RED HAT FORUM | September 10 2019

QUARKUS: SUPERSONIC, SUBATOMIC JAVA - HOW TO GET STARTED

DIMITRIS ANDREADIS, SOFTWARE ENGINEERING, RED HAT

QUARKUS

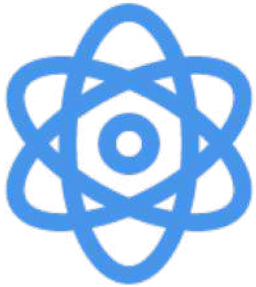
Supersonic. Subatomic. Java.

Dimitris Andreadis
Engineering Director
Red Hat

@dandreadis

A stack to write Java apps

Cloud Native,



Microservices,

Serverless

QUARKUS?

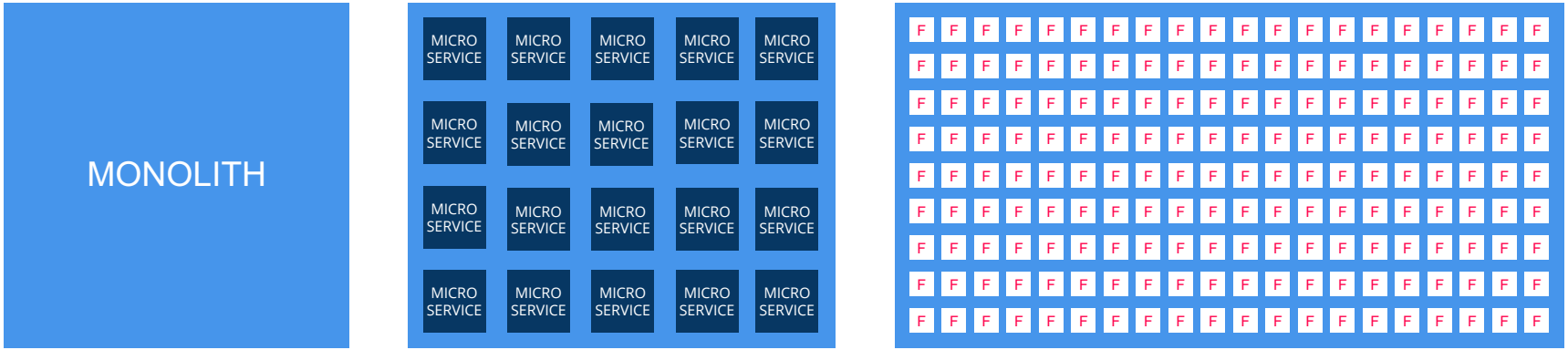
QUARK: elementary particle / **US**: hardest thing in computer science

