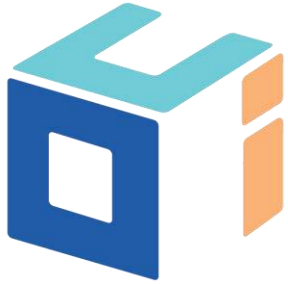


OCI | WE ARE SOFTWARE ENGINEERS.

WEBINAR

Introduction to Micronaut

Ultra-Lightweight Microservices for the JVM

Graeme Rocher | November 14, 2018

ABOUT THE PRESENTER: GRAEME ROCHER

- Creator: Grails (grails.org)
- Creator: Micronaut (micronaut.io)
- Author: *The Definitive Guide to Grails* (Apress, 2009)
- Senior Software Engineer at Object Computing, Inc. (objectcomputing.com)
- 2018 Oracle Groundbreaker Award Winner

AGENDA

- How We Got Here
- Microservice Challenges
- Microservice Framework Landscape
- Micronaut Demos

WE ARE
SOFTWARE
ENGINEERS.

HOW WE GOT HERE

THEN AND NOW

- Since 2008, a lot has changed
- 10 years is a long time in technology
- Everybody was building monoliths
- No Angular, no React, no Docker, no microservices

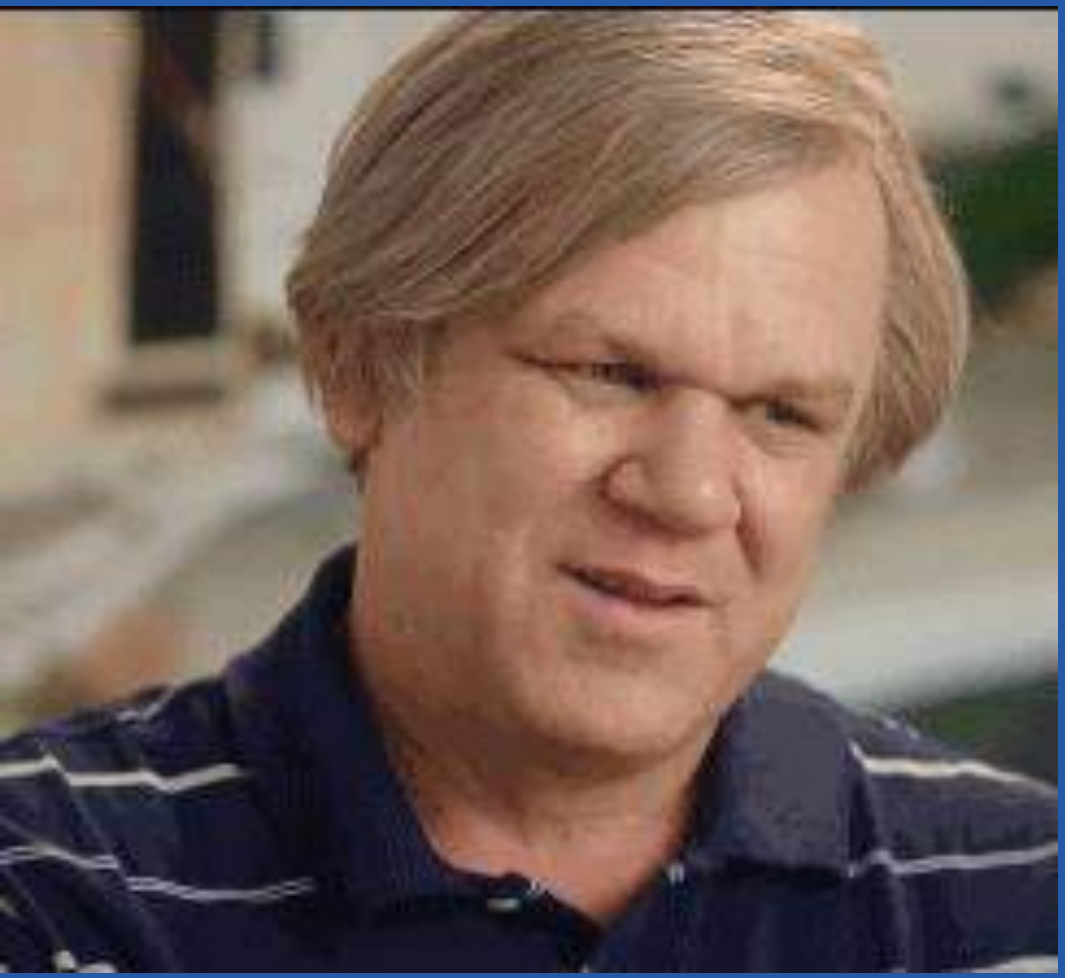
OCI | WE ARE SOFTWARE ENGINEERS.

