Ndongo Tonux SAMB

Context?

Containers

kubernetes

FaaS

serverless
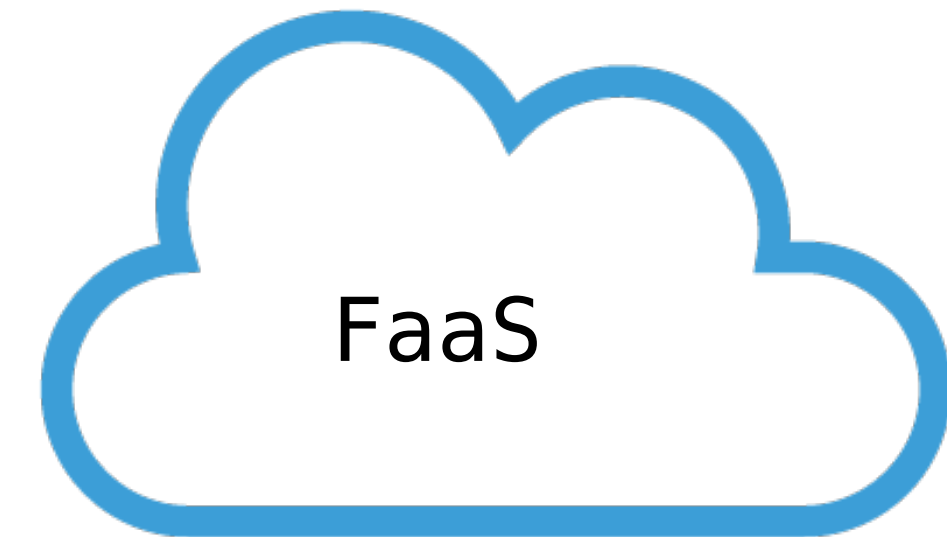
Microservices

Quarkus?

TOOLKIT

and

FRAMEWORK

For writing Java applications

Quarkus?

*.class

QUARKUS

Maven/Gradle plugin

Optimized jar

JVM

GraalVM™

Native
executable

Quarkus core?

Jandex	Gizmo	Graal SDK	Weld Arc
Deploy (Build)		**Runtime**	
Maven		SmallRye Config	

RESTEasy · Undertow · Hibernate · Bean Validation · Narayana · Agroal · JBoss Logging · MP Health · MP Rest Client · Fault Tolerance · MP Metrics · MP Open API · OpenSSL

