Quarkus: Fast, Small, Innovative, and Native?

QUARKUS

rafael.benevides@oracle.com

@rafabene

Link → http://bit.ly/quarkus

Rafael Benevides

Cloud-Native Developer Advocate

rafael.benevides@oracle.com

@rafabene

Java Certifications:
SCJA / SCJP / SCWCD / SCBCD / SCEA
JBoss Certifications:
JBCD / JBCAA
Red Hat Certifications:
OpenShift / Containers / Ansible
Other Certifications:
SAP Netweaver / ITIL / IBM Software Quality

Break New Ground

https://developer.oracle.com

Break New Ground

https://cloudnative.oracle.com

ORACLE

Visite o nosso estande!

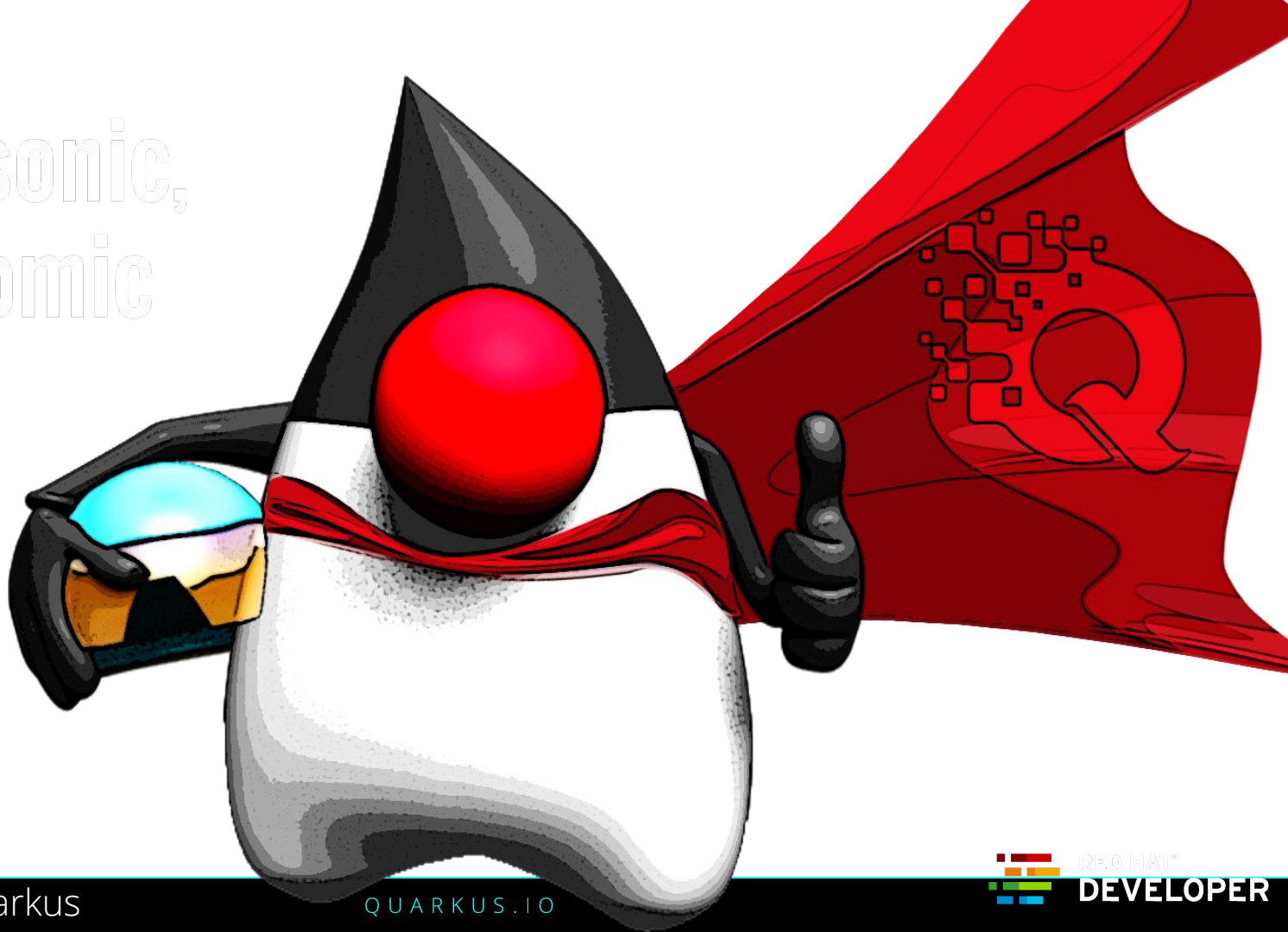
- Faça um trial e ganhe um brinde
- Faça um hands on e ganhe outro brinde

