QUARKUS

EPISODE IV

QUARKUS. A NEW HOPE

EMERGING TECH VALENCIA

Juan Zuriaga
Fabián Martínez

About us

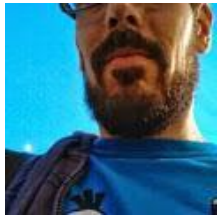
Fabián Martínez

famartin@redhat.com

Quality Engineer, Red Hat

Backgroud of Backend Developer, Java, climber, musician,...

@famargon13



Juan Zuriaga

jzuriaga@redhat.com

Certification Software Engineer

Red Hat

Preferences ... Vim editor, US kb layout, bright colors, climbing, cooking, and strong coffee

@juazugas

Content

- Why Java? Why Quarkus?
- What is Quarkus?
- Quick Demo “Hello Mario”
- How Quarkus works
- Demo 2. CRUD
- More demos. Performance & resources, ...
- Q&A

Why Java?
Why Quarkus?

Java, 25 years with us

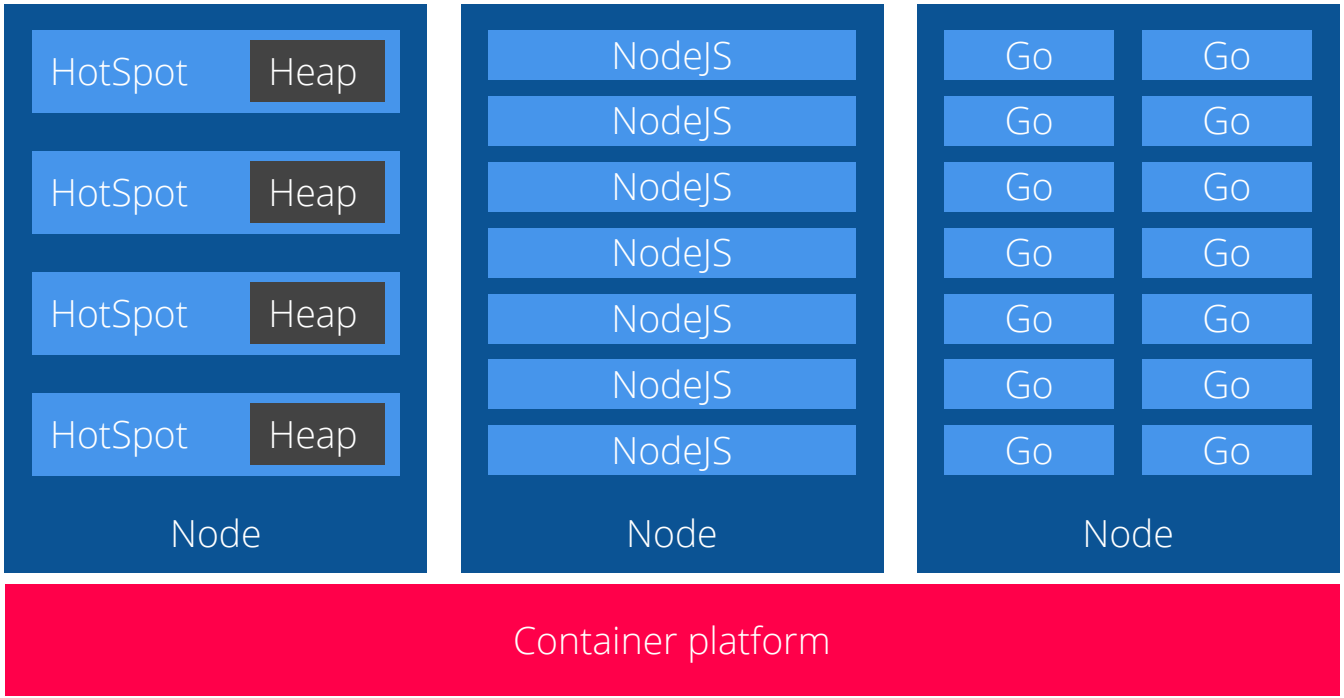
JVM & frameworks init on start-up

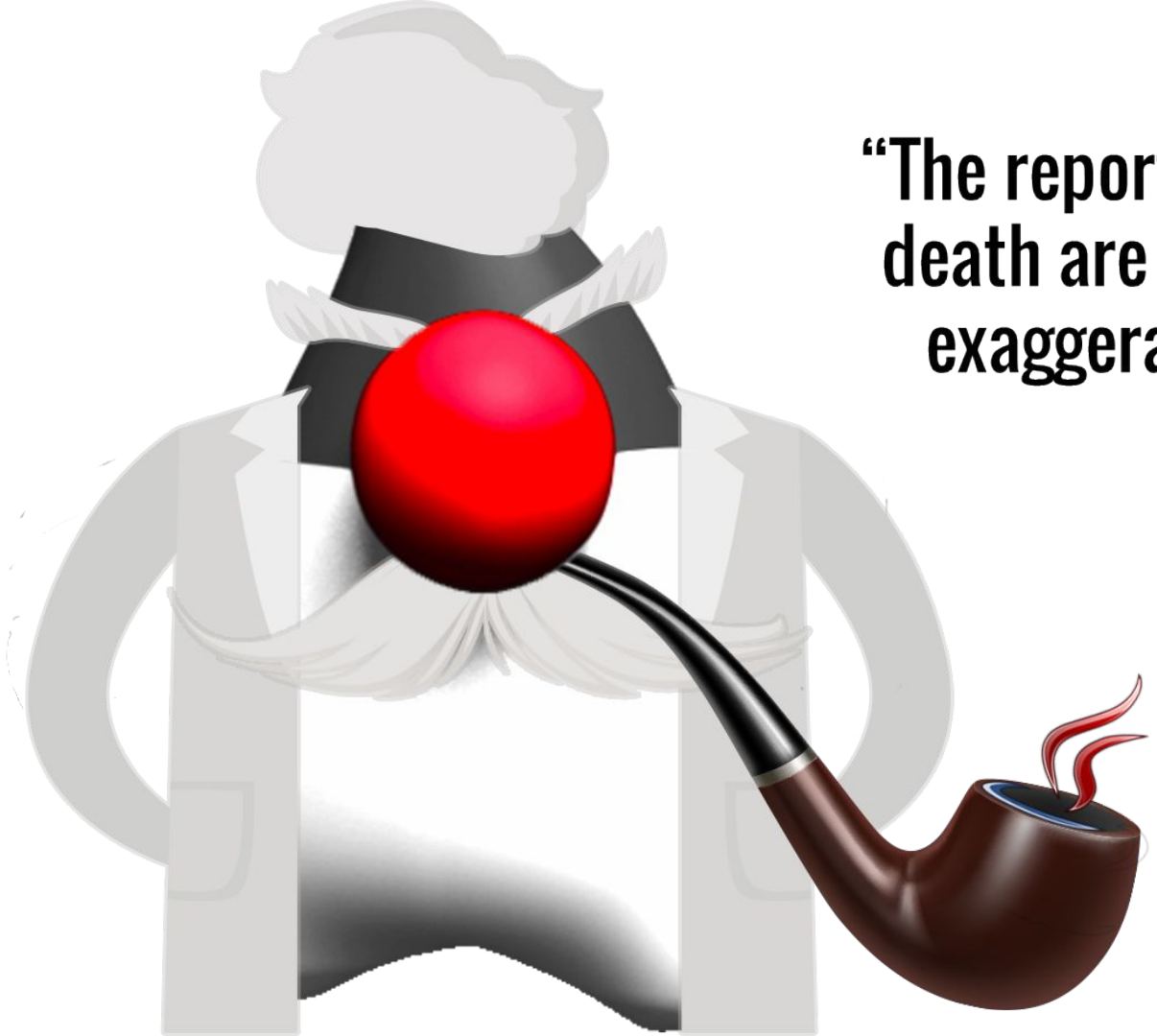
Okayish on-premises, bad for the cloud

- High memory consumption
- Long start-up time

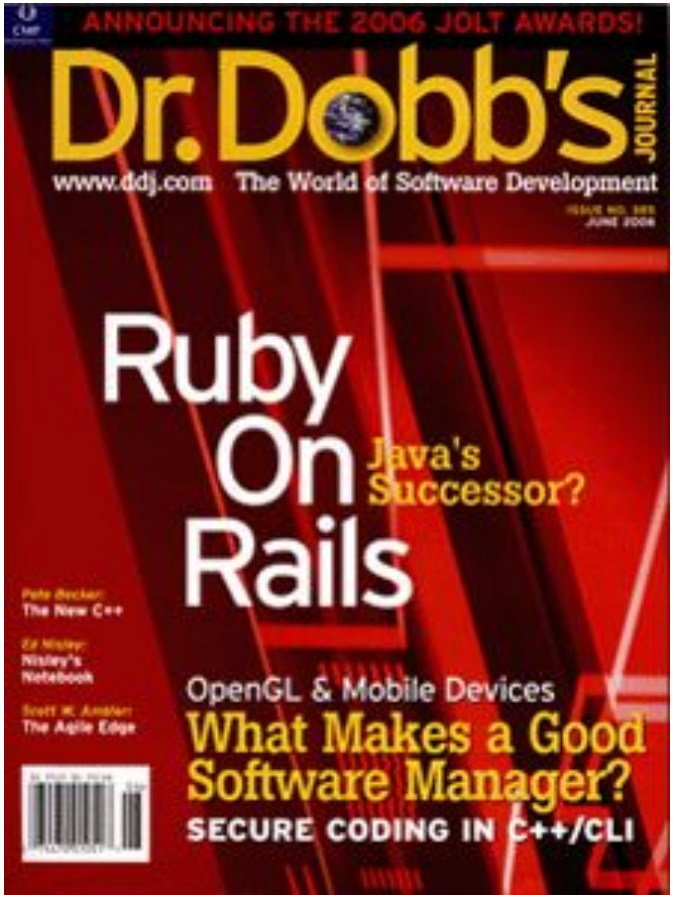
What about Java in the Cloud?

