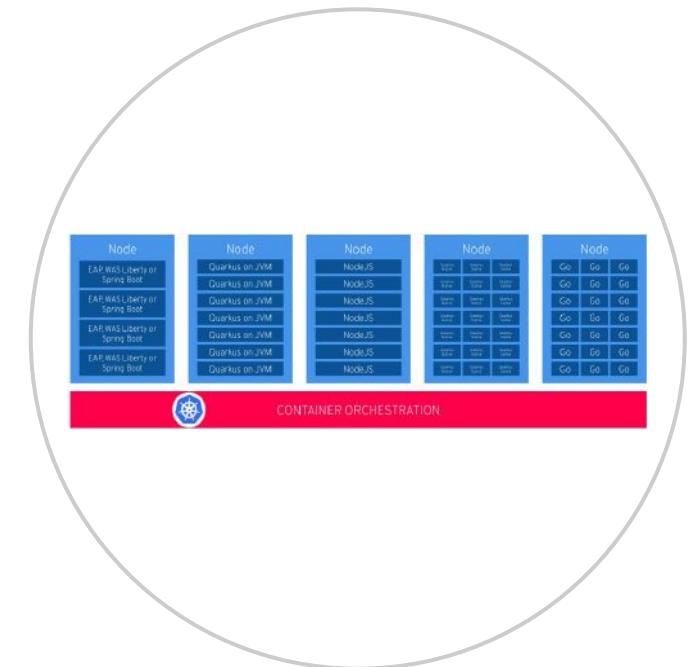
Reduced Memory Footprint



Fast Startup Time

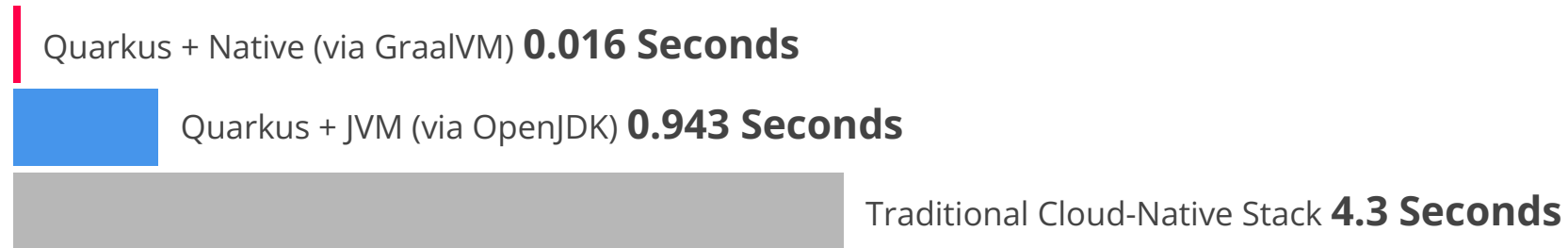


Smaller Disk Footprint

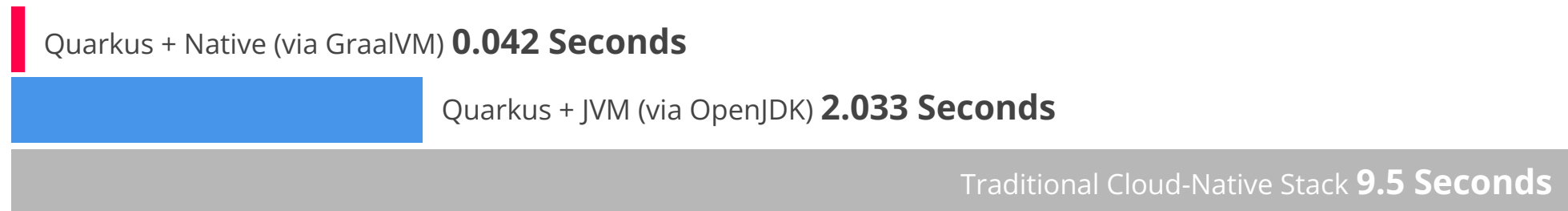


Supersonic, Subatomic Java

REST



REST + CRUD

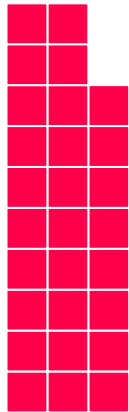


Time to first response

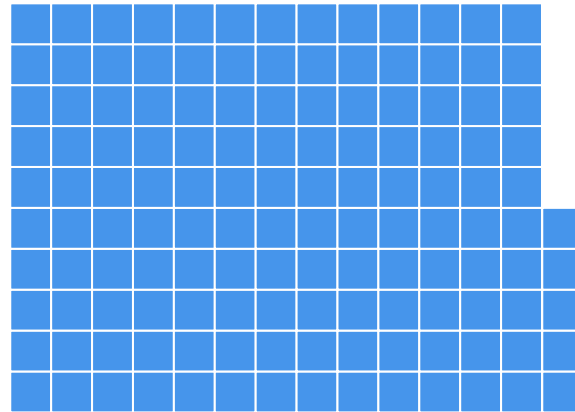


Supersonic, Subatomic Java

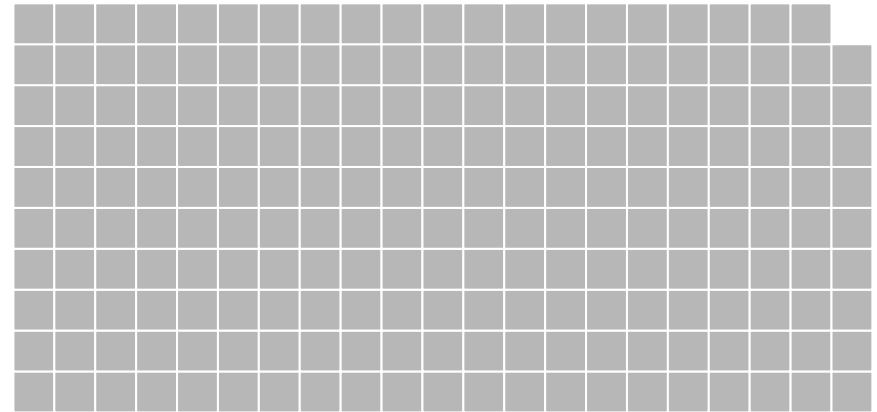