Extensions
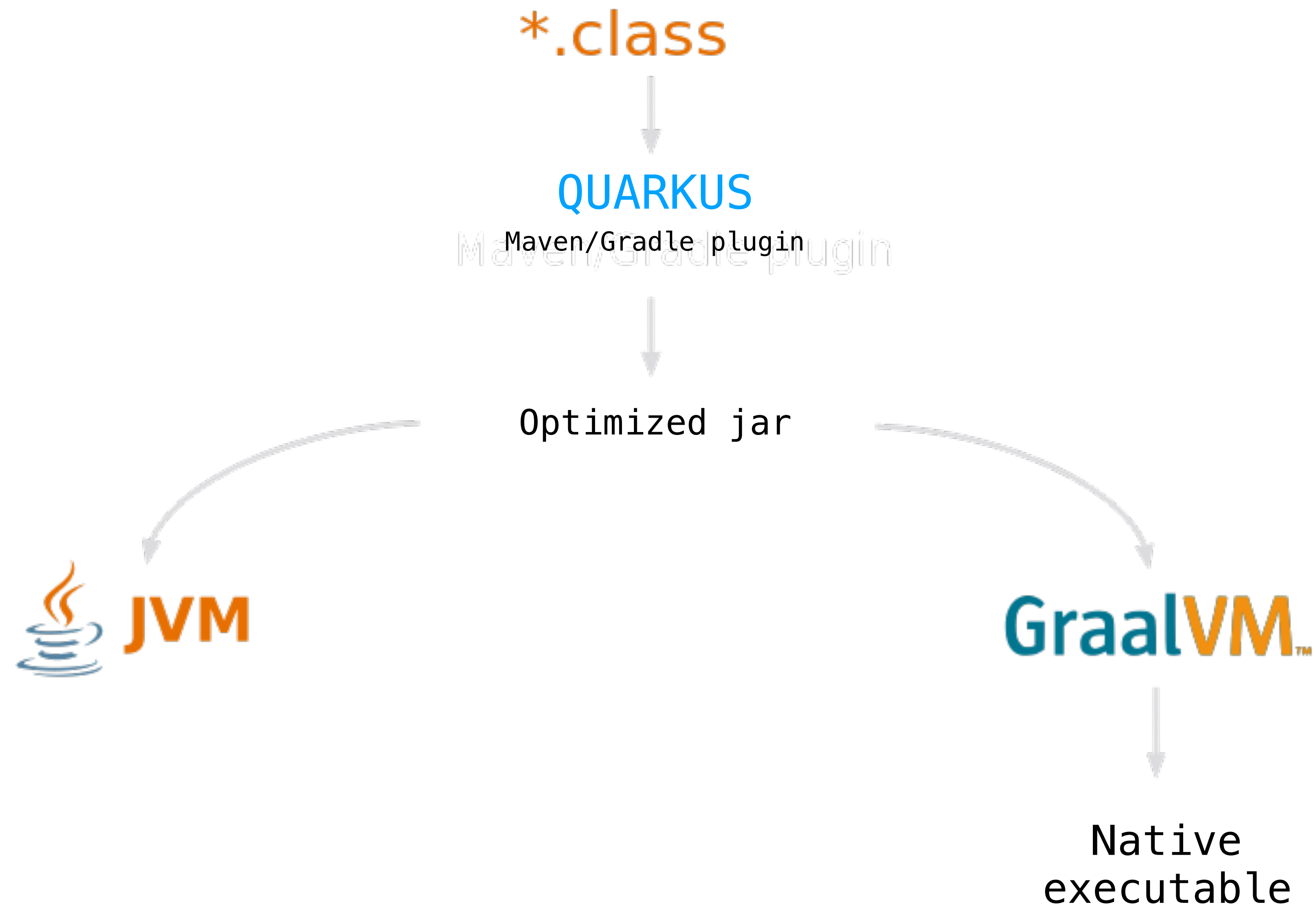