SO WE TRY TO ADAPT

- Let’s try to adapt existing legacy technologies to microservices
- Technologies like Spring, Jakarta EE, and others were never optimized for low-memory footprint microservices

MICROSERVICE CHALLENGES

OCI | WE ARE SOFTWARE ENGINEERS.



WHAT TO DO, WHAT TO DO?

Shall we:

1. Try to convince people that something never designed for microservices is still okay?

or . . .

2. Go back to the drawing board?

THE GOAL

Create a new framework designed from the ground up for microservices and serverless computing

- Blazing-fast startup time
- Low-memory footprint
- As small as possible JAR sizes
- Zero dependency
- 12 Factor (https://12factor.net)

MICROSERVICE FRAMEWORK LANDSCAPE

THE ANALYSIS

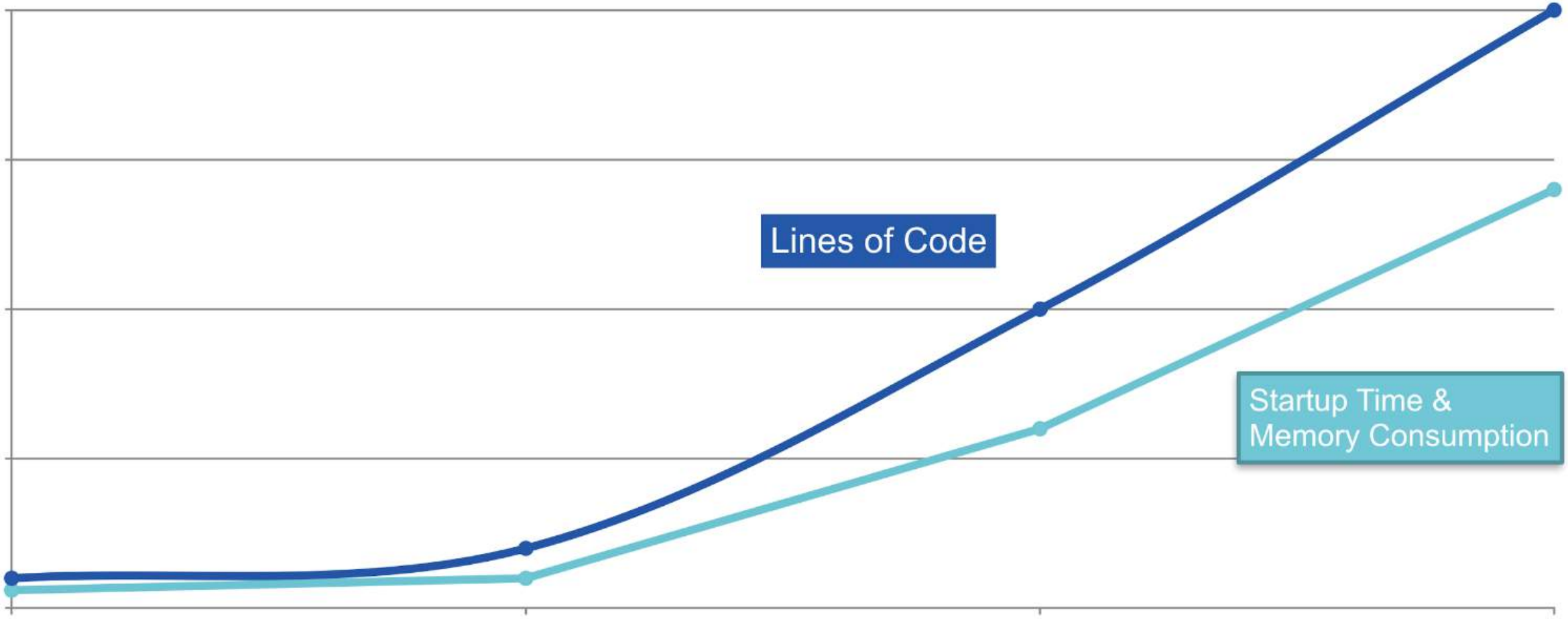
To meet this goal, we performed an analysis of Spring and Grails and the challenges of using them to develop microservice applications.

