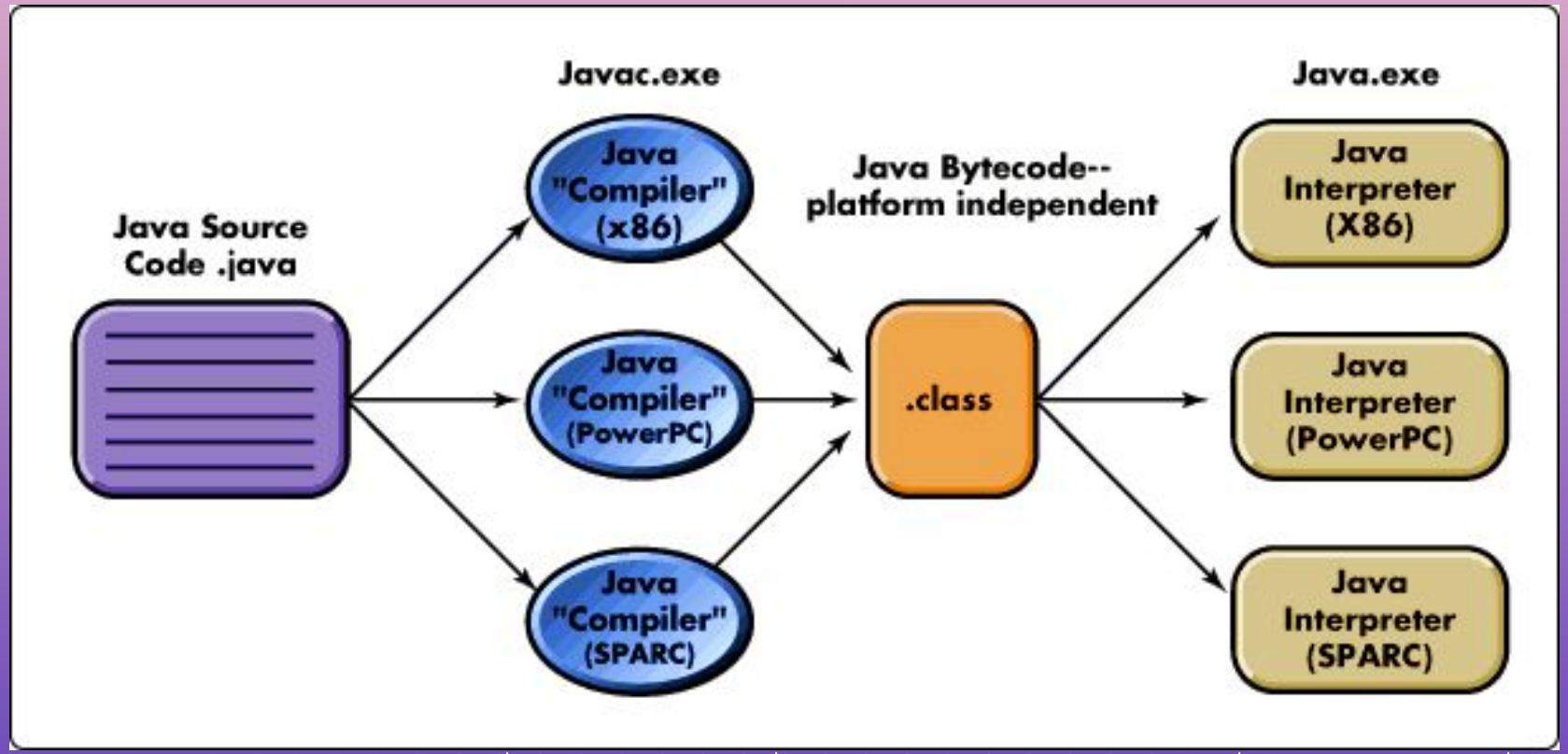
JAVA IS DEAD
LONG LIVE JAVA

QUARKUS

kubernetes

A Brief History of Java

- **James Gosling, Mike Sheridan, and Patrick Naughton** initiated the Java language project in June 1991. The small team of sun engineers called Green Team.
- Originally designed for small, embedded systems in electronic appliances like set-top boxes.
- Firstly, it was called “Greentalk” by James Gosling, and file extension was .gt.
- After that, it was called Oak and was developed as a part of the Green project.
- Renamed to Java after a trademark name conflict with Oak Technologies.


