ORACLE®

GraalVM™ : Run 2. Programs 1. Faster 3. Anywhere

Yudi Zheng

Senior Researcher
Graal Compiler Team, Oracle Labs
Apr. 2019

ORACLE

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, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

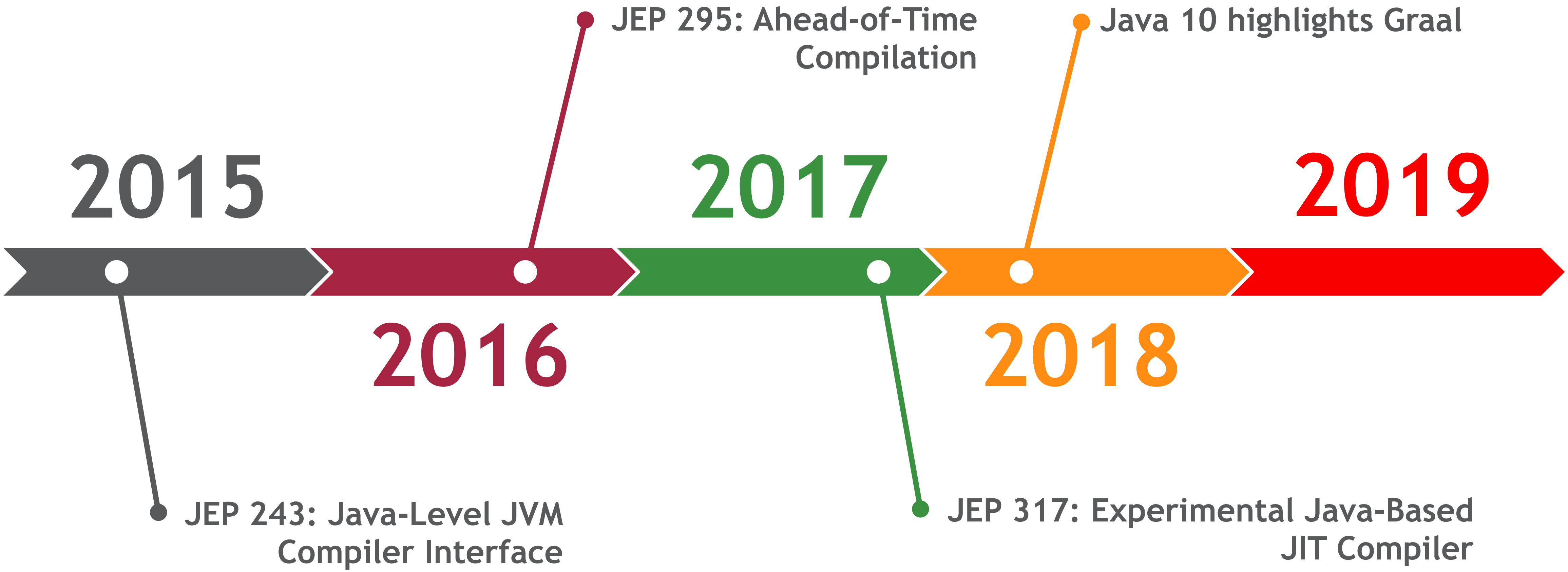
About Me

- Yudi Zheng (郑雨迪)
- yudi.zheng@oracle.com
- Graal compiler team, Oracle Labs Zurich
- Research area: dynamic compilation, program analysis

GraalVM™

GraalVM™ x OpenJDK™

GraalVM™ x OpenJDK™



2015

JEP 243: Java-Level JVM Compiler Interface

2016

JEP 295: Ahead-of-Time Compilation

2017

JEP 317: Experimental Java-Based JIT Compiler

2018

Java 10 highlights Graal

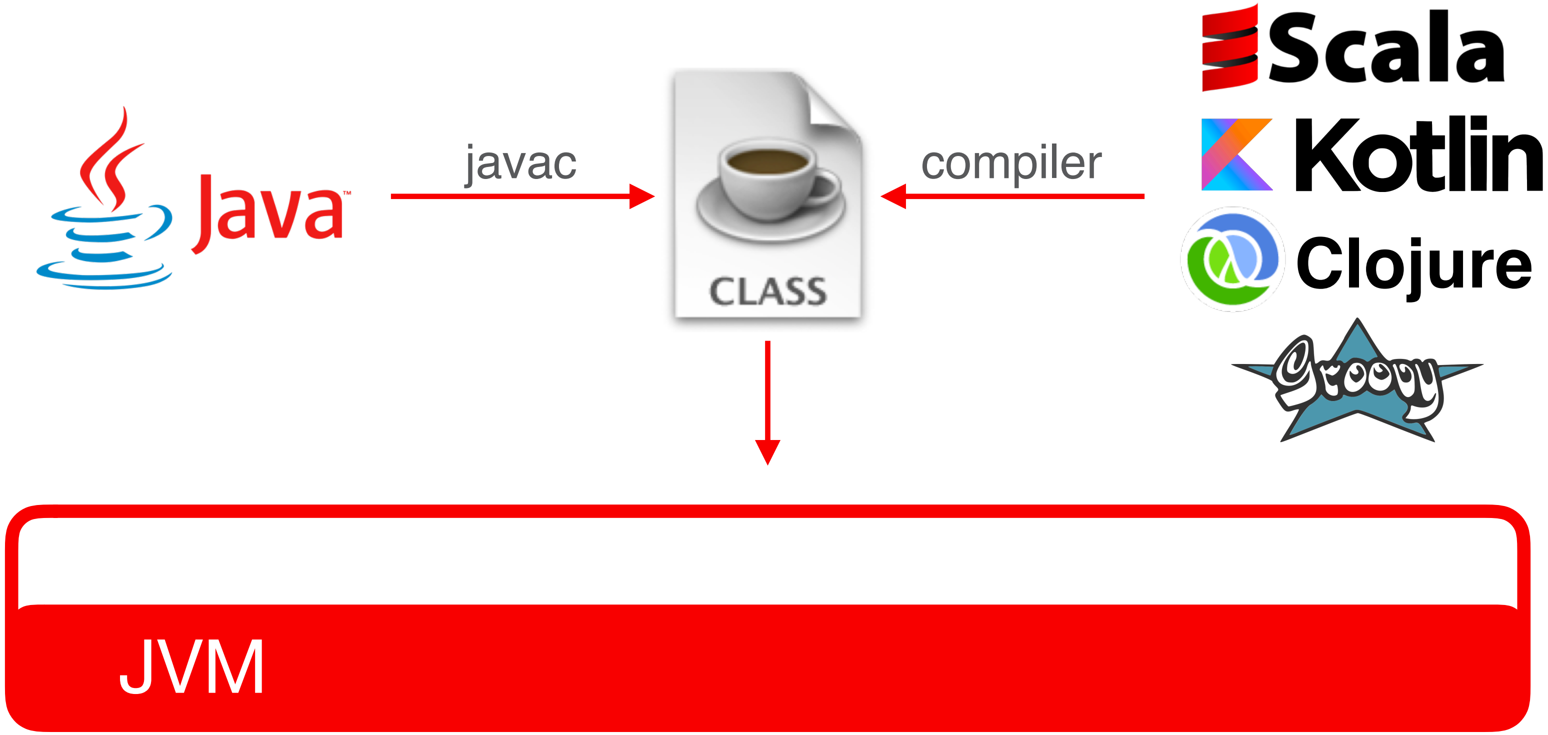
2019

“We’ll start with the ~~soup~~ Java again.”