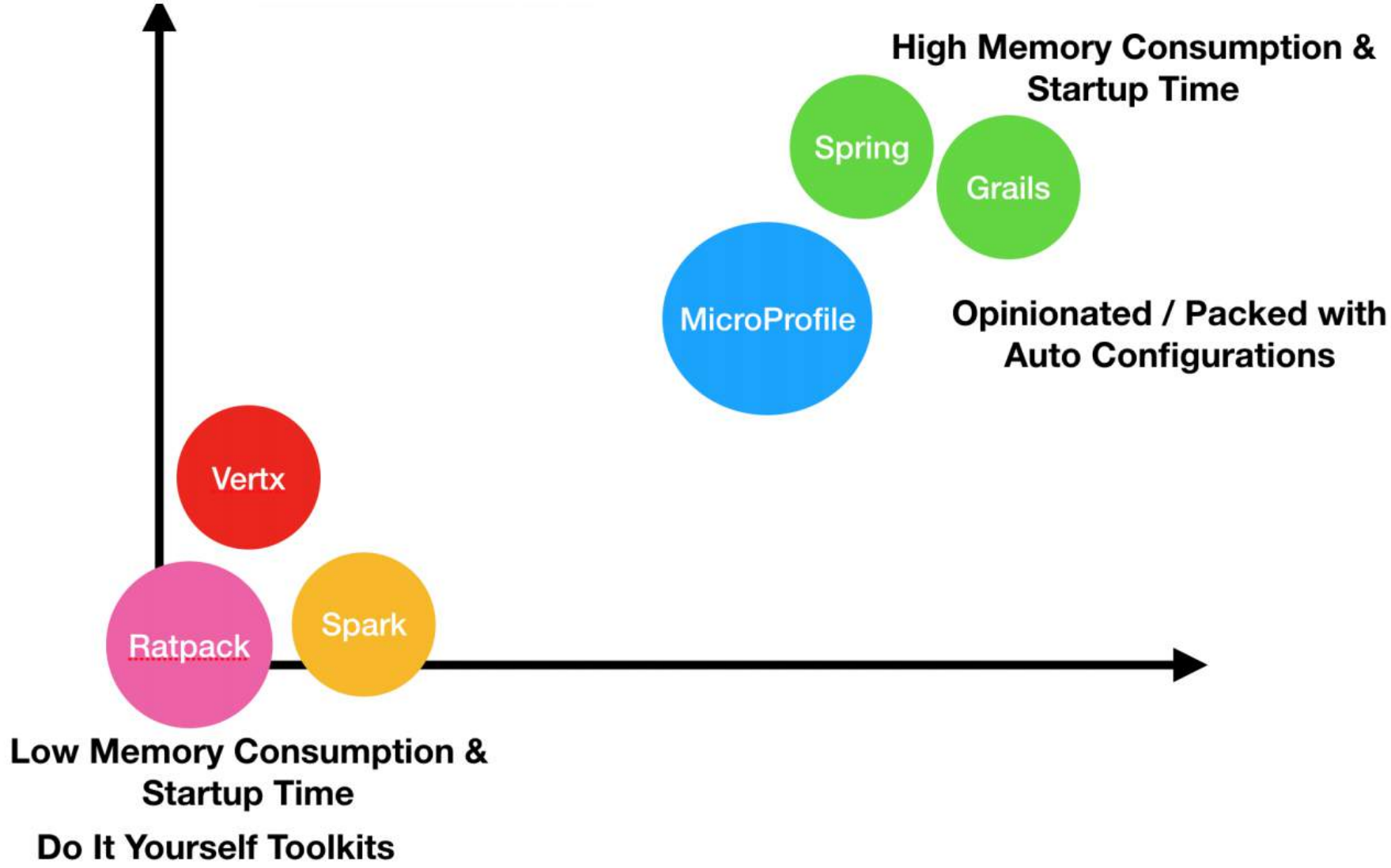
WHAT SPRING AND JAKARTA EE DO

Spring and Jakarta EE are amazing technical achievements. They do all of the following . . . but they do them **at runtime**.

- **Read the byte code** of every bean they find
- **Synthesize new annotations** for each annotation on each bean method, constructor, field, etc., to support annotation metadata
- **Build reflective metadata** for each bean for every method, constructor, field, etc.

SO WHAT'S THE PROBLEM?





THE MICRO REALITY

- Frameworks based on reflection and annotations become fat.
- But we love the programming model and productivity, so we live with it.
- So … how can we be more efficient?

