ORACLE

ORACLE

New Opportunities for Developers with GraalVM

Alina Yurenko

GraalVM Developer Advocate

Oracle Labs

October 01, 2019

Safe harbor statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

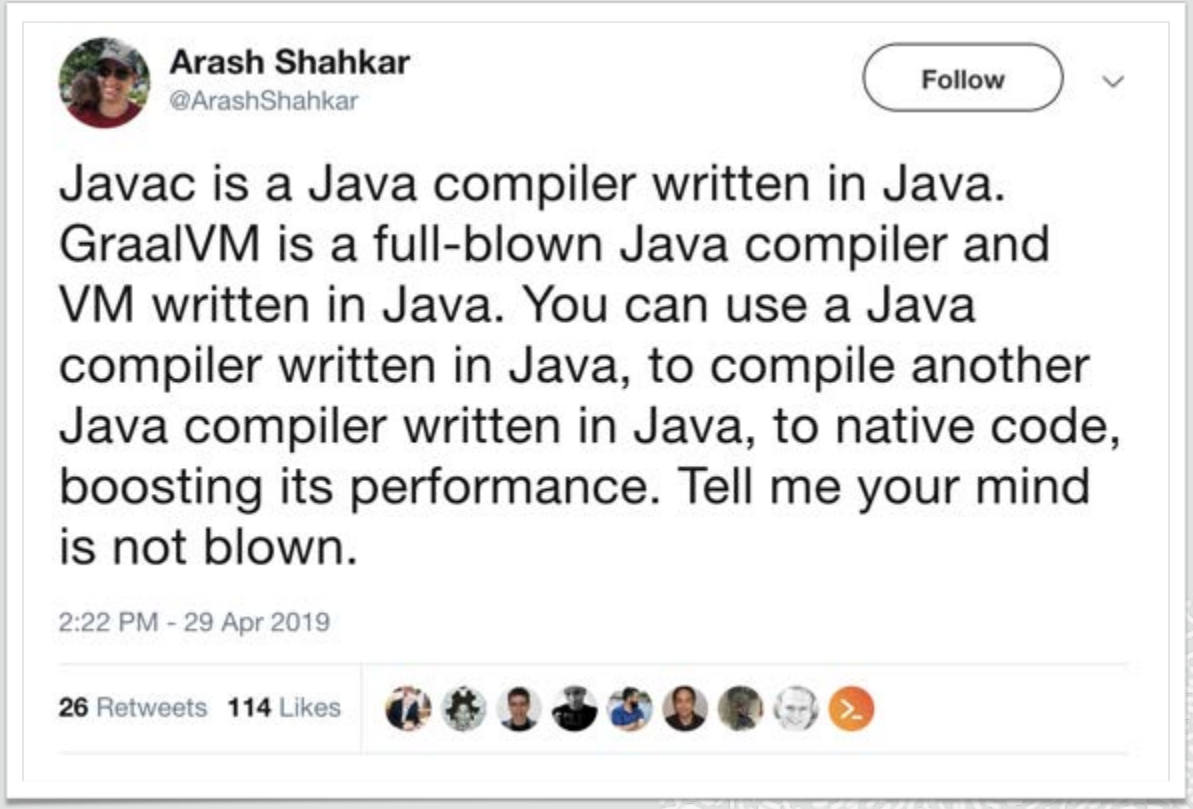
Agenda

1. What's under the hood
2. Performance Optimization
3. Fast startup for Java programs
4. Industry use cases
5. What's next

Agenda

1. What's under the hood
2. Performance Optimization
3. Fast startup for Java programs
4. Industry use cases
5. What's next

