On the Hunt for the Holy Graal:

One VM to rule them all?

Dr. Marco Bungart

Software Engineer

2019-09-25

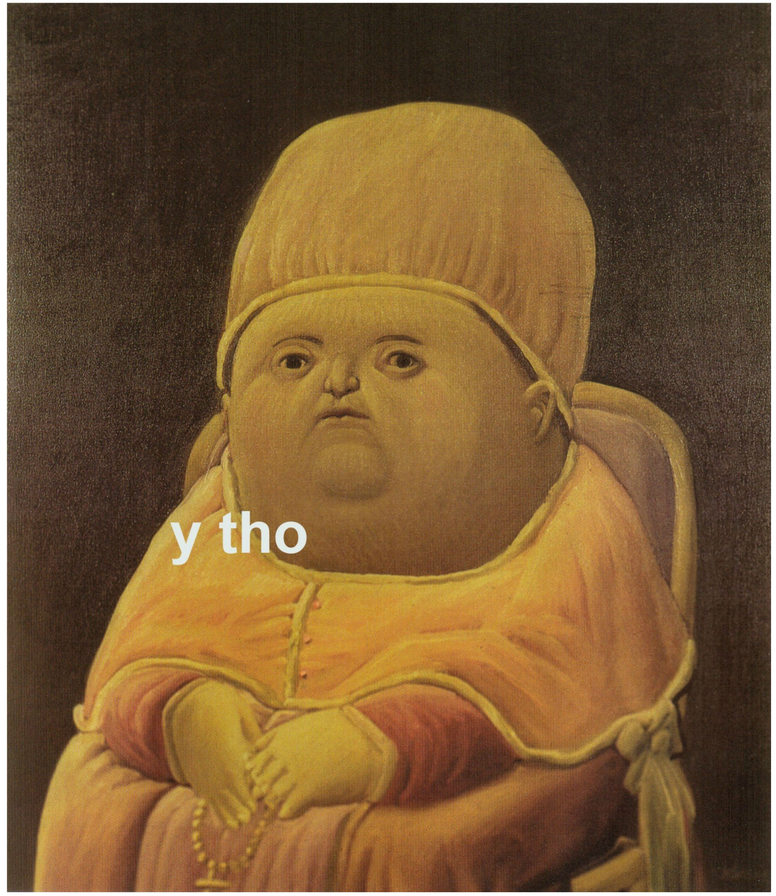
Safe Harbor

The following presentation is intended to outline the features of GraalVM and basic use cases. Neither ConSol Consulting & Solutions Software GmbH nor Dr. Marco Bungart work for or with Oracle on the GraalVM project and are therefore not resposible to fulfill any primises made by the GraalVM project.

Features shown may be marked as „experimental“ or „technical previews“. They are subject to change and may never be delivered.

Opinions presented are my own and neither those of the GraalVM team nor those of ConSol Consulting & Solutions Software GmbH.

Another JVM?