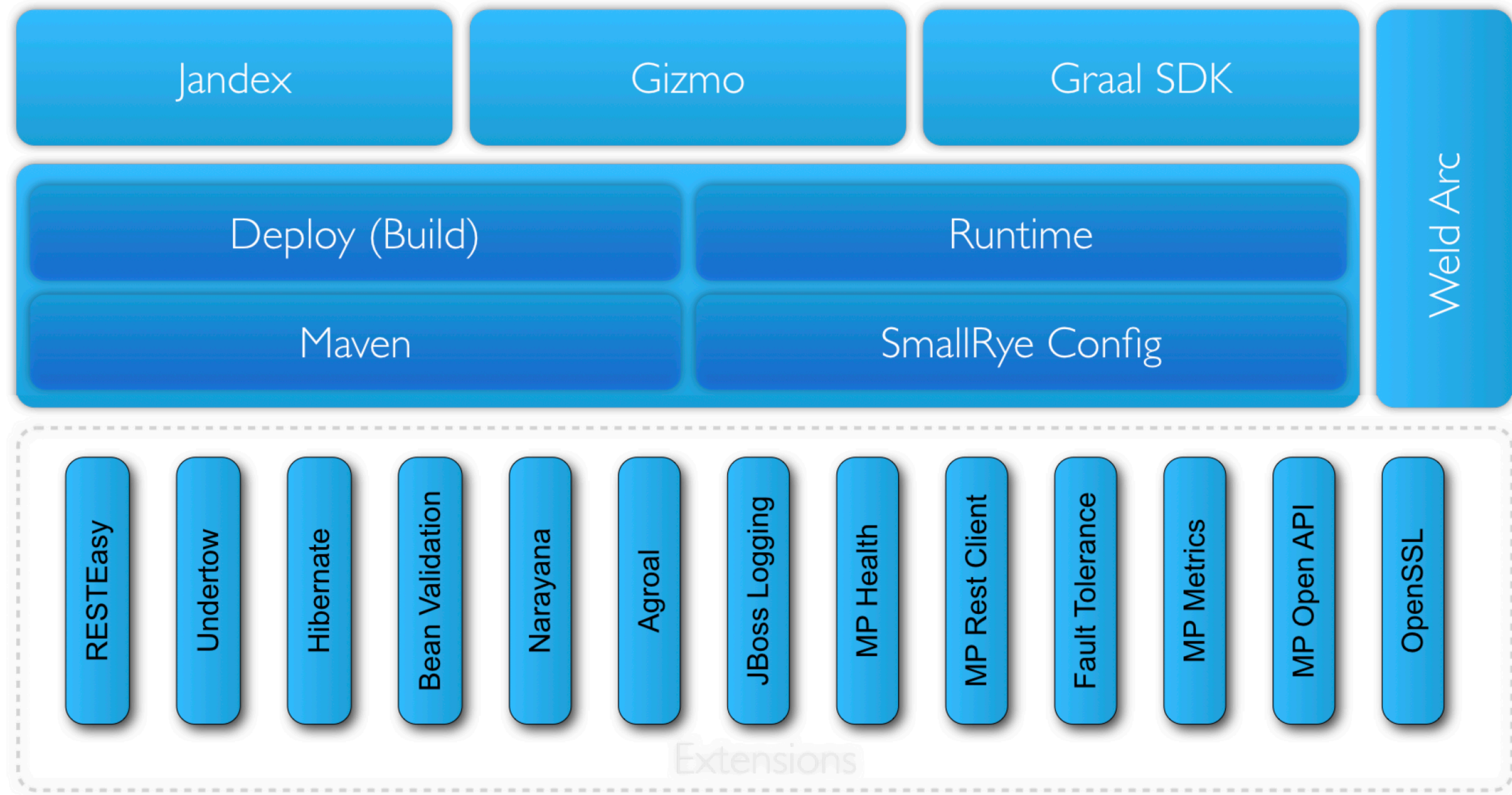
Existing Standards