GraalVM magic in one tweet



Why GraalVM

High Performance

Optimize application performance with GraalVM compiler

Fast Startup

Compile your application AOT and start instantly

Polyglot

Mix & match languages with seamless interop

Open Source

See what's inside, track features progress, contribute

GraalVM™

OpenJDK™

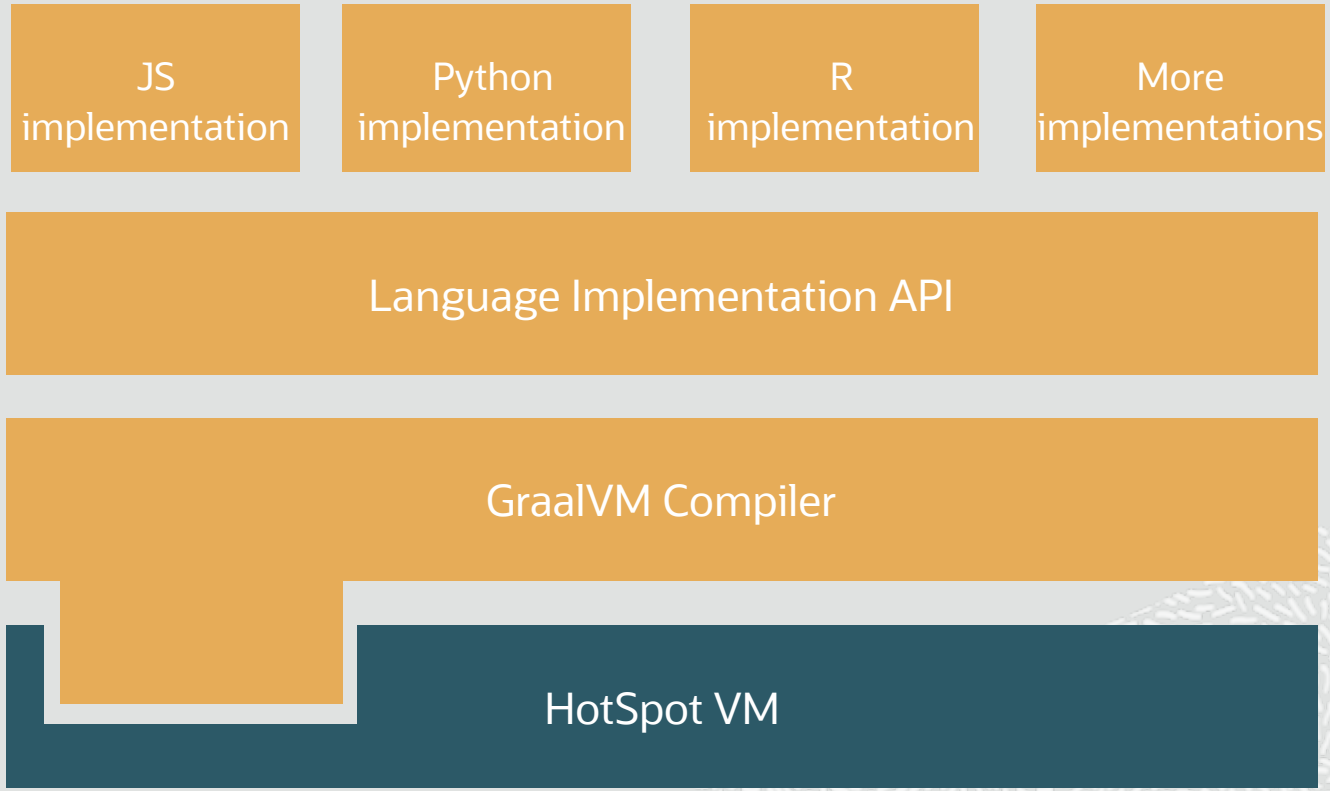
node

database

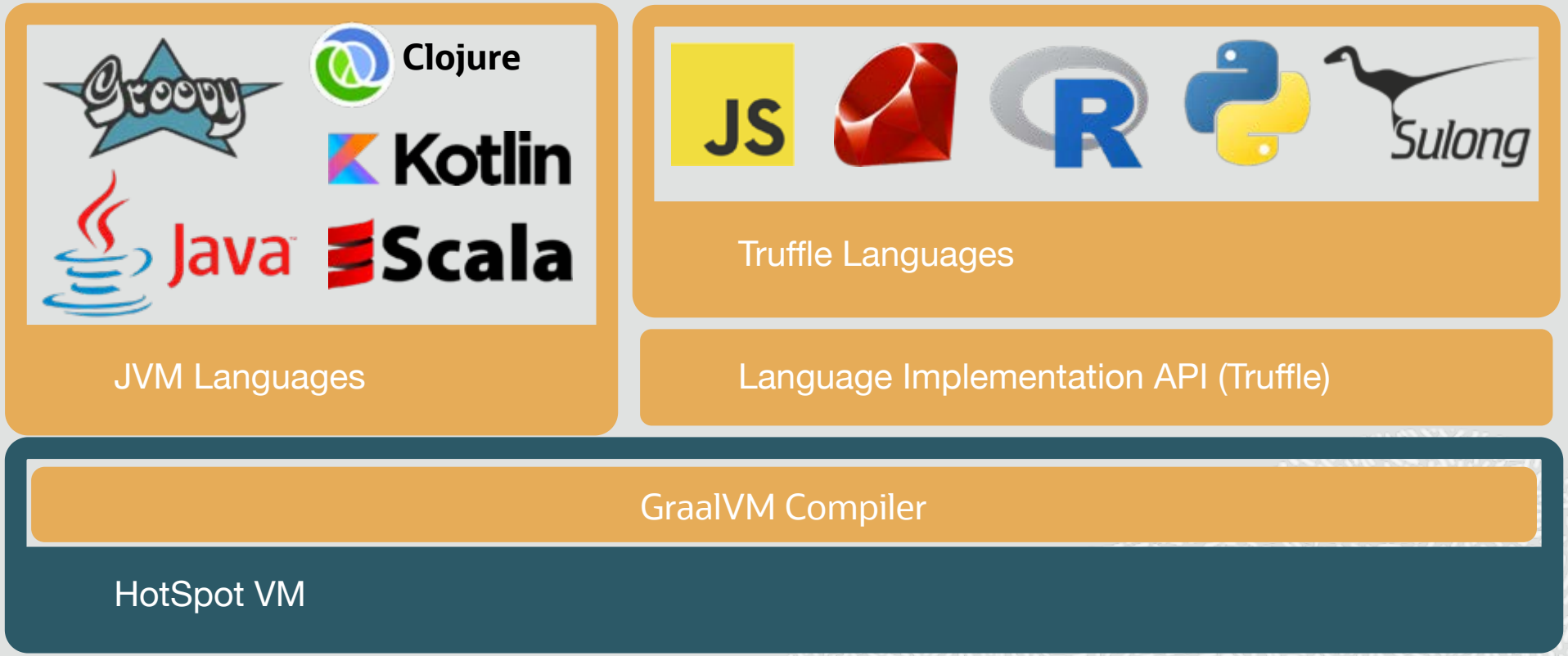
standalone

Production-ready!🎉

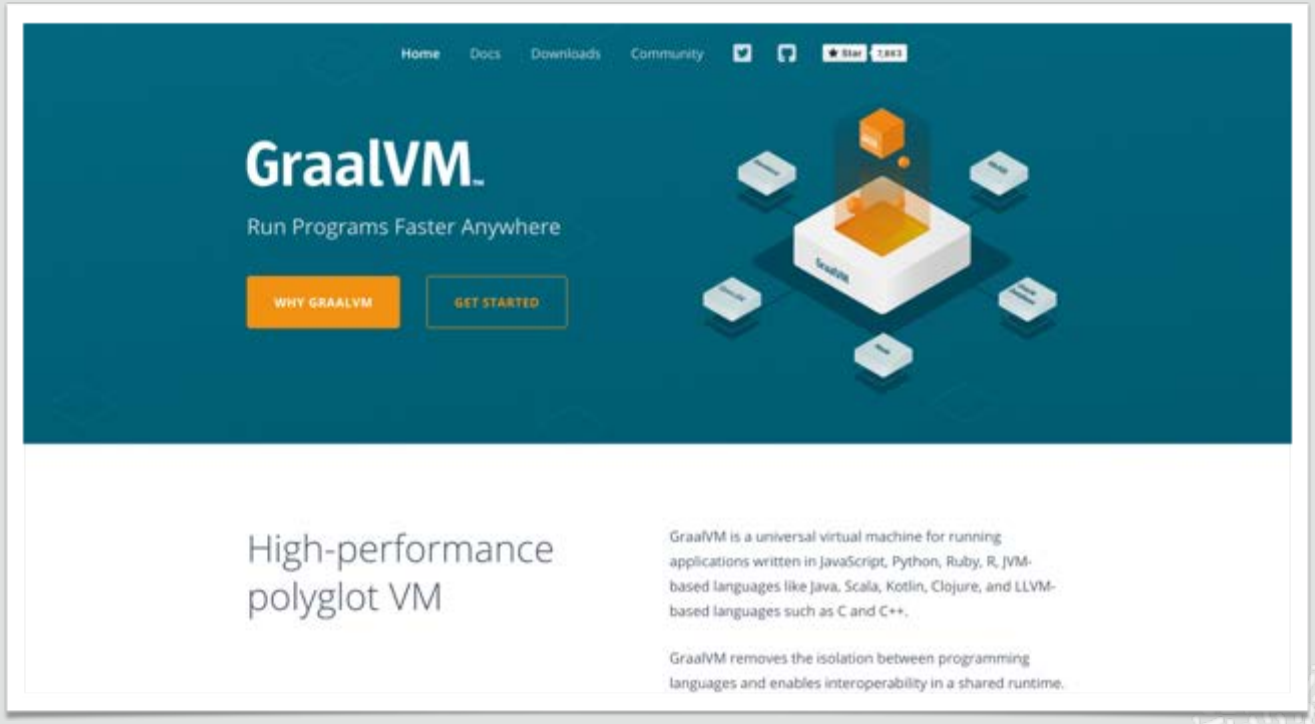
Architecture



Architecture

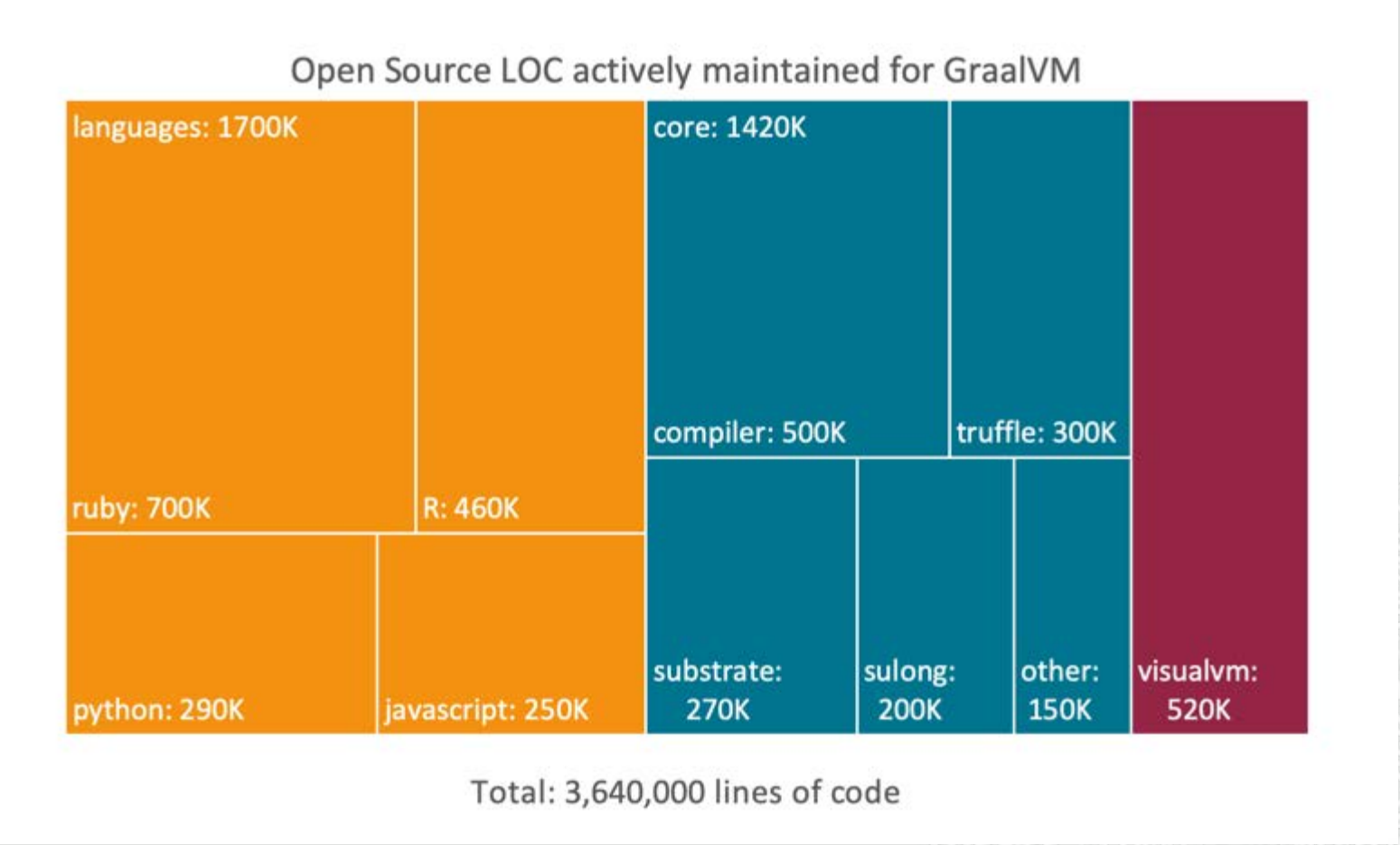


Get Started



- Downloads