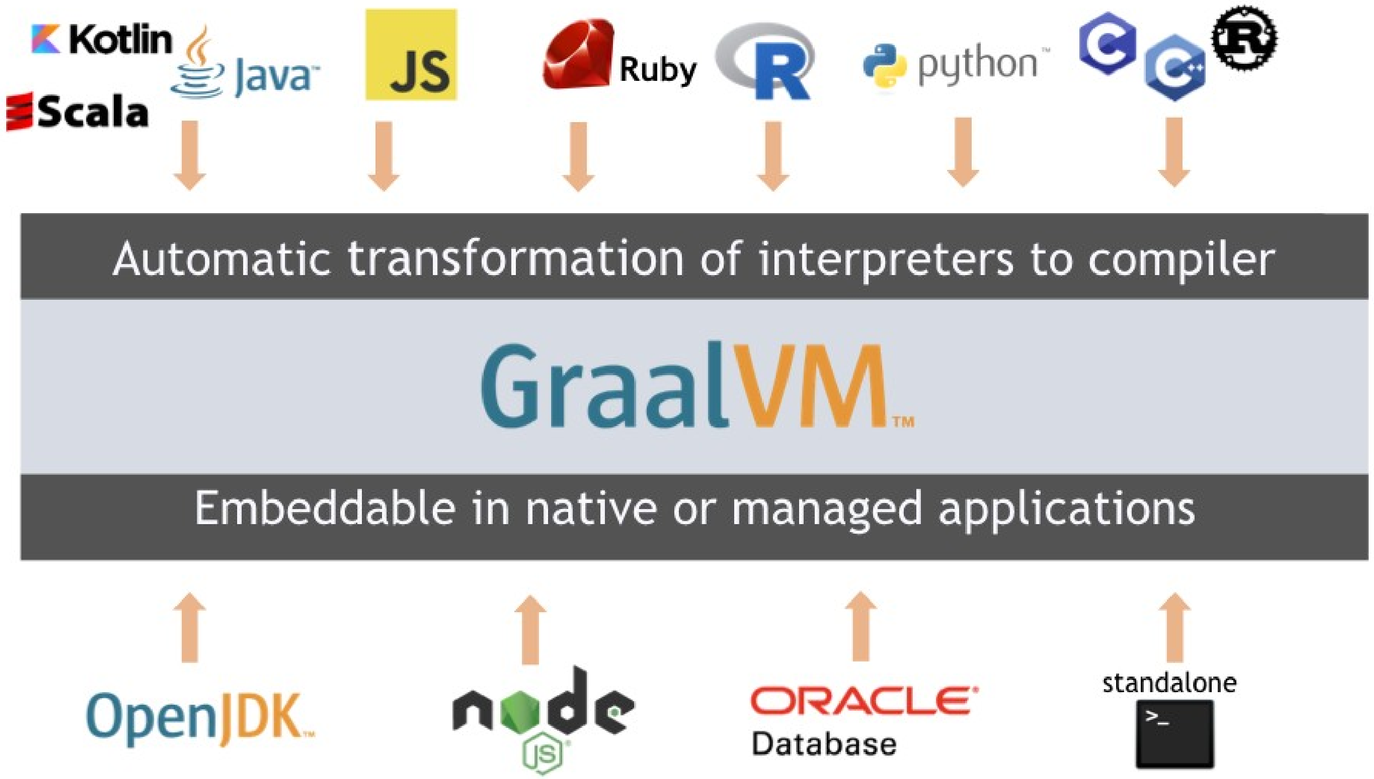


Source: https://knowyourmeme.com

- What offers GraalVM?
  - Extendable Polyglot-capabilities
  - Native compilation
  - Lower memory footprint
- Why use GraalVM?
  - Polyglot-capabilites
  - Better performance
- When to use GraalVM?
  - Polyglot projects
  - Containerized Applications

What: Internals of GraalVM

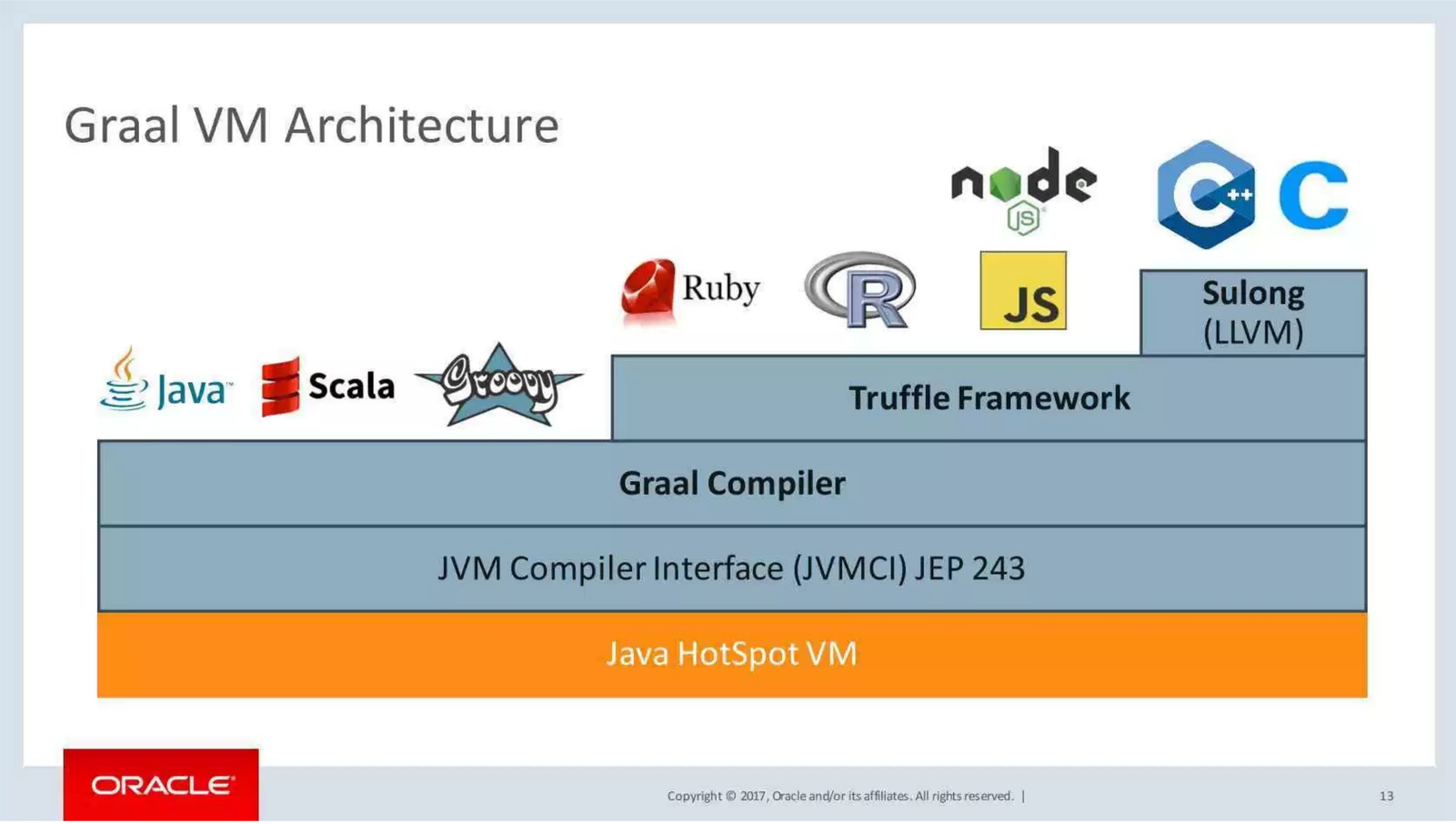
- On a high abstraction level:



Source: https://www.graalvm.org/

What: Internals of GraalVM

- In more detail:



Source: https://jaxenter.de

Licensing of GraalVM

Community Edition

GraalVM Community is available for free for evaluation, development and production use. It is built from the GraalVM sources available on GitHub. We provide pre-built binaries for Linux, macOS X, and Windows platforms on x86 64-bit systems. Windows support is experimental.

DOWNLOAD FROM GITHUB

Enterprise Edition

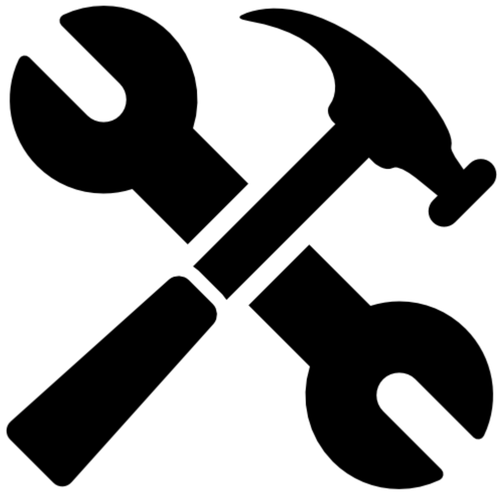
GraalVM Enterprise provides additional performance, security, and scalability relevant for running applications in production. It is free for evaluation uses and available for download from the Oracle Technology Network. We provide binaries for Linux, macOS X, and Windows platforms on x86 64-bit systems. Windows support is experimental.

DOWNLOAD FROM OTN

Source: https://www.graalvm.org/downloads/

Why: benefits GraalVM?