Javac.exe
Java.exe
Java Source Code .java
Java "Compiler" (x86)
Java "Compiler" (PowerPC)
Java "Compiler" (SPARC)
Java Bytecode-- platform independent
.class
Java Interpreter (X86)
Java Interpreter (PowerPC)
Java Interpreter (SPARC)
Written in C
Written in C

Java and the Web

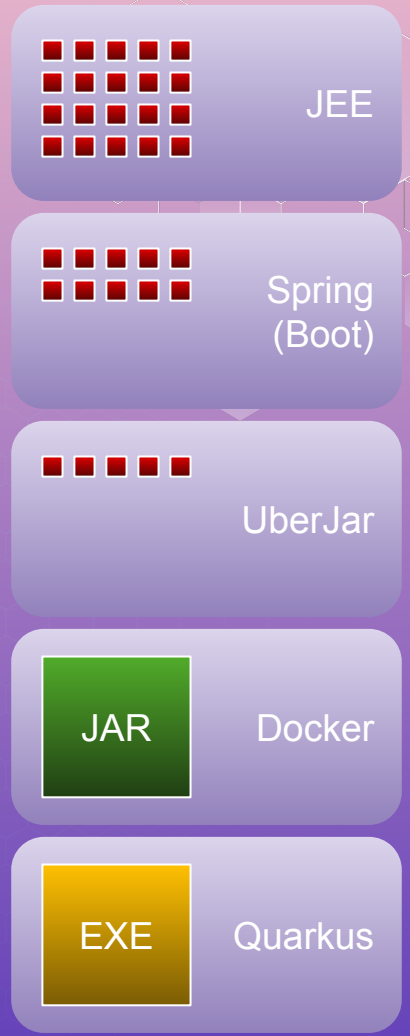
- Before 1995 web pages were just static HTML
- In 1995 Sun launched HotJava a pure Java Web Browser
- HotJava enabled animation and client side events
- Also in 1995 JavaScript was invented by Brendan Eich at Netscape
- The ‘browser wars’ saw JavaScript sidelined for a decade
- Google killed Java by releasing V8 of the Chrome JavaScript engine which used JIT compiling to native code

Java and the Enterprise

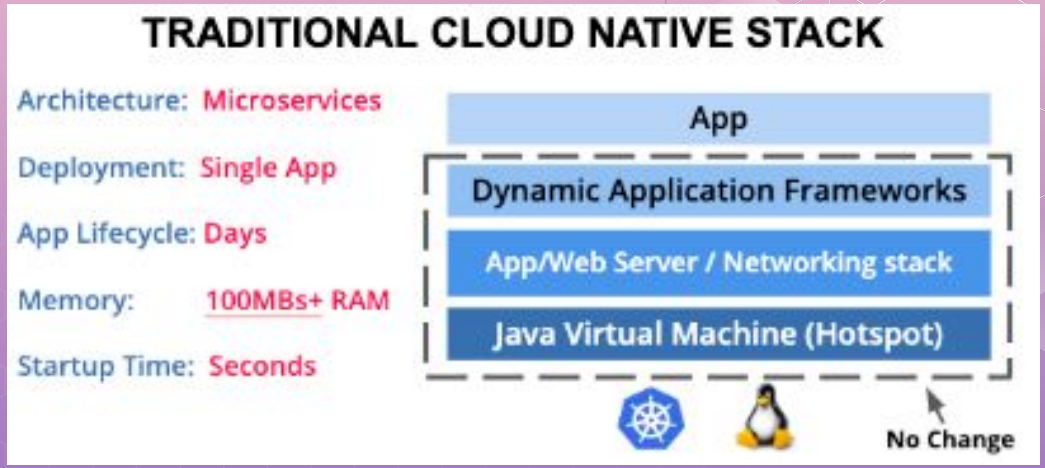
- Sun Microsystems offered JRE (SE) free but charged for JDK
- JEE is a ***specification*** set of API extensions to JSE
- JEE v1.2 had just 10 specifications: Servlets, EJB, JSF, JMS, RMI-IIOP, JDBC, JTA, JNDI, JavaMail, JavaBeans
- Spring released in 2003, lightweight container. Innovations in ORM, XML and Dependency Injection
- Many JVM languages; Clojure, Groovy, Kotlin, Scala
- Many programming paradigms; OOP, Functional, DSL