The hidden truth about Java + containers





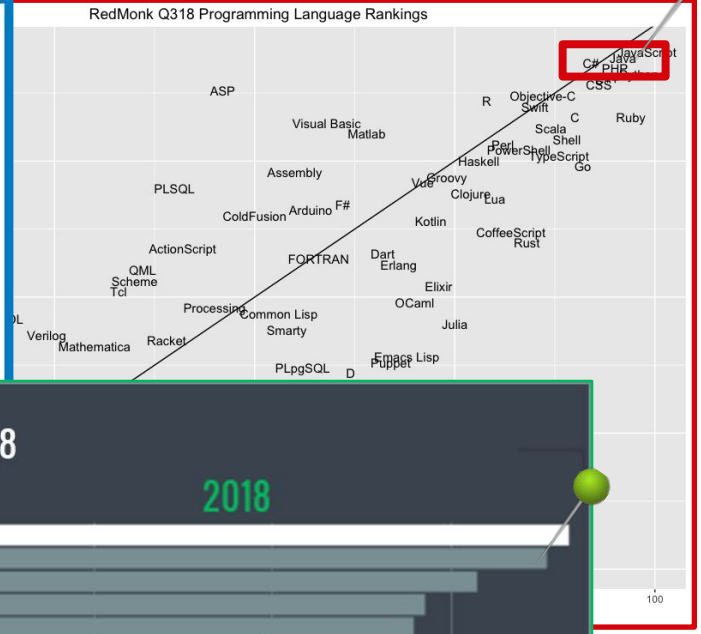
"The reports of my death are greatly exaggerated."

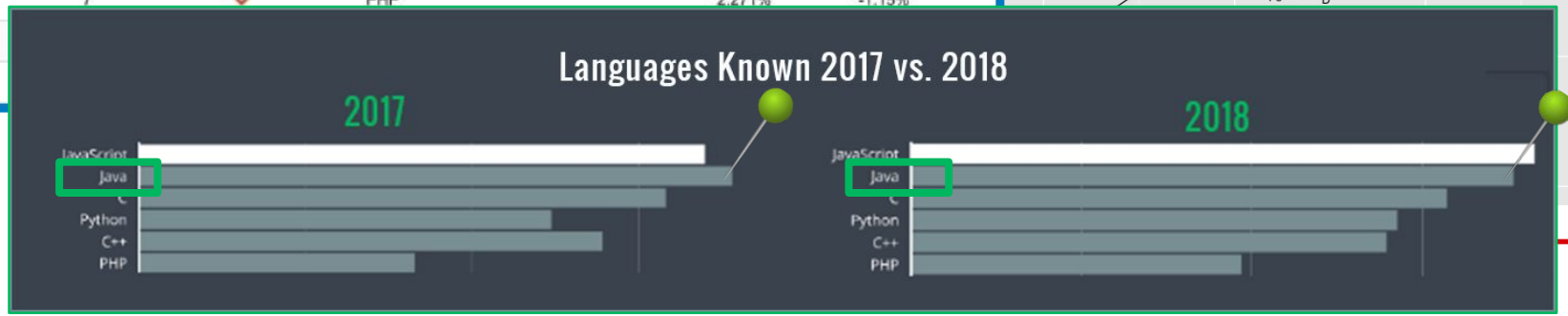

ANNOUNCING THE 2006 JOLT AWARDS!
Dr. Dobb's JOURNAL
www.ddj.com The World of Software Development
JUNE 2006
Ruby On Rails
Java's Successor?
Pete Becker:
The New C++
Ed Nisley:
Nisley's Notebook
Scott W. Ambler:
The Agile Edge
OpenGL & Mobile Devices
What Makes a Good Software Manager?
SECURE CODING IN C++/CLI