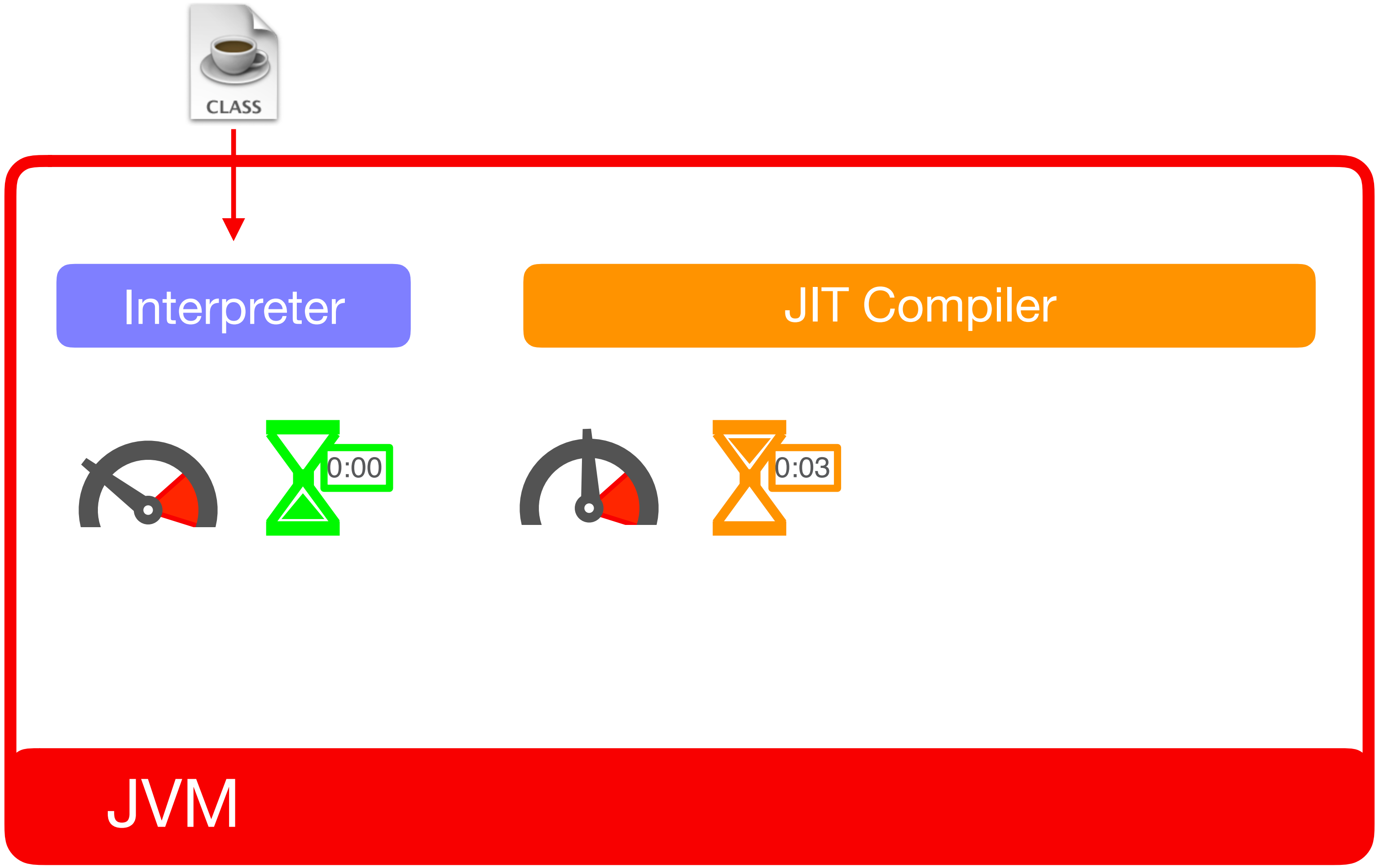
ORACLE

Java Execution in a Nutshell

Java Execution in a Nutshell

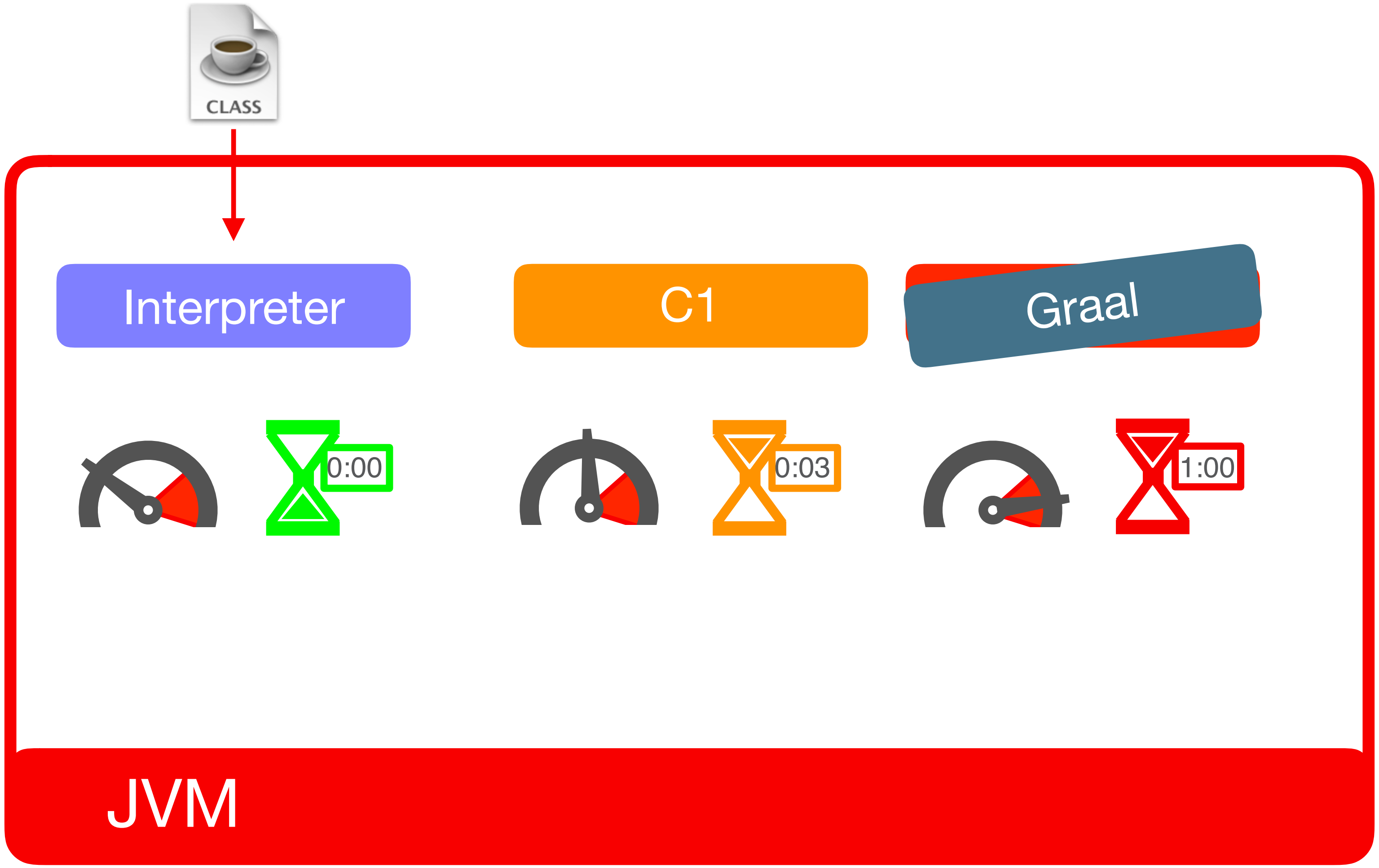


Java Execution in a Nutshell

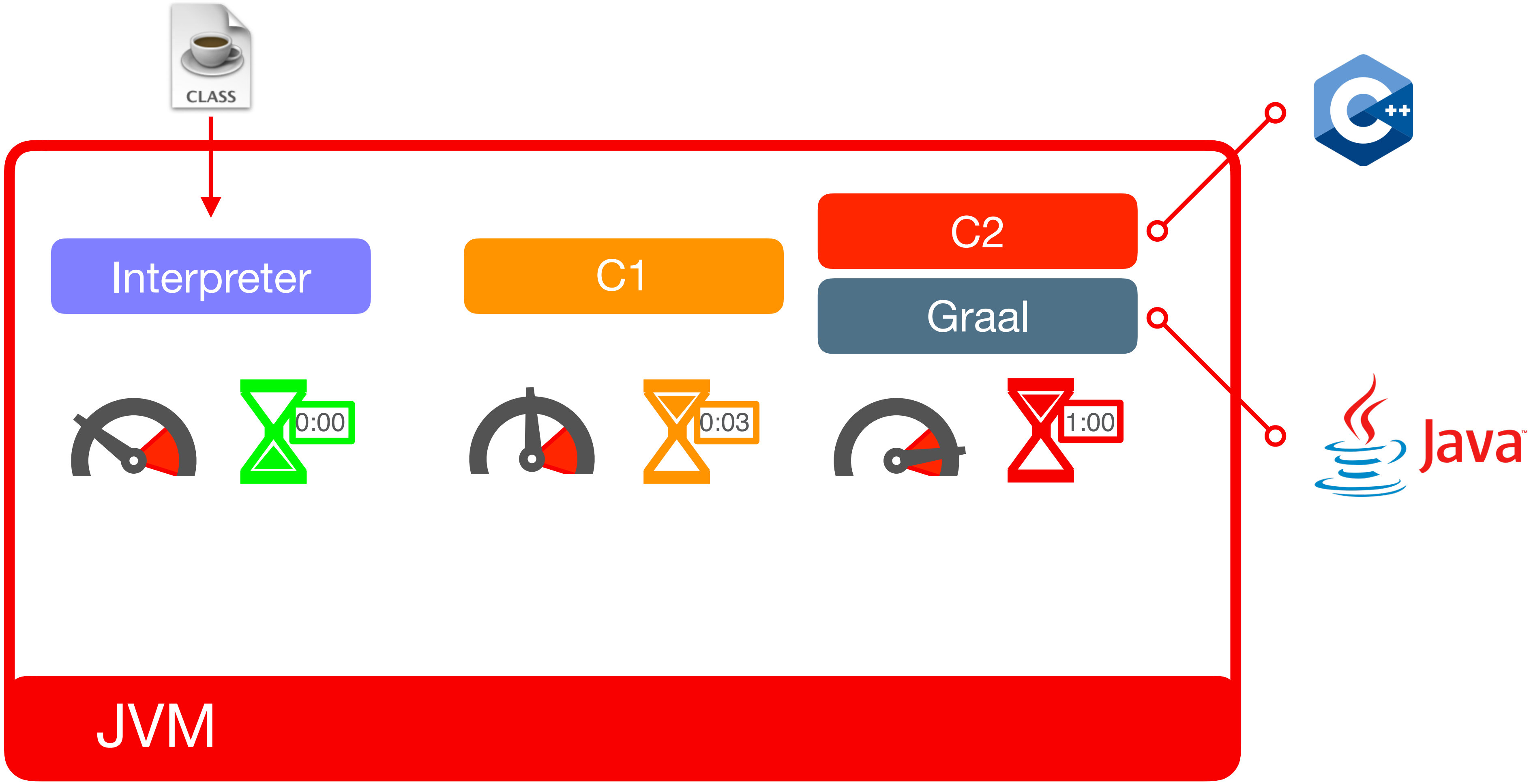


Copyright © 2019, Oracle and/or its affiliates. All rights reserved.

Java Execution in a Nutshell



Java Execution in a Nutshell



Graal & C2: Escape Analysis

```
class Add {
  int x, y;
  Add(int x, int y) {
    this.x=x; this.y=y;
  }
  int apply() {return x+y;}
}
```

```