Imagine if Kubernetes or Docker had been written in Spring or Jakarta EE instead of Go …

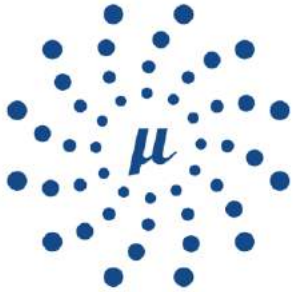
ALREADY SOLVED BY AHEAD-OF-TIME (AOT) COMPILATION

The Android community already solved the problem

- AOT compilation used extensively
- Google Dagger 2.x
  - Compile-time dependency injector
  - Reflection free
  - Limited in scope to just DI

INTRODUCING MICRONAUT

- Designed from the ground up with microservices in mind
- Ultra-lightweight and reactive, based on Netty
- Uses AOT compilation
- HTTP client and server
- Support for Java, Kotlin, and Groovy



MICRONAUT

OCI | WE ARE SOFTWARE ENGINEERS.

MICRONAUT DEMOS

OCI | WE ARE SOFTWARE ENGINEERS.

Hello Micronaut

HELLO MICRONAUT

```
@Controller
class HelloController {
      @Get("/hello/{name}")
      String hello(String name) { return "Hello " + name; }
}

@Client("/") // Client Generated at Compile Time
interface HelloClient {
      @Get("/hello/{name}")
      String hello(String name);
}
```

HOW SMALL?

- Smallest Micronaut Hello World JAR is 10MB when written in Java and 12MB in Groovy
- Can be run with as little as 10MB max heap with Kotlin and Java (22 for Groovy)
- Startup time is around a second for Kotlin and Java (a little more for Groovy)
- All dependency injection, AOP, and proxy generation happens at compile time

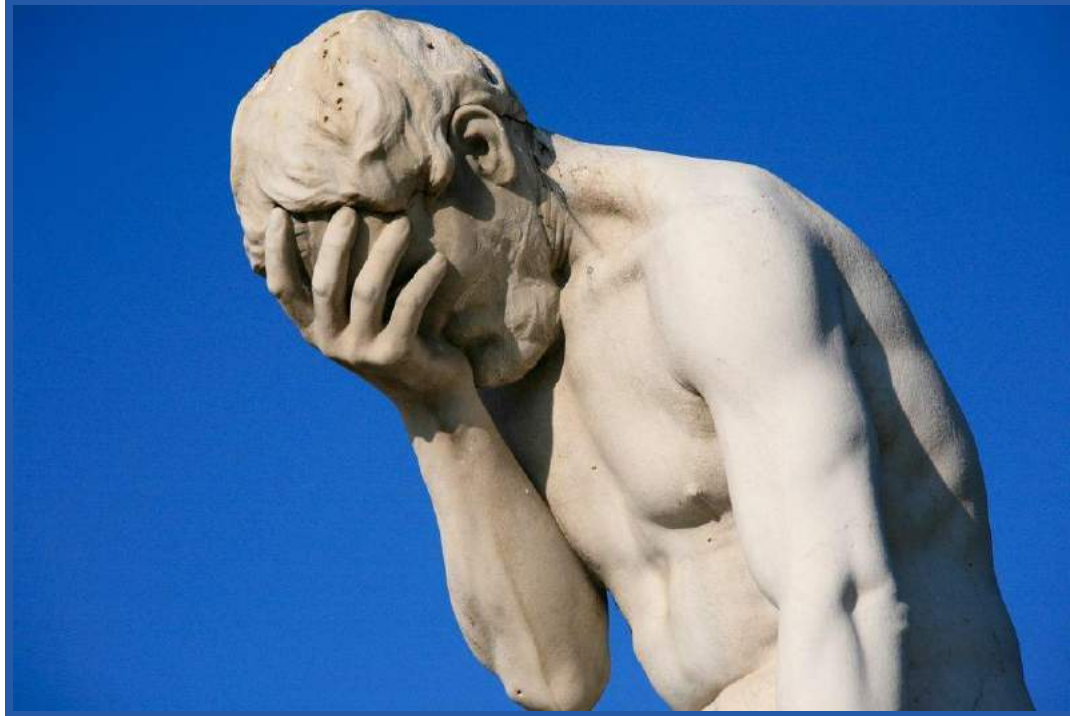
WHAT MICRONAUT COMPUTES COMPILE TIME

- All dependency and configuration injection
- Annotation metadata (meta annotations)
- AOP proxies
- Essentially all framework infrastructure (i.e., what Spring/CDI do at runtime)

Essentially, Micronaut is an AOT framework

NOT ANOTHER SERVER!?