MICROPROFILE™

✦Servlet
✦JAX-RS
✦JPA, JDBC
✦CDI
✦Bean Validation
✦Transactions
✦Logging

✦OpenAPI
✦OpenTracing
✦Reactive Messaging
✦Rest Client

✦Fault Tolerance
✦Health
✦JWT
✦Metrics

Libs and Frameworks

Eclipse Vert.x · Hibernate · RESTEasy · Apache Camel · Eclipse MicroProfile · Netty

Kubernetes · OpenShift · Jaeger · Prometheus · Apache Kafka · Infinispan

Flyway · Neo4j · MongoDB · MQTT · KeyCloak · Apache Tika

Red Hat

Imperative and Reactive

Combine both the familiar imperative
code and the non-blocking reactive
style when developing applications

```java
@Inject
SayService say;

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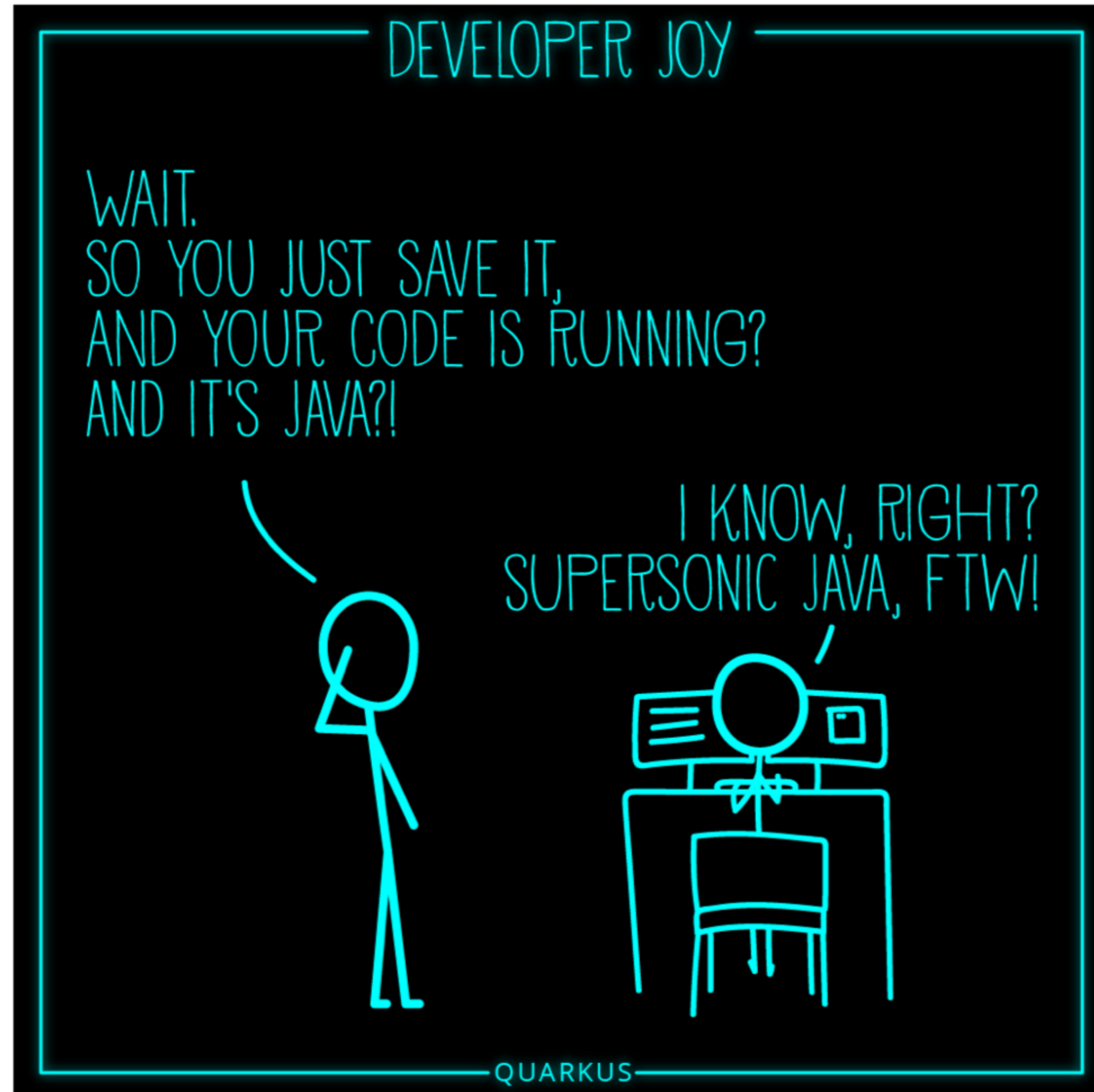
Imperative