REST + CRUD*



Quarkus + Native
(via GraalVM)
28 MB



Quarkus + JVM
(via OpenJDK)
145 MB



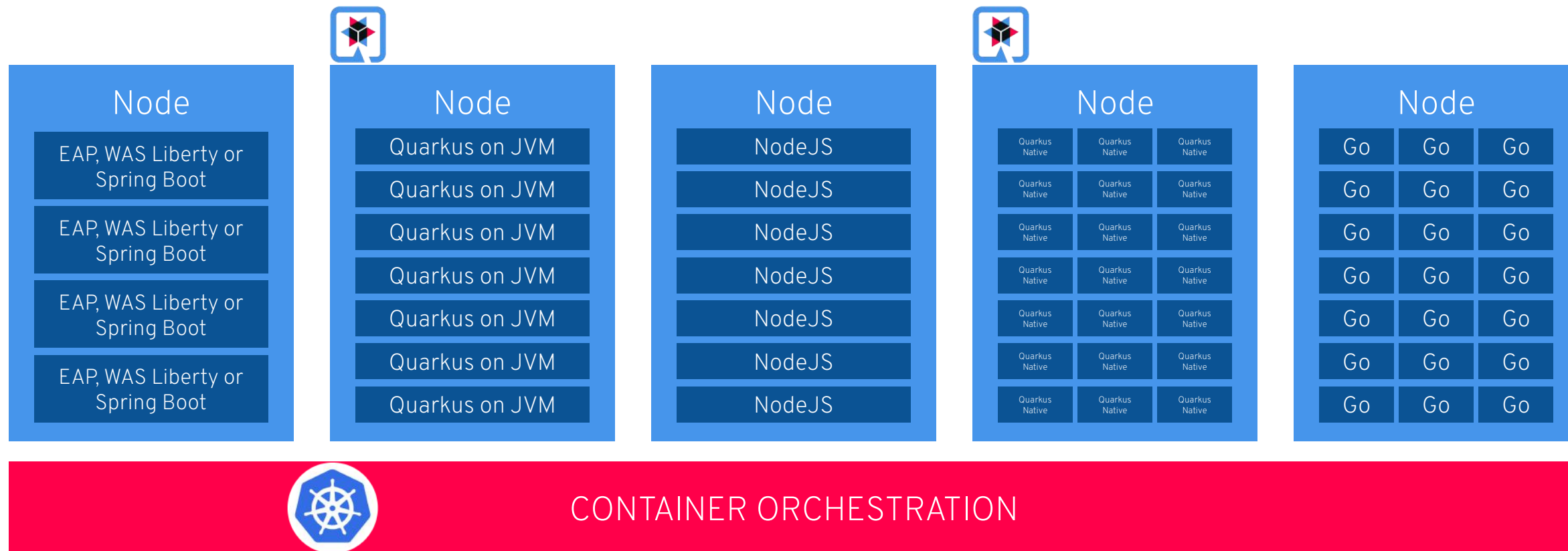
Traditional
Cloud-Native Stack
209 MB



*Memory (RSS) in Megabytes, tested on a single-core machine



Cloud Native Java Stack + Containers



*“We could run **3 times** denser deployments without sacrificing **availability** and **response times** of services”*