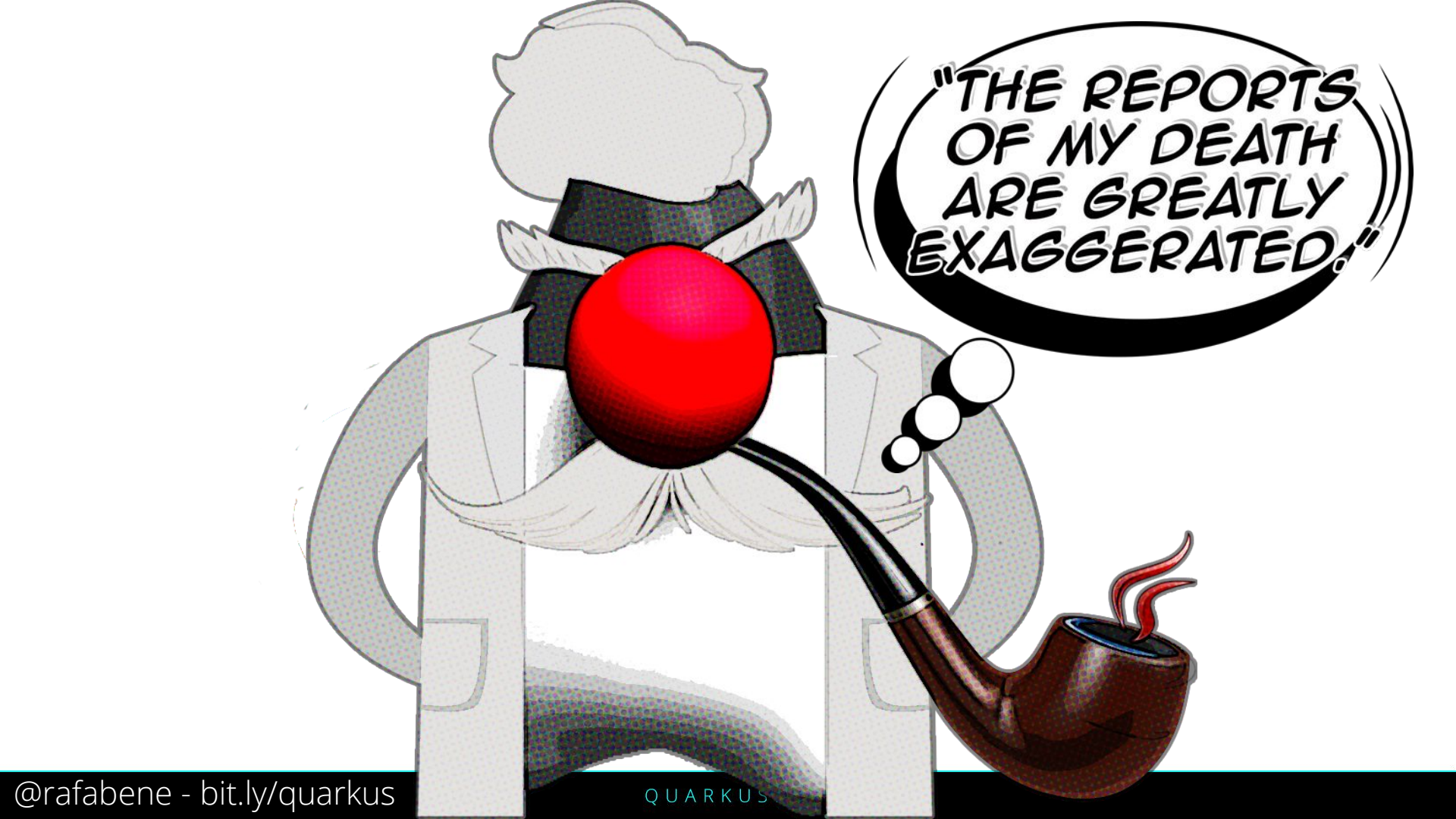
MicroProfile

microprofile.io

Supersonic, Subatomic Java

QUARKUS.IO




"THE REPORTS OF MY DEATH ARE GREATLY EXAGGERATED."

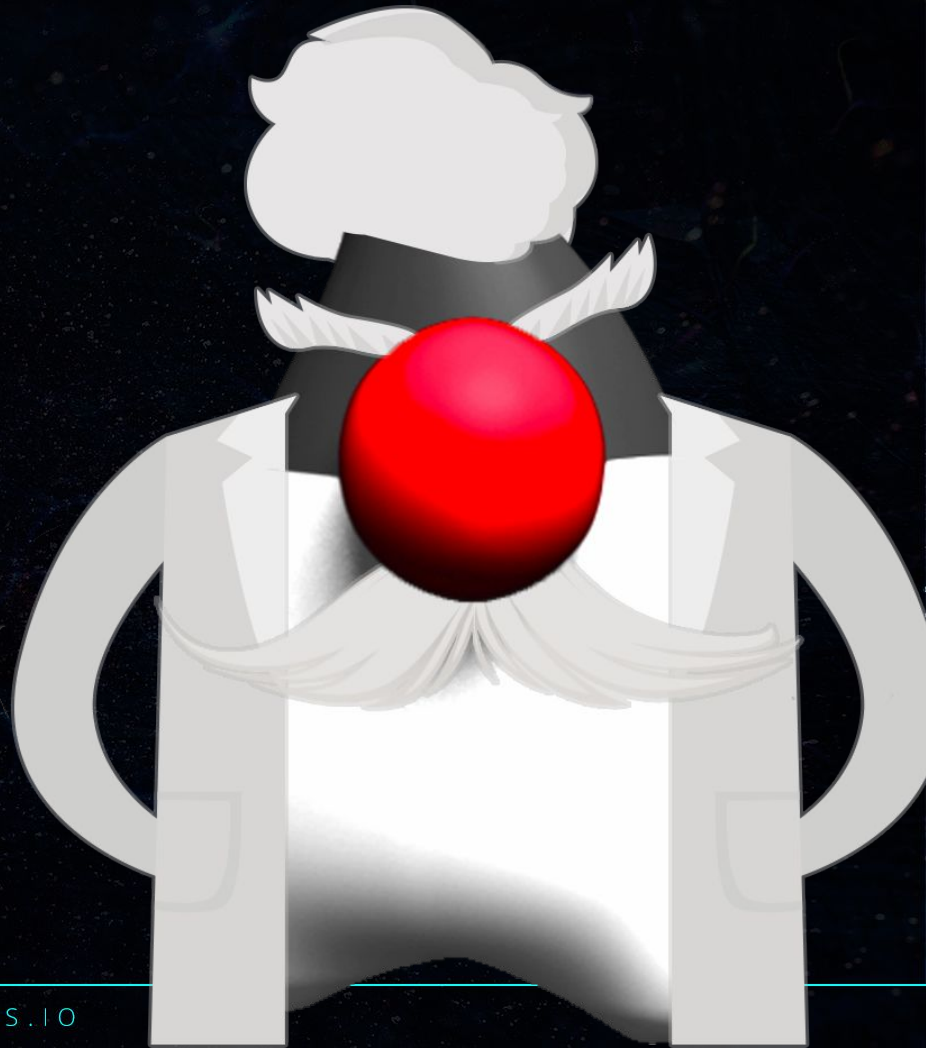
"Today, Java is old hat."

Computer World, May 22, 2006

FRANK HAYES ■ FRANKLY SPEAKING

Not Dead Yet

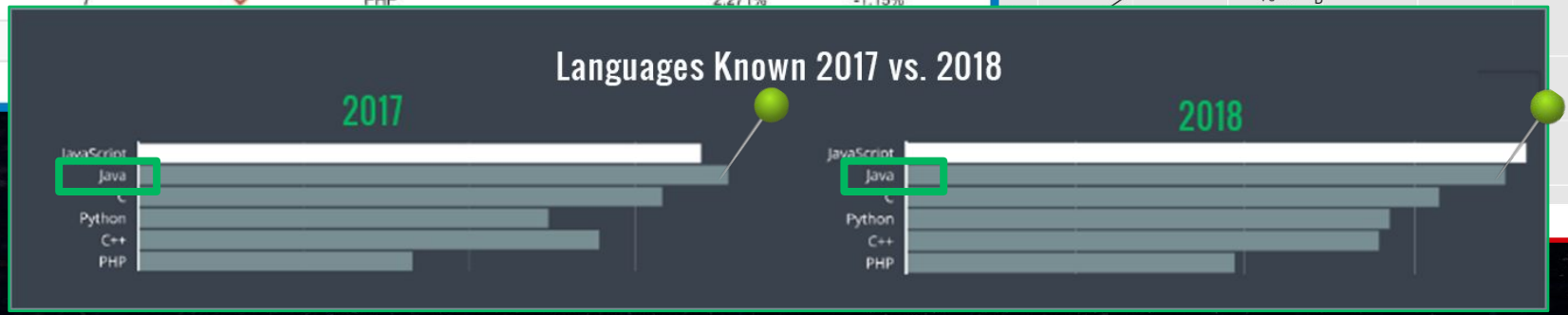
IS JAVA DEAD? Come on, seriously — why else would Sun Microsystems be offering it up to the open-source crowd? (See story, page 1.) A decade ago, Java was the hottest, most exciting thing in IT, a certified Windows-killer that was going to wipe out Microsoft's monopoly and revolutionize the way software was made, distributed and run. Today? Today, Java is old hat. It's been eclipsed by open-source, the *new* hottest thing in IT that's going to wipe out Microsoft's monopoly and revolutionize the way software is made, distributed and run.