```java
@Inject @Stream("kafka")
Publisher<String> reactiveSay;

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

Reactive

Developper Joy

- Easy to start with
- Live reload, a.k.a. dev mode
- Java or Kotlin
- Maven or Gradle
- Unified configuration
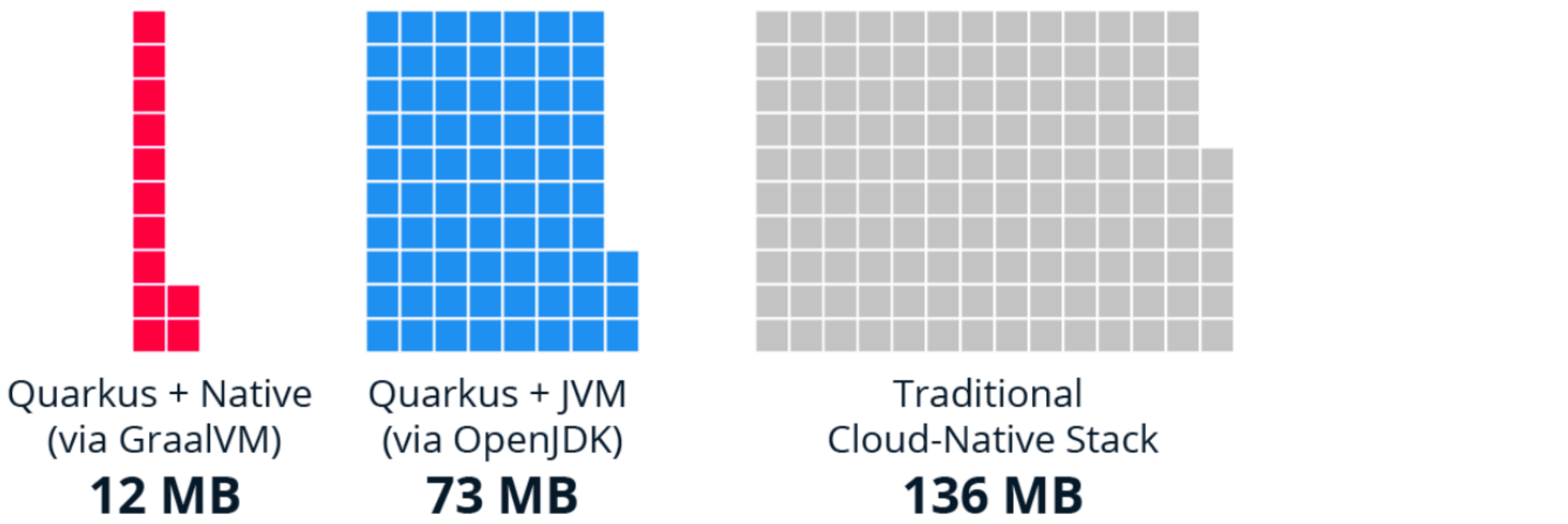- Runners for JUnit 4 and 5
- Isolation from GraalVM CLI/API
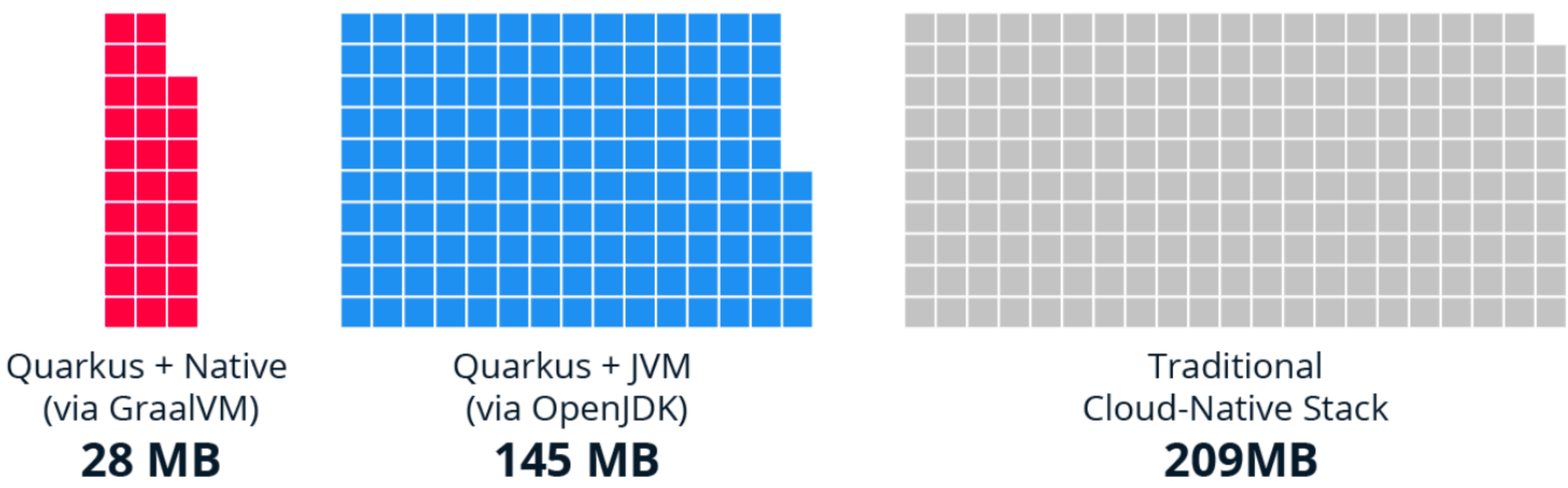- …