…6 months ago

What is the best way to write Cloud Native Apps

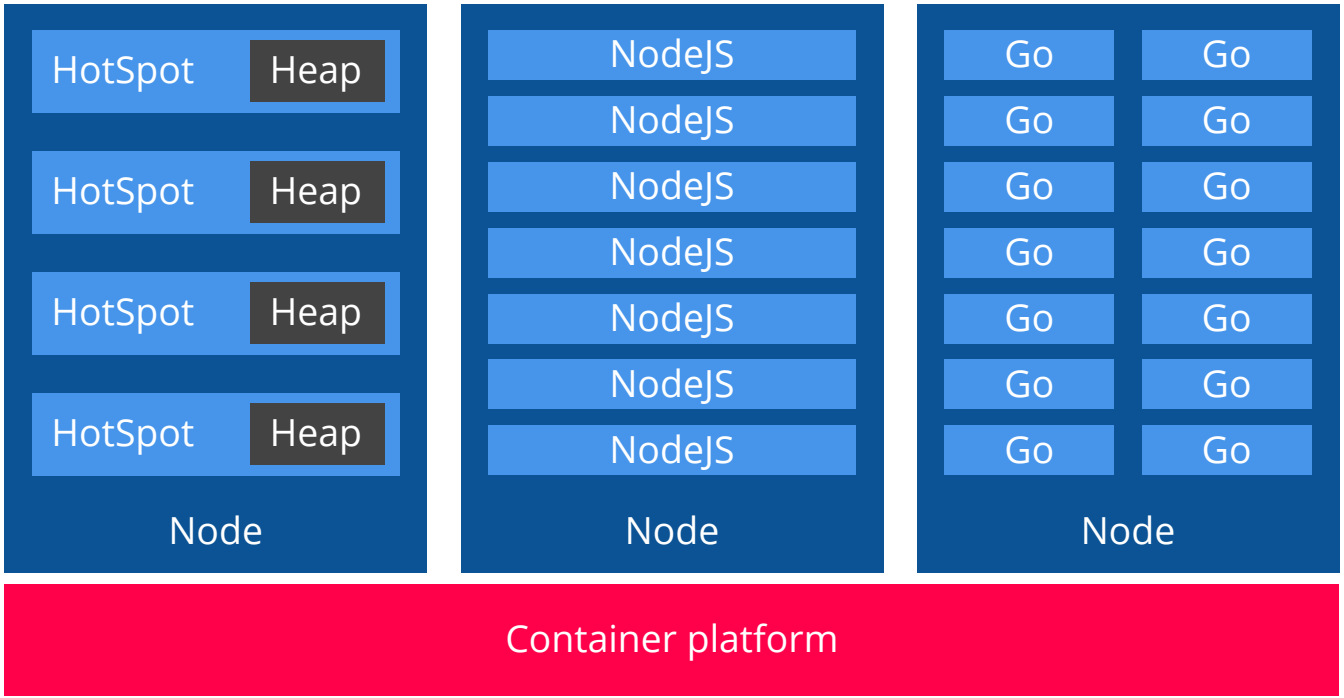
in Java

A Deployment Continuum



- 1 monolith ≈ 20 microservices ≈ 200 functions
- Long running process(es), vs scale up/down, vs scale infinitely and back to 0
- Start up time and density becomes key

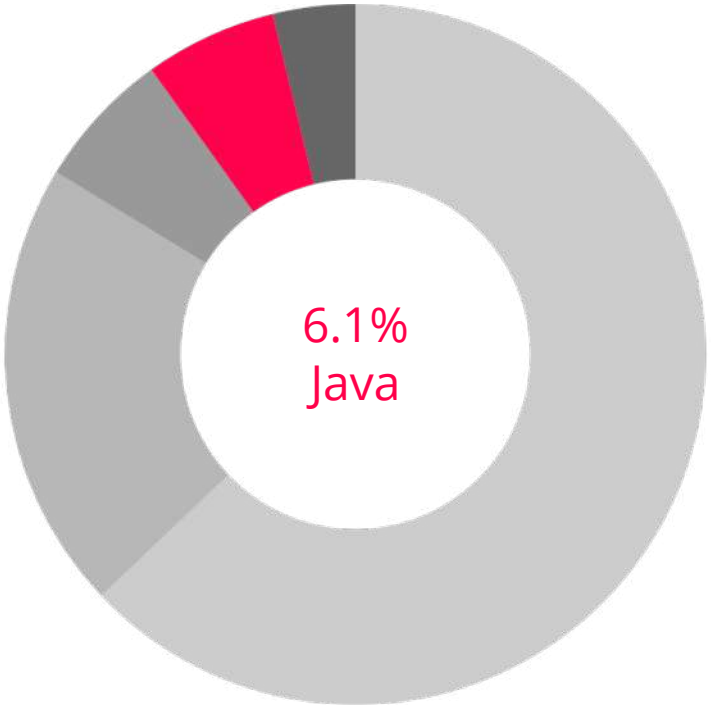
Deployment density matters



The simple truth about Java + containers

- Startup overhead
  - # of classes, bytecode, JIT
- Memory overhead
  - # of classes, metadata, compilation