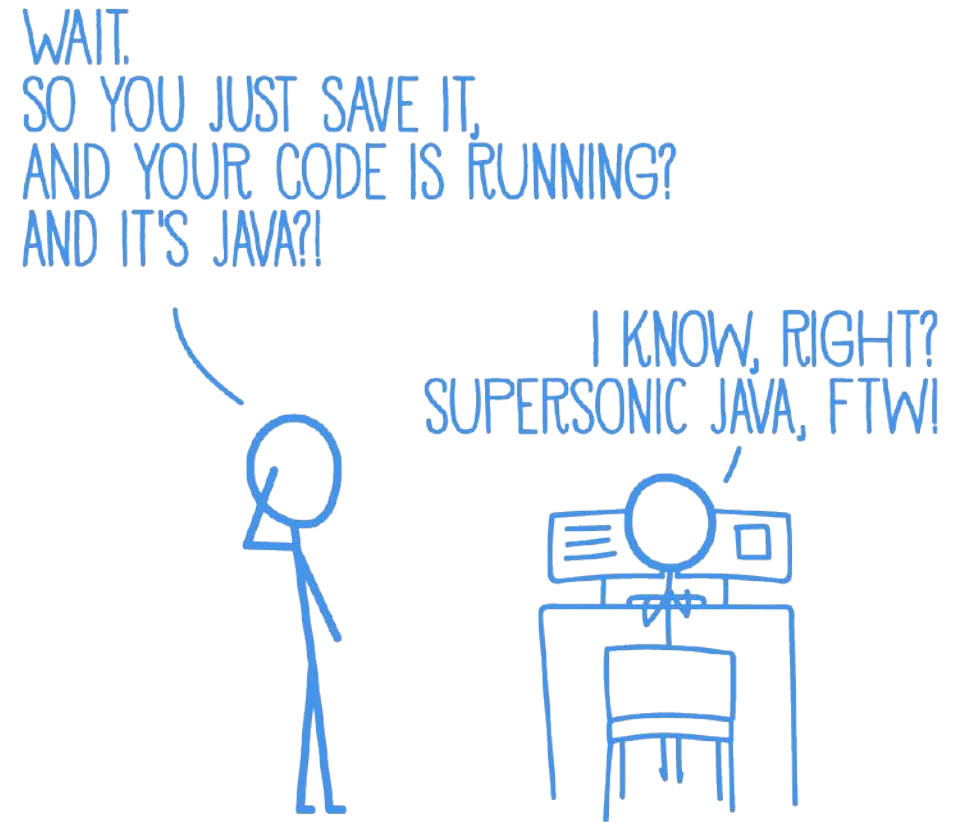


Benefit No. 2: Developer Joy

*“Our developers used to wait **2 to 3 mins** to see their changes. **Live coding** does away with this.”*

A cohesive platform for optimized developer joy:

- Based on standards and more
- Unified configuration
- Live coding
- Streamlined code for the 80% common usages, flexible for the 20%
- No hassle native executable generation
- Zero configuration with dev services
- Continuous testing for instant feedback



Benefit No. 3: Unifies Imperative and Reactive

```
@Inject  
SayService say;  
  
@GET  
@Produces(MediaType.TEXT_PLAIN)  
public String hello() {  
    return say.hello();  
}
```

```
@Inject @Stream("kafka")  
Publisher<String> reactiveSay;  
  
@GET  
@Produces(MediaType.SERVER_SENT_EVENTS)  
public Publisher<String> stream() {  
    return reactiveSay;  
}
```

- Combine both Reactive and imperative development in the same application
- Inject the EventBus or the Vertx context
- Use the technology that fits your use-case
- Key for reactive systems based on event driven apps



Benefit No. 4: Best of Breed Frameworks & Standards

"When you adopt Quarkus, you will be productive from day one since you don't need to learn new technologies."

The logo for Eclipse Vert.x, featuring the word "VERT.x" in a bold, sans-serif font. "VERT" is in dark blue and ".x" is in purple.

Eclipse Vert.x



Hibernate



RESTEasy



Apache Camel



Eclipse MicroProfile



Netty



Kubernetes



OpenShift



Jaeger



Prometheus



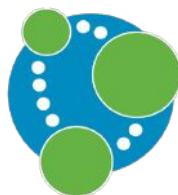
Apache Kafka



Infinispan



Flyway



Neo4j



MongoDB



MQTT



KeyCloak



Apache Tika



Use Cases



NET NEW

Low memory footprint + lightning fast startup time + small disk footprint = an ideal runtime for Kubernetes-native microservices



MONO 2 MICRO

Quarkus is a great choice to modernize existing monolithic applications by breaking it into smaller, loosely coupled microservices.



SERVERLESS

Scaling up or down (0) is extremely fast with Quarkus making it an ideal runtime for creating serverless applications.



EVENT-DRIVEN/REACTIVE

Quarkus utilizes an asynchronous, reactive event loop that makes it easy to create reactive applications.





Demo

HOW DOES QUARKUS WORK?



Quarkus - Optimizing the Java Stack

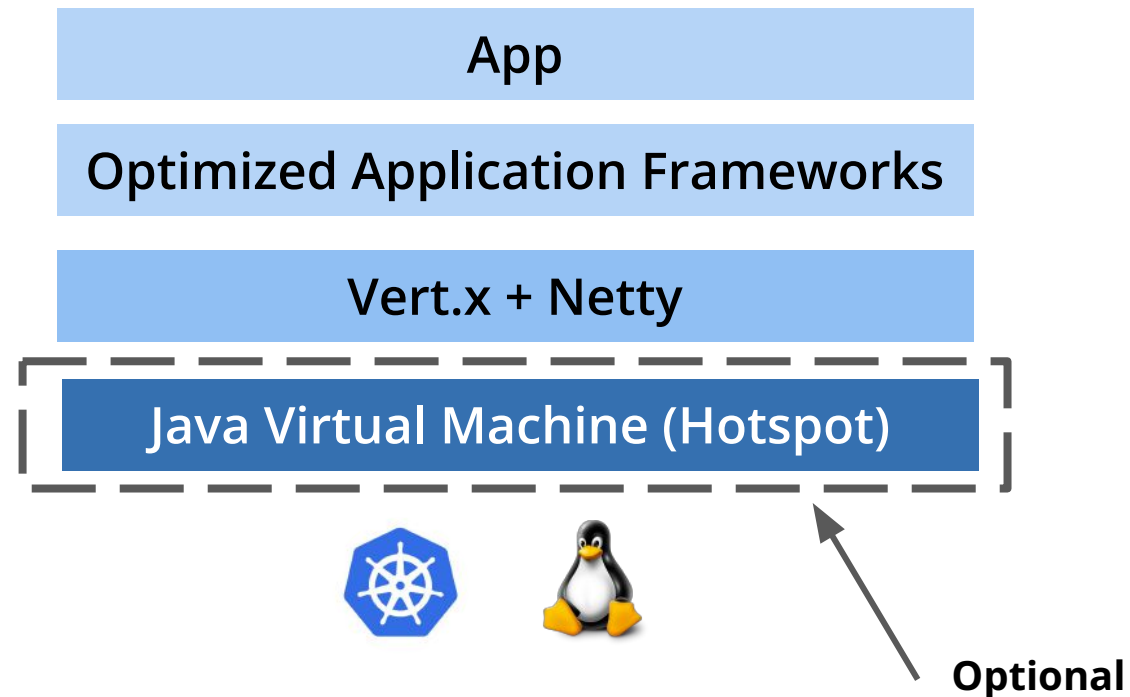
Architecture: **Microservices, Serverless**