TIOBE

Feb 2019	Feb 2018	Change	Programming Language	Ratings	Change
1	1		Java	15.876%	+0.89%
2	2		C	12.424%	+0.57%
3	4	^	Python	7.574%	+2.41%
4	3	v	C++	7.444%	+1.72%
5	6	^	Visual Basic .NET	7.095%	+3.02%
6	8	^	JavaScript	2.848%	-0.32%
7	5	v	C#	2.346%	-1.61%
8	7	v	PHP	2.271%	-1.15%
9					
10					

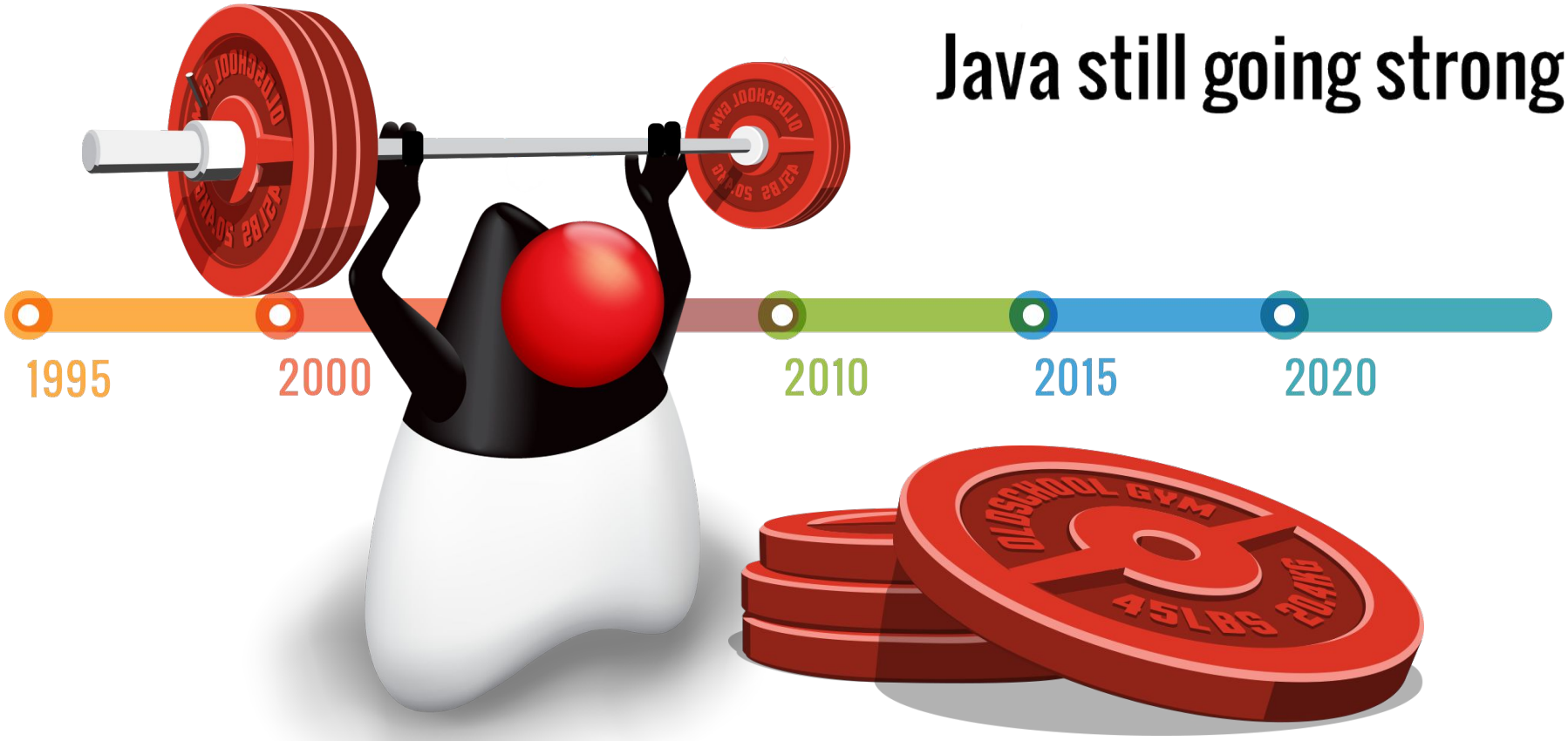
RedMonk





HackerRank

Java still going strong



In 2019 Red Hat announces...

QUARKUS

What is Quarkus?