- Documentation
- Community support

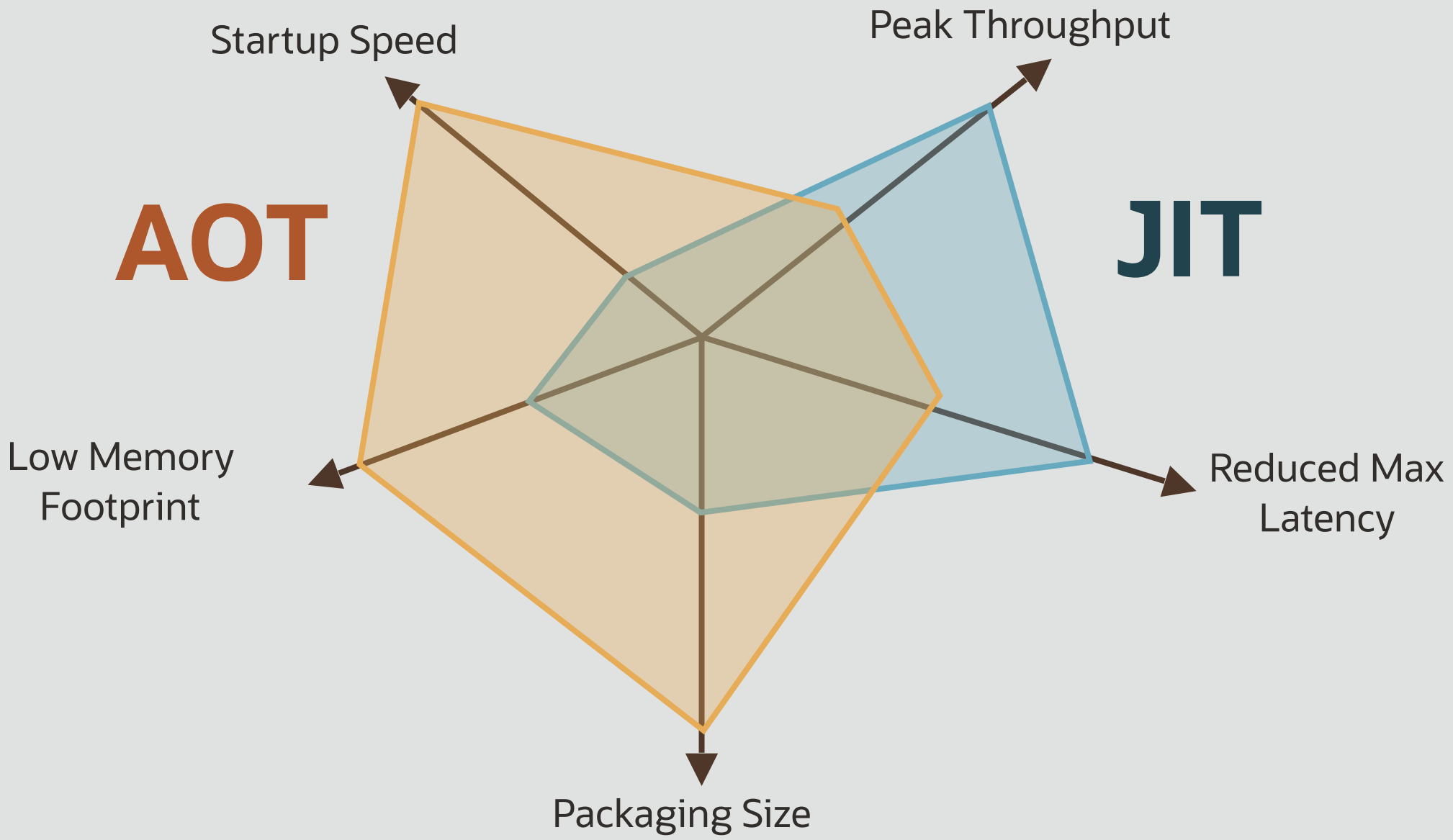
GraalVM Open Source



Agenda

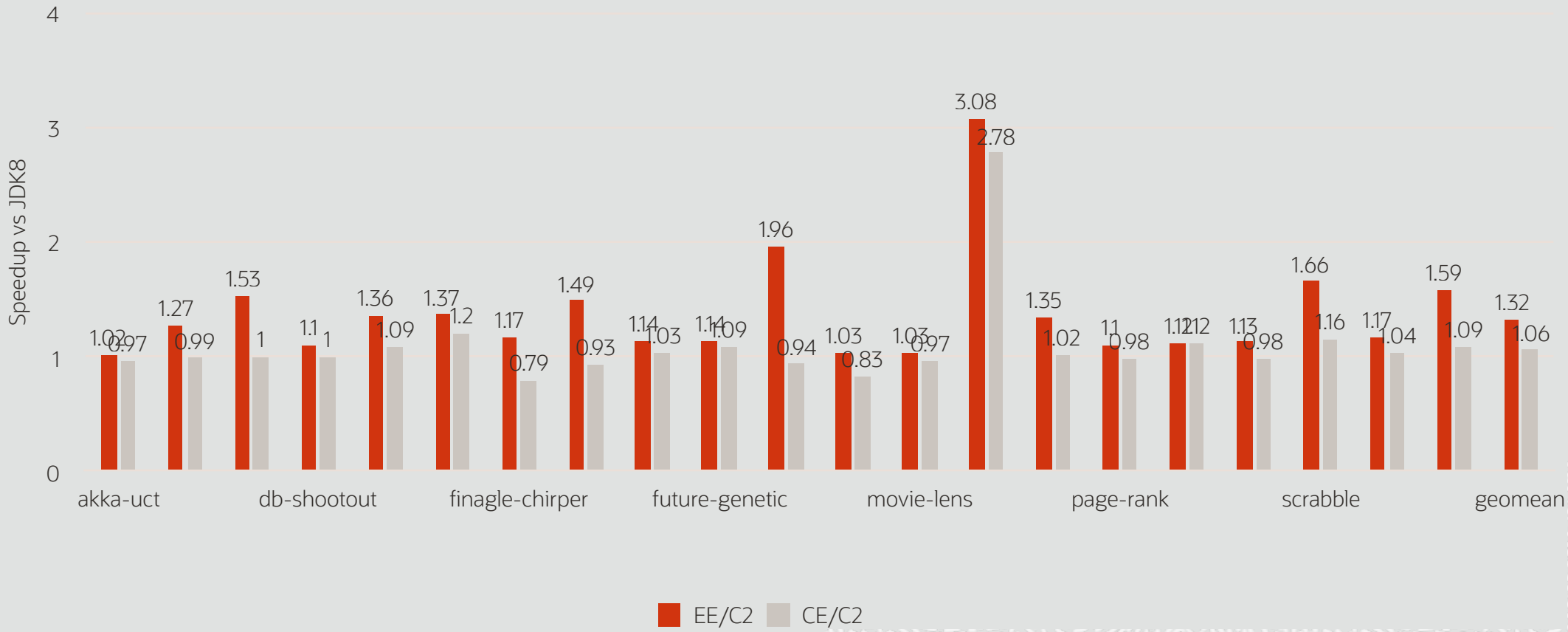
1. What's under the hood
2. Performance optimization
3. Fast startup for Java programs
4. Industry use cases
5. What's next

Measure performance?