TIOBE

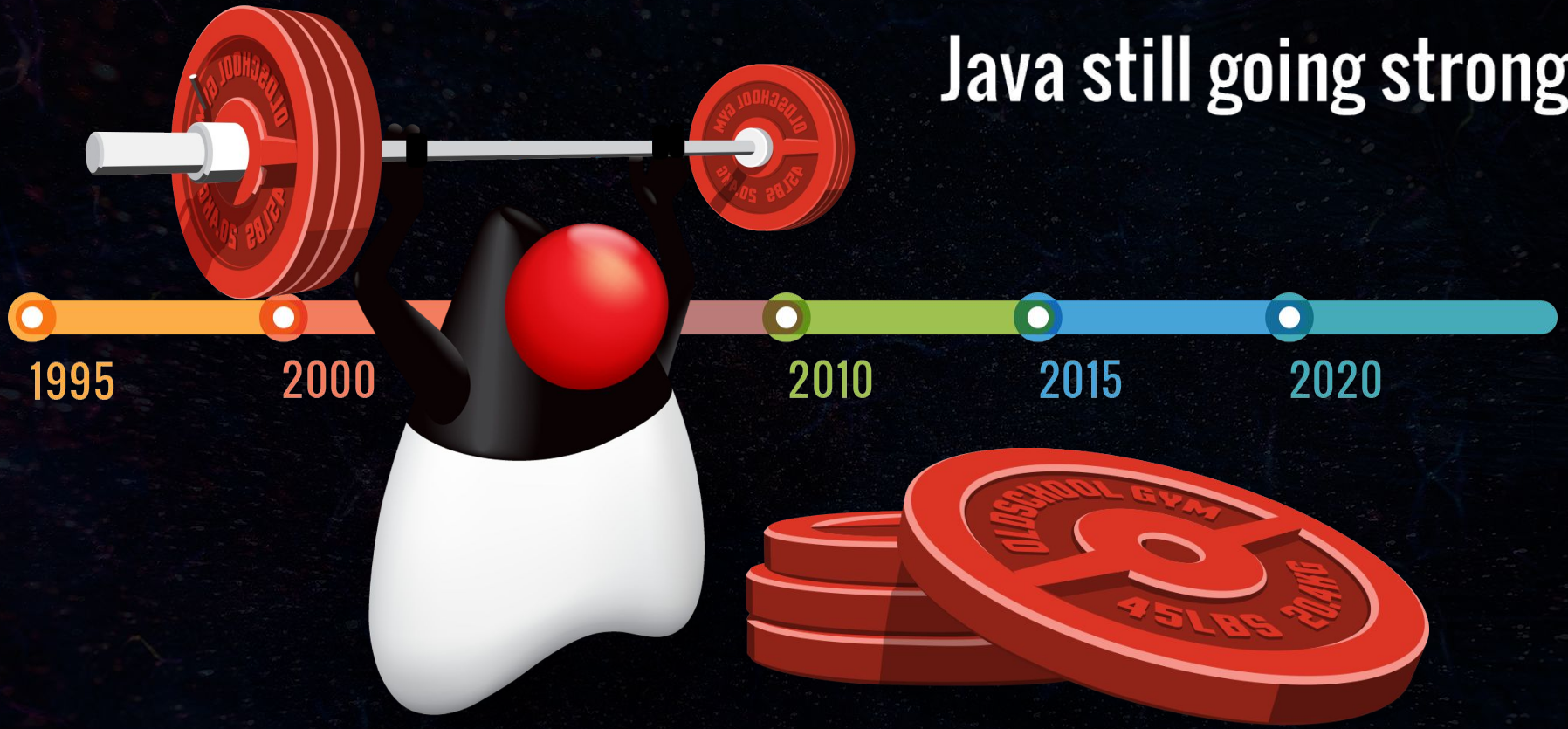
Feb 2019	Feb 2018	Change	Programming Language	Ratings	Change
1	1		Java	15.876%	+0.89%
2	2		C	12.424%	+0.57%
3	4	^	Python	7.574%	+2.41%
4	3	v	C++	7.444%	+1.72%
5	6	^	Visual Basic .NET	7.095%	+3.02%
6	8	^	JavaScript	2.848%	-0.32%
7	5	v	C#	2.846%	-1.61%
8	7	v	PHP	2.271%	-1.15%
9					
10					