Serverless Adoption



62.9% Node.js

20.8% Python

6.4% Go

6.1% Java

3.8% C#

https://serverless.com/blog/2018-serverless-community-survey-huge-growth-usage/

Looking for solutions

Java on cloud native: a Red Hat journey

Standards

- Java EE streamlining
- Eclipse MicroProfile

Runtimes

- WildFly on Raspberry Pi
- WildFly on OpenShift
- WildFly-Swarm / Thorntail

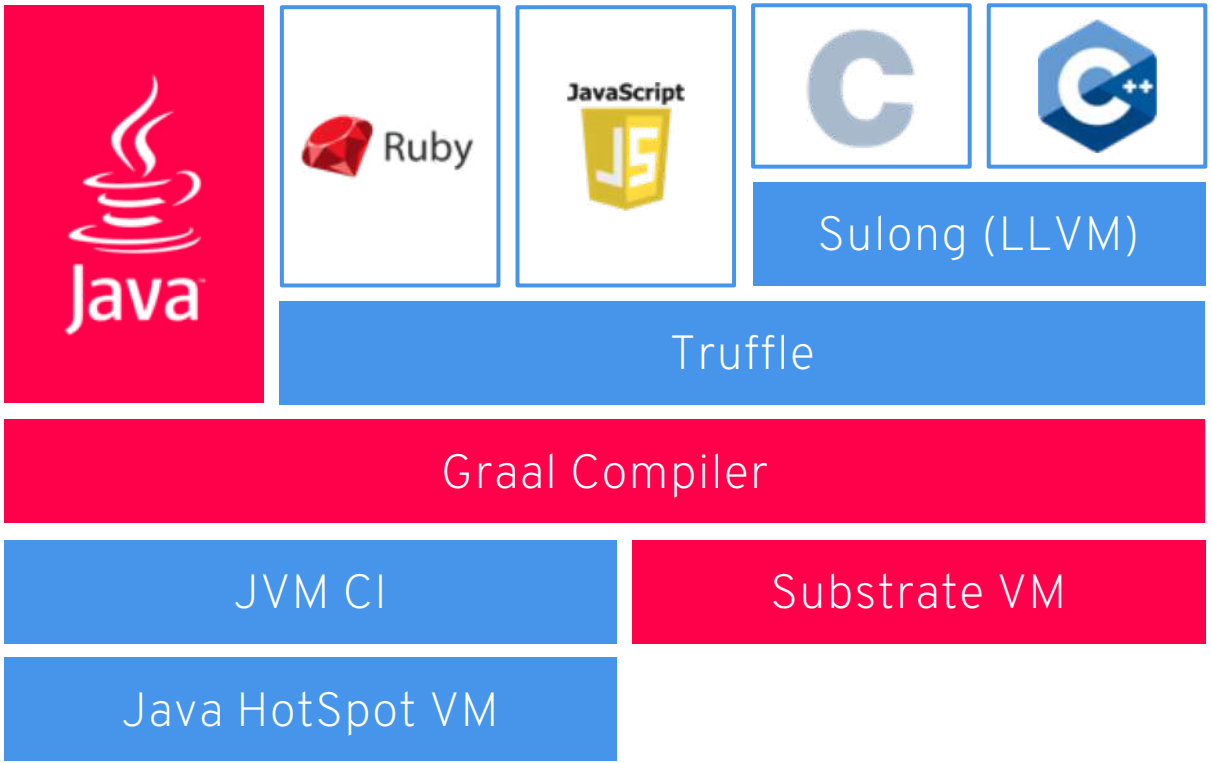
Ahead of time compilation

- Lead gcj (GNU Compiler for Java)
- Looked at Dalvik, Avian, Excelsior JET

OpenJDK

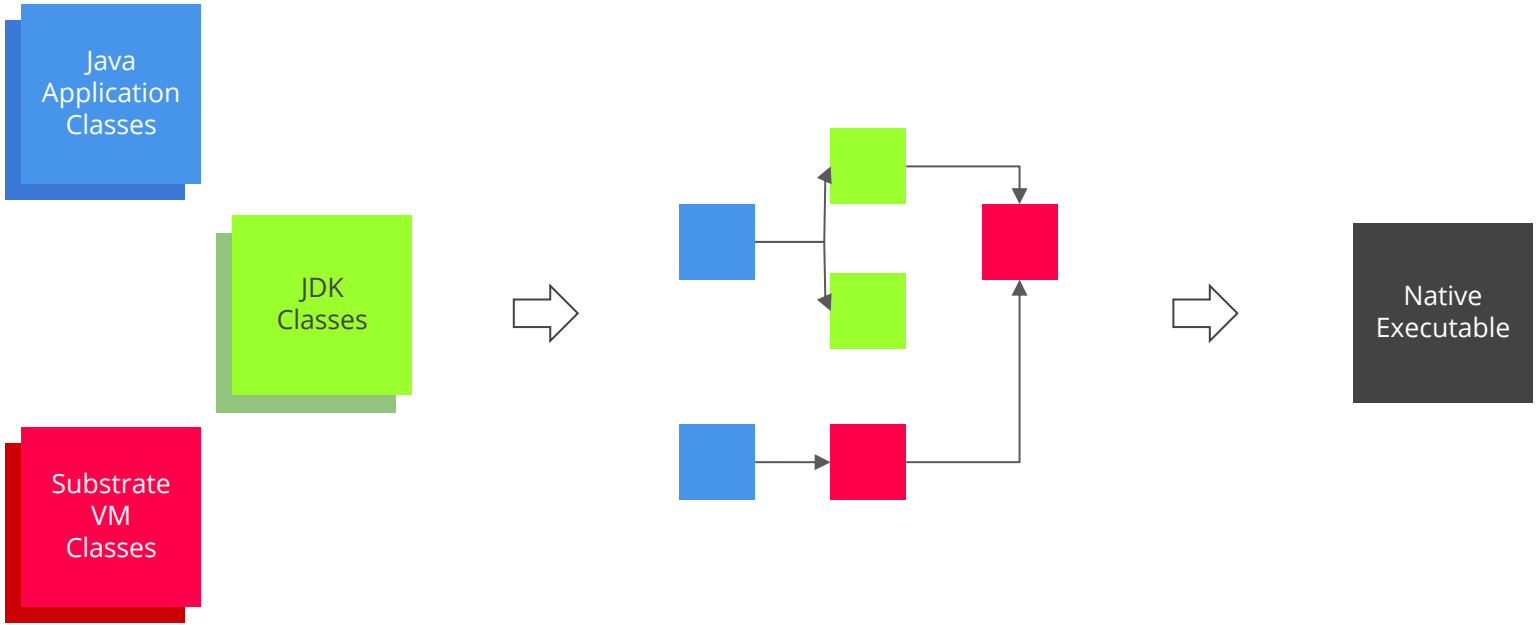
- Container ergonomics
- JVM metadata reduction