Demo time

GraalVM JIT Performance: Renaissance.dev

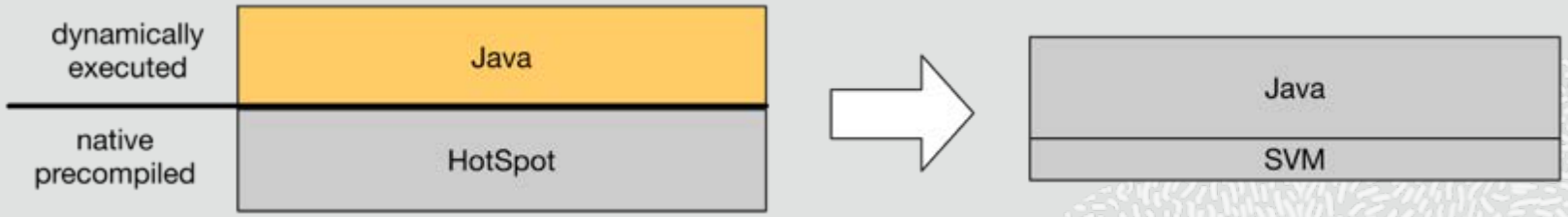


Agenda

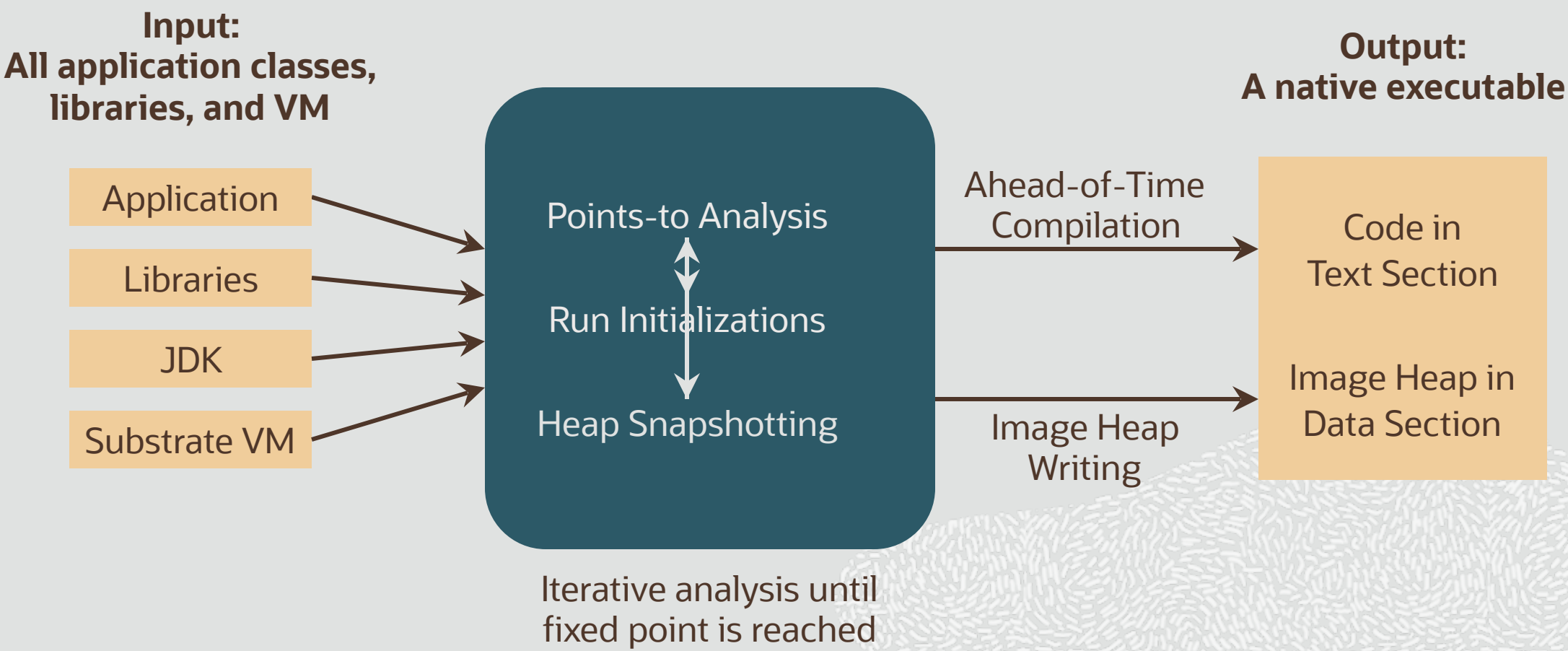
1. What's under the hood
2. Performance Optimization
3. Fast startup for Java programs
4. Industry use cases
5. What's next

GraalVM Native Images

- Instant startup;
- Low memory footprint;
- Works with memory management;
- AOT-compiled using the GraalVM compiler.