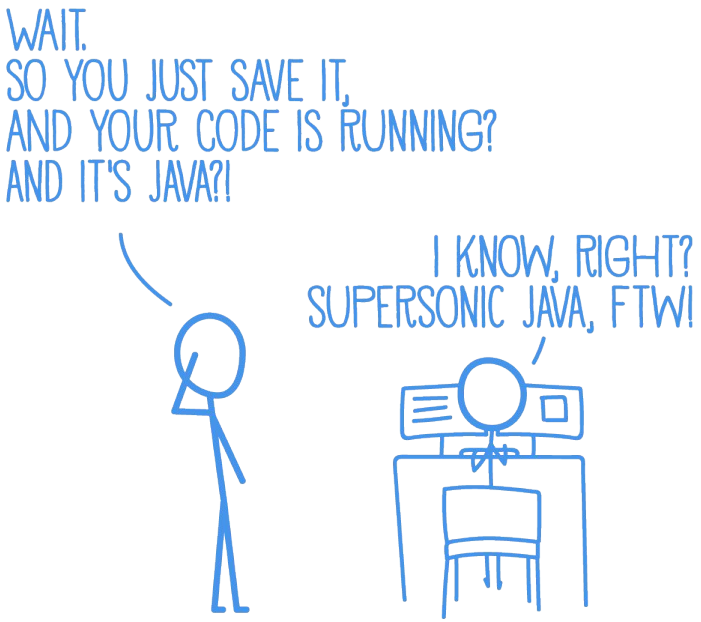
Supersonic. Subatomic. Java.

“A Kubernetes Native Java stack tailored for GraalVM & OpenJDK HotSpot, crafted from the best of breed Java libraries and standards” quarkus.io

Benefit No. 1: Developer Joy

A cohesive platform for optimized developer joy:

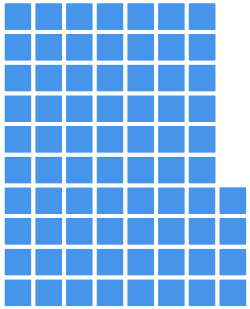
- Based on standards, but not limited
- Unified configuration
- Zero config, **live reload** in the blink of an eye
- Streamlined code for the 80% common usages, flexible for the 20%
- **No hassle native executable generation**