The Holy
GraalVM
You keep using that word.
I do not think it means what you think it means.


Java
Ruby
JavaScript
Sulong (LLVM)
Truffle
Graal Compiler
JVM CI
Substrate VM
Java HotSpot VM

Dead code elimination

Closed-world assumption



The Dark Side

AOT is not That Simple

Not supported

- Dynamic classloading
- InvokeDynamic & Method handles
- Finalizers
- Security manager
- JVMTI, JMX, native VM Interfaces

OK with caveats in usage

- Reflection (requires configuration)
- Dynamic proxy (requires configuration)
- JNI (requires configuration)
- Static initializers (eager)
- Lambda, Threads (Okay)
- References (mostly supported)

https://github.com/oracle/graal/blob/master/substratevm/LIMITATIONS.md

Quarkus to the rescue!

Application initialization takes time!

What does a framework do at startup time?

- Parse config files
- Scan the classpath & classes themselves
- Discover annotations, getters or other metadata
- Build framework metamodel objects
- Prepare reflection and build proxies
- *Start and open IO, threads etc*

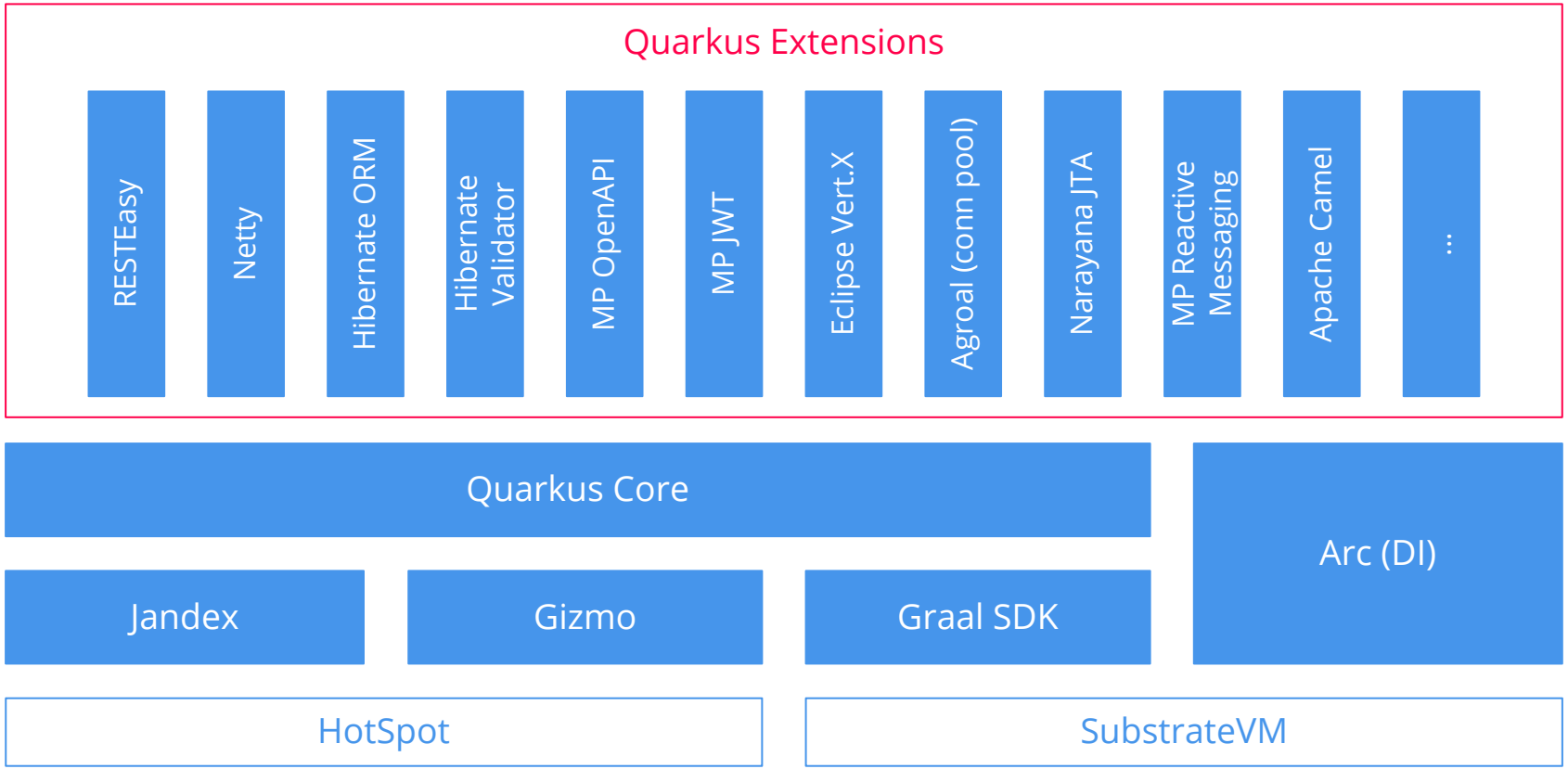
Why not Initialize at Build time (vs Runtime)?