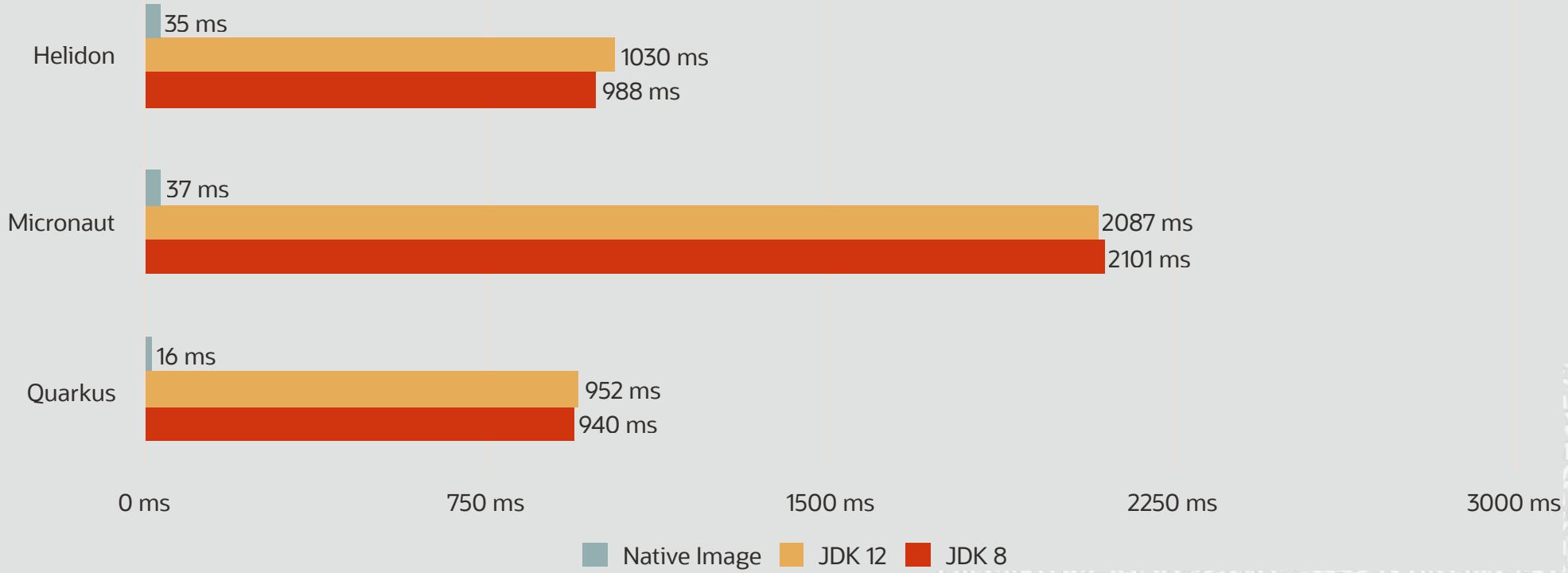


Native Image: Details

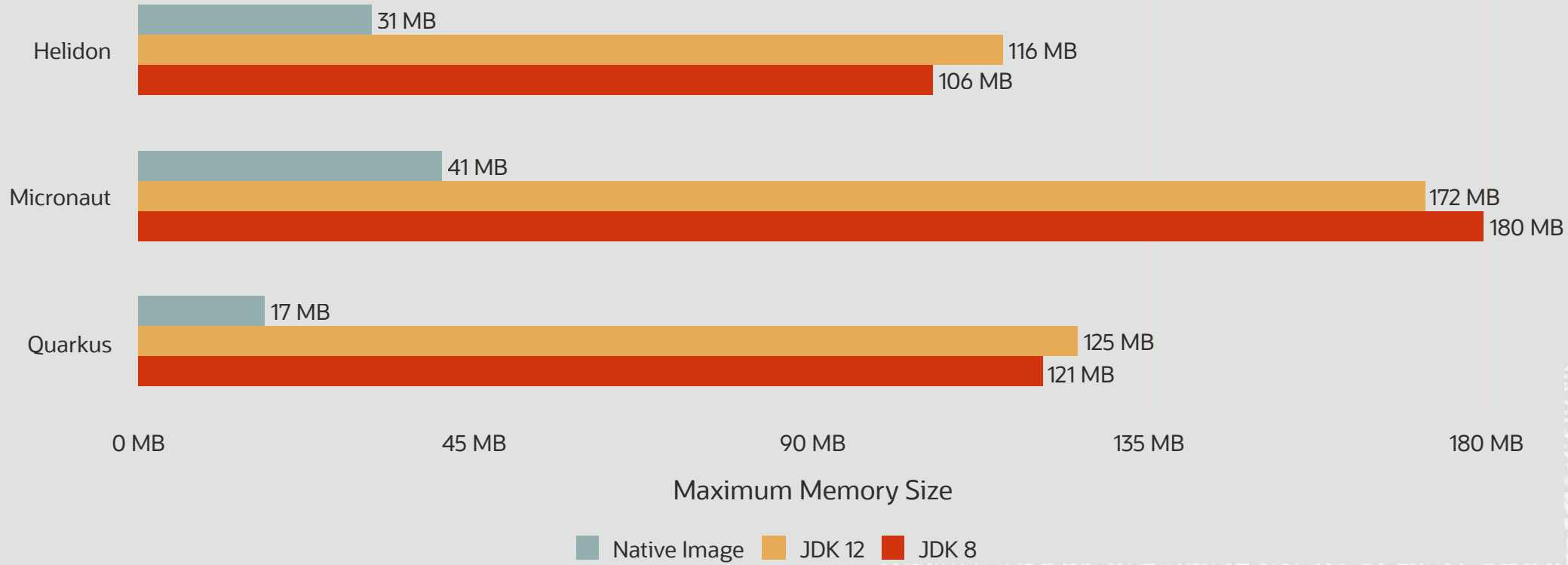


Demo time

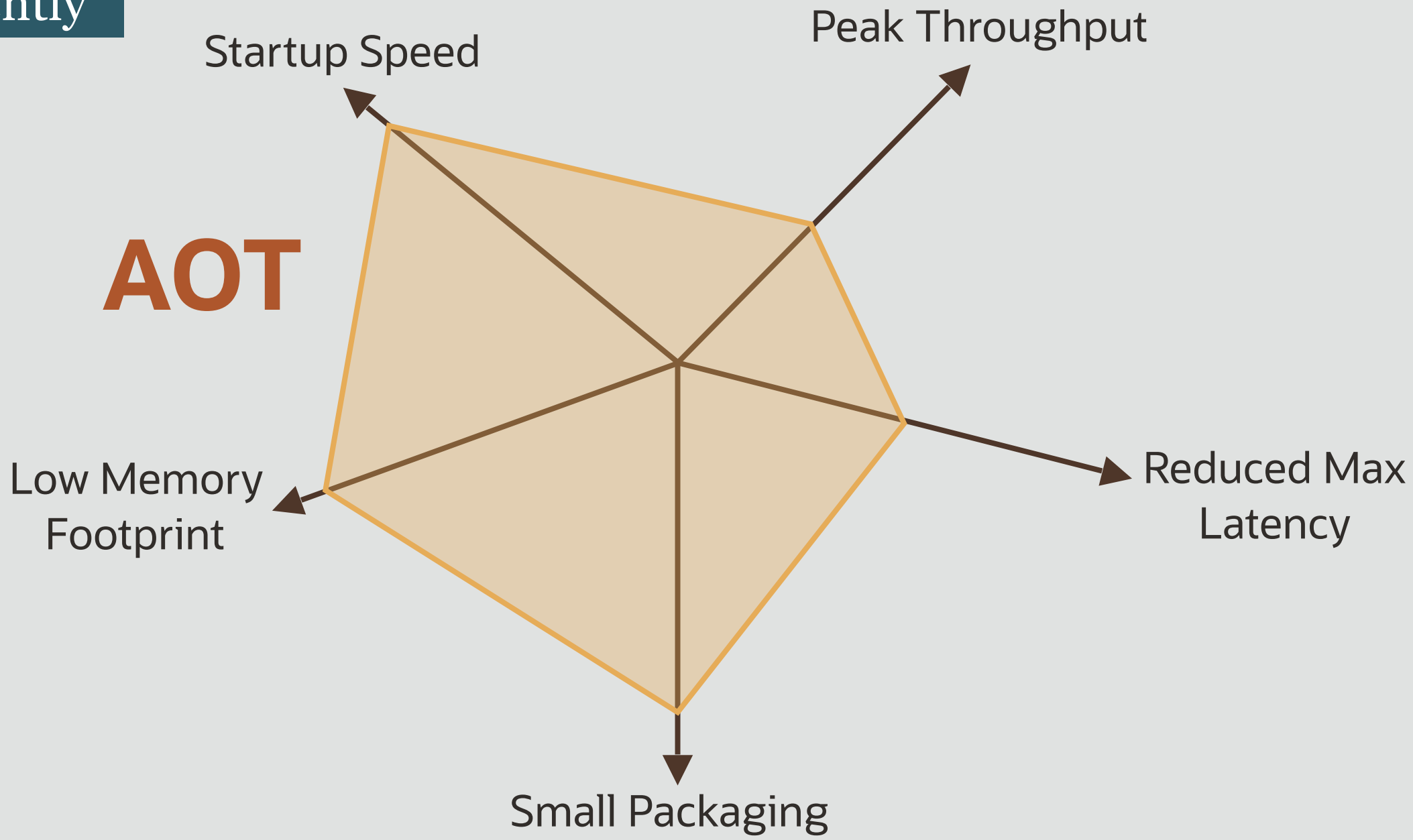
Microservice Frameworks: Startup Time



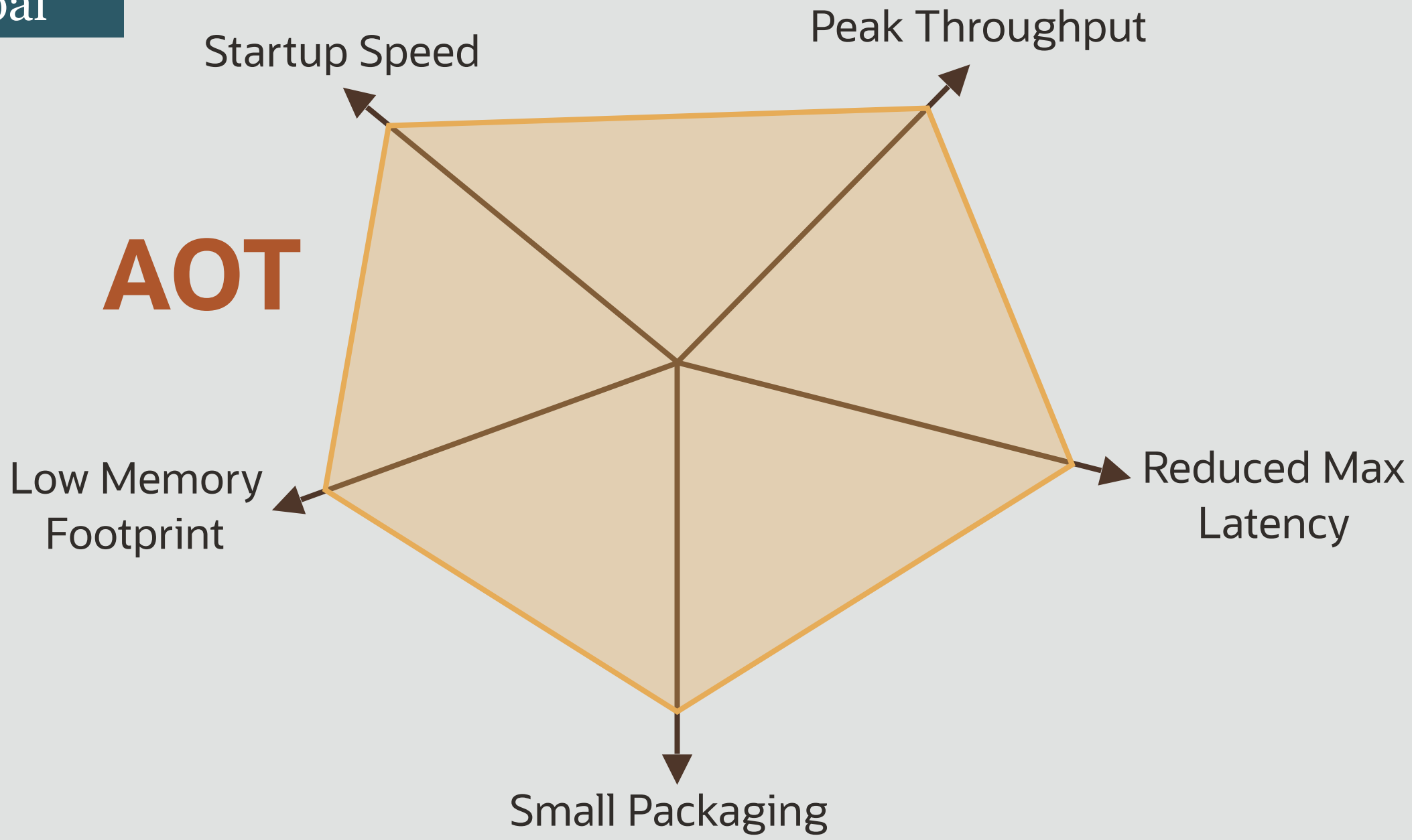
Microservice Frameworks: Memory Usage



Currently


Startup Speed
Peak Throughput
AOT
Low Memory Footprint
Reduced Max Latency
Small Packaging

Goal


AOT
Startup Speed
Peak Throughput
Low Memory Footprint
Reduced Max Latency
Small Packaging

Native Image: Profile-Guided Optimizations (PGO)

The GraalVM compiler is built ground-up with profiles in mind

Collecting profiles is essential for performance of native images

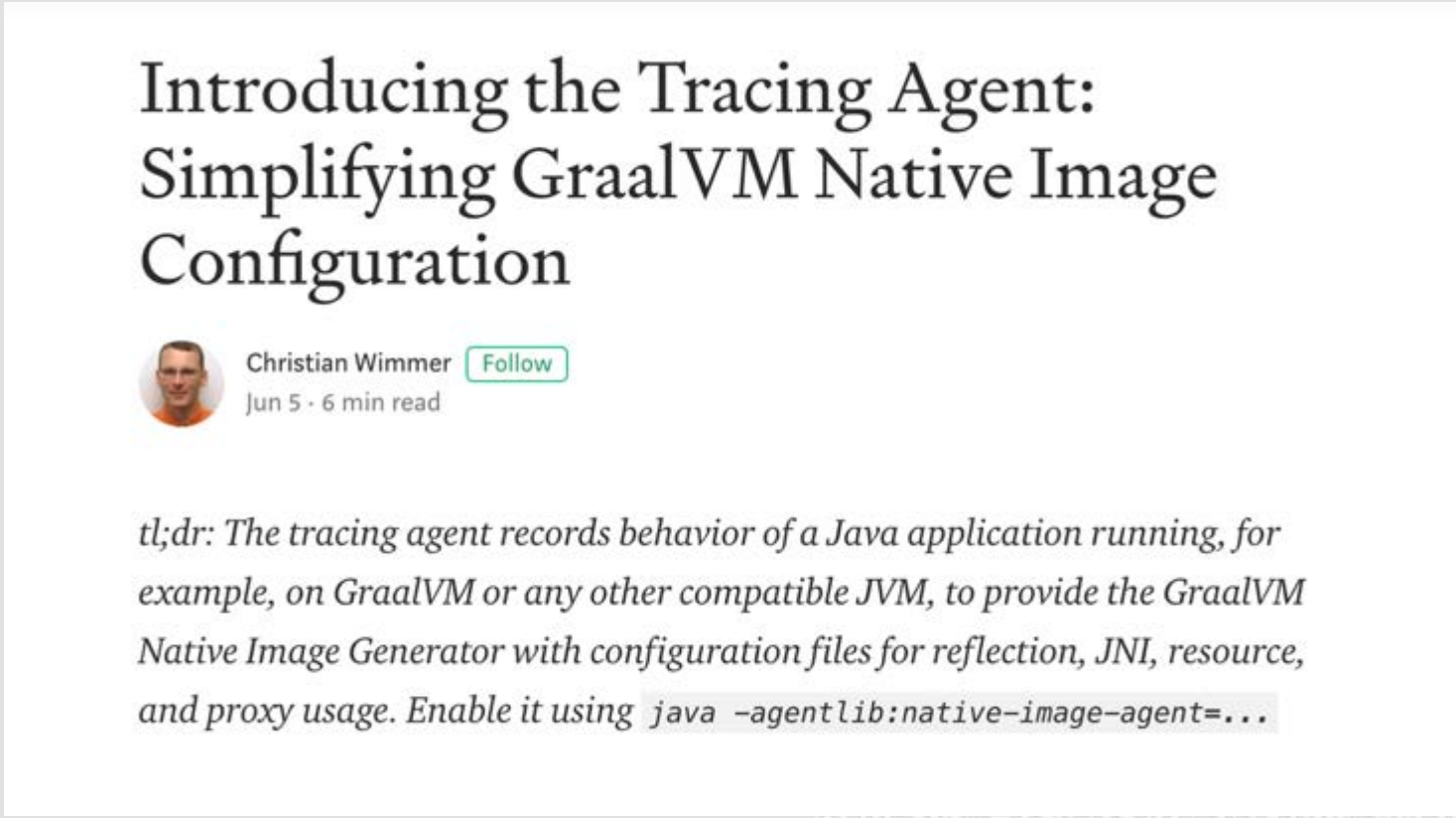
PGO requires running relevant workloads before building an image

```
$ java -Dgraal.PGOInstrument=myclass.iprof MyClass
```

```
$ native-image --pgo=myclass.iprof MyClass
```

```
$ ./myclass
```

Simplifying the Native Image Configuration



Introducing the Tracing Agent: Simplifying GraalVM Native Image Configuration

Christian Wimmer Follow
Jun 5 · 6 min read

tl;dr: The tracing agent records behavior of a Java application running, for example, on GraalVM or any other compatible JVM, to provide the GraalVM Native Image Generator with configuration files for reflection, JNI, resource, and proxy usage. Enable it using `java -agentlib:native-image-agent=...`

