The holy grail or just another Sau, die durchs Dorf getrieben wird?

Thilo Ognibeni
Köln, 03. März 2020
Have you ever heard of...?
Thilo Ognibeni
Software Architect / Developer

Usually feeling at home in the (Java) middleware
... with an exception/hiccup of Vue.js
Born in Aalen, studied in Ulm, ended up in KA
Worshipping agile development
Proudly playing the accordion
GraalVM – Behind the scenes

GraalVM – Practical usage
Disadvantages of Java

› You need a JRE to run Java programs 😐
› Java is slow! ... Is it??
› High memory consumption
› Long start-up times
› “Warm-up phase” of the JVM
› Not really viable for serverless computing
GraalVM, a VM, or what?
GraalVM – a VM ... more *and* less!

› Name of the whole project at Oracle
› Implementation of the Java Virtual Machine
› Support for Native Images
› Truffle Language Implementation Framework
› LLVM Runtime and JavaScript Runtime
› Polyglot
› Embeddable
› Written in Java
GraalVM – History and Options

- Development started around 7 years ago at Oracle Labs and Johannes Kepler University Linz
- New compiler for the 20 year old HotSpot Compiler
- First production-ready version, GraalVM 19.0, released in May 2019
- Since 19.3.0 based either on OpenJDK 1.8.0 or 11
- Available as Community or Enterprise Edition
Java Virtual Machines for Beginners

Graal Compiler

https://dzone.com/articles/jvm-architecture-explained
Java Virtual Machines for Beginners

https://jrebel.com/rebellabs/graal-and-truffle-for-polyglot-languages-on-jvm/
GraalVM Compiler

› Modern JIT Compiler
› Replacing C2 Compiler in HotSpot
› Written in Java (modular and extendable)
› Aggressive use of
  › Inlining
  › Polymorphic Inlining (using profiling data)
  › Partial Escape Analysis
  › etc.
Truffle Language Implementation Framework

› Allows implementing languages on top of GraalVM
› Language abstract syntax tree interpreter
› Write a language interpreter in Java, automatically get a JIT optimized compiler ("Futamura Projection")
› Makes complicated creation of compilers obsolete
› Typical flow looks like
  Program Code $\rightarrow$ AST $\rightarrow$ Truffle $\rightarrow$ Graal $\rightarrow$ Machine code
› Written in Java
Wanna mix it? – GraalVM’s polyglot capabilities

› Zero overhead interoperability between programming languages
› Graal provides complete Node.js environment
› You can use all Graal features also for polyglot apps
› Implements Chrome DevTools Protocol
GraalVM Native Image

› Allows Ahead-of-Time (AoT) Compilation
› Produces executable native binaries
› JVM is not necessary any more
› SubstrateVM bundled in binary
› Reduces
› start-up time
› memory footprint
› Increases compile time ... a lot 😱
Compiling, Interpreting, or what?

- Java is slow because it’s only interpreting the Bytecode 😐
- JIT compiler compiling “hot spots” at runtime
  - Bad: Program needs to “warm-up” first
  + Good: Bytecode is platform independent
  + Good: Interpretation/JIT compilation allows late binding
  + Good: Reduced compilation time
- Precompiled code can be slower than JIT compiled code!
AoT compilation

› Compiles to (machine dependent) optimized machine code
› Complex and advanced (= time consuming) optimizations
› Eliminates the need for JIT compilation
› Drops a useful fraction of a runtime environment
› No profile guided optimization ... out of the box
› Static analysis of **everything** at build time
  › Reflection?
  › Dynamic proxies?
Let’s go serverless!
GraalVM in the serverless environment

› All current Frameworks (try to) support Native Images
  › Micronaut
  › Quarkus
  › Spring Boot (with restrictions)

› Amazon Lambda
  › Support for “custom runtimes”
    (= BYOR, bring your own runtime)
Let’s do it!
Serverless – Findings

› The problem is not the “serverless framework”
› Native image should only be created during CI
  › otherwise the build takes way too long
  › *but*, many problem only occur in the native image 😞
› **Long build time** complicates bug finding and fixing
› Fast paced development → outdated documentation
Interesting reads

› [https://www.graalvm.org/docs/] (Official GraalVM Docs)
› [https://jaxenter.de/neuer-compiler-fuer-jvm-graalvm-83252] (Nice overview article)
› [https://github.com/neomatrix369/awesome-graal] (Huge list of Graal resources)
› [https://github.com/chrisseaton/graalvm-ten-things] (Top 10 things to do with GraalVM)
› [https://blog.plan99.net/graal-truffle-134d8f28fb69] (Graal and Truffle)
› [https://medium.com/@mathiasdpunkt/fighting-cold-startup-issues-for-your-kotlin-lambda-with-graalvm-39d19b297730] (Fighting cold startup issues)
› [https://micronaut-projects.github.io/micronaut-aws/latest/guide/#customRuntimes] (Micronaut AWS custom GraalVM native runtimes)
› [https://github.com/spring-projects/spring-framework/wiki/GraalVM-native-image-support] (Spring framework support for GraalVM native images)
› [https://blog.codecentric.de/2019/04/quarkus-macht-java-fit-fuer-die-cloud/] (Quarkus)
› [https://docs.aws.amazon.com/lambda/latest/dg/runtimes-walkthrough.html] (AWS custom runtimes)
› [https://github.com/trivial-pursuit/spring-kotlin-demo/tree/lambda-graal] (Example code, Disruptive Guestbook)
Vielen Dank

Thilo Ognibeni
Software Architect / Software Developer

inovex GmbH
Ludwig-Erhard-Allee 6
76131 Karlsruhe

thilo.ognibeni@inovex.de