RedMonk

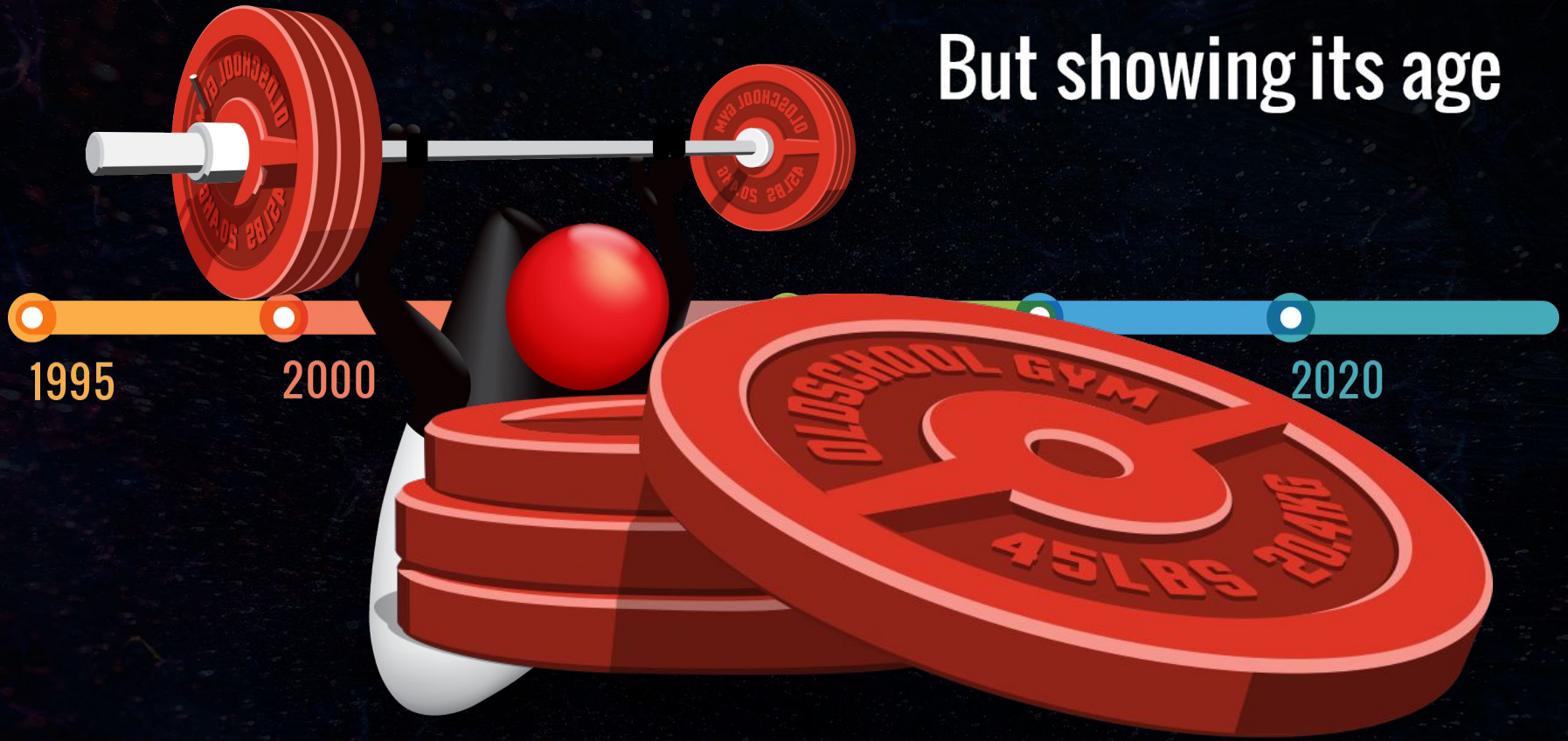


HackerRank

Java still going strong

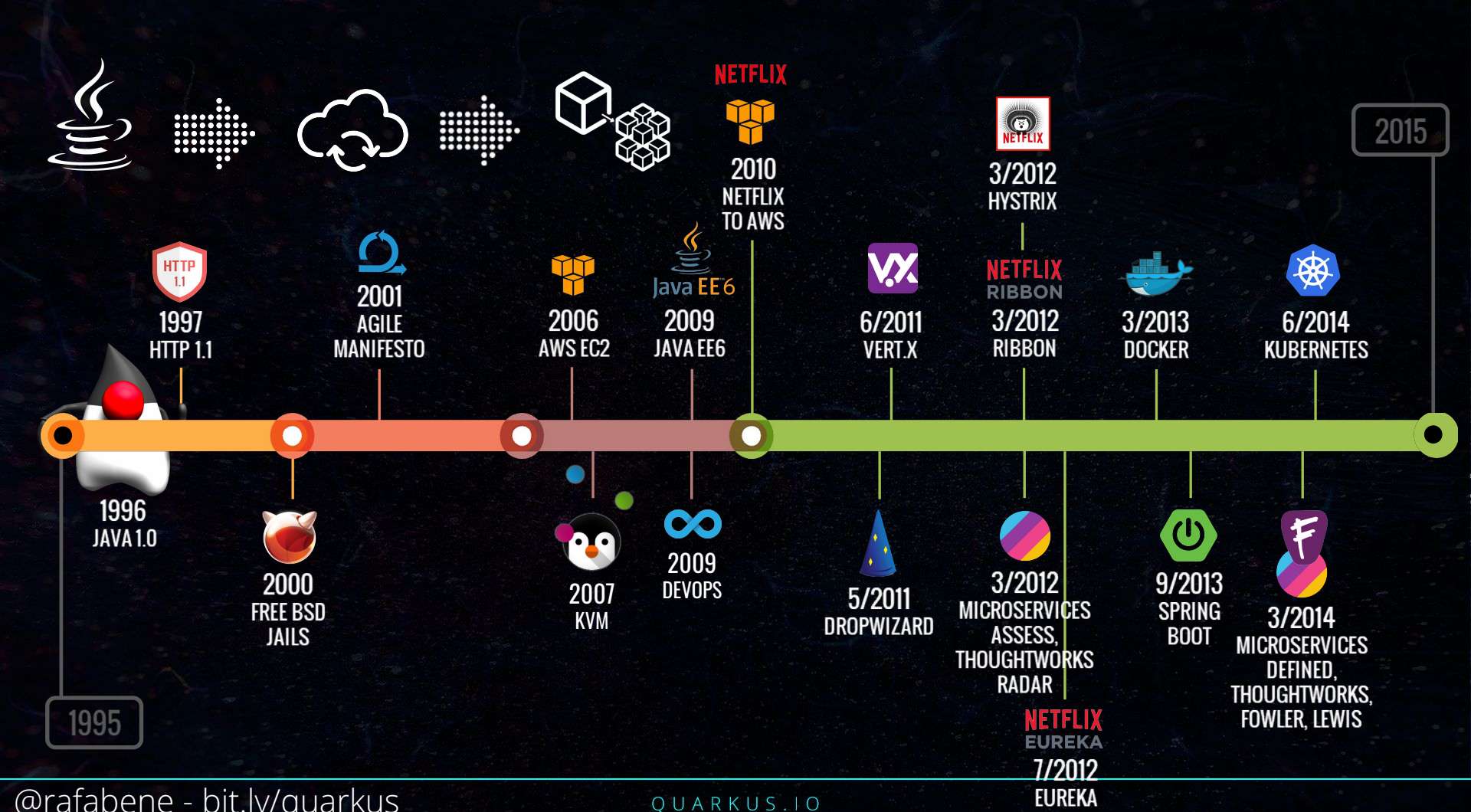


But showing its age




Hey, is it getting a
little tight in here?


BOOM!