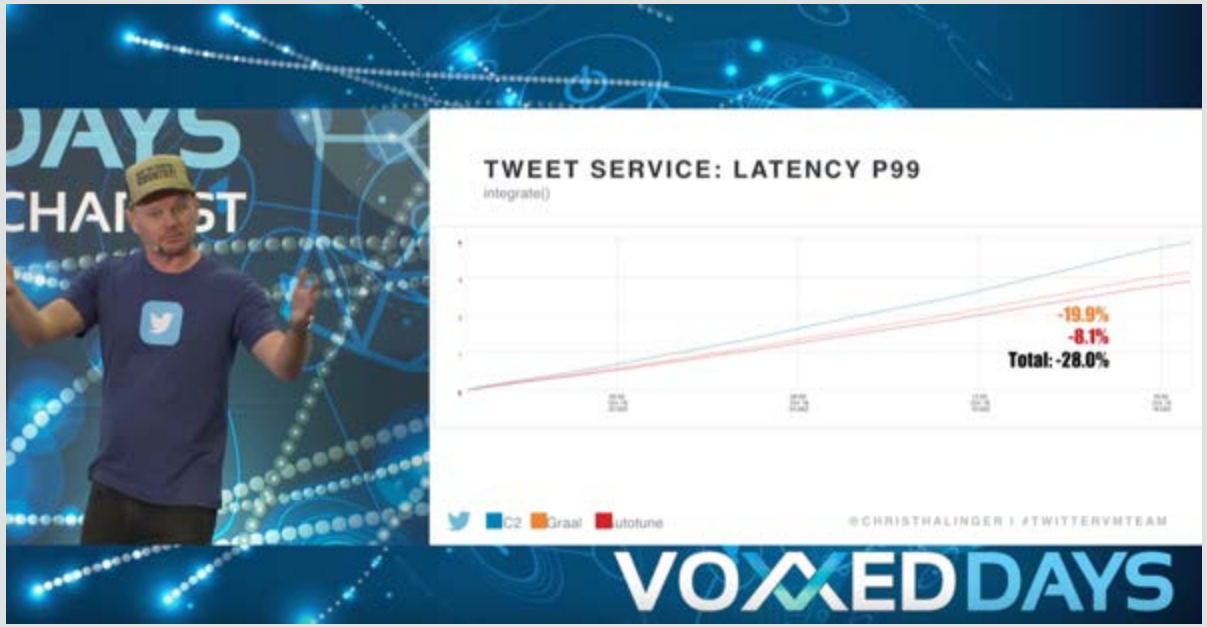
Continue Learning About GraalVM Native Images

- Reference manual: graalvm.org/docs/reference-manual/aot-compilation/
- Improving performance of GraalVM native images with PGO: https://medium.com/graalvm/improving-performance-of-graalvm-native-images-with-profile-guided-optimizations-9c431a834edb
- GraalVM Native Images: The Best Startup Solution for Your Applications: https://www.youtube.com/watch?v=z0jedLjcWjI

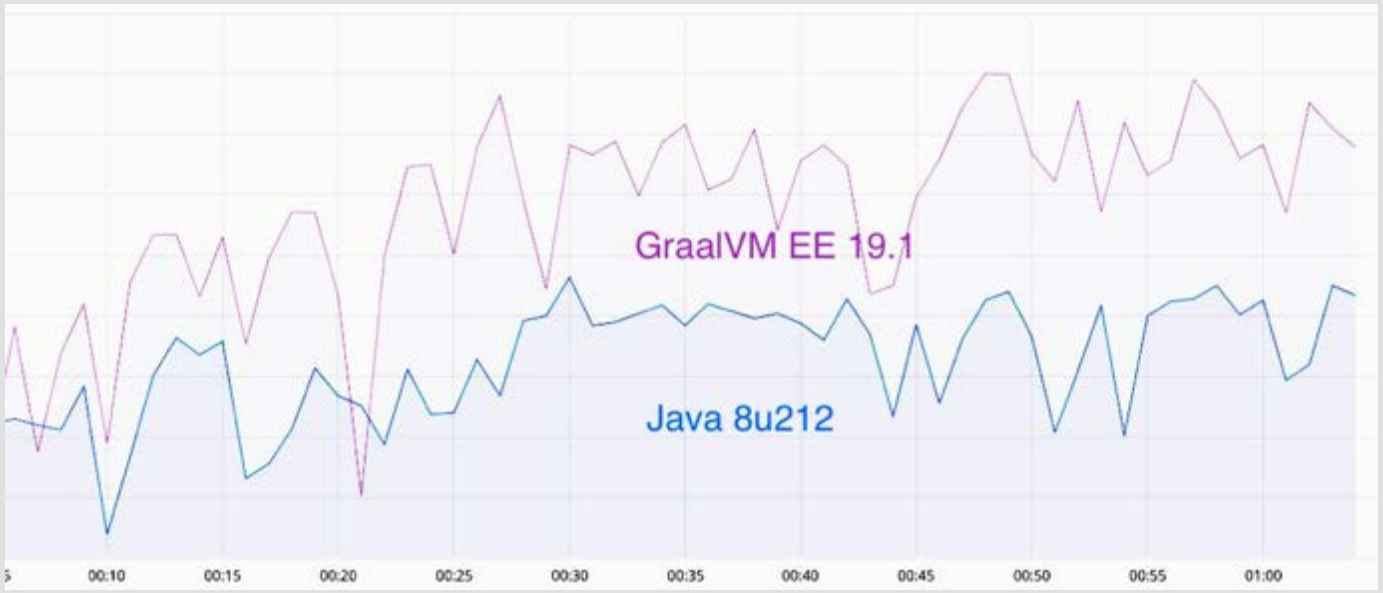
Agenda

1. What's under the hood
2. Performance optimization
3. Fast startup for Java programs
4. Industry use cases
5. What's next

Twitter uses GraalVM compiler in production to run their Scala microservices



- Peak performance: +10%
- Garbage collection time: -25%
- Seamless migration



ORACLE
Cloud Infrastructure

The rich ecosystem of CUDA-X libraries is now available for GraalVM applications.

GPU kernels can be directly launched from GraalVM languages such as R, JavaScript, Scala and other JVM-based languages.

Agenda

1. What's under the hood
2. Performance optimization
3. Fast startup for Java programs
4. Industry use cases
5. What's next for GraalVM

Recent Updates

- Updated profile-guided optimizations for native images;
- Support for JFR in Graal VisualVM;
- Throughput improvements in native images;
- LLVM toolchain;
- VS Code plugin preview;
- Class Initialization changes in native images.

What's next for GraalVM

- JDK-11 based builds;
- ARM64 and Windows support;
- Low-latency, high-throughput, and parallel GC for native images;
- Work with the community to support important libraries;
- New languages and platforms;
- Your choice – contribute!

What's next for you

- Download:
  graalvm.org/downloads
- Follow updates:
  @GraalVM / #GraalVM
- If you need help:
- gitter.im/graalvm
- graalvm-users @oss.oracle.com

Thank you!

Alina Yurenko / @alina_yurenko

GraalVM Developer Advocate
Oracle Labs