Container first

💾 Small size on disk   ✓ Small container images

🚀 Fast boot time   ✓ Instant scale up

🔬 Low RSS   memory   ✓ More containers with the same RAM

Memory

REST

Quarkus + Native
(via GraalVM)
12 MB

Quarkus + JVM
(via OpenJDK)
73 MB

Traditional
Cloud-Native Stack
136 MB

REST
+ CRUD

Quarkus + Native
(via GraalVM)
28 MB

Quarkus + JVM
(via OpenJDK)
145 MB

Traditional
Cloud-Native Stack
209MB

* Tested on a single-core machine

Boot + First Response Time

REST

Quarkus + Native
(via GraalVM)
0.016 Seconds

Quarkus + JIT
(via OpenJDK)
0.943 Seconds

Traditional
Cloud-Native Stack
4.3 Seconds

① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨ ⑩

REST + CRUD

Quarkus + Native
(via GraalVM)
0.042 Seconds

Quarkus + JIT
(via OpenJDK)
2.033 Seconds

Traditional
Cloud-Native Stack
9.5 Seconds
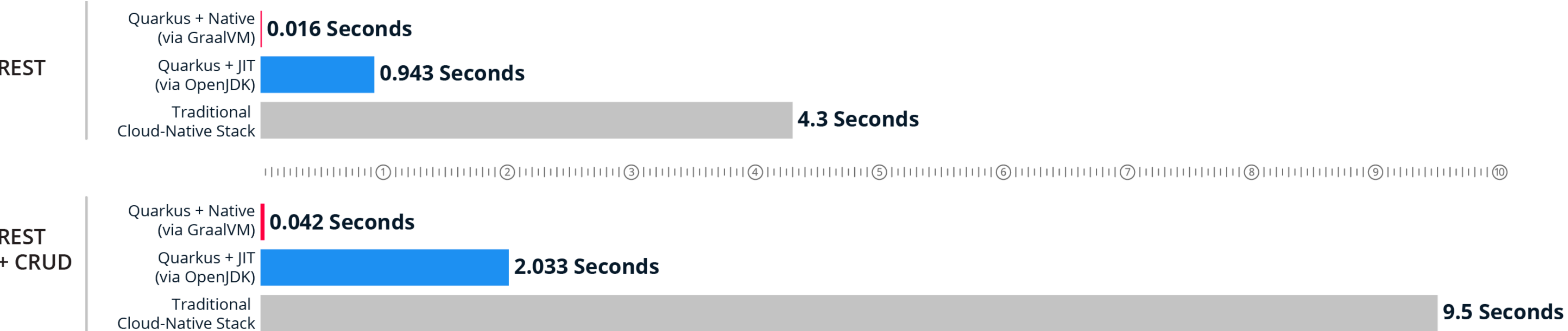
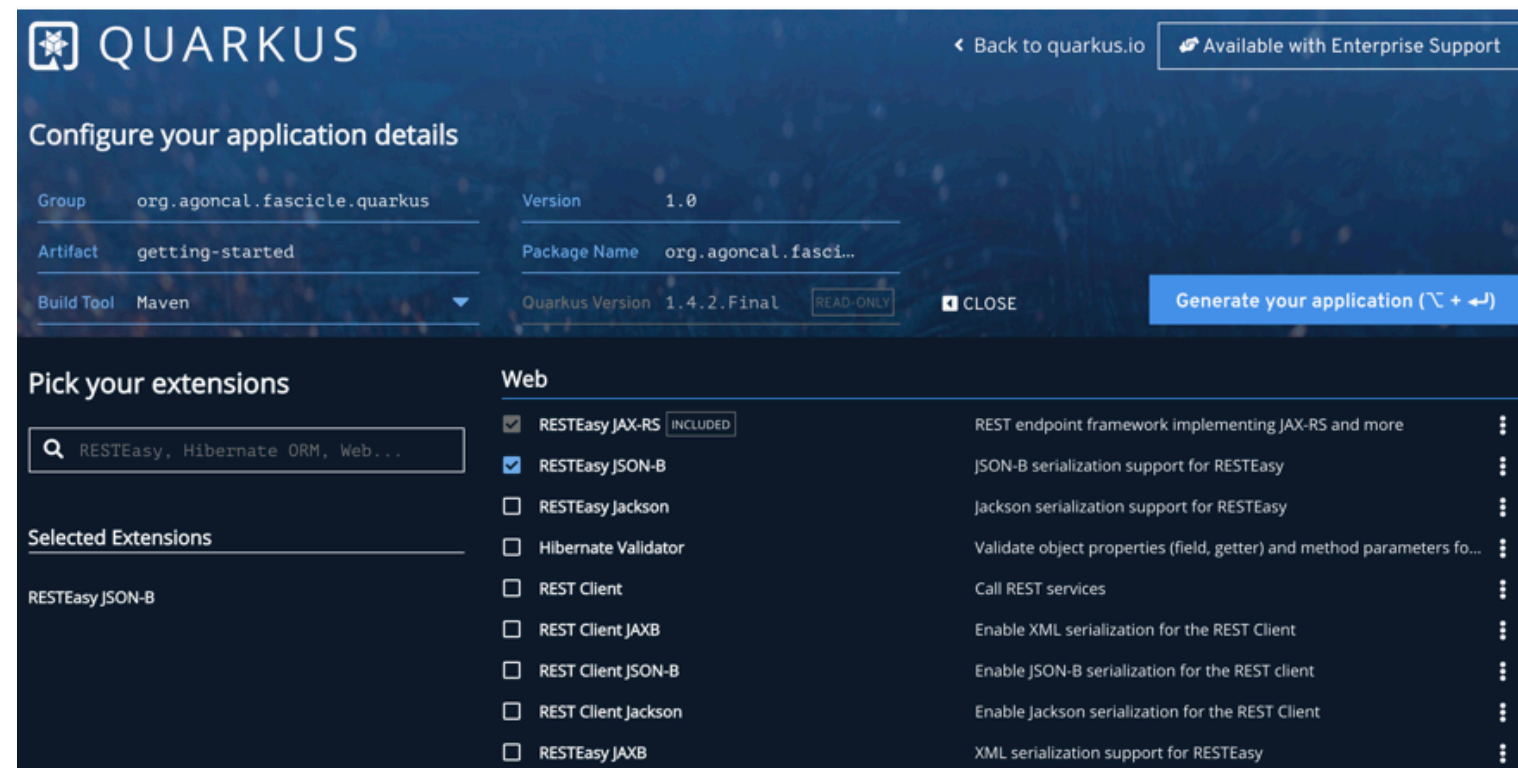* Tested on a single-core machine