NETFLIX
2010
NETFLIX
TO AWS
3/2012
HYSTRIX
2015
1997
HTTP 1.1
2001
AGILE
MANIFESTO
2006
AWS EC2
2009
JAVA EE6
6/2011
VERT.X
NETFLIX
RIBBON
3/2012
RIBBON
3/2013
DOCKER
6/2014
KUBERNETES
1996
JAVA 1.0
2000
FREE BSD
JAILS
2007
KVM
2009
DEVOPS
5/2011
DROPWIZARD
3/2012
MICROSERVICES
ASSESS,
THOUGHTWORKS
RADAR
9/2013
SPRING
BOOT
3/2014
MICROSERVICES
DEFINED,
THOUGHTWORKS,
FOWLER, LEWIS
NETFLIX
EUREKA
7/2012
EUREKA
1995

Cost of a Java-based Web App circa 1999

```
     $18,000  Sun Sparc App Server Box (4 CPUs, 2GB of RAM)
+    $60,000  BEA Weblogic
+    $92,000  Sun Spark DB Server Box (8 CPUs)
+   $243,000  Oracle RDBMS
+    $50,000  Symantec Visual Café for 10 developers
-------------------------------------------------------
    $463,000 (capex) + ~$80,000 annual maint (opex)
```

m5ad.4xlarge	16	N/A	64 GiB	2 x 300 NVMe SSD	$0.824 per Hour
m5ad.12xlarge	48	N/A	192 GiB	2 x 900 NVMe SSD	$2.472 per Hour
m5ad.24xlarge	96	N/A	384 GiB	4 x 900 NVMe SSD	$4.944 per Hour
m5d.large	2	8	8 GiB	1 x 75 NVMe SSD	$0.113 per Hour
m5d.xlarge	4	16	16 GiB	1 x 150 NVMe SSD	$0.226 per Hour
m5d.2xlarge	8	31	32 GiB	1 x 300 NVMe SSD	$0.452 per Hour

MEMORY	VCPUS	SSD DISK	TRANSFER	PRICE
1 GB	1 vCPU	25 GB	1 TB	$5/mo $0.007/hr
2 GB	1 vCPU	50 GB	2 TB	$10/mo $0.015/hr
3 GB	1 vCPU	60 GB	3 TB	$15/mo $0.022/hr
2 GB	2 vCPUs	60 GB	3 TB	$15/mo $0.022/hr
1 GB	3 vCPUs	60 GB	3 TB	$15/mo $0.022/hr
4 GB	2 vCPUs	80 GB	4 TB	$20/mo $0.030/hr
8 GB	4 vCPUs	160 GB	5 TB	$40/mo $0.060/hr
16 GB	6 vCPUs	320 GB	6 TB	$80/mo $0.119/hr

INSTANCE	VCPU	RAM	TEMPORARY STORAGE	PAY AS YOU GO
D2 v3	2	8 GiB	50 GiB	$0.096/hour
D4 v3	4	16 GiB	100 GiB	$0.192/hour
D8 v3	8	32 GiB	200 GiB	$0.384/hour

IMHO you really should not be using Spring Boot for Lambdas ...
Spring is designed for long running services. Lambda's are literally functions that get called and then are gone. Different paradigm. Lambda's should be extremely tiny and simple too, if you want long running more complex microservices you're better off using ECS/EKS.

— nutrecht July 2018

The bigger problem with using spring (and java in general) is that lambda execution costs rise with memory usage. A lot of modern java frameworks assume that memory is abundant and cheap which isn't the case on lambda.

- davewritescode July 2018

The problem with java etc in a lambda is the start up time.

- morematttymattmatt July 2018

https://www.reddit.com/r/java/comments/8xv3ha/does_anybody_use_spring_boot_with_aws_lambda_do/

Quarkus

Supersonic java

WTF

is a

Quarkus ?

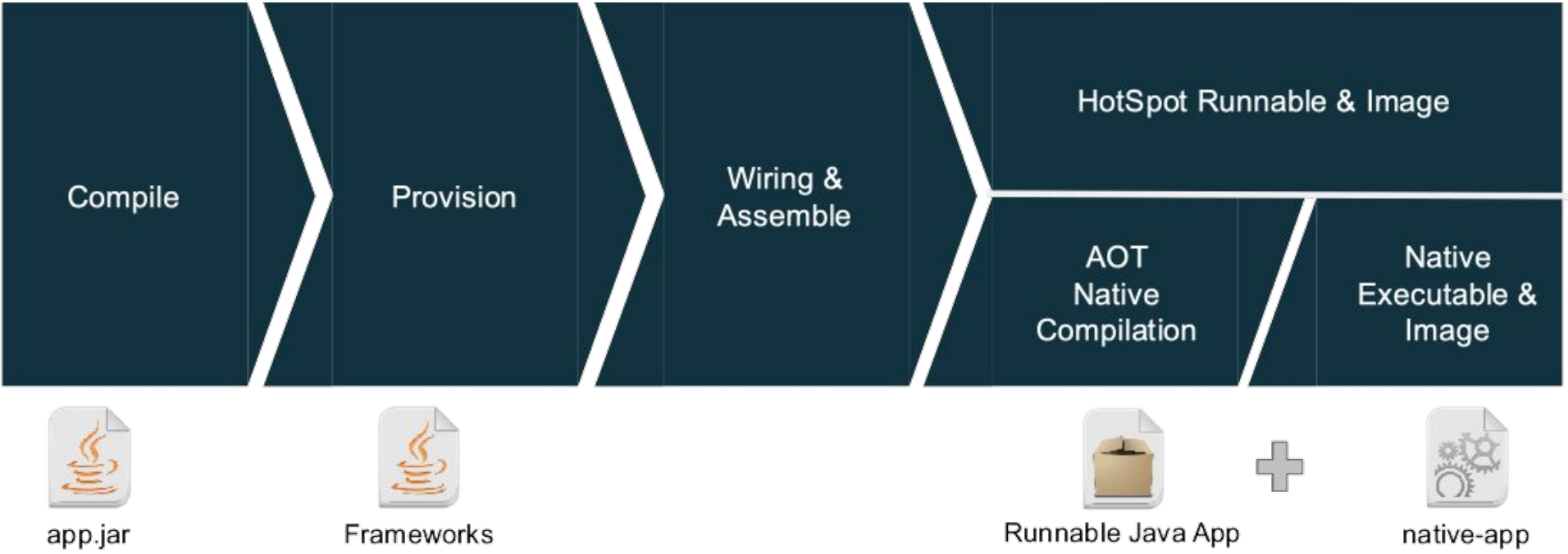
Quark any of a number of subatomic particles carrying a fractional electric charge

And "us"
(hardest problem in software)

Build a new Project

```
mvn io.quarkus:quarkus-maven-plugin:0.18.0:create
```

Quarkus Build process



Framework Optimizations

- Moved as much as possible to build phase
- Minimized runtime dependencies
- Maximize dead code elimination
- Introduced clear metadata contracts
- Spectrum of optimization levels
  (all -> some -> no runtime reflection)