int add(int x, int y) {
  return new Add(x, y).apply();
}
```

→

```
int add(int x, int y) {
  return x+y;
}
```

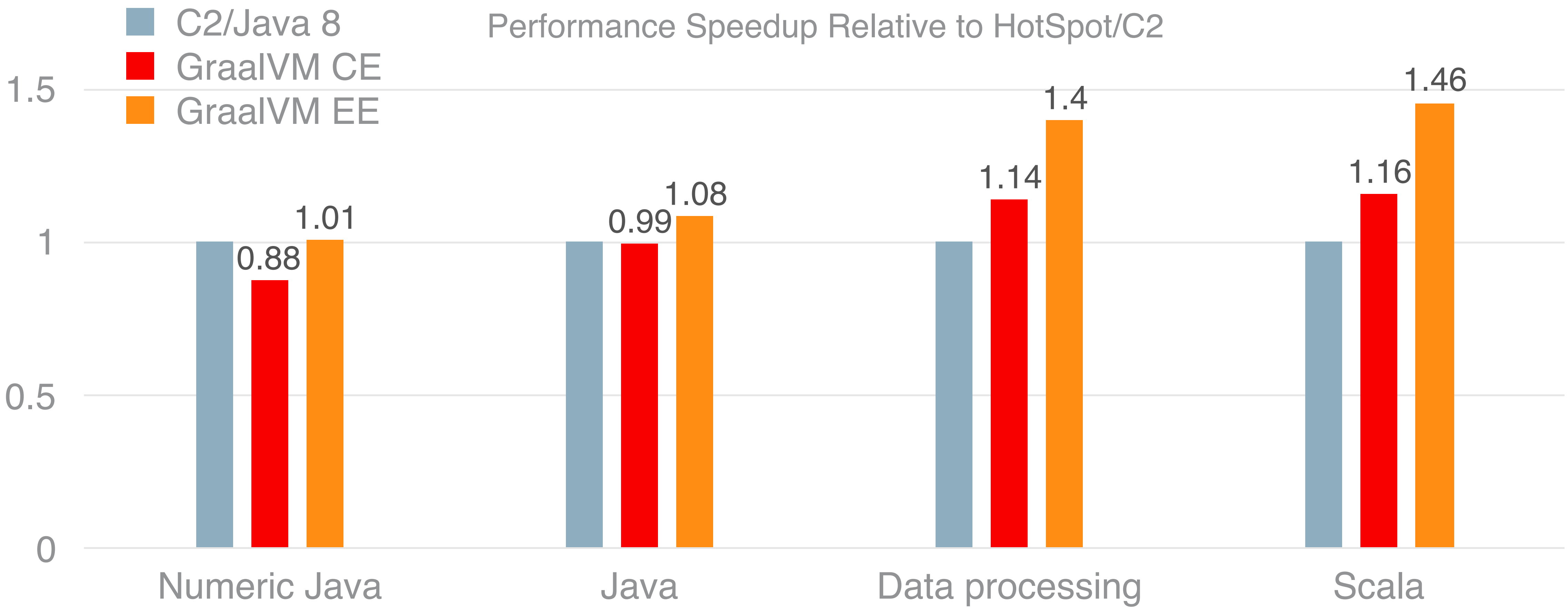
Graal-Only: Partial Escape Analysis

```java
static Add cachedOp;

int add(int x, int y,
        boolean cond) {
  Add op = new Add(x, y);
  if (cond)
    cachedOp = op;
  return op.apply();
}
```

→

```java
static Add cachedOp;

int add(int x, int y,
        boolean cond) {
  if (cond){
    Add op = new Add(x, y);
    cachedOp = op;
  }
  return x + y;
}
```

