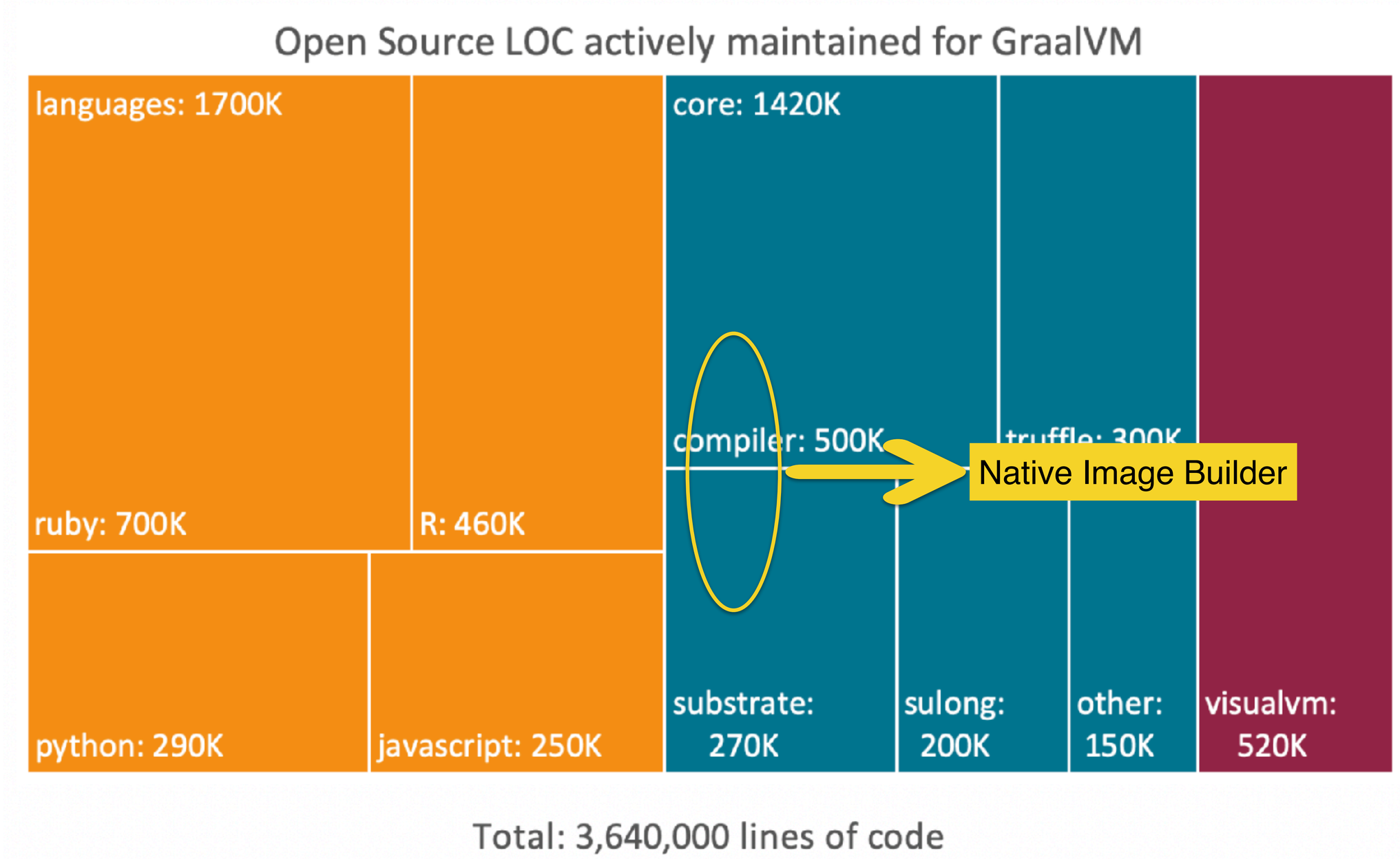
Boost Your Java Application's Performance with Native Images

Elder Moraes

A little bit of context



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

Getting Started

What is a native image?

It’s a technology that uses AOT compilation to create a standalone executable

How a native image is generated?

- Using the Native Image Builder
- It performs a static analysis of all classes (inside a project) and its dependencies

How a native image is generated?

- All methods and libraries used at runtime are mapped
- So it Ahead of Time compiles this code generating a native executable for a specific OS and architecture

What is embedded in a native image?

- Application classes
- Classes dependencies
- Classes of runtime libraries
- Static code native from JDK

Do I need a JVM to run a native image?

Nope!

Serious? How it works?

- Native images have a runtime system called Substrate VM
- Some of its components are: memory management, thread scheduling, GC, etc.

Native Image vs JVM

Native Image vs JVM

Faster startup

- No classloading (all classes were loaded, linked and initialized)
- No code being interpreted
- No JIT
- Part of the heap were generated during the compilation

Native Image vs JVM

Smaller memory footprint

- No metadata for loaded classes
- No profiling data for JIT optimizations
- No cache for interpreted code
- No JIT structures

So… why not using it for everything?

Because there are no silver bullets!

Why not using native images for everything?

There’s no support for:

- JVM TI (Java Virtual Machine Tool Interface)
- Java Agents
- JMX (Java Management Extensions)

Why not using native images for everything?

The bigger the heap, the worst the performance:

- GC is performed by the SerialGC available at SubstrateVM - very limited if compared to a "full" GC
- If the heap is huge, the GC break the application performance

Why not using native images for everything?

There’s no runtime optimization

- No JIT
- All optimizations made during the compile time are final

Why not using native images for everything?

There’s no support for head dump and thread dump

- Or there is… but only for Enterprise Edition

When should I use native images?

When should I use native images?

When your application:

- Is small
- Can be invoked multiples times in a short period of time
- Has a short life cycle
- Ex: CLI apps and serverless functions

Quarkus & Native Images

Quarkus & Native Images

- All Quarkus optimizations are*__independent__* of using native images or not
- But, if you want to use it, Quarkus is 100% compatible (including its extensions)
- You only need the Native Image Builder

Thank you