Optimizations benefit both GraalVM (SVM) and HotSpot

Build a native binary

```
mvn clean package -Pnative
```

Polyglot, Native or JVM, Embeddable

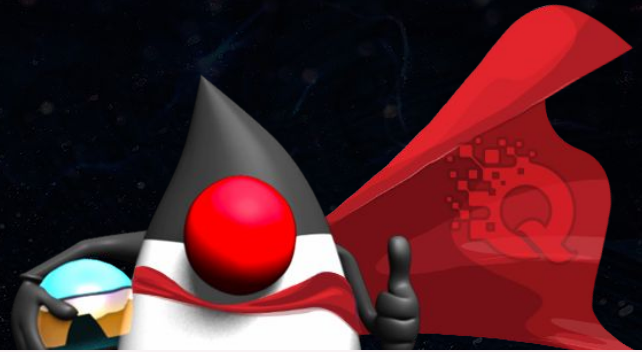

Java
Ruby
JavaScript
C
C++
Sulong (LLVM)
Truffle
Graal Compiler
JVM CI
Substrate VM
Java HotSpot VM

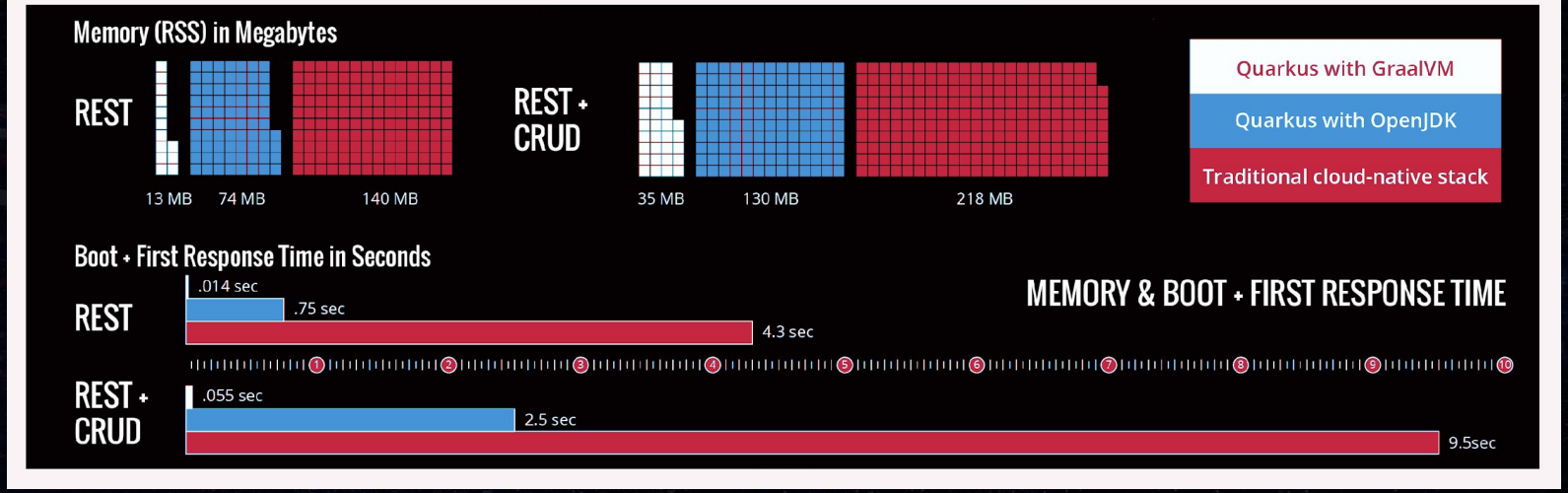
GraalVM Limitations

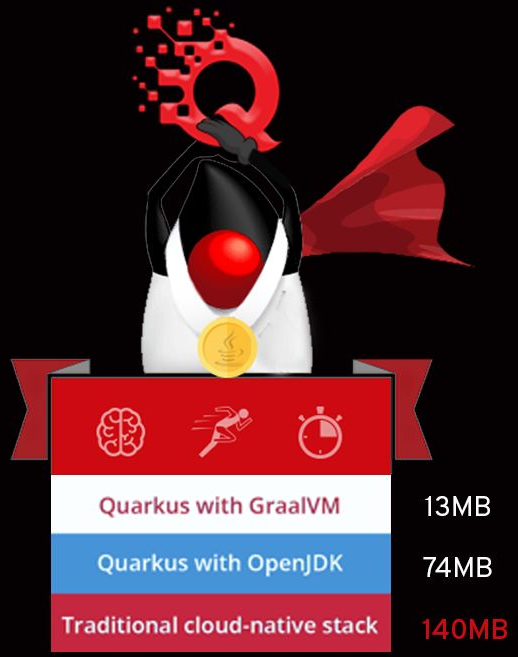
- Dynamic Class Loading
  - Deploy JARS, WARS at Runtime is impossible
- JVMTI, JMX + Other native VM interfaces
  - No Agents -> No JRebel, Byteman, profilers, tracers
- Others
  - Security Manager
  - finalize() (deprecated anyway)
  - InvokeDynamic and MethodHandler

GraalVM Limitations

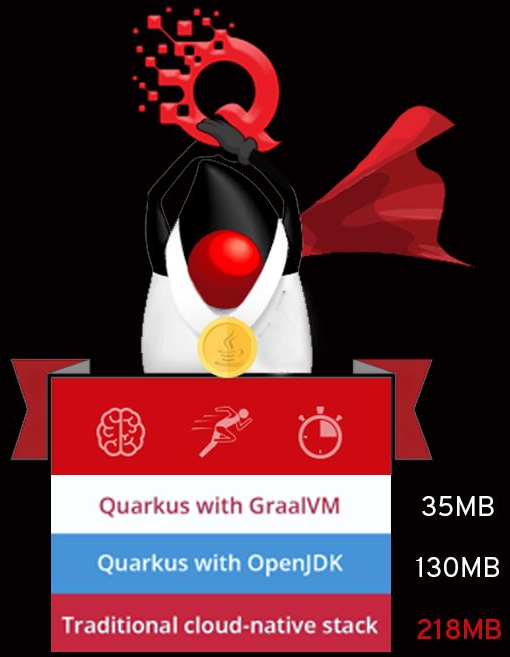
- Reflection (LIMITED)
  - Requires registration
  - -H:ReflectionConfigurationFiles=/path/to/reflectconfig