Peak Performance: Graal as JIT Compiler

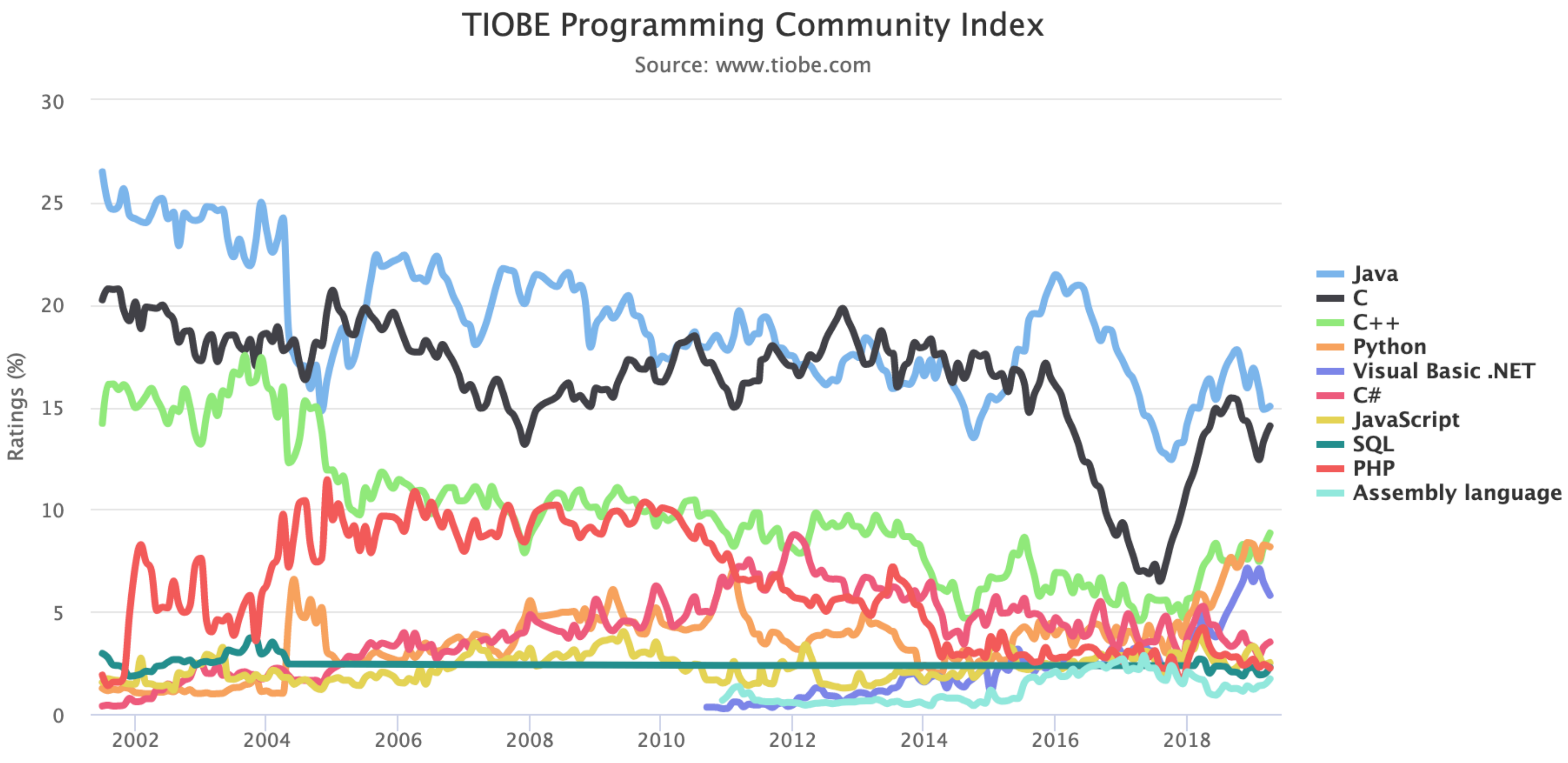


* Numeric Java: numeric benchmarks from SPECjvm2008
* Java: real-life applications from SPECjvm2008, DaCapo, Renaissance
* Data processing: Java8 streams, Spark, ML algorithms from Renaissance
* Scala: Scala applications from ScalaBench, Renaissance

in Java

“Can’t I stay a little longer?
I’m so happy here.”