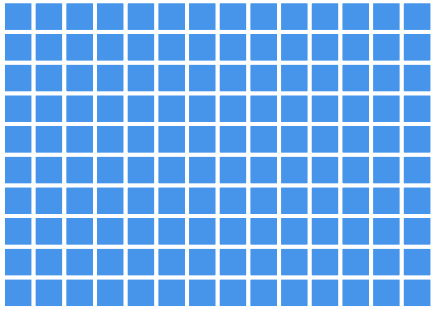
Benefit No. 2: Container first

REST

Quarkus + AOT
19 MB

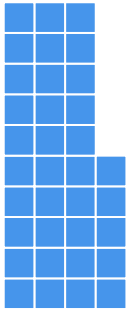


Quarkus + OpenJDK (JIT)
77 MB

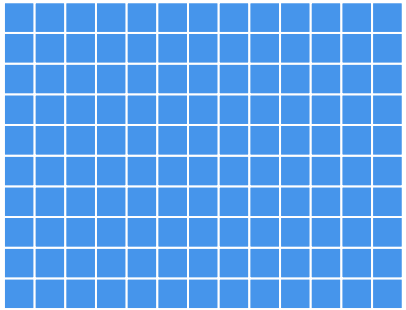


Traditional Cloud-Native Stack
140 MB

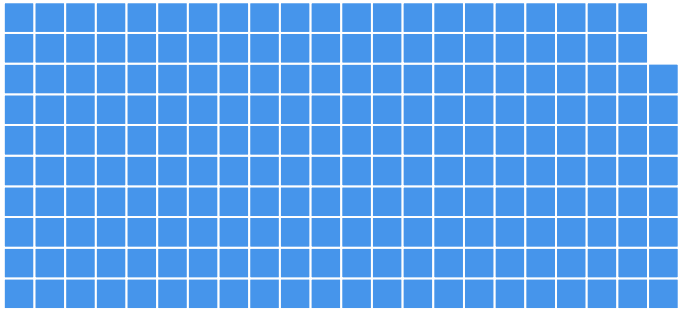
REST + CRUD



Quarkus + AOT
35 MB

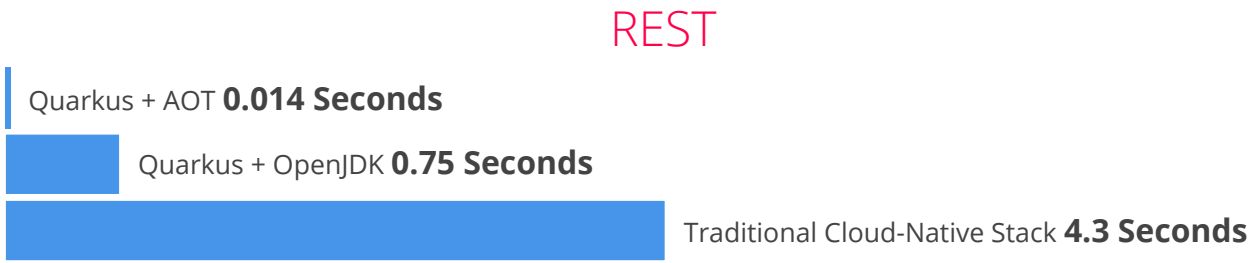


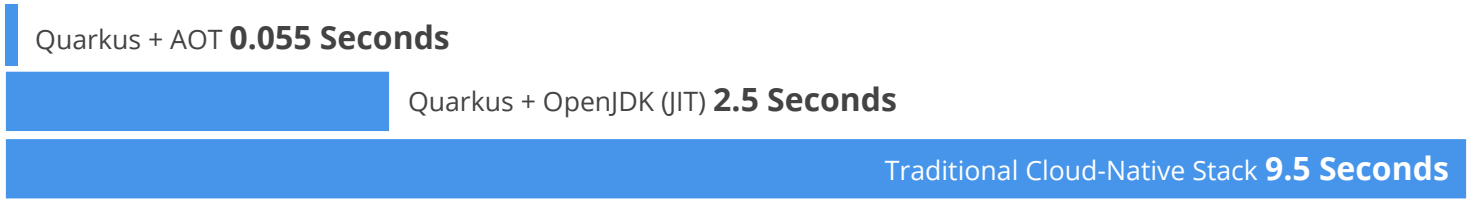
Quarkus + OpenJDK (JIT)
130 MB



Traditional Cloud-Native Stack
218 MB

Benefit No. 2: Container first





Time to first response

Benefit No. 3: Unifies Imperative and Reactive

```
@Inject
SayService say;

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

```
@Inject @Channel("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
- 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

Eclipse Vert.x

Eclipse MicroProfile

Spring Compat

Hibernate

RESTEasy

Apache Camel

Kubernetes

OpenShift

Jaeger

Prometheus

Apache Kafka

Netty

Demo 1
“It’s me, Mario”

How Quarkus works

Move start up time tasks to build time

Move startup time to build time

What does a framework do at startup time

- Parse config files
- Classpath & classes scanning
  - for annotations, getters or other metadata
- Build framework metamodel objects
- Prepare reflection and build proxies
- *Start and open IO, threads etc*