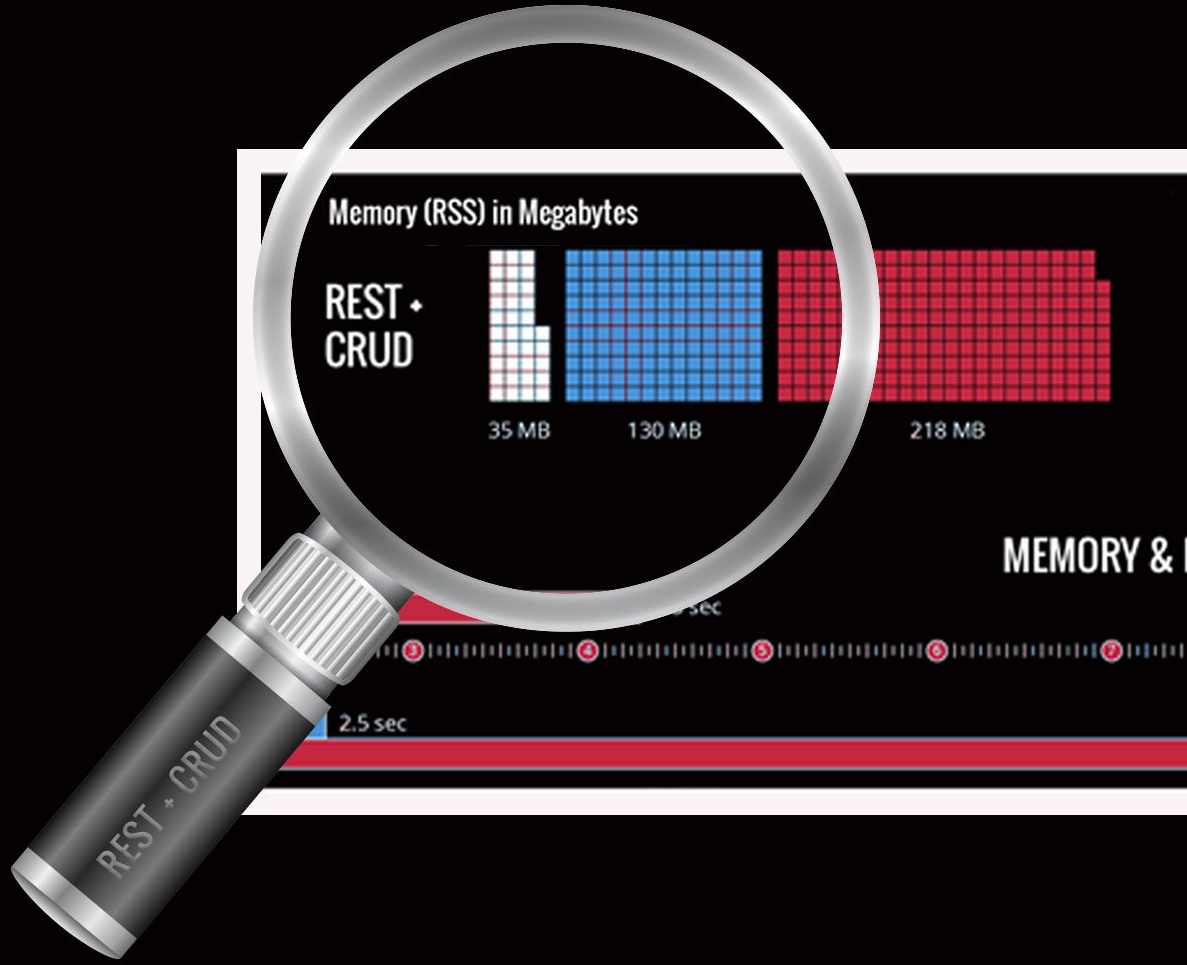

Memory (RSS) in Megabytes
REST
13 MB
74 MB
140 MB
REST + CRUD
35 MB
130 MB
218 MB
Quarkus with GraalVM
Quarkus with OpenJDK
Traditional cloud-native stack
Boot + First Response Time in Seconds
MEMORY & BOOT + FIRST RESPONSE TIME
REST
.014 sec
.75 sec
4.3 sec
REST + CRUD
.055 sec
2.5 sec
9.5sec