Deployment: **Single App**

App Lifecycle: **Milliseconds to Days**

Memory: **10MBs+ RAM**

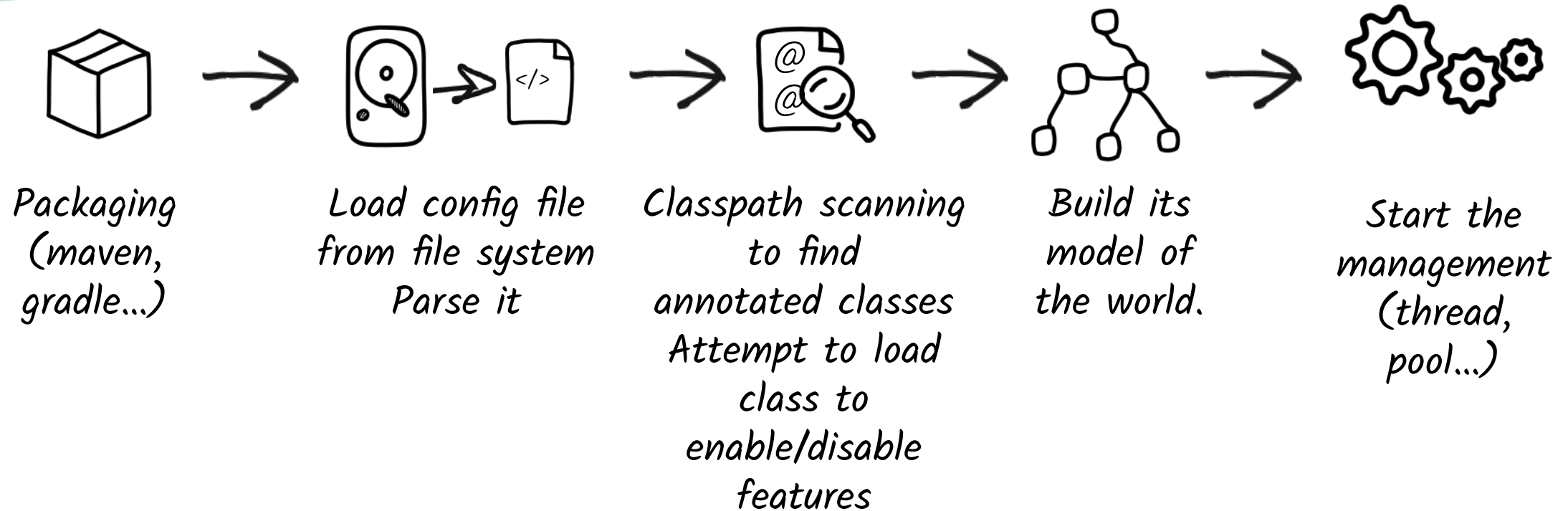
Startup Time: **Milliseconds**



How Does a Framework Start?

Build Time

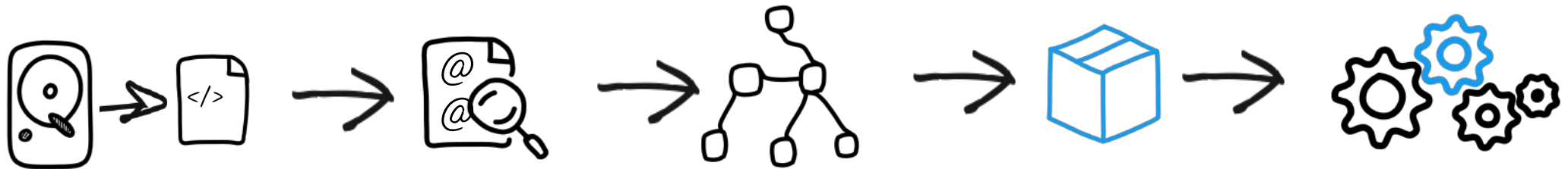
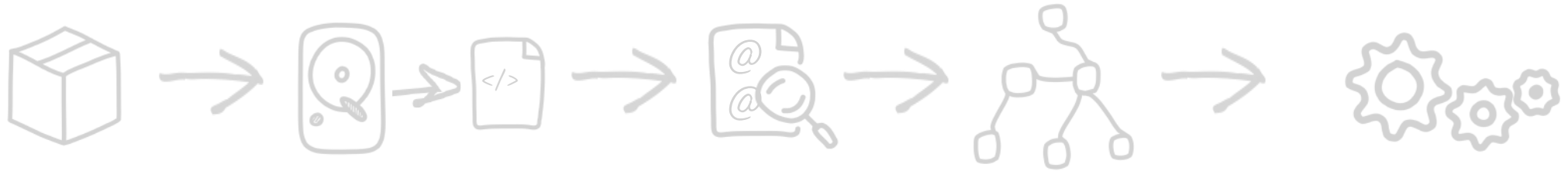
Runtime