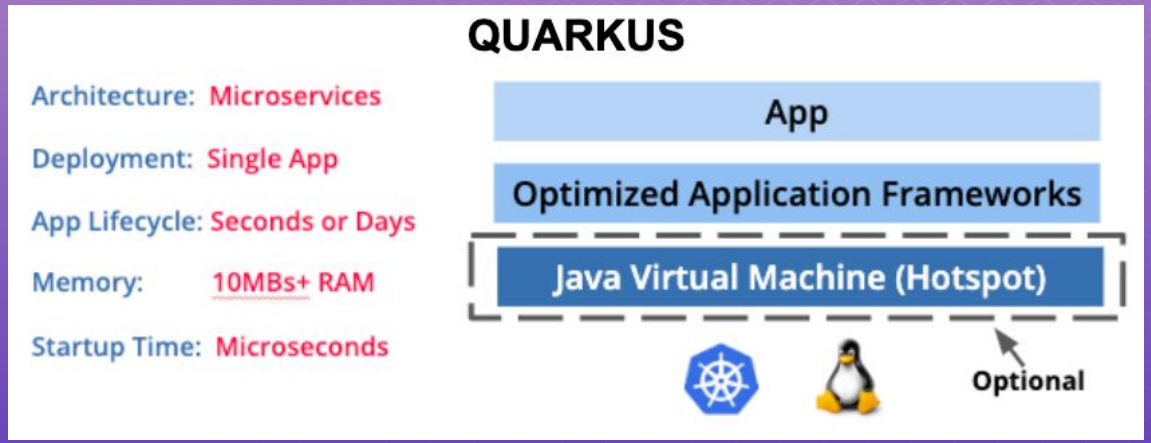
Monoliths to Microservices

- JEE was rooted in the monolithic Application Server
- Rise of consumer transactions on the web required Highly Available and Distributed Architectures
- JEE clusters, Distributed Cache, OSGi
- Docker, Kubernetes, OpenShift and Cloud Foundry
- And now….. Quarkus!



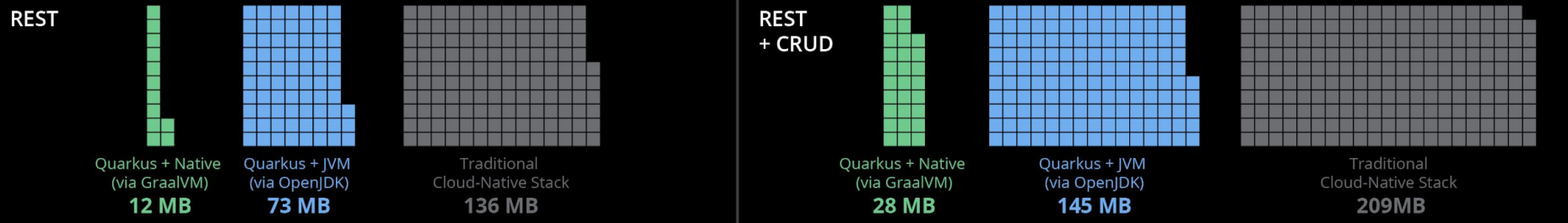
Quarkus


TRADITIONAL CLOUD NATIVE STACK
Architecture: Microservices
Deployment: Single App
App Lifecycle: Days
Memory: 100MBs+ RAM
Startup Time: Seconds
App
Dynamic Application Frameworks
App/Web Server / Networking stack
Java Virtual Machine (Hotspot)
No Change