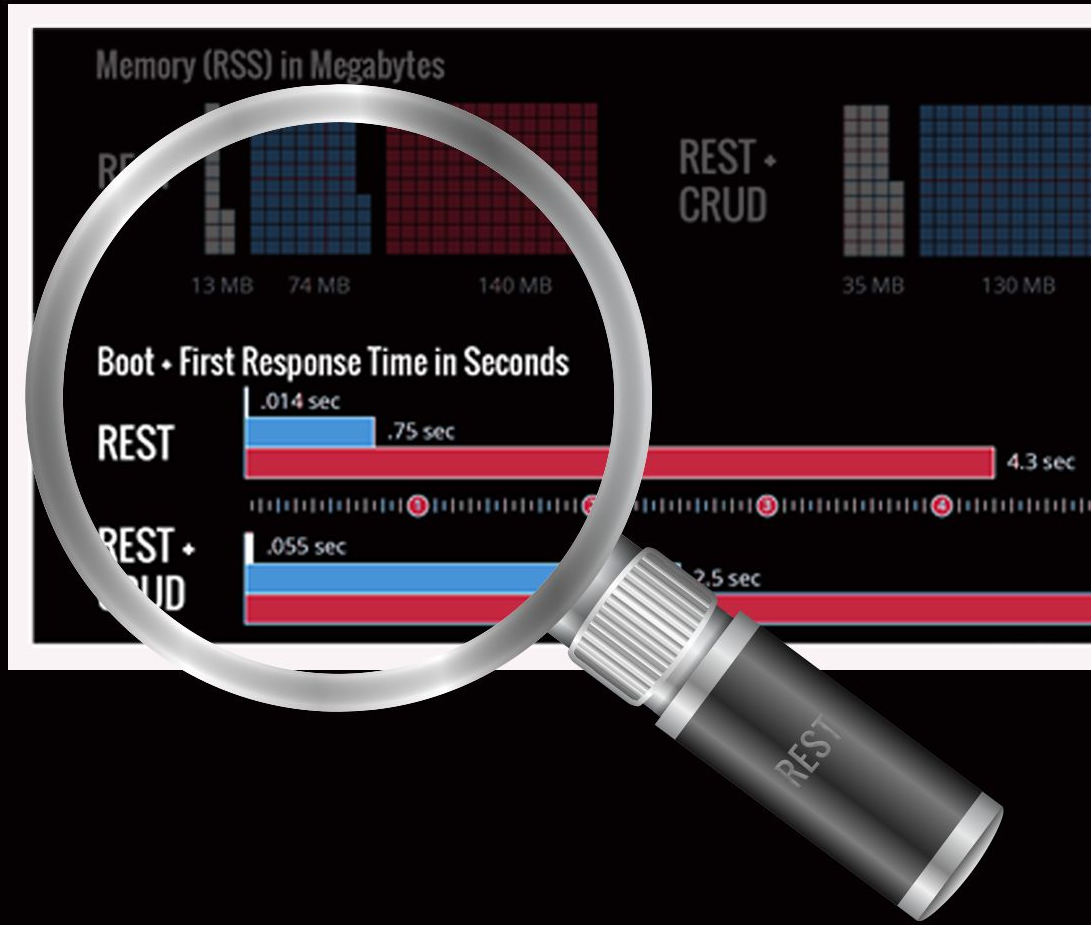


Memory (RSS) in Megabytes



Memory (RSS) in Megabytes

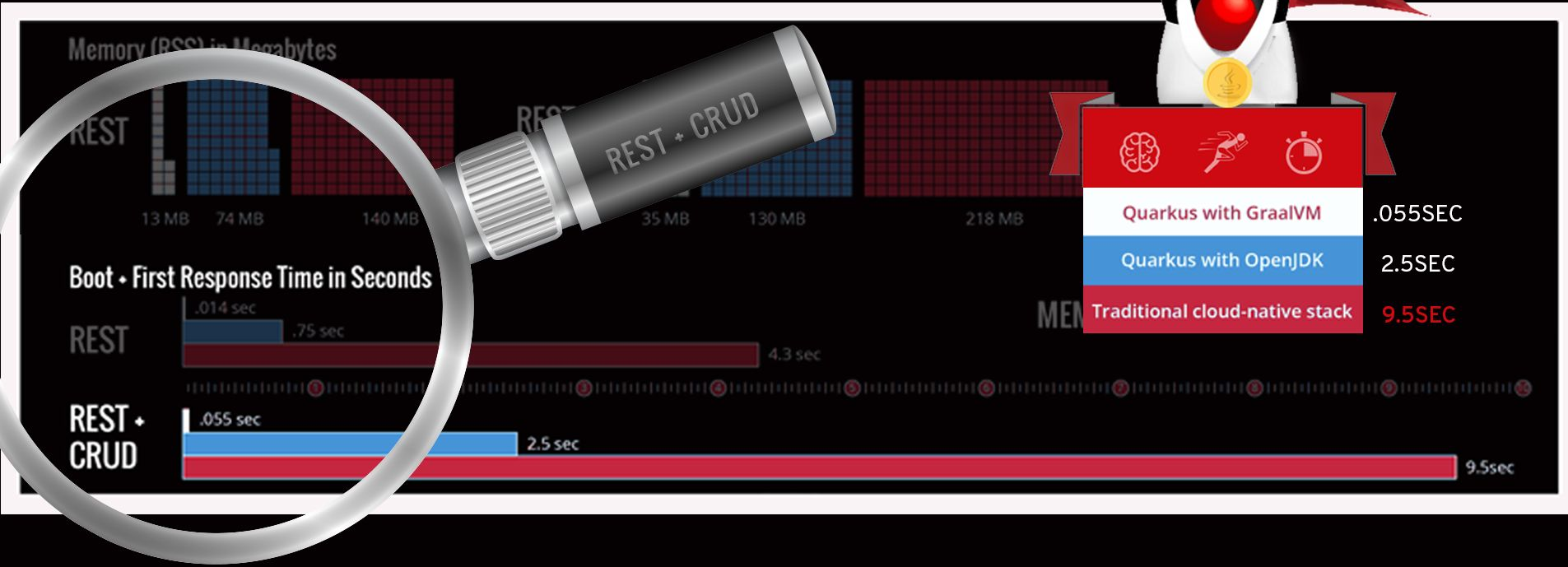




Boot + First Response Time in Seconds

★★★★★

Boot + First Response Time in Seconds



Build the Native Docker Image

```
mvn package -Pnative \
  -Dnative-image.docker-build=true
```

developer

Enough ^ love to go around.

Live reload

Imperative and reactive

Serverless

Microservices

Fat jars and native executables

Optimized for JAX-RS & JPA (Hibernate) ...


DEMO

Connect to Database

```
quarkus.datasource.driver=org.mariadb.jdbc.Driver
quarkus.hibernate-orm.dialect=org.hibernate.dialect.MariaDBDialect
quarkus.datasource.url=jdbc:mariadb://localhost:3306/quarkus
quarkus.datasource.username = rafael
quarkus.datasource.password = 123456
quarkus.hibernate-orm.log.sql=true
quarkus.hibernate-orm.database.generation=update
```

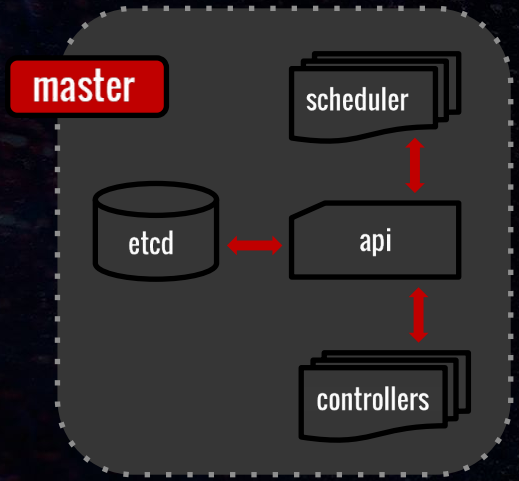
Build the Image

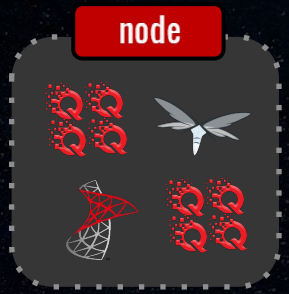
```
eval $(minikube docker-env)

docker build -f src/main/docker/Dockerfile.native -t myquarkus .
```

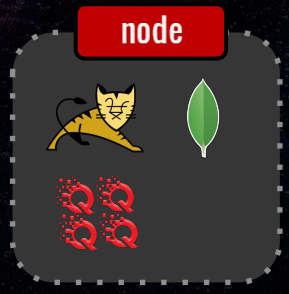
Instantiate the Image

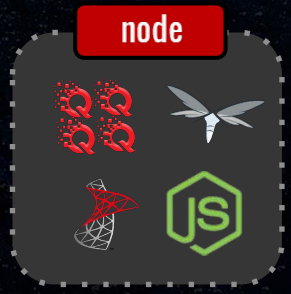
```
kubectl run myquarkus-app --image=myquarkus --port=8080
--image-pull-policy=IfNotPresent

kubectl expose deployment myquarkus-app --type=NodePort

curl $(minikube service myquarkus-app --url)/hello
```

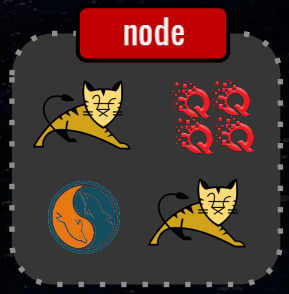
```
Image:
quay.io/images/custservice:1.1.0
Replicas:
20
Labels:
customerservice=prod,ci_build=1213
ConfigMap:
cust_config
```







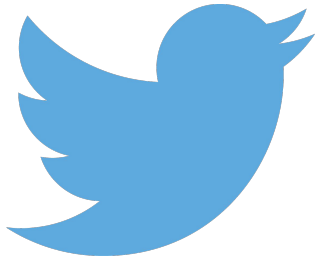




10x smaller.

Up to 100x faster.


It’s just the beginning ...
The End



@RAFABENE