- New little HTTP frameworks appearing all the time
- If all we had achieved was another HTTP server, Micronaut wouldn't be very interesting
- What else does it do?



NATIVELY CLOUD NATIVE

- Service Discovery – Consul, Eureka, Route 53, and Kubernetes
- Configuration Sharing – Consul supported and Amazon ParameterStore
- Client-Side Load Balancing – Integrated or Netflix-Ribbon supported
- Support for serverless computing; AWS Lambda, OpenFaas, Fn supported; Azure coming

OCI | WE ARE SOFTWARE ENGINEERS.

Micronaut Pet Store

SERVERLESS COMPUTING

- Write functions and run them locally or as regular server applications
- Deploy functions to AWS Lambda; after warmup, functions execute in milliseconds

```
@Field @Inject Twitter twitter

@CompileStatic
URL updateStatus(Message status){
    Status s = twitter.updateStatus(status.text)
    String url = "https://twitter.com/$s.user.screenName/status/${s.id}"
    return new URL(url)
}
```

GraalVM

- New Polyglot VM from Oracle
- Runs JS, Java, Ruby, R, etc.
- Ability to turn Java code native
- http://www.graalvm.org

GraalVM NATIVE

Works well when:

- Little or no runtime reflection is used
- There's limited or no dynamic classloading
- You plan ahead
- Third-party libraries are used selectively

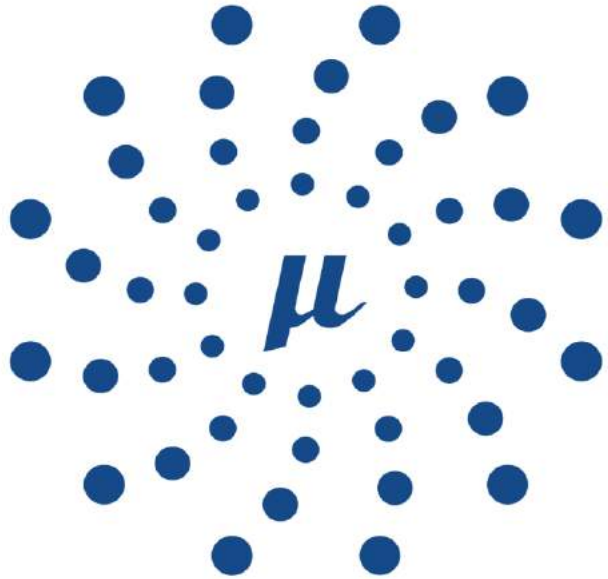
Micronaut + GraalVM™

MICRONAUT + GraalVM

- Like Graal itself at the experimental phase
- Micronaut AOT compilation and reflection-free model makes it easier
- A lot of Micronaut already working:
  - HTTP server, client, and serverless
  - Service discovery
  - DI and AOP

MICRONAUT 1.0 OUT NOW

- Compile-time DI and AOP
- HTTP client and server
- Service discovery
- Distributed tracing
- Serverless functions
- Data access: SQL, MongoDB, Redis, Cassandra, etc.



MICRONAUT 1.0 ON SDKman!

The Micronaut CLI is now available via SDKman!

```
$ curl -s "https://get.sdkman.io" | bash
$ source "$HOME/.sdkman/bin/sdkman-init.sh"
$ sdk install micronaut
$ mn create-app hello-world
```

MICRONAUT RESOURCES

- Gitter Community: https://gitter.im/micronautfw
- User Guide: http://micronaut.io/documentation.html
- Micronaut Guides: http://guides.micronaut.io
- FAQ: http://micronaut.io/faq.html
- GitHub: https://github.com/micronaut-projects/micronautcore
- Examples: https://github.com/micronaut-projects/ micronaut-examples

UPCOMING MICRONAUT EVENTS

- Micronaut Deep Dive
  - 3-day online workshop
  - https://objectcomputing.com/training/catalog/micronaut-training/micronaut-deep-dive
- Devnexus 2GM 2019
  - 2GM (Groovy, Grails and Micronaut) to be featured at Devnexus 2019
  - https://devnexus.com/
- Micronaut Summit 2019
  - Micronaut-focused conference
  - https://micronautsummit.com/

SUMMARY

- Micronaut aims to provide the same "WOW" factor for microservices that Grails did for monoliths
- Built by the people that created Grails, leveraging over 10 years experience in framework development
- Uses AOT compilation to support low memory footprint
- Micronaut 1.0 is available now