Build time benefits

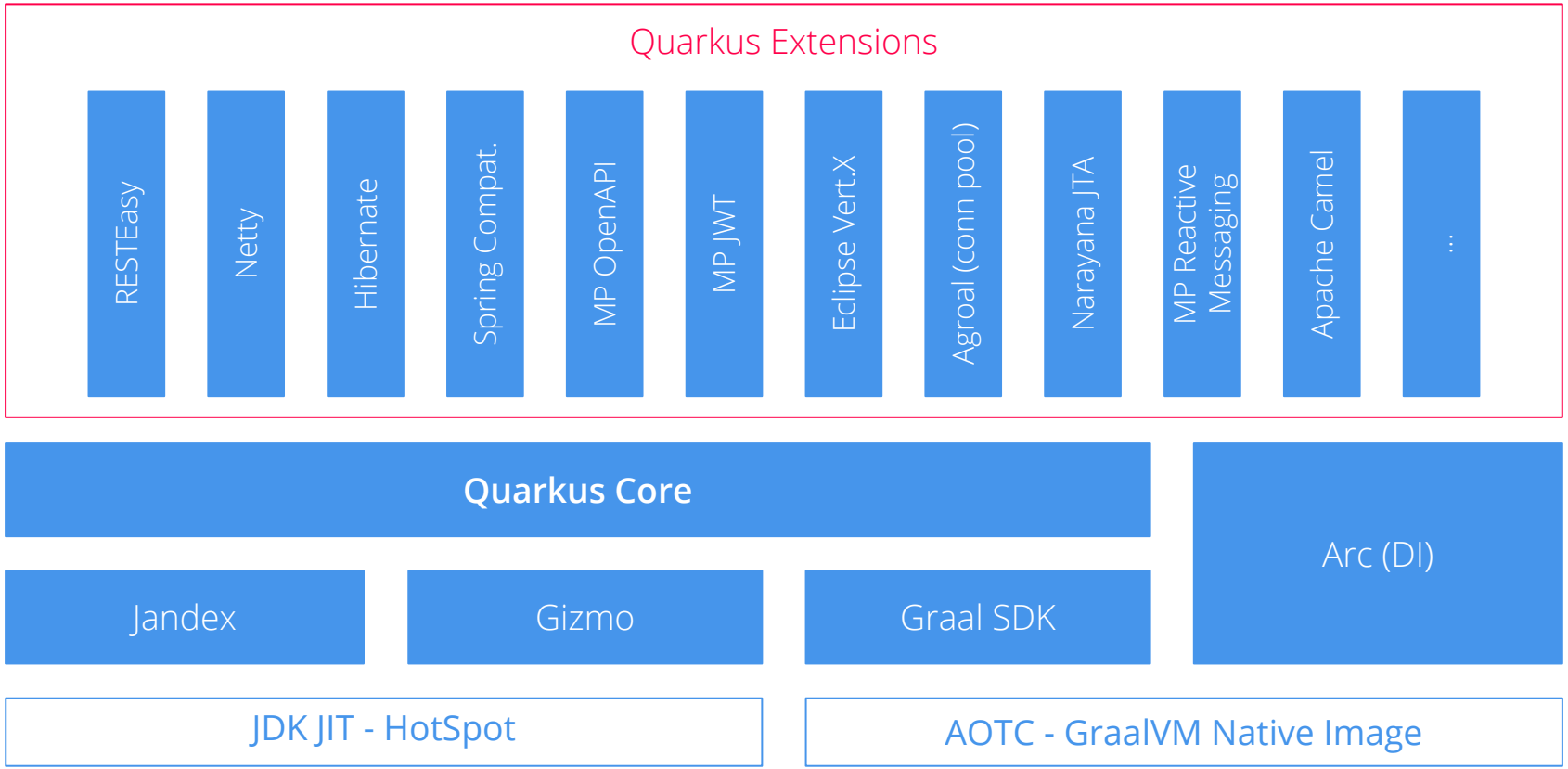
Do the work once, not at each start

All the bootstrap classes are no longer loaded

Less time to start, less memory used

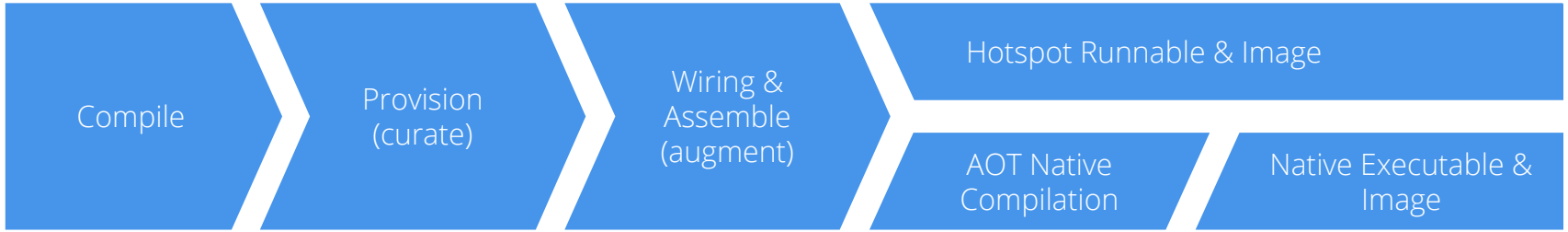
Less or no reflection nor dynamic proxy

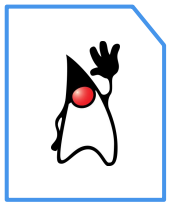
Extensions


Quarkus Extensions
RESTEasy
Netty
Hibernate
Spring Compat.
MP OpenAPI
MP JWT
Eclipse Vert.X
Agroal (conn pool)
Narayana JTA
MP Reactive Messaging
Apache Camel
...
Quarkus Core
Arc (DI)
Jandex
Gizmo
Graal SDK
JDK JIT - HotSpot
AOTC - GraalVM Native Image