The Quarkus Way

Build Time

Runtime

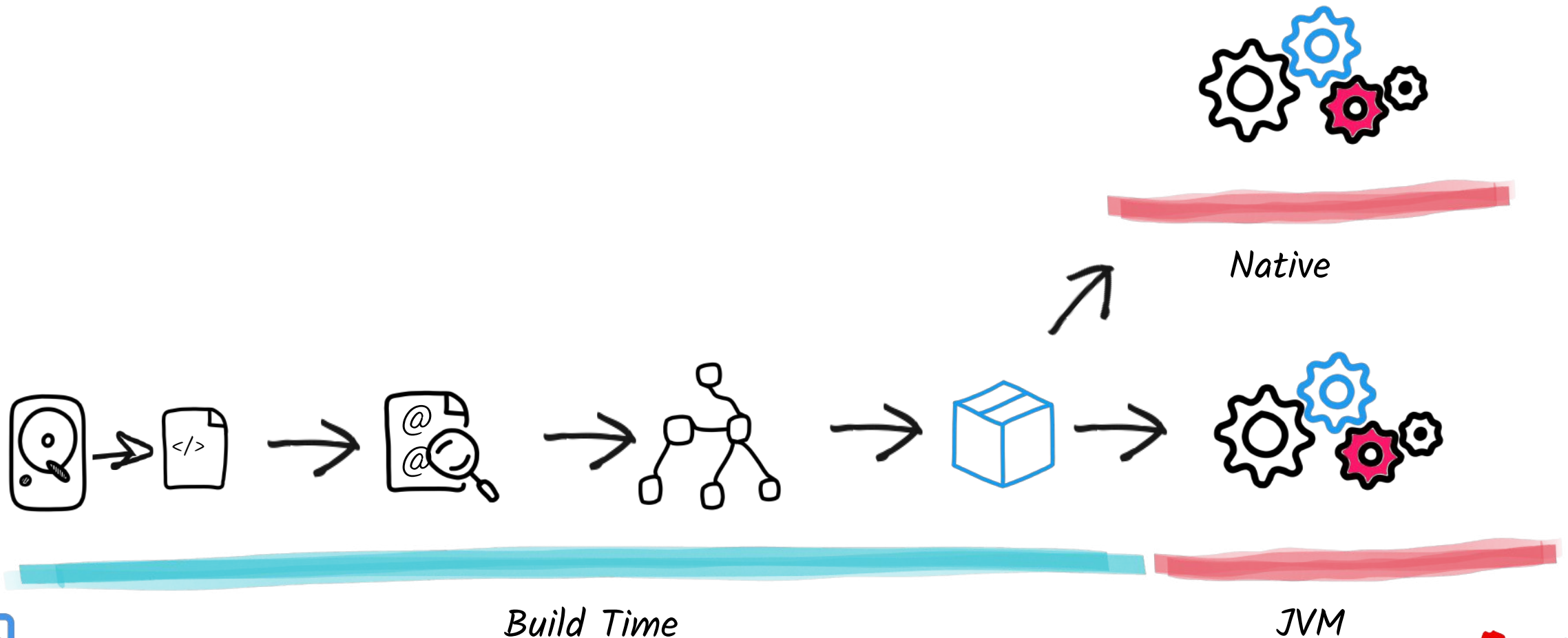


Build Time

Runtime



The Quarkus Way enables Native compilation



Build Time

JVM

Customers using Quarkus Today



"We could run 3 times denser deployments without sacrificing availability and response times of service"

Thorsten Pohl

Lufthansa Technik AVIATAR
Product Owner Automation &
Platform Architect



"When you adopt Quarkus, you will be productive from day one since you don't really need to learn new technologies."

Roberto Cortez

Talkdesk Principal Architect



"Quarkus seemed to provide the performance boost we needed while at the same time having a good backer (**Red Hat**) and relying on battle-tested technologies"

Christos Sotiriou

DXL technical lead at Vodafone Greece



“Quarkus seemed to provide the performance boost we needed while at the same time having a good backer (**Red Hat**) and relying on battle-tested technologies”

Christos Sotiriou

DXL technical lead at Vodafone Greece

Challenge

Running 140 microservices, with heavy spikes in traffic, caused delays and pause while booting new containerized applications leading to waste of marketing efforts.

Solution

After initial tests indicated that Quarkus would reduce application boot times, reduce CPU and memory usage, and make the entire development process run faster, Vodafone decided to port their most essential libraries and microservices to this new stack.

Why Quarkus

The main criteria for their selection process to find a replacement for Spring Boot were developer efficiency, lower cloud resource consumption and shorter applications boot-up times. A great impact on cloud resource consumption costs as well as user experience improvement. Their trust of Red Hat combined with its credibility in the software market gave them the assurance that they were making the right choice by selecting Quarkus, whose sponsor is Red Hat.