Q & A

LEARN MORE ABOUT OCI EVENTS & TRAINING

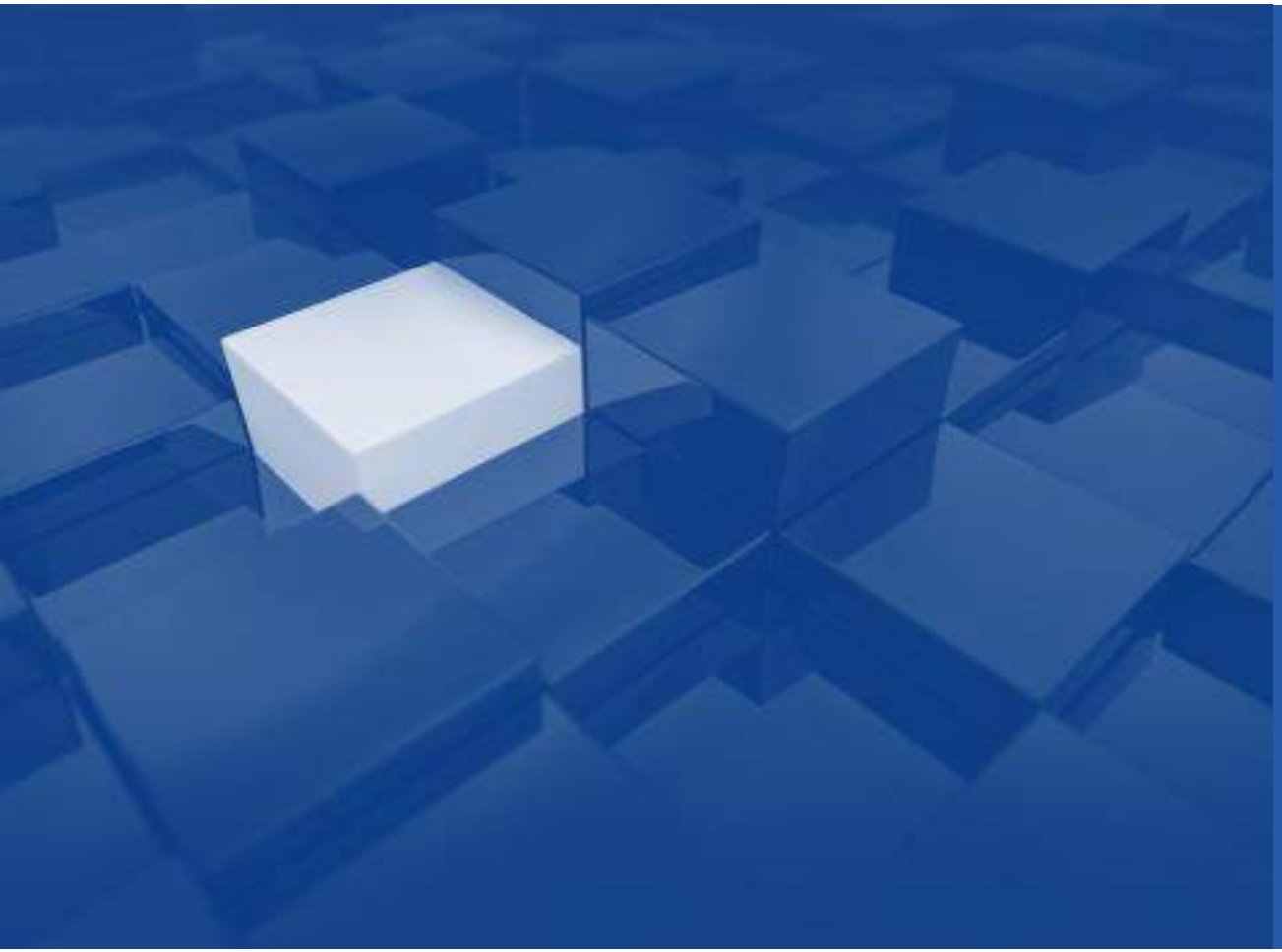
Events:

- objectcomputing.com/events

Training:

- objectcomputing.com/training
- grailstraining.com
- micronauttraining.com

Or email info@ocitraining.com to schedule a custom training program for your team online, on site, or in our state-of-the-art, Midwest training lab.



OCI | WE ARE SOFTWARE ENGINEERS.

CONNECT WITH US

1+ (314) 579-0066

@objectcomputing

objectcomputing.com