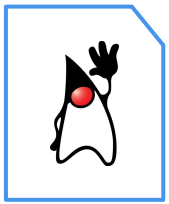
Quarkus
The nitty-gritty

An ahead-of-time, build-time, runtime





app.jar



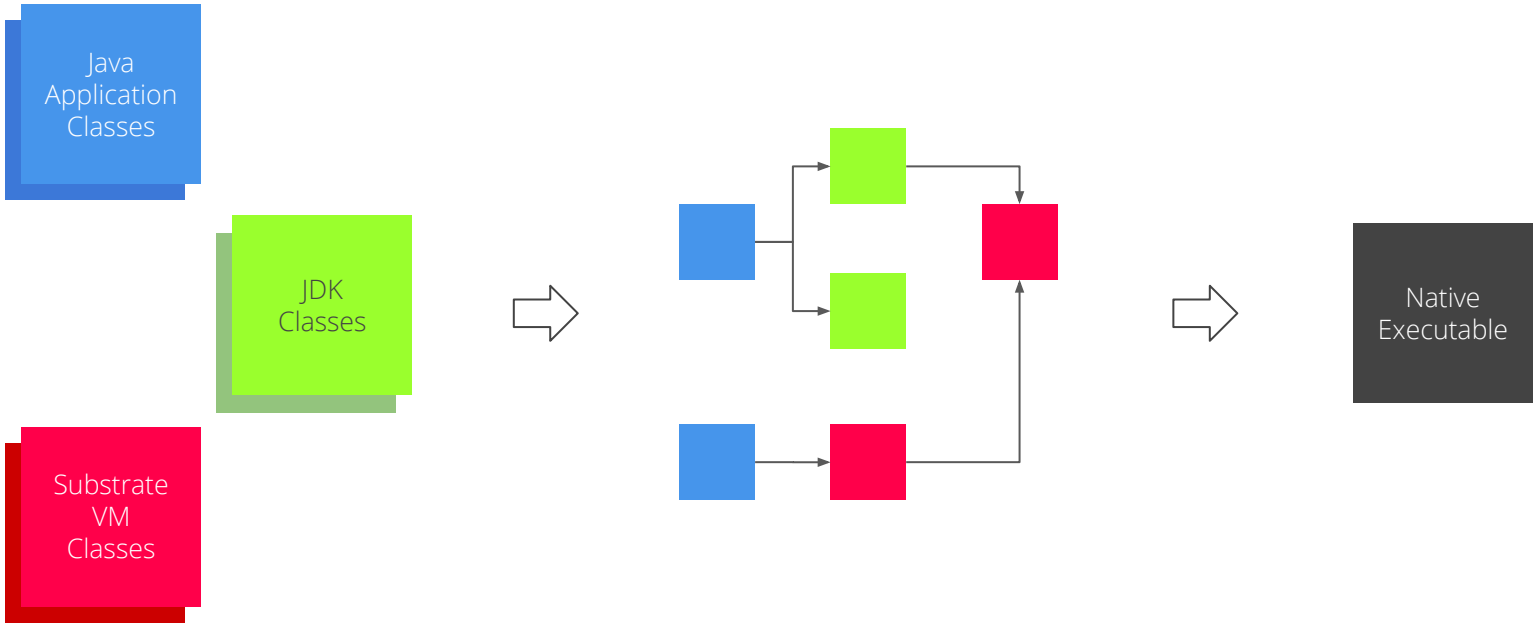
frameworks

Runnable java app

native-app

AOTC - GraalVM - Dead code elimination

Closed-world assumption



Quarkus eases GraalVM utilization

Drives the gathering of metadata needed by GraalVM

- based on framework knowledge
- Classes using reflection, resources, etc

Minimize dependencies

Help dead code elimination

Developer doesn’t even realize it’s using Graal VM

When to use which VM with Quarkus

JIT - OpenJDK HotSpot

High memory density requirements
High request/s/MB
Fast startup time

Best raw performance (CPU)
Best garbage collectors
Higher heap size usage

Known monitoring tools
Compile Once, Run anywhere
Libraries that only works in standard JDK

AOT - GraalVM native image

Highest memory density requirements
Highest request/s/MB
    for low heap size usages
Faster startup time
    10ths of ms for Serverless

Demo 2

CRUD application

More demos

the more the better

Thank you

Survey Query for next talk/s ...

- Spring plus Quarkus plus Vert.x
- Event driven apps with Quarkus
- Quarkus: Secure your apps
- OpenShift integration:
  Deployment, monitoring and observability

Suggestions?

Q&A

Resources

https://quarkus.io

https://quarkusio.zulipchat.com

@quarkusio

Code Quarkus Demo code

https://github.com/emerging-tech-vlc/quarkus-intro

Quarkus why, how and what

by Emmanuel Bernard

https://www.youtube.com/watch?v=SQDR34KoC-8&t=693s

Katacoda: Developing with Quarkus

https://www.katacoda.com/openshift/courses/middleware/middleware-quarkus/

GraalVM

https://www.graalvm.org/

Quarkus Reaches Version 1.0:

Q&A with Thomas Qvarnstrom

https://www.infoq.com/articles/redhat-release-quarkus-1-0/

QUARKUS

Supersonic. Subatomic. Java.

Juan Zuriaga
Fabián Martínez