The War of Programming Languages

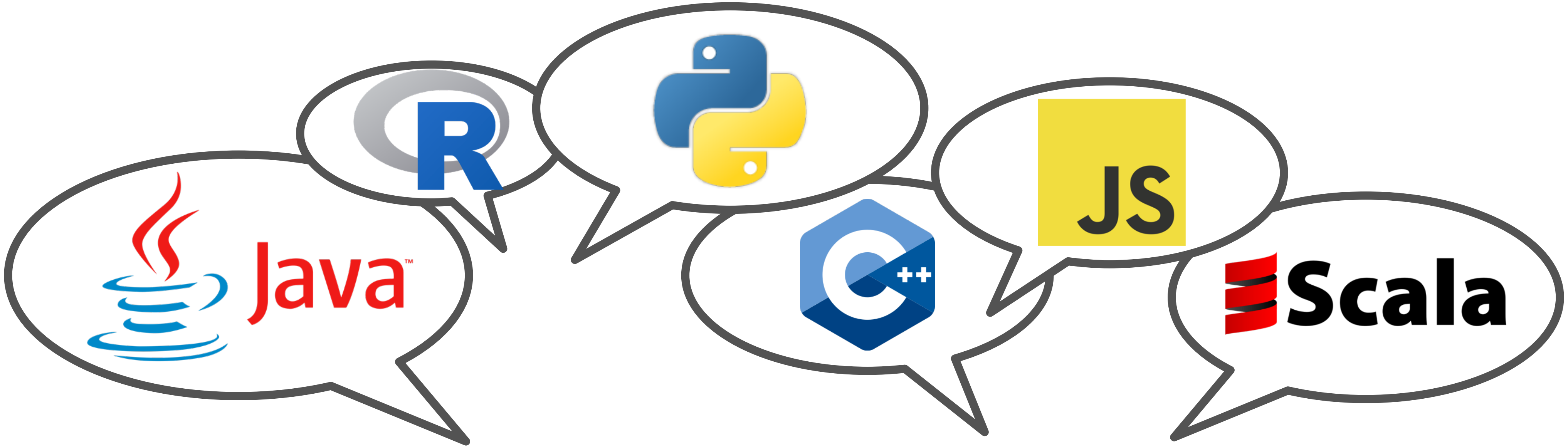


ORACLE

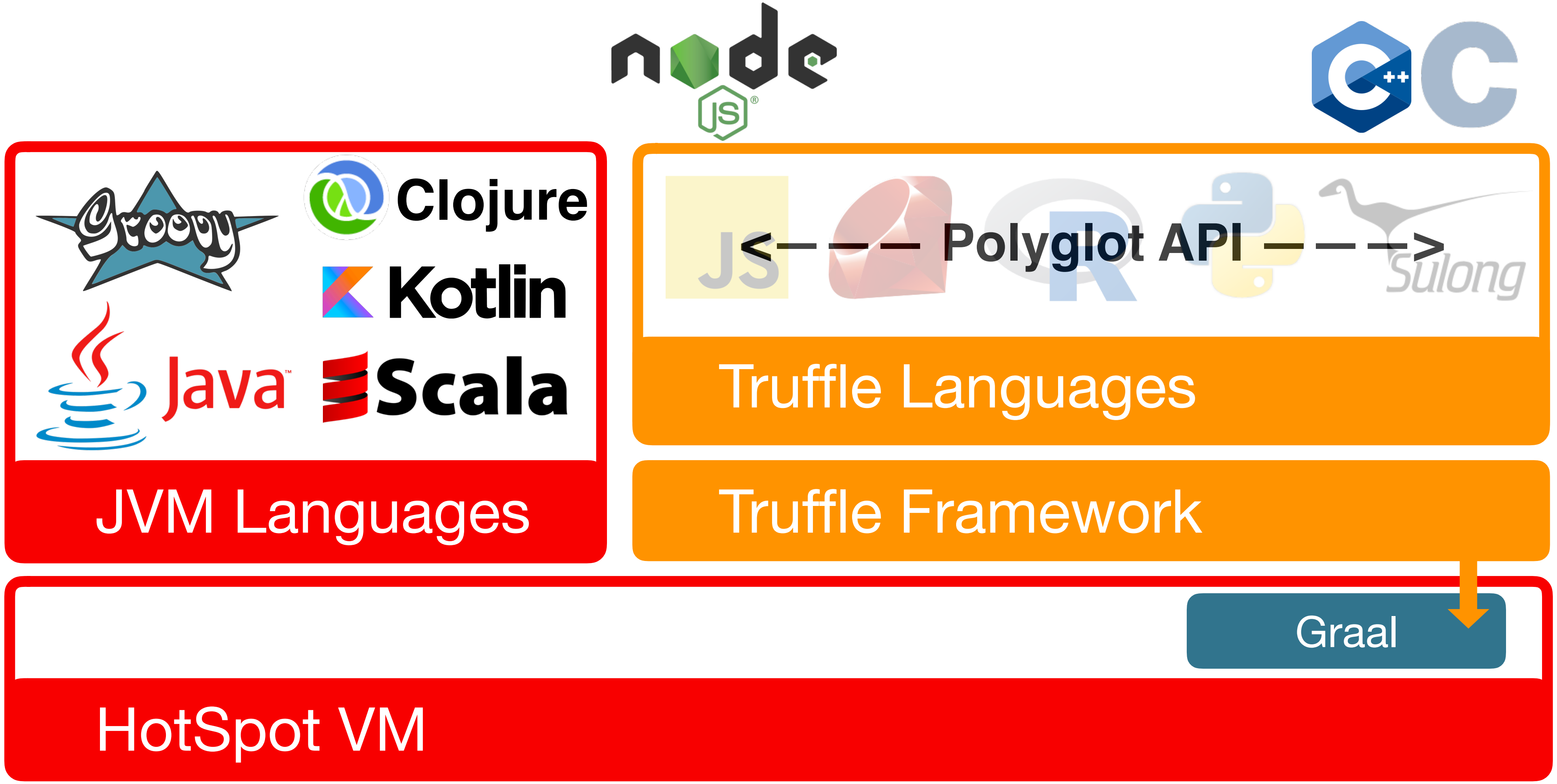
Enhance Your Favorite Language…

- Project Amber
  - JEP 286: Local-Variable Type Inference (JDK 10)
  - JEP 325: Switch Expressions (preview, JDK 12)
- Project Loom: Fibers, Continuations and Tail-Calls for the JVM
- Project Panama: Interconnecting JVM and native code

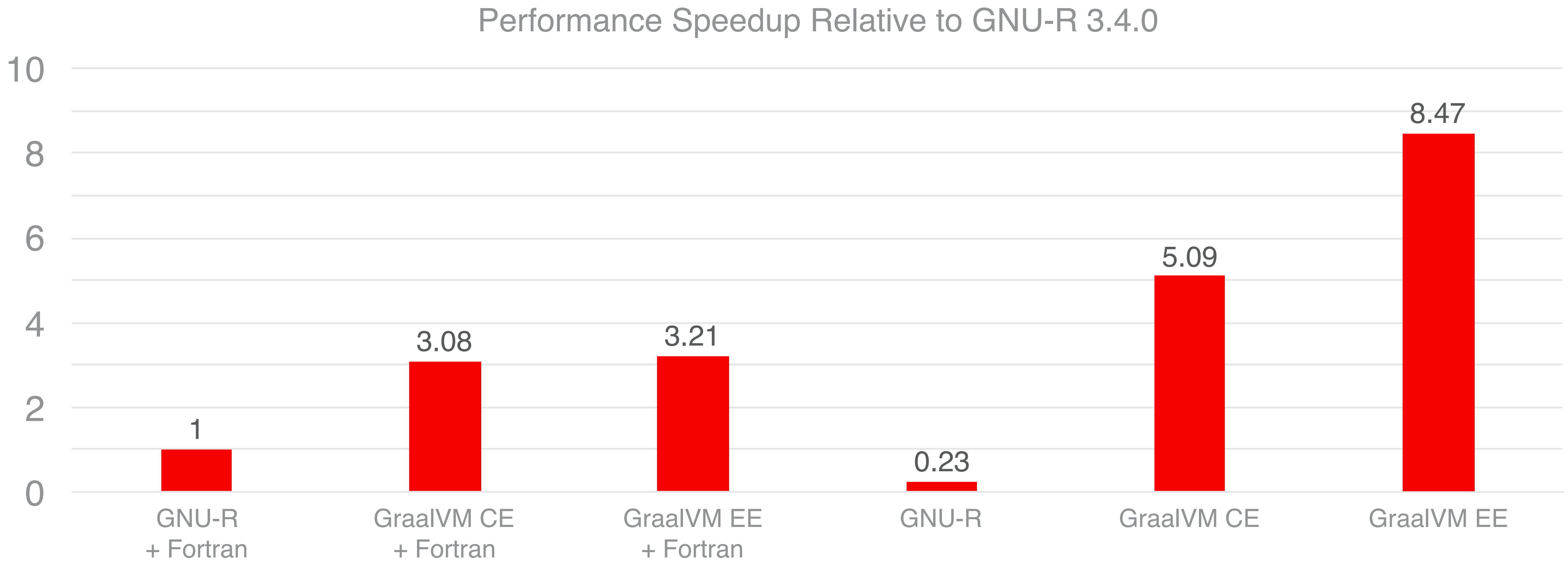
… or Polyglot!