- Programming languages are tools in your toolbox

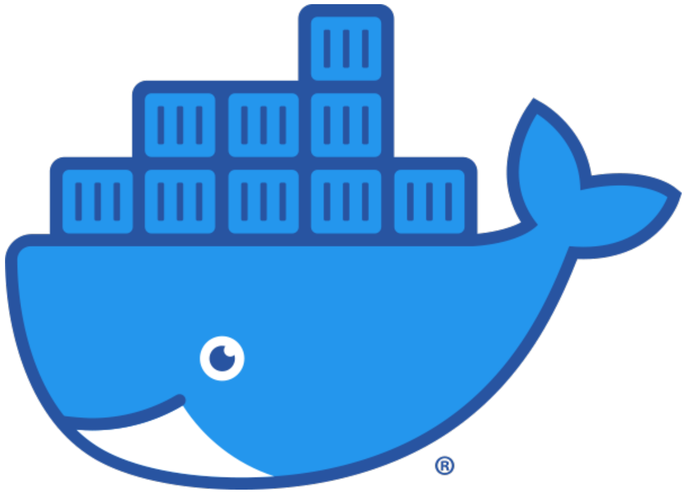


Source: https://www.freepik.com

- Performance improvement

Source: https://insights.jmw.com

- Native compilation for containerization

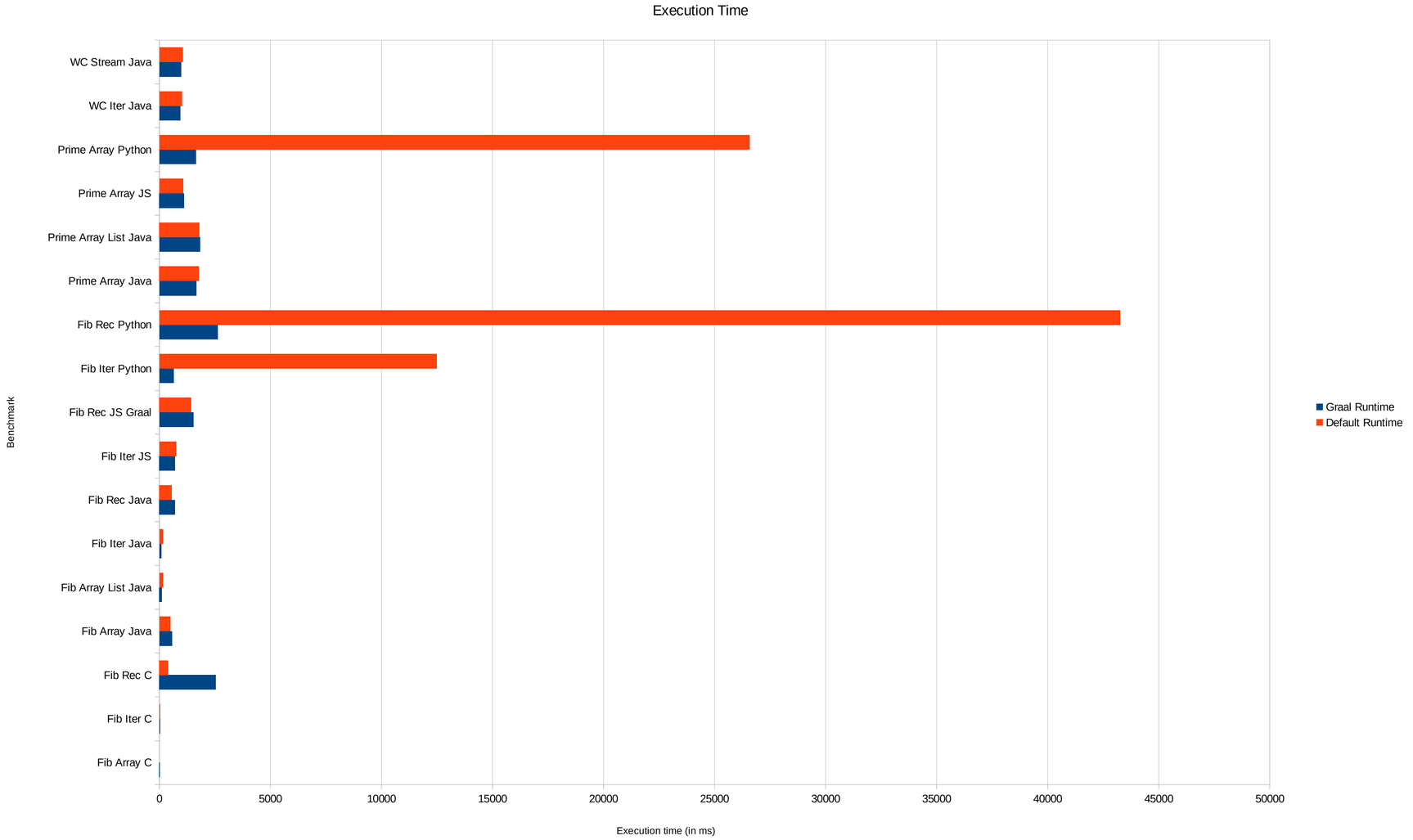


Source: https://www.docker.com

How: to use Polyglot with GraalVM?

- Demo Time!