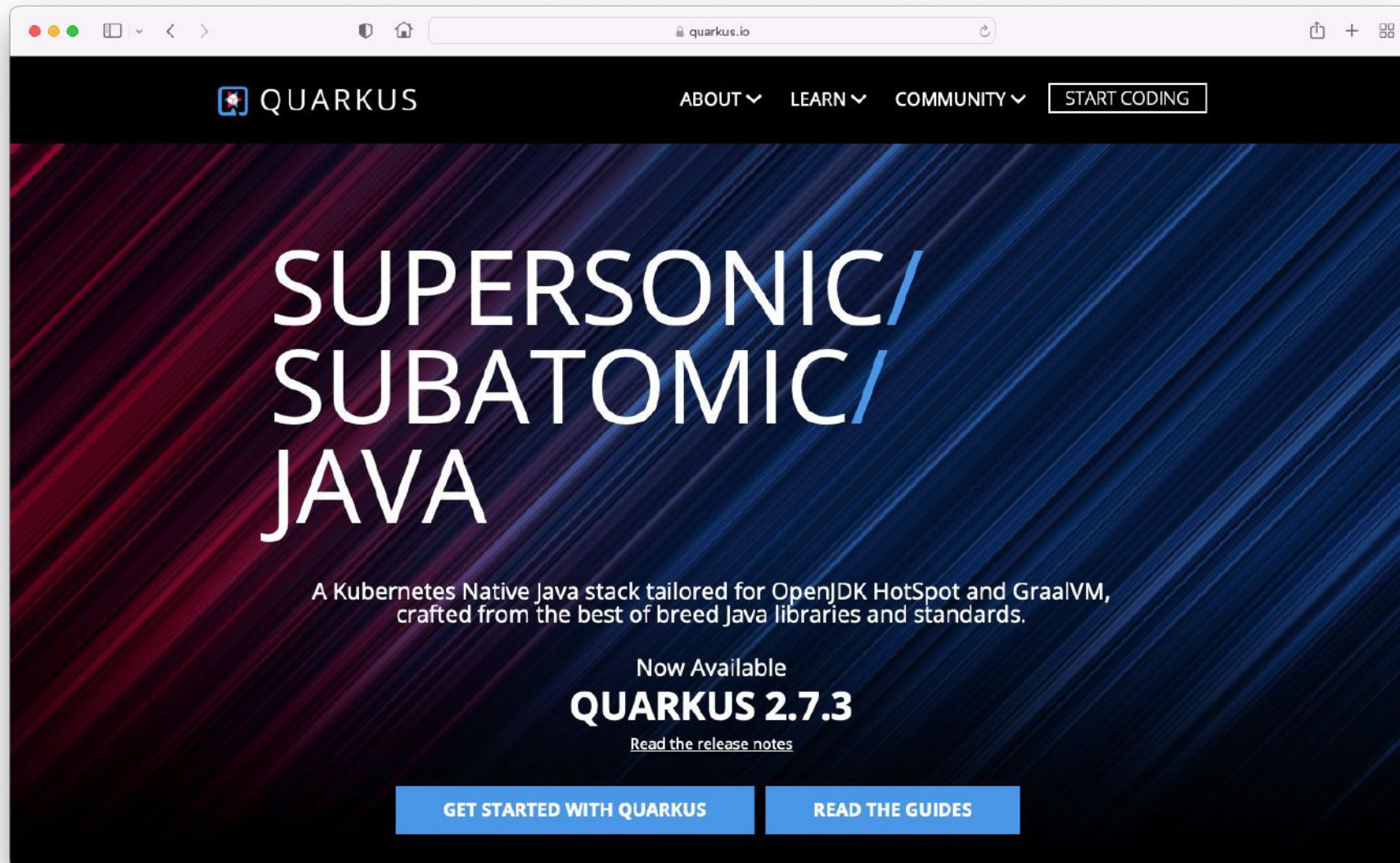
Results

- Start-up times have been reduced to almost a quarter without any optimization
- Memory resource consumption was cut in half in JVM mode
- The use of the Quarkus live coding capability (a.k.a. dev mode) resulted in an increase of developer productivity
- Migrating from Spring Boot to Quarkus didn't require a lot of effort for their Spring developers, resulting in a small learning curve
- Far healthier cluster overall, as it is no longer experiencing difficulty in handling the sudden traffic spikes driven by the company's direct marketing campaigns