GraalVM™ and Its Ecosystem



Performance: Raytracing on R

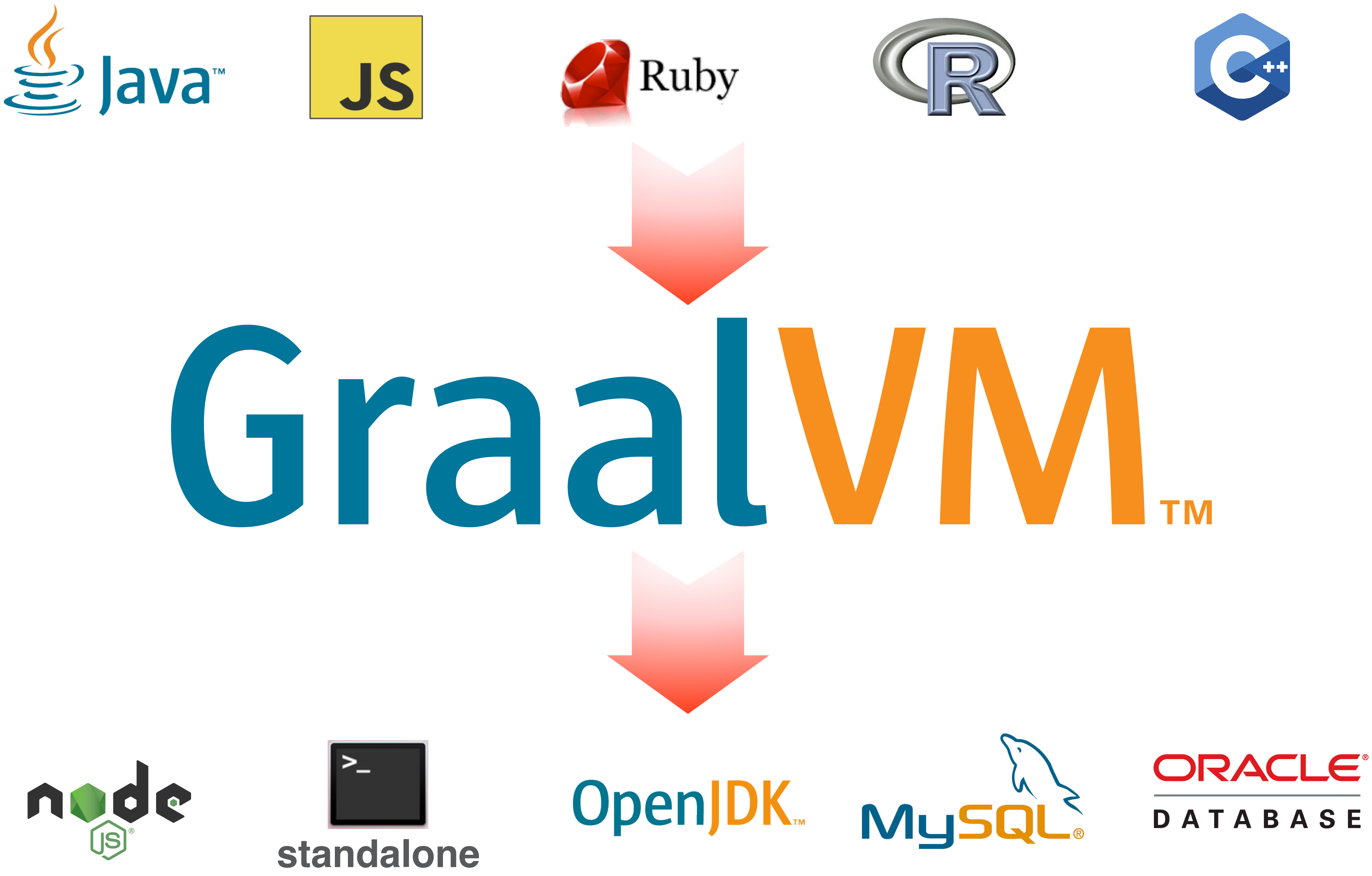


* ref https://medium.com/graalvm/faster-r-with-fastr-4b8db0e0dceb

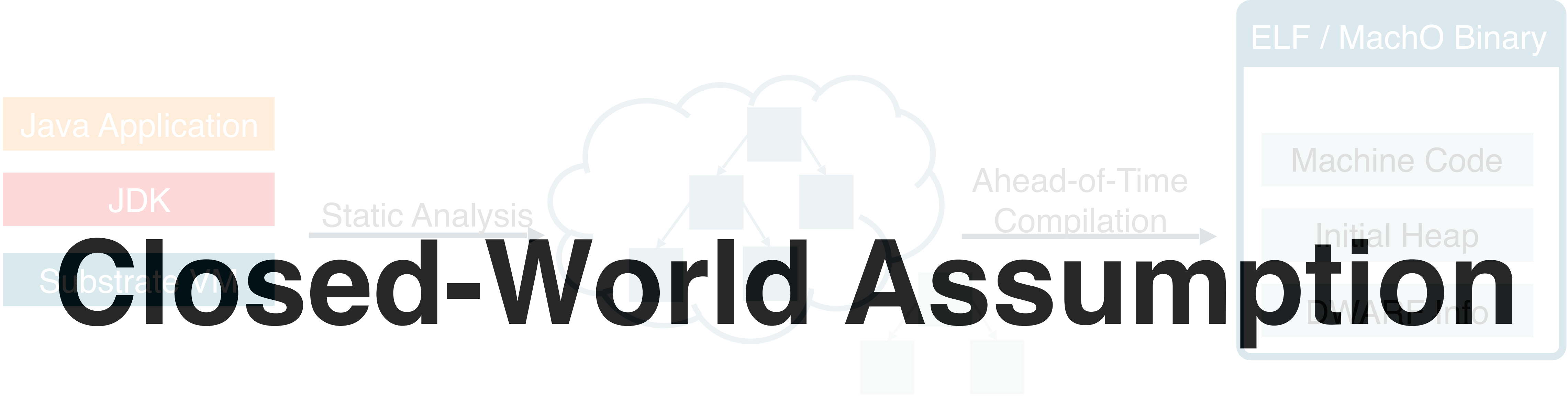
JVM

“Now is your chance to escape!”

ORACLE


Java™
JS
Ruby
R
C++
GraalVM™
node JS
standalone
OpenJDK™
MySQL®
ORACLE® DATABASE

Substrate VM



Java Application

JDK

Substrate VM

Static Analysis

Ahead-of-Time Compilation

ELF / MachO Binary

Machine Code

Initial Heap

DWARF Info

All Java classes from application, JDK, and Substrate VM

Reachable methods, fields, and classes

Application running without dependency on JDK and without Java class loading

Closed-World Assumption

"Hello World" in C, Java

	C	Java/JVM	Java/SVM
🕒	<10 ms	40 ms	<10 ms
RAM	450 KB	24 MB	850 KB
101 011	100 K	140 M	220 K

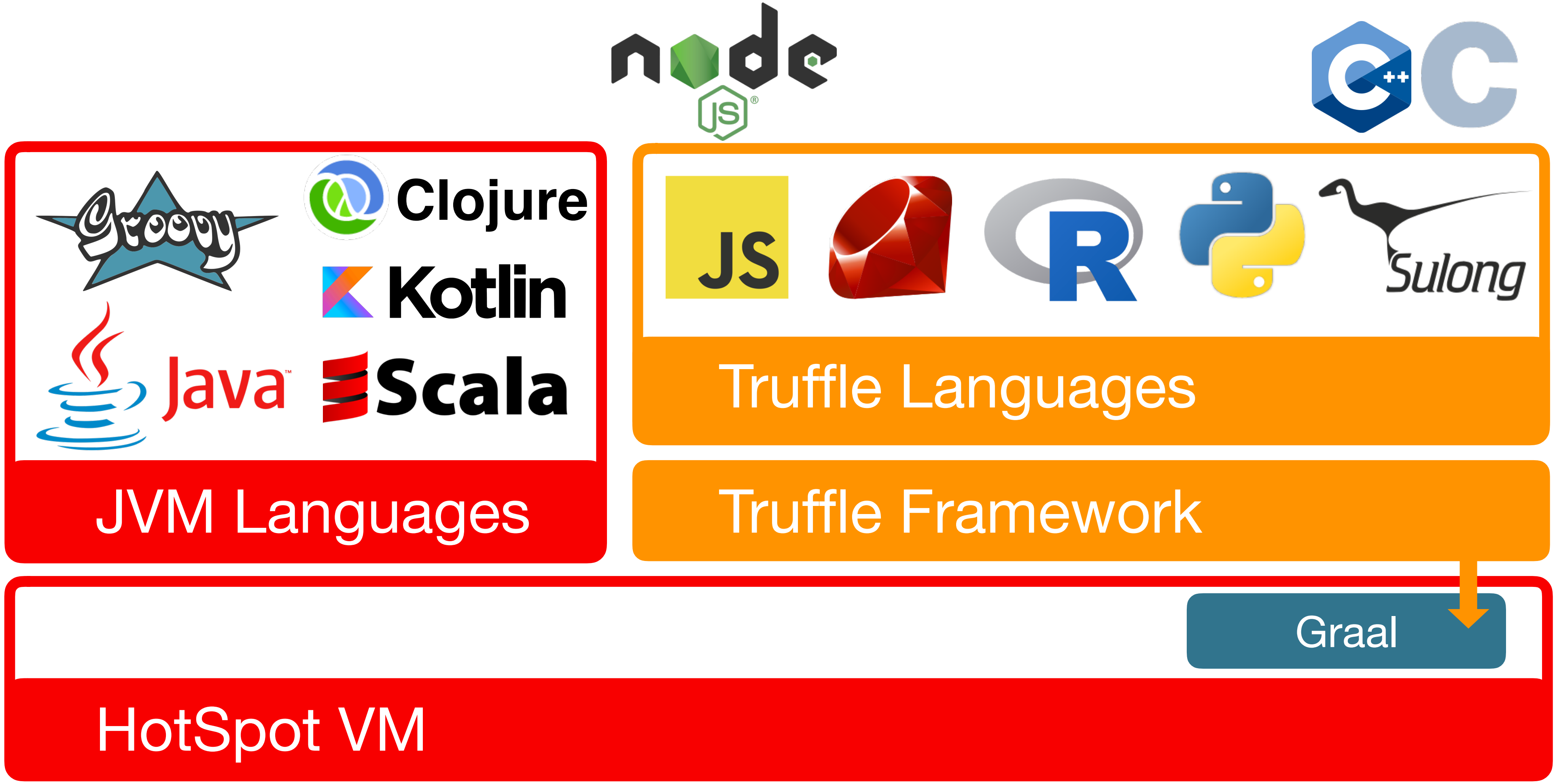
* *Operating system*: **Linux**; *Time,* Memory: **/usr/bin/time** ... ; *Instructions*: **valgrind** --tool=callgrind ...;

"Hello World" in JavaScript

	V8	Nashorn	SVM
🕒	<10 ms	450 ms	<10 ms
RAM	18 MB	56 MB	4 MB
101 011	10 M	N/A	520 K