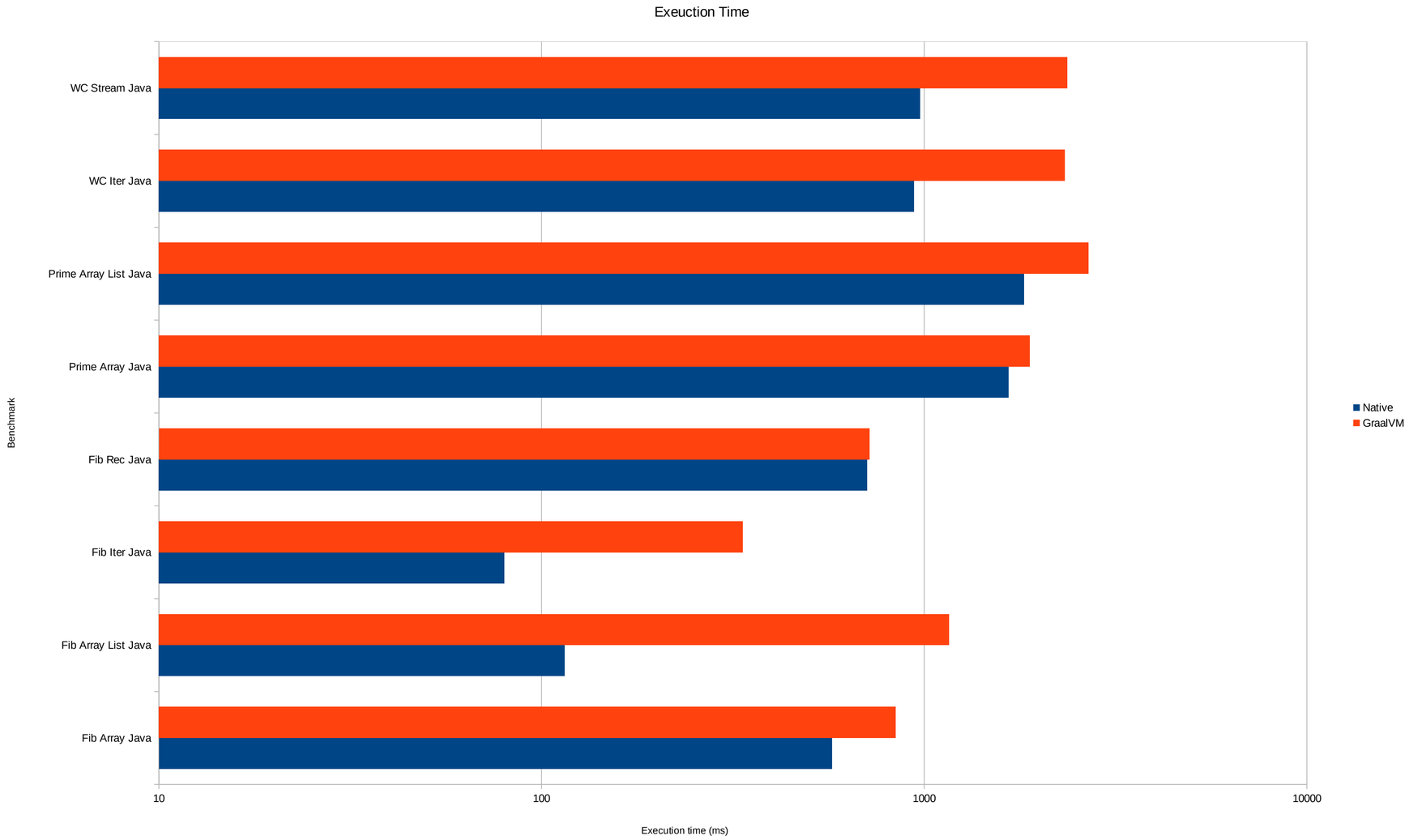
Performance of GraalVM



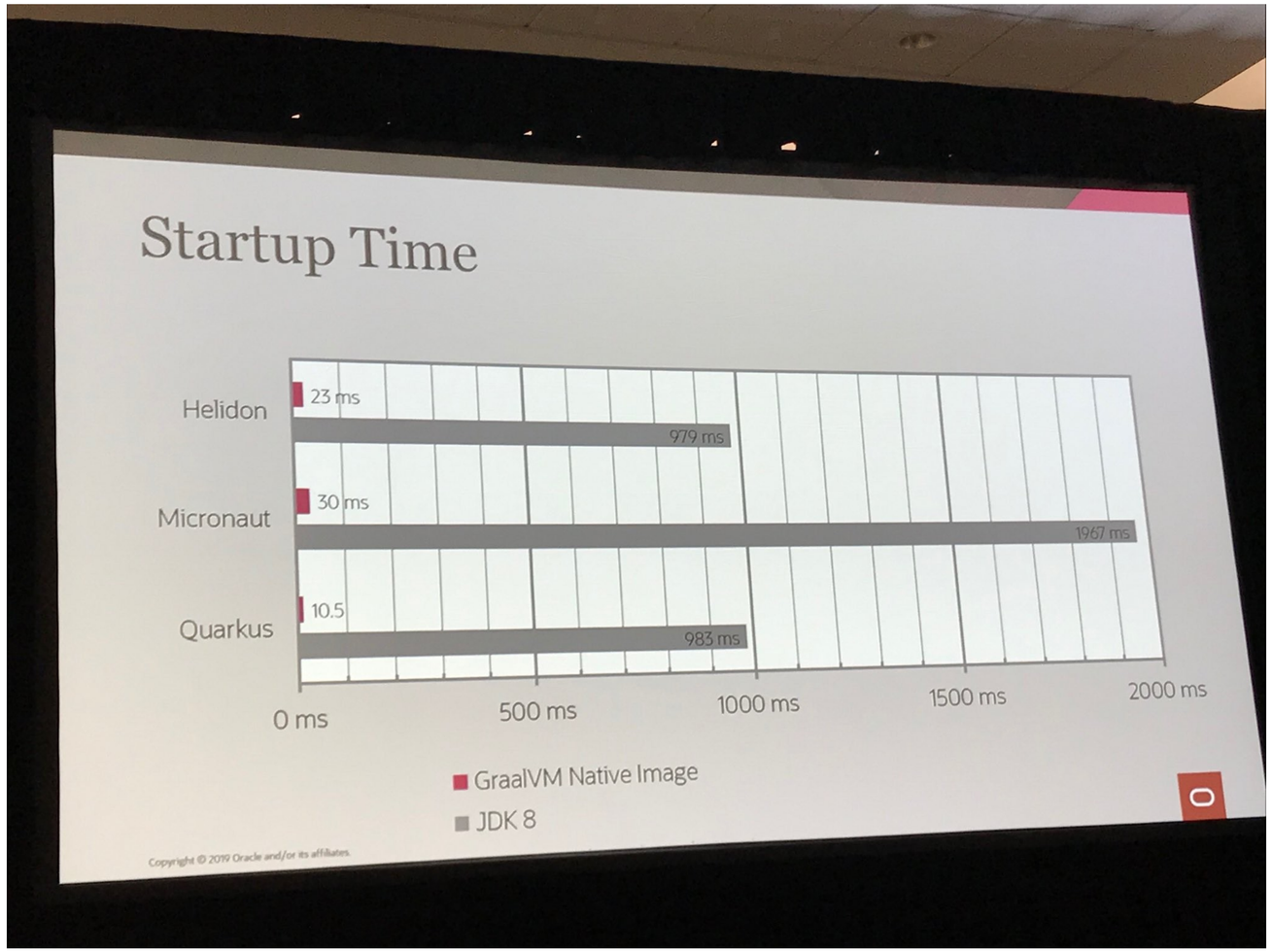
Native Compilation

- Uses SubstrateVM
  - Closed-World assumption
  - Compilation is slow and memory intensive
  - Memory consumption lower
- Guest-languages (JS, Python, …) are **not** compiled natively
- Can compile binary with only static dependencies
- Execution time is **not** at the core of optimization

Native Compilation



Native Compilation



Source: [cesar_saavedr]

Native Compilation

- Demo Time!

References & Further Readings

- **Here at JCON:**
- Peter Palaga: Quarkus: Supersonic Subatomic Java (Cinema B, right here!)

- **References:**
- Interop & benchmark samples:
- Polyglot javascript-R webserver: graalvm demos github repository
- Spring native compilation: spring graal feature github repository

- **Further readings:**
- https://www.graalvm.org/docs/
- Chris Seaton: Top 10 Things To Do With GraalVM

Thank You For Your Attention!

Questions?

Dr. Marco Bungart
Software Engineer

Kanzlerstraße 8
D-40472 Düsseldorf, Germany
Phone: +49-211-339903-0
marco.bungart@consol.de
www.consol.com
Twitter: @turing85