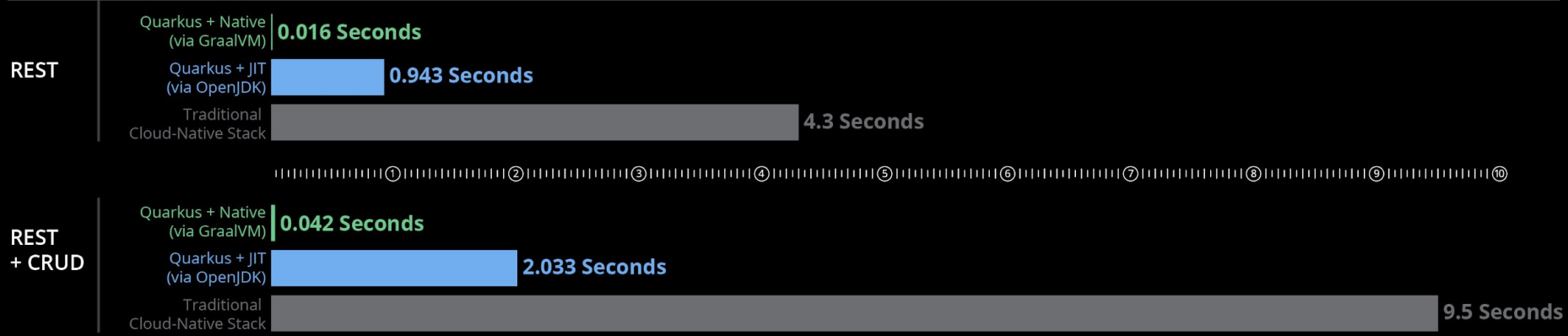

QUARKUS
Architecture: Microservices
Deployment: Single App
App Lifecycle: Seconds or Days
Memory: 10MBs+ RAM
Startup Time: Microseconds
App
Optimized Application Frameworks
Java Virtual Machine (Hotspot)
Optional