Do the work once, not at each start

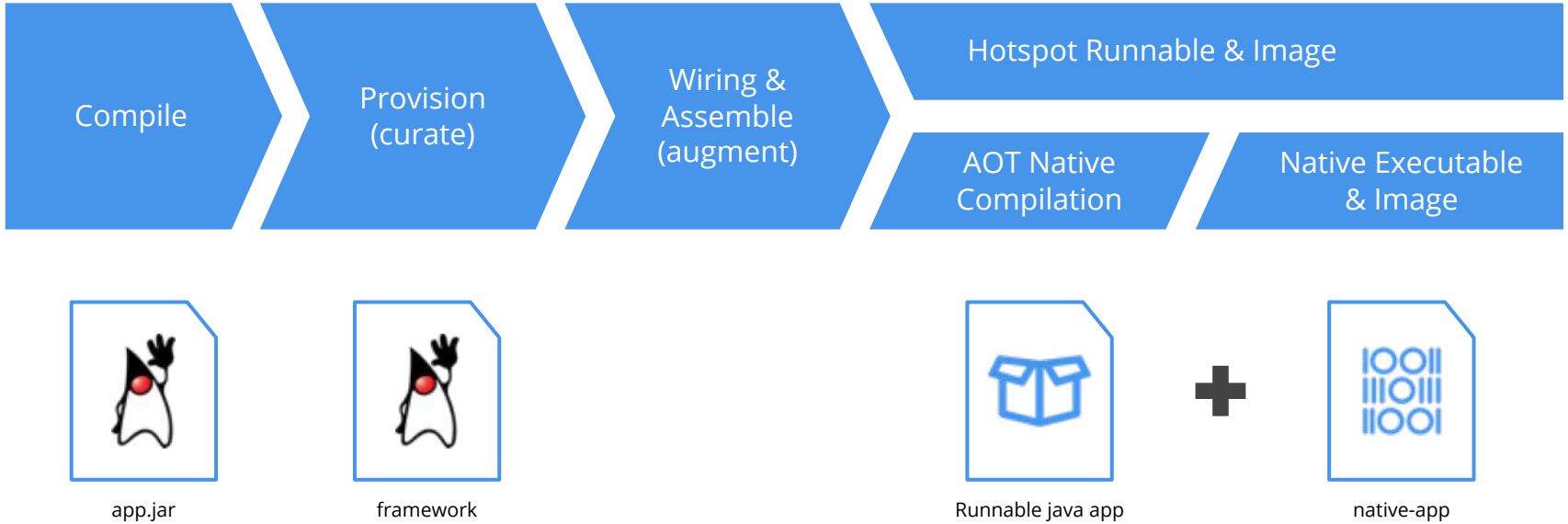
All the bootstrap classes are no longer loaded