GraalVM

GraalVM is a universal virtual machine for
running applications written in
JavaScript, Python, Ruby, R, JVM-based
languages like Java, Scala, Groovy,
Kotlin, Clojure, and LLVM-based languages
such as C and C++
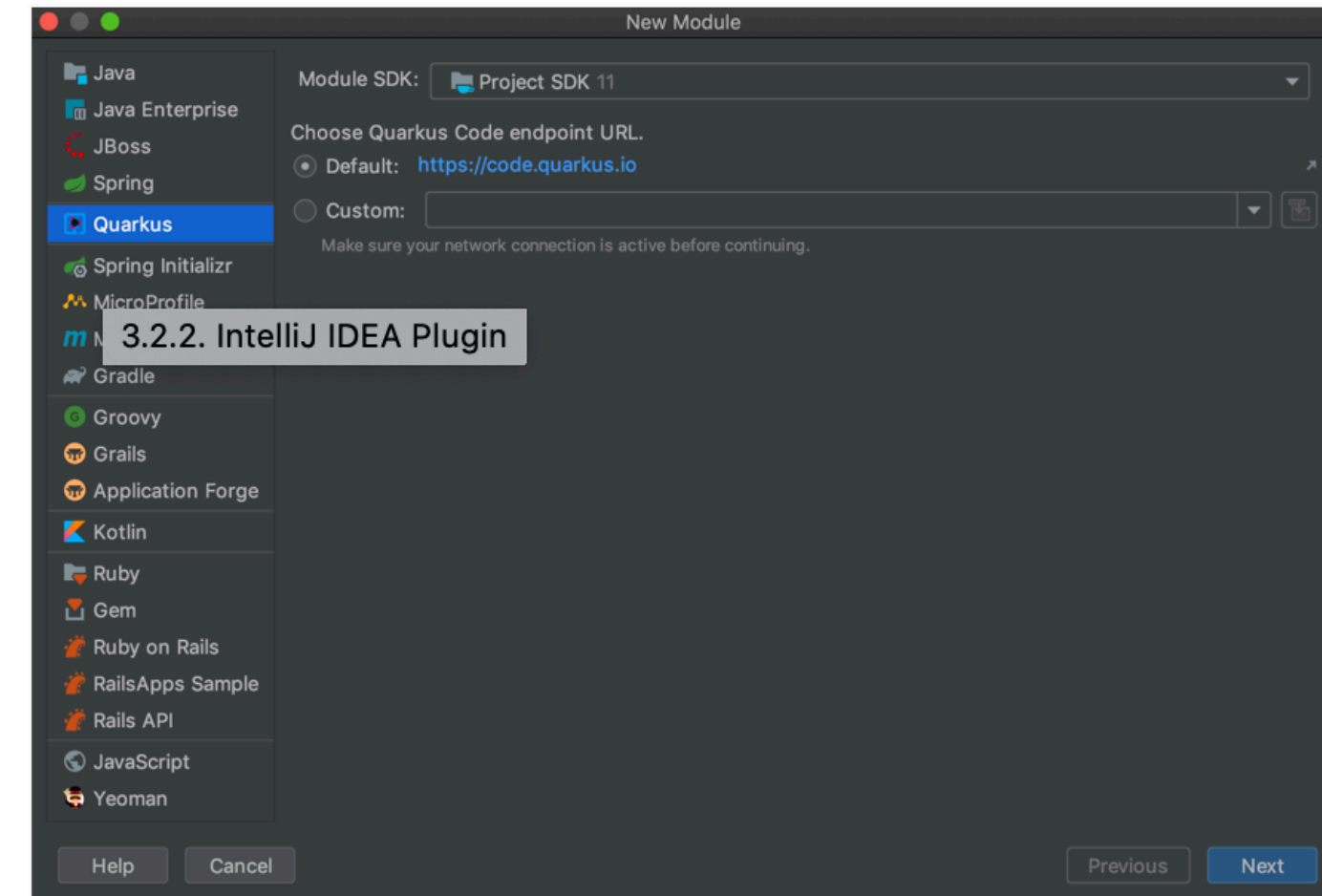
Demo