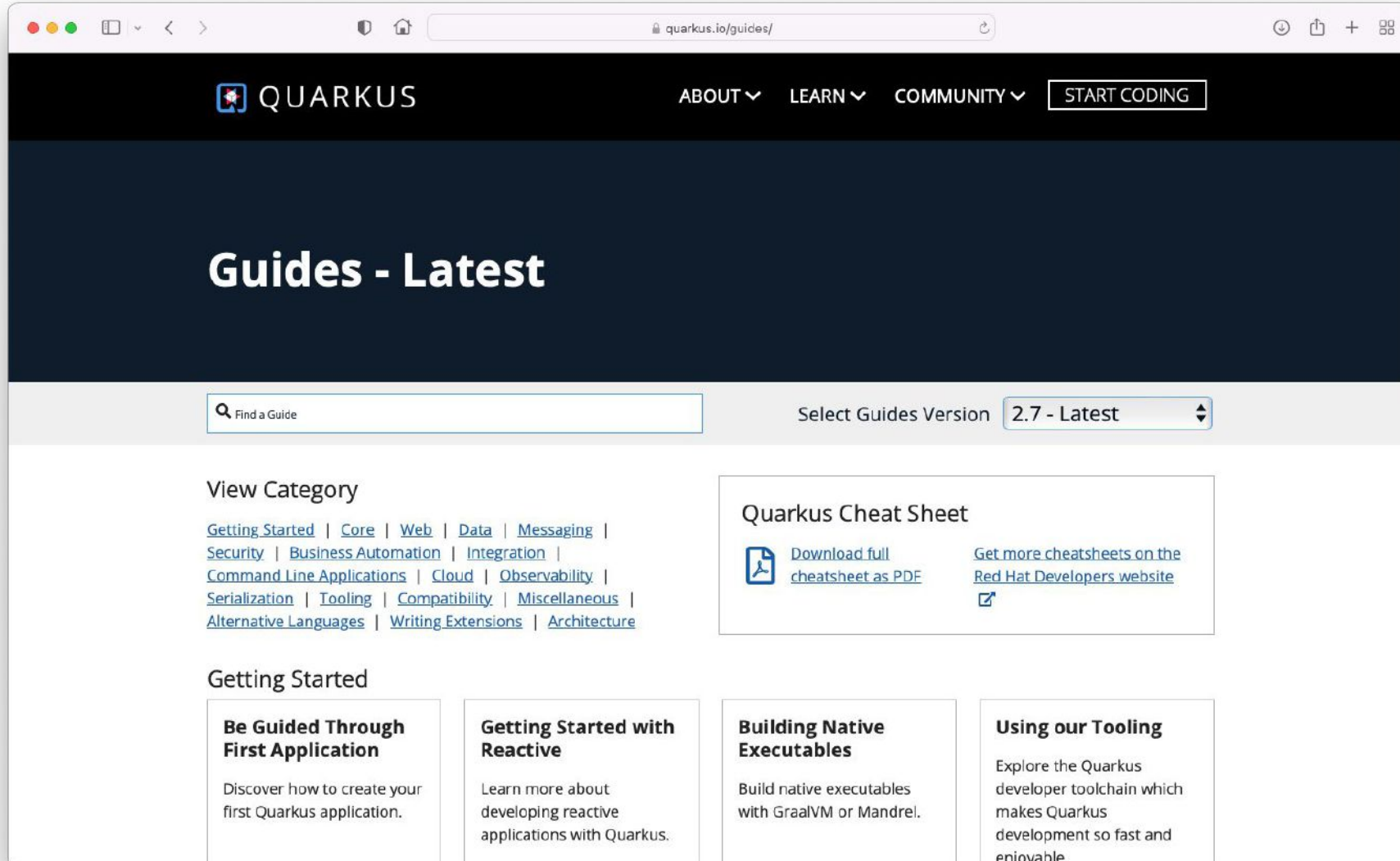


Where to learn more?

https://quarkus.io



https://quarkus.io/guides/

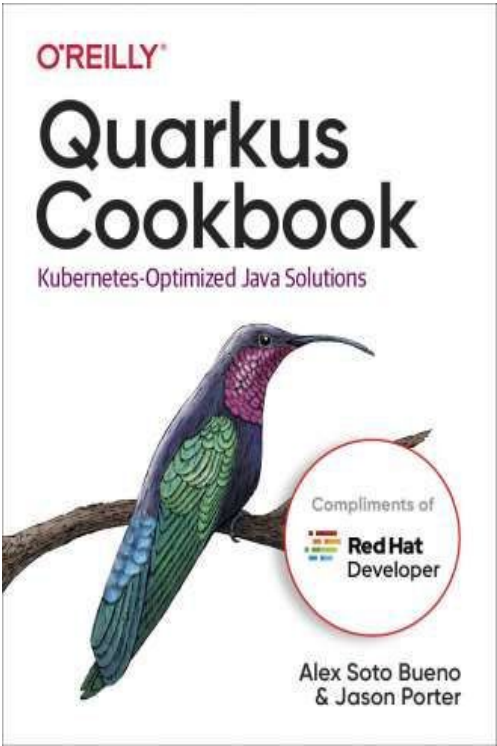
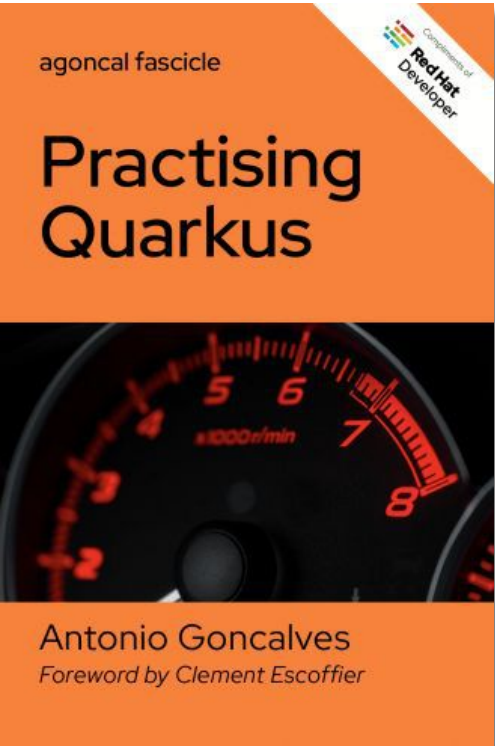
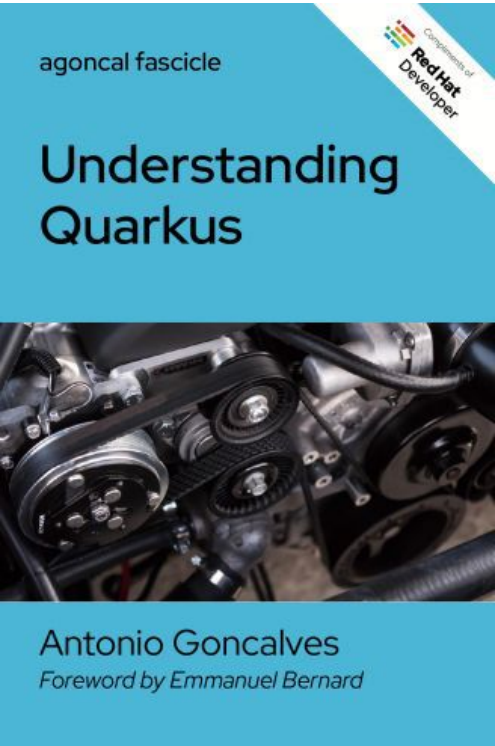
A screenshot of the Quarkus website's guides page. The browser address bar shows 'quarkus.io/guides/'. The page has a dark blue header with the Quarkus logo and navigation links: 'ABOUT', 'LEARN', 'COMMUNITY', and a 'START CODING' button. Below the header, the main section is titled 'Guides - Latest'. There is a search bar labeled 'Find a Guide' and a dropdown menu for 'Select Guides Version' currently set to '2.7 - Latest'. The content area is divided into two columns. The left column, titled 'View Category', lists various guide categories like 'Getting Started', 'Core', 'Web', 'Data', 'Messaging', 'Security', 'Business Automation', 'Integration', 'Command Line Applications', 'Cloud', 'Observability', 'Serialization', 'Tooling', 'Compatibility', 'Miscellaneous', 'Alternative Languages', 'Writing Extensions', and 'Architecture'. The right column, titled 'Quarkus Cheat Sheet', includes a PDF download link and a link to 'Get more cheatsheets on the Red Hat Developers website'. Below these, the 'Getting Started' section features four cards: 'Be Guided Through First Application', 'Getting Started with Reactive', 'Building Native Executables', and 'Using our Tooling', each with a brief description of the guide's content.