Less time to start, less memory used

Less or no reflection nor dynamic proxies


Quarkus Extensions
RESTEasy
Netty
Hibernate ORM
Hibernate Validator
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
HotSpot
SubstrateVM

Quarkus Build Process

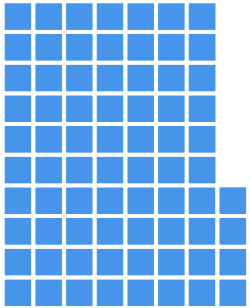


DEMO

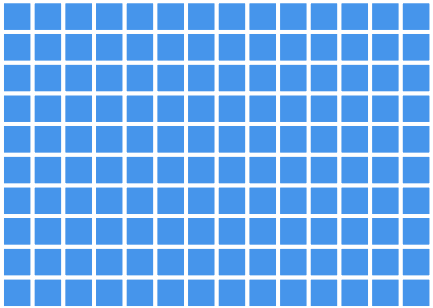
Benefit No. 1: Supersonic Subatomic Java

REST

Quarkus + GraalVM
13 MB



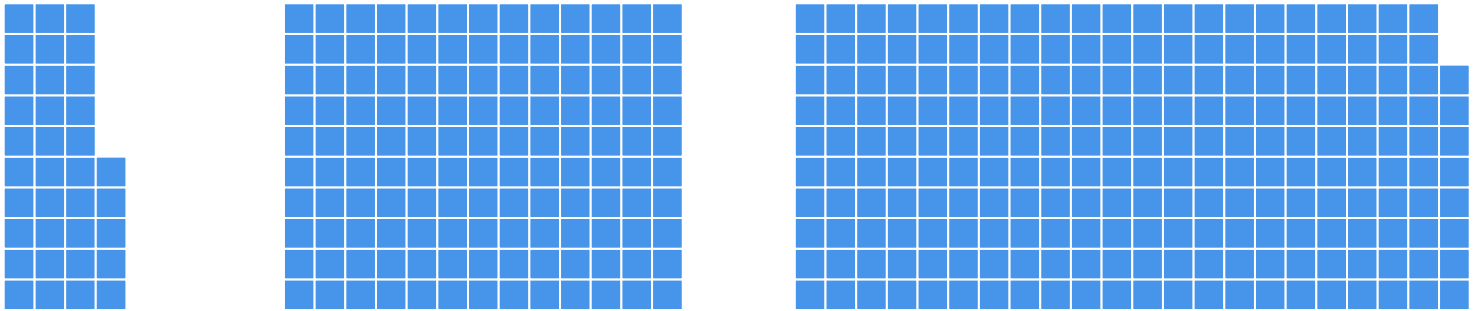
Quarkus + OpenJDK
74 MB



Traditional Cloud-Native Stack
140 MB

Benefit No. 1: Supersonic Subatomic Java

REST + CRUD



Quarkus + GraalVM
35 MB

Quarkus + OpenJDK
130 MB

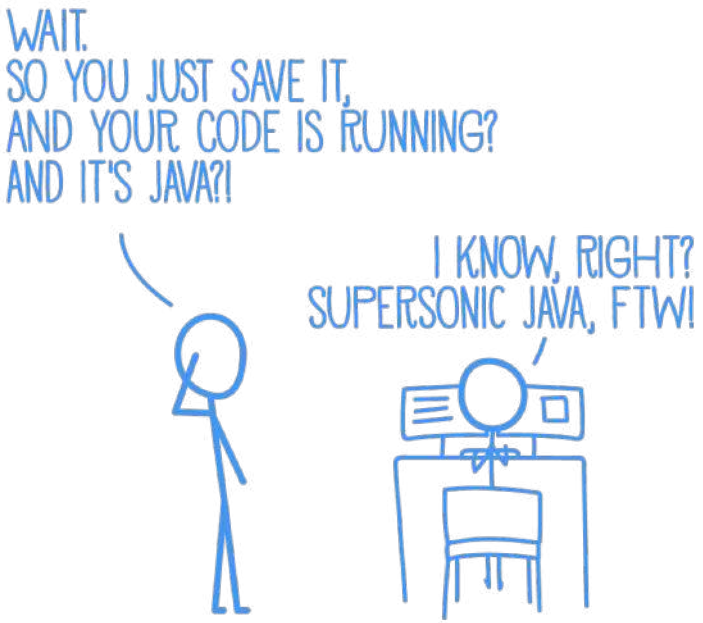
Traditional Cloud-Native Stack
218 MB

Benefit No. 1: Supersonic Subatomic Java

Benefit No. 2: 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. 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