•Web Interface

https://code.quarkus.io

•IntelliJ IDEA Plugin

https://code.quarkus.io
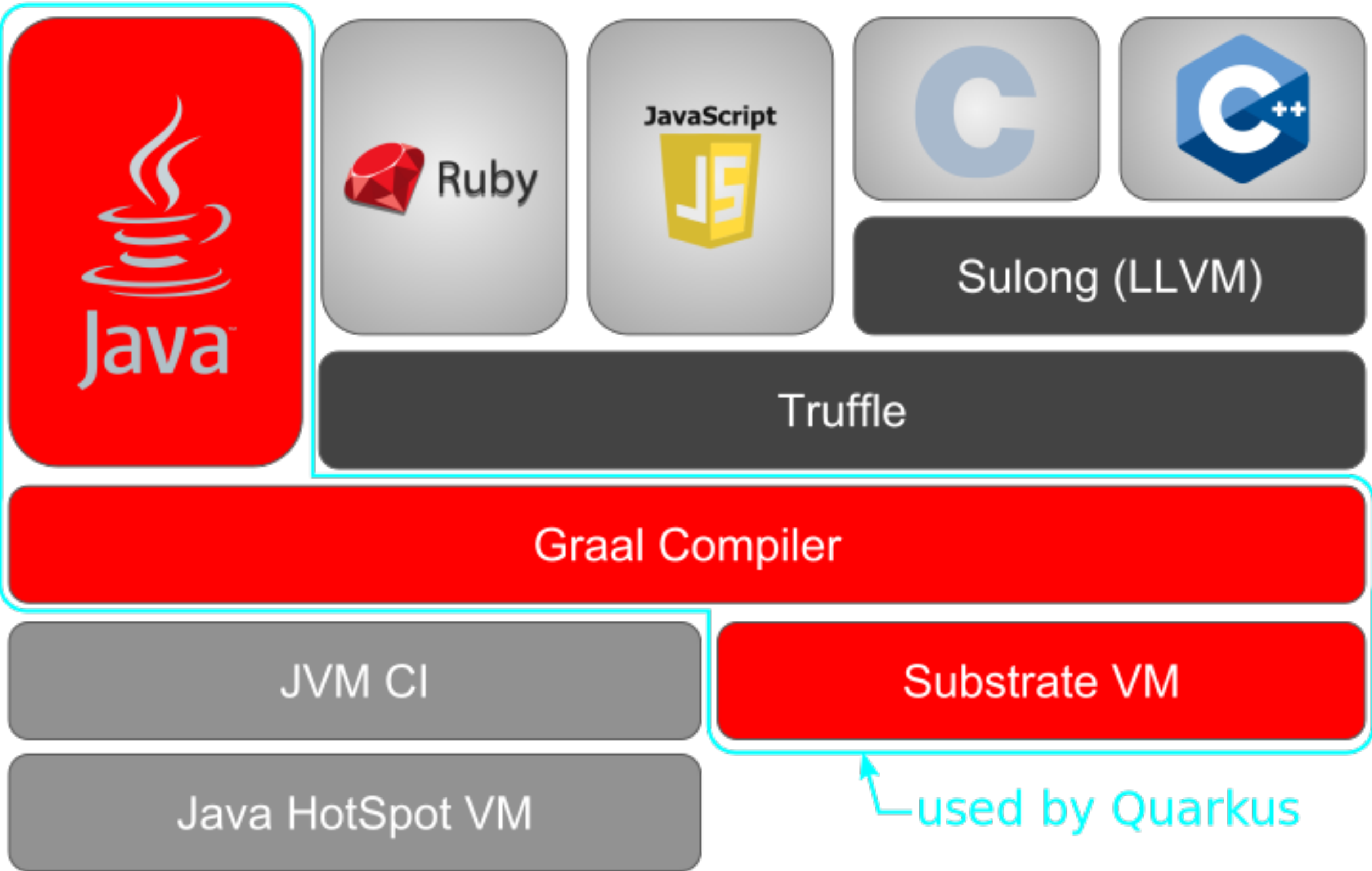
•Maven Plugin

```
mvn "io.quarkus:quarkus-maven-plugin:1.10.5.Final:create" \
   -DprojectGroupId="com.drb.developers" \
   -DprojectArtifactId= "quarkus-app " \
   -DprojectVersion="1.0-SNAPSHOT" \
   -DclassName="HelloResource" \
   -Dpath="hello"
cd quarkus-app
```

Any Questions???