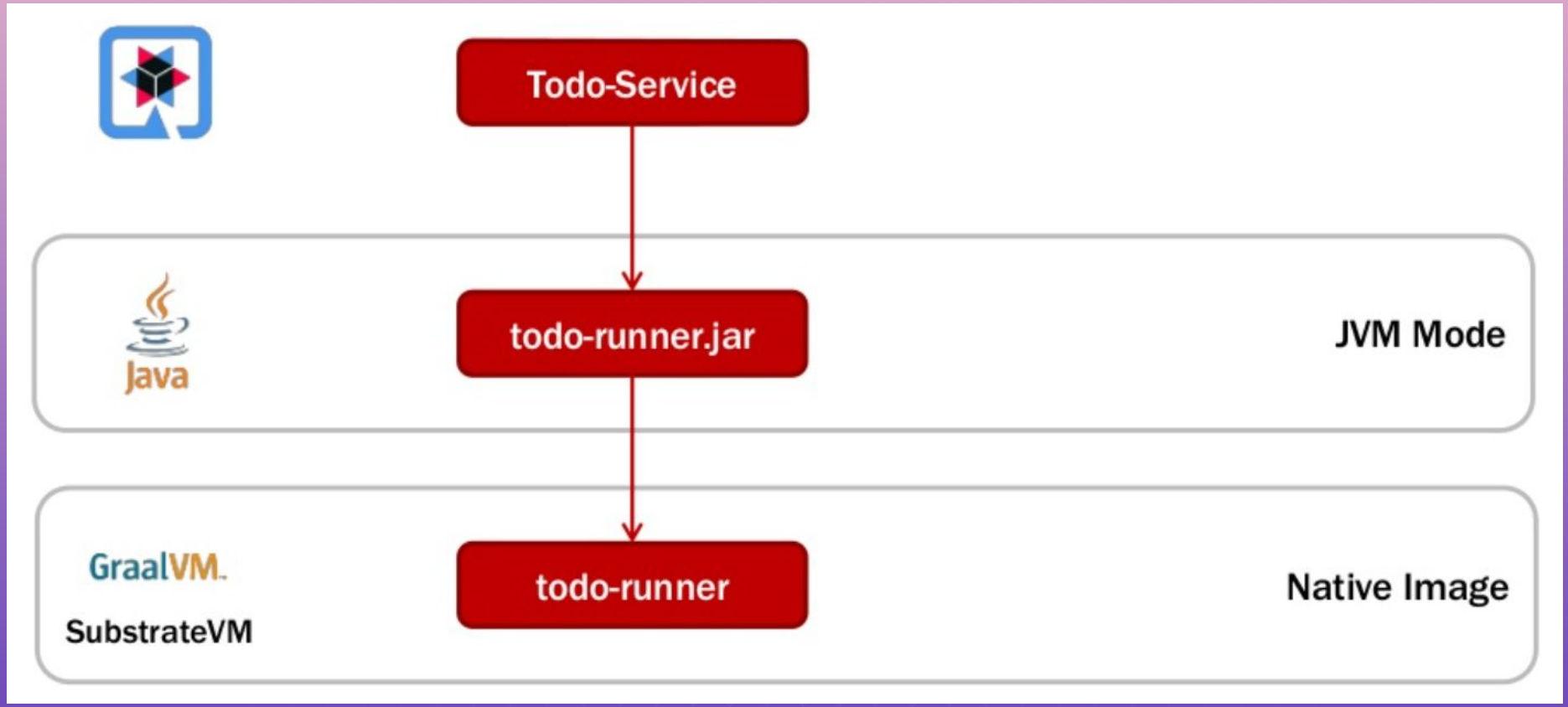
Memory (RSS) in Megabytes*

*Tested on a single-core machine

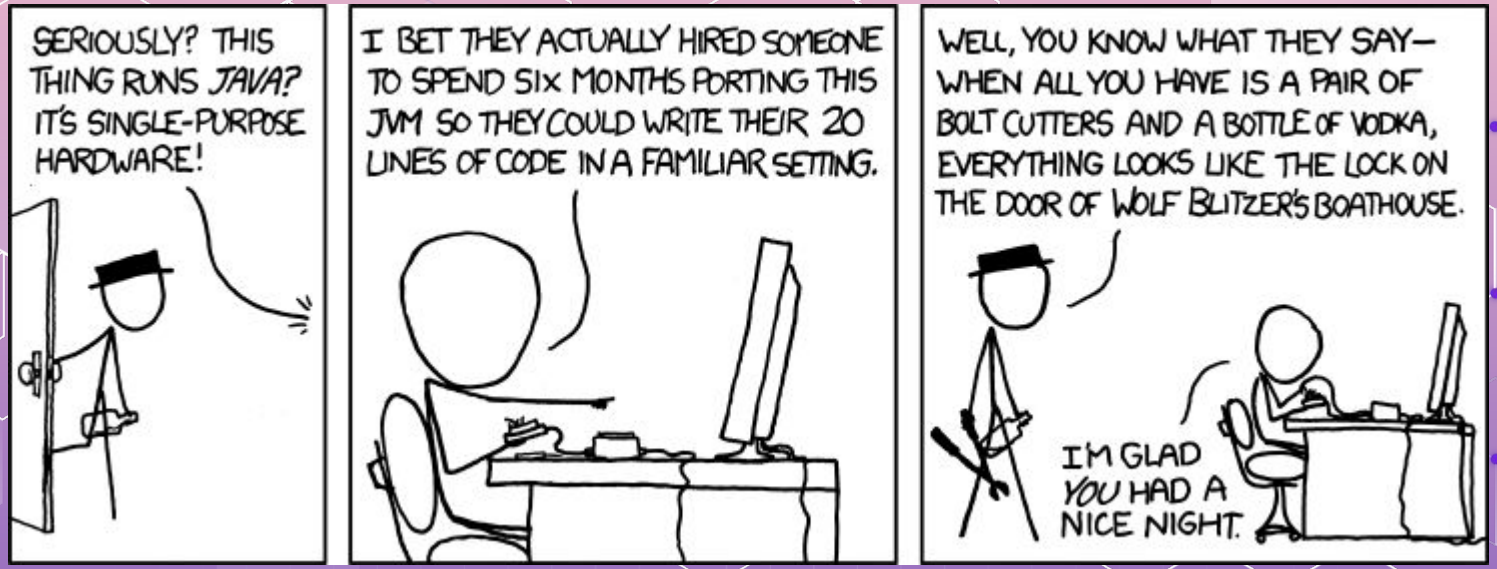


BOOT + First Response Time



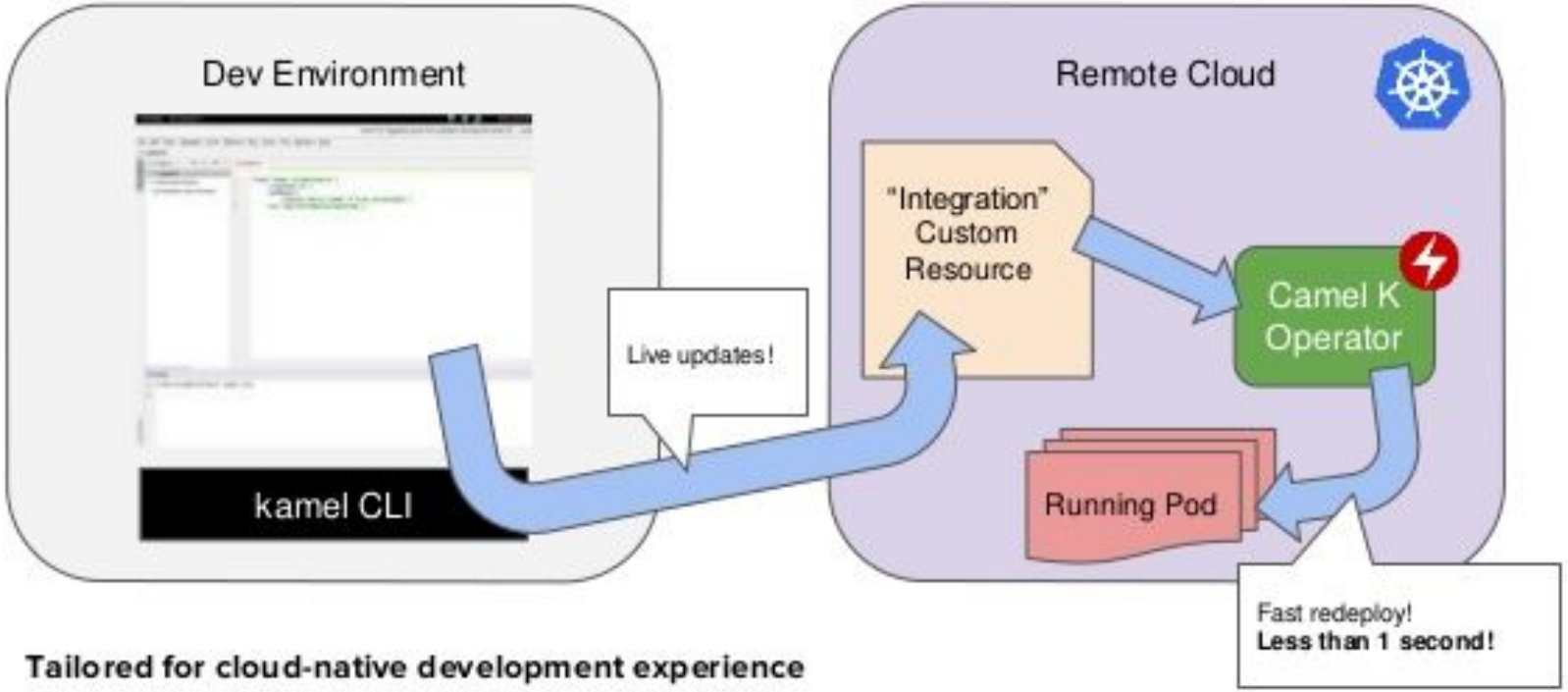

Todo-Service
Java
todo-runner.jar
JVM Mode
GraalVM
SubstrateVM
todo-runner
Native Image

DEMO


SERIOUSLY? THIS THING RUNS JAVA? IT'S SINGLE-PURPOSE HARDWARE!
I BET THEY ACTUALLY HIRED SOMEONE TO SPEND SIX MONTHS PORTING THIS JVM SO THEY COULD WRITE THEIR 20 LINES OF CODE IN A FAMILIAR SETTING.
WELL, YOU KNOW WHAT THEY SAY— WHEN ALL YOU HAVE IS A PAIR OF BOLT CUTTERS AND A BOTTLE OF VODKA, EVERYTHING LOOKS LIKE THE LOCK ON THE DOOR OF WOLF BLITZER'S BOATHOUSE.
I'M GLAD YOU HAD A NICE NIGHT.

THANKS

Architecture of Camel K



Tailored for cloud-native development experience