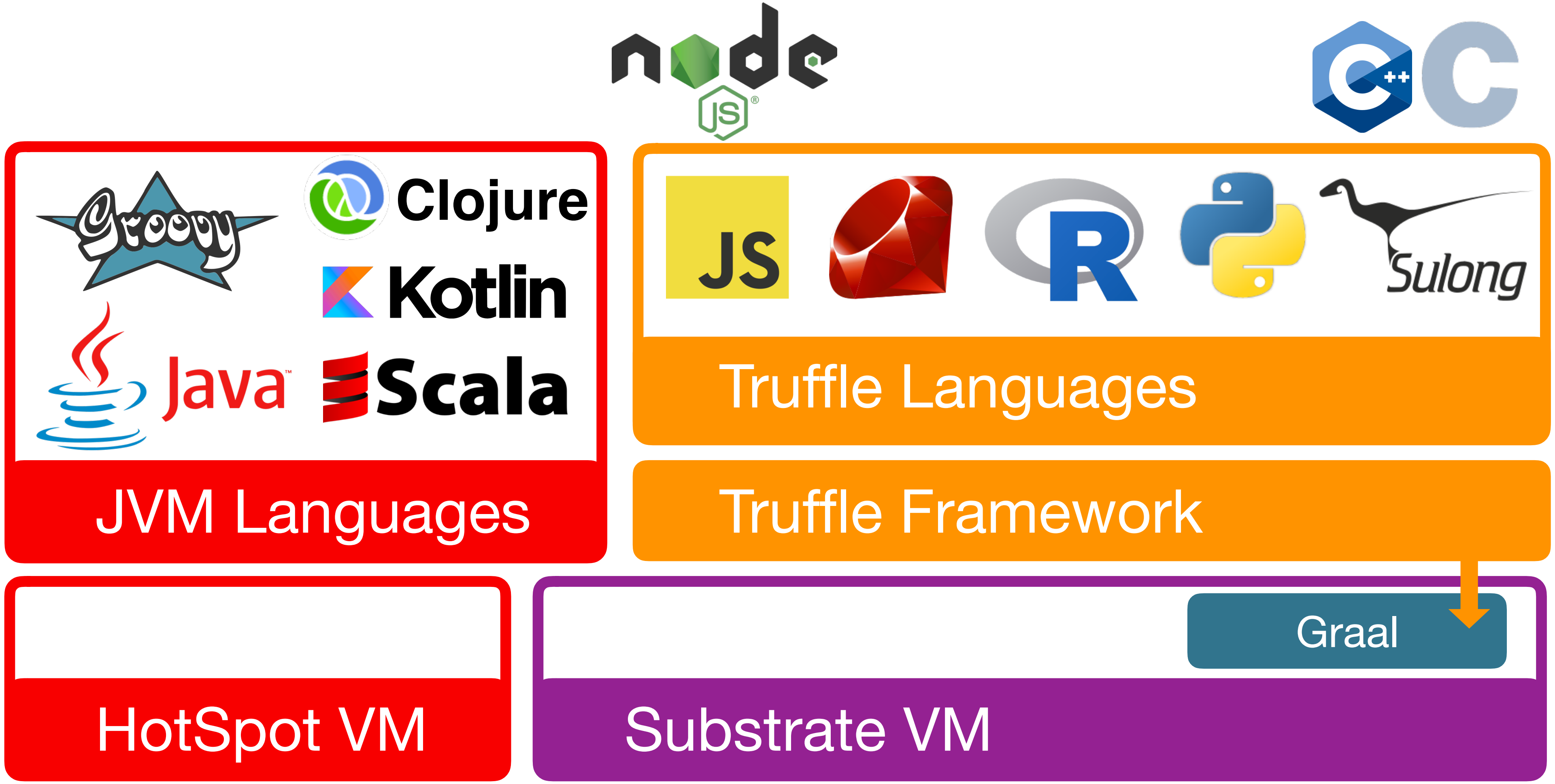
* *Operating system*: **Linux**; *Time,* Memory: **/usr/bin/time** ... ; *Instructions*: **valgrind** --tool=callgrind ...;