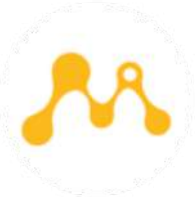
Benefit No. 4: Best of Breed Frameworks & Standards

Eclipse Vert.x

Hibernate

RESTEasy

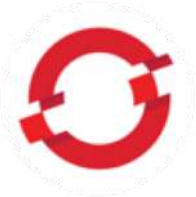
Apache Camel



Eclipse MicroProfile

Netty

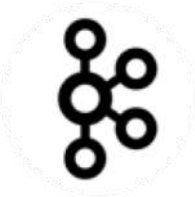
Kubernetes



OpenShift

Jaeger

Prometheus



Apache Kafka

Infinispan

Want to learn more?

QUARKUS

https://quarkus.io

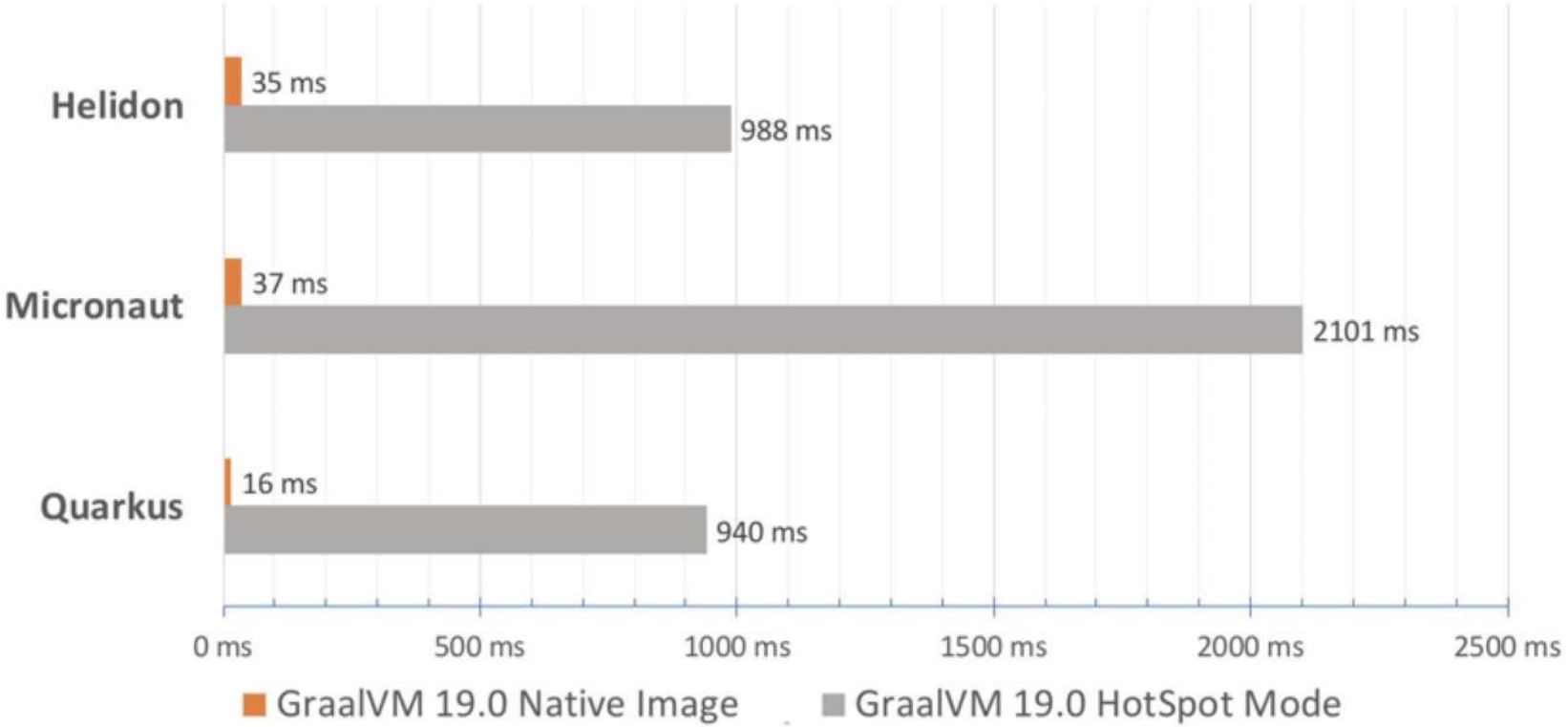