Red Hat Cloud-native Microservices Development with Quarkus (DO378)

free e-books

<https://developers.redhat.com/e-books/>



Developer Sandbox



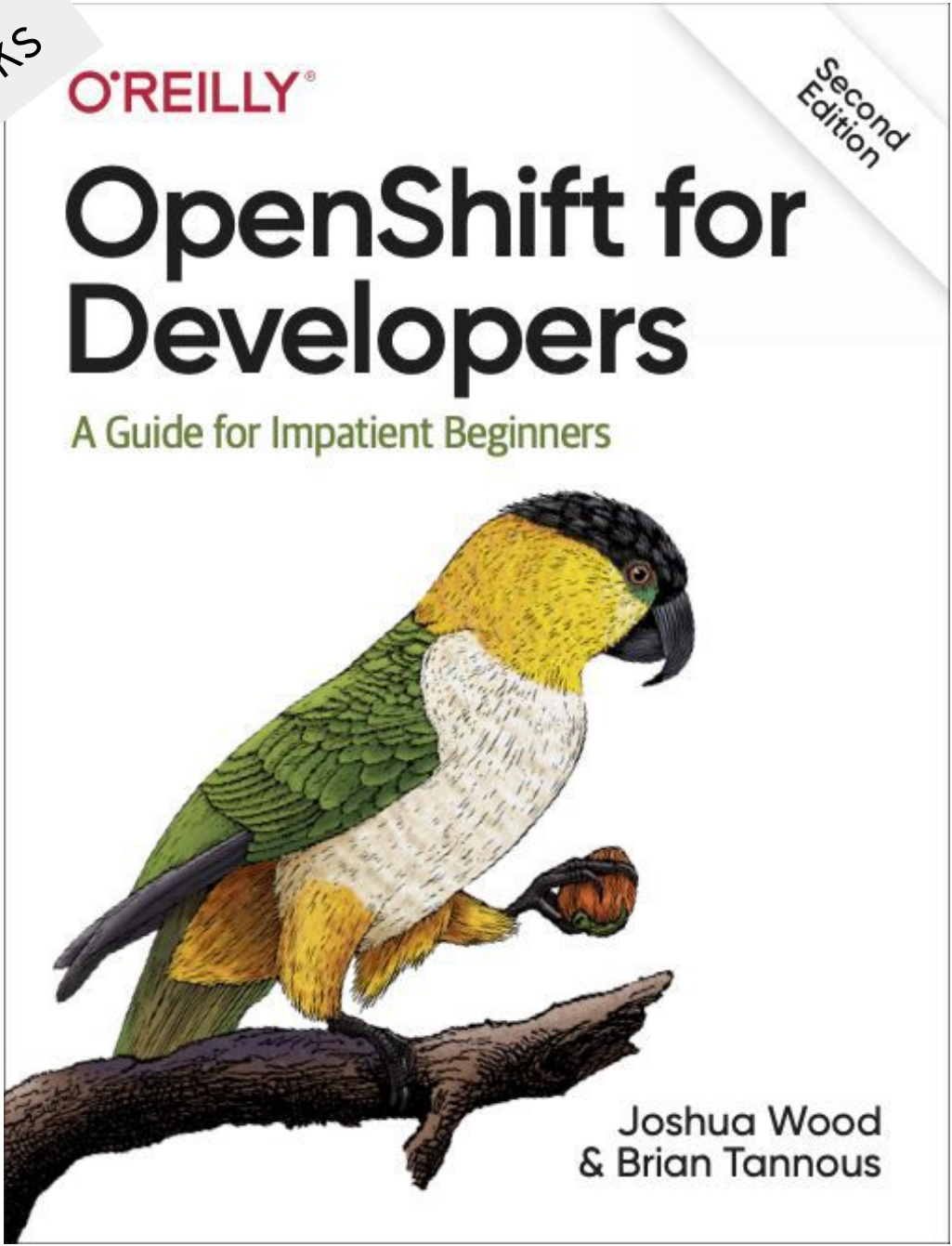
Get **free access** for renewable **30 days** to a self-service, cloud-hosted **Kubernetes** experience with **Developer Sandbox** for **Red Hat OpenShift**.

<https://developers.redhat.com/developer-sandbox>

```
[your@sandbox ~]$ lscpu  
RAM: 7GB  
Storage: 15GB  
Time limit: 30 days  
Awesome: YES
```



free e-books



[Download](https://red.ht/3lxJCzY)

<https://red.ht/3lxJCzY>





Summary

Summary

- ▶ New architectures and design principles drive new needs on technology. New requirements on Java to stay relevant.

e.g. Cloud and Edge Computing, Containers & K8S, MSA, EDA, Serverless/FaaS

- ▶ Quarkus has superfast startup times and low memory consumption, and at the same time provide a very pleasant and productive experience for developers.

- ▶ Red Hat is investing in upstream projects that modernise Java to meet new needs.



- ▶ Helps organisations to protect Java investments & skill sets to modernise legacy as well as develop the next generation of cloud native applications.

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.



linkedin.com/company/red-hat



youtube.com/user/RedHatVideos



facebook.com/redhatinc



twitter.com/RedHat