https://quarkusio.zulipchat.com

@quarkusio

Thank you

Java Microservice: Startup Time

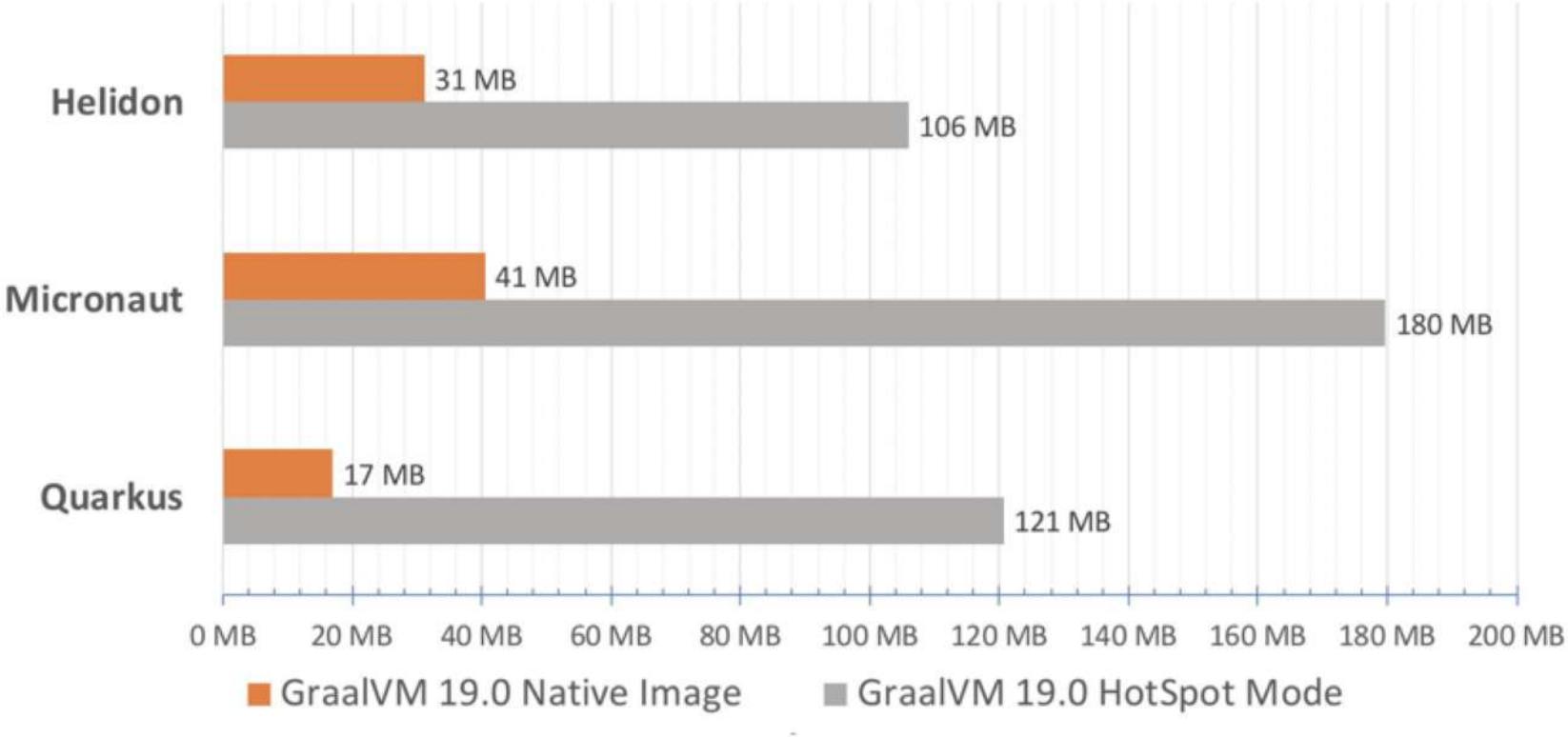
~50x faster



https://twitter.com/graalvm/status/1133448633943306240/photo/1

Java Microservice: Memory Footprint

~5x lower



https://twitter.com/graalvm/status/1133448633943306240/photo/2


Red Hat
RED HAT FORUM | Next Up 14:50
Manage Windows with Ansible
by Roland Wolters