GraalVM™ and Its Ecosystem

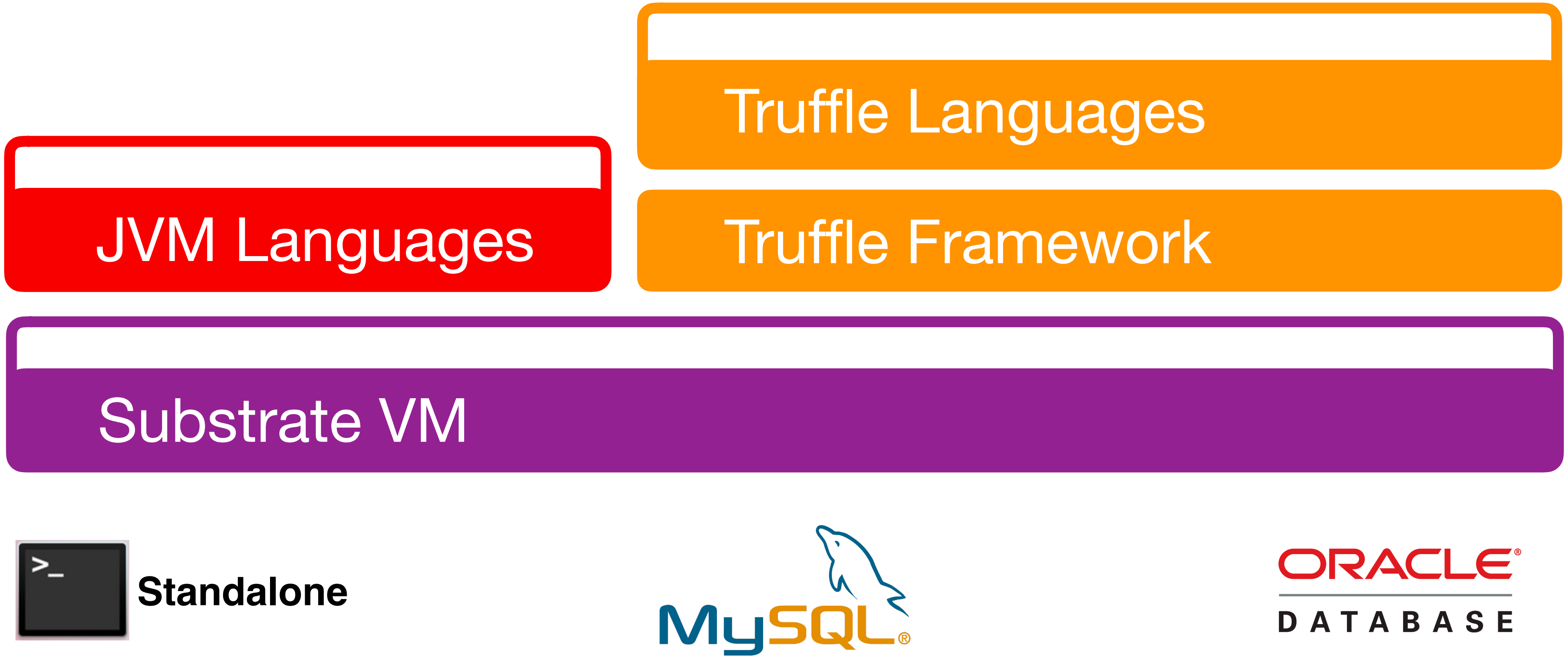


ORACLE

GraalVM™ and Its Ecosystem



GraalVM™ and Its Ecosystem



Truffle Languages

JVM Languages

Truffle Framework

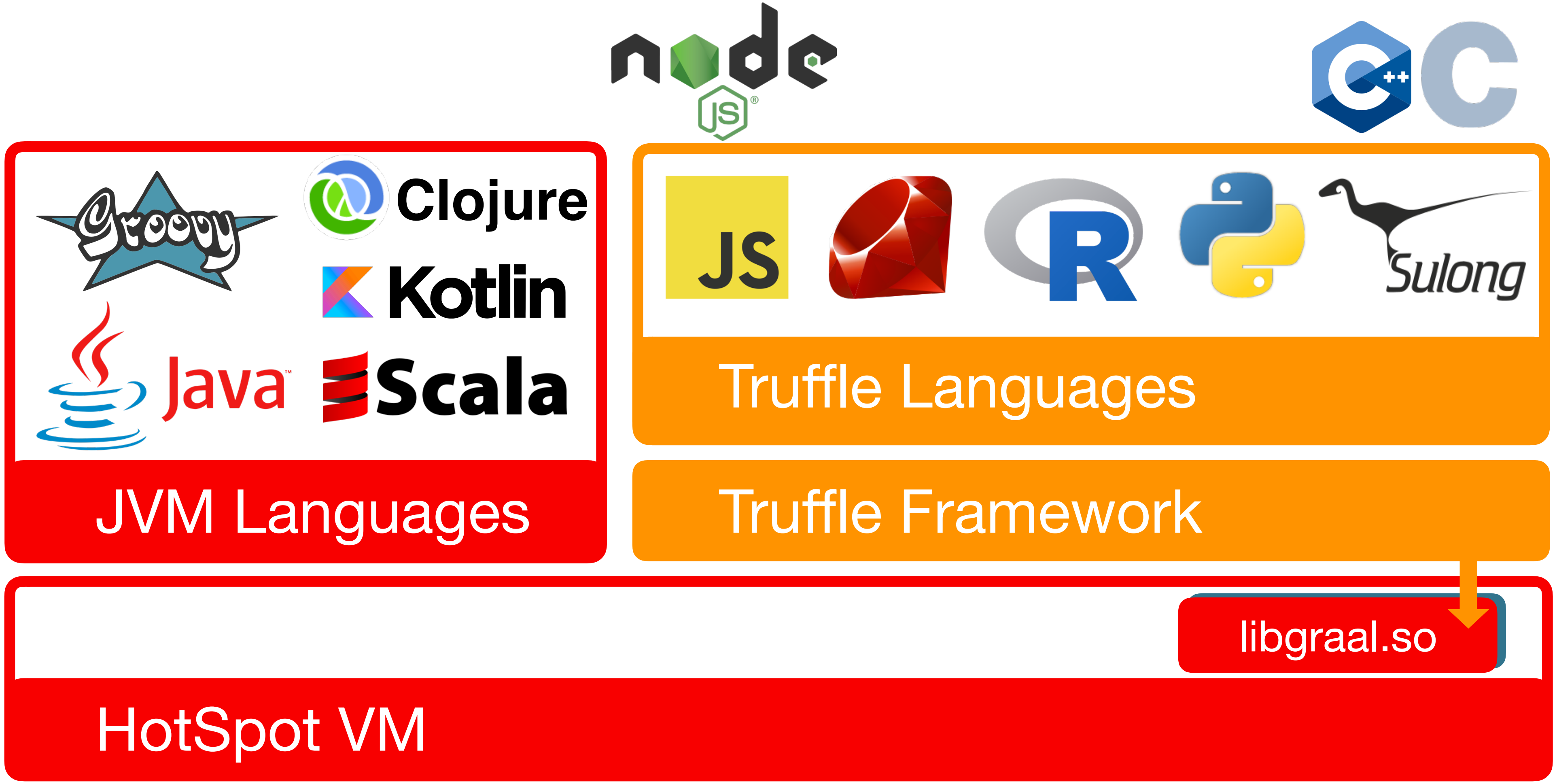
Substrate VM

Standalone

MySQL®

ORACLE® DATABASE

AOT-Graal: libgraal.so



JVM Languages

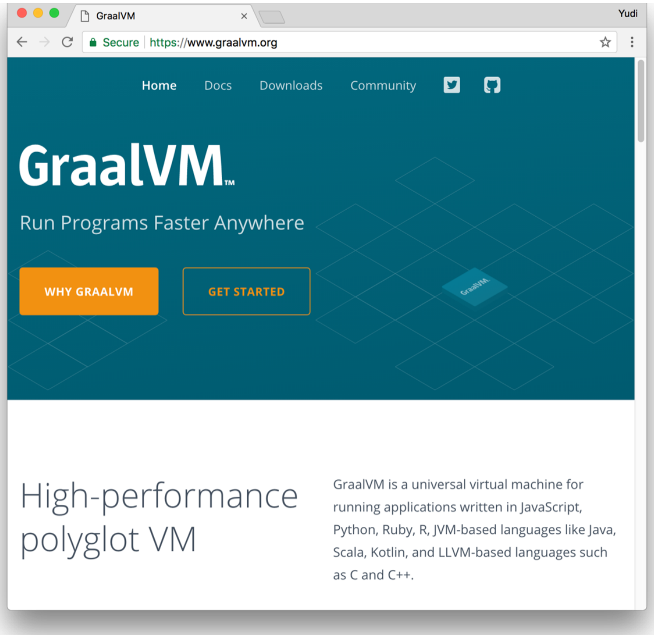
Truffle Languages

Truffle Framework

libgraal.so

HotSpot VM

Get Started!



GraalVM™

Run Programs Faster Anywhere

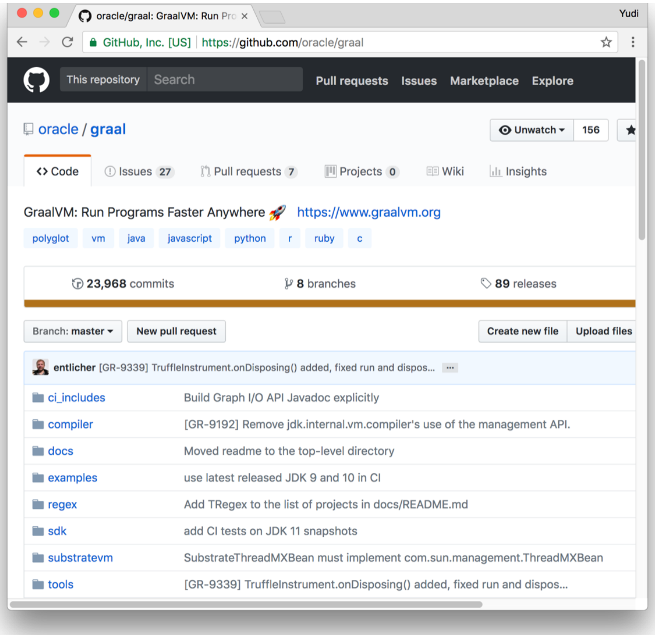
High-performance polyglot VM

GraalVM is a universal virtual machine for running applications written in JavaScript, Python, Ruby, R, JVM-based languages like Java, Scala, Kotlin, and LLVM-based languages such as C and C++.

Documentation

https://www.graalvm.org/

https://medium.com/graalvm



oracle / graal

GraalVM: Run Programs Faster Anywhere 🚀 https://www.graalvm.org

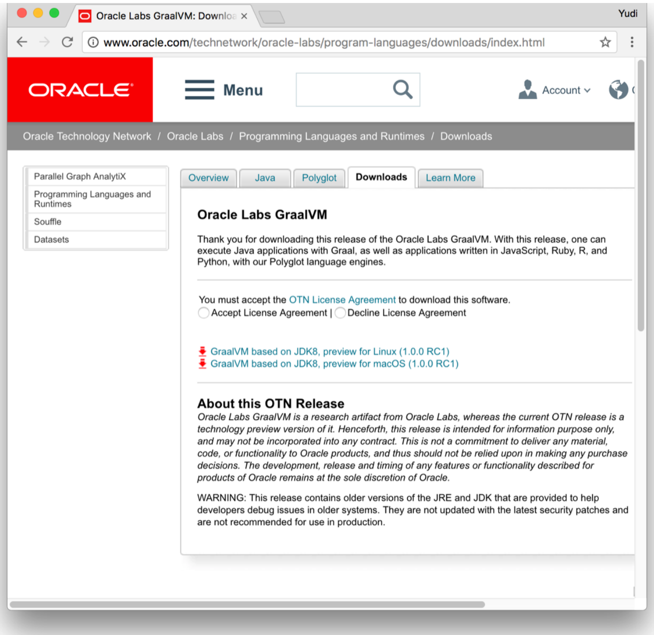
23,968 commits · 8 branches · 89 releases

entlicher [GR-9339] TruffleInstrument.onDisposing() added, fixed run and dispos...

ci_includes	Build Graph I/O API Javadoc explicitly
compiler	[GR-9192] Remove jdk.internal.vm.compiler's use of the management API.
docs	Moved readme to the top-level directory
examples	use latest released JDK 9 and 10 in CI
regex	Add TRegex to the list of projects in docs/README.md
sdk	add CI tests on JDK 11 snapshots
substratevm	SubstrateThreadMXBean must implement com.sun.management.ThreadMXBean
tools	[GR-9339] TruffleInstrument.onDisposing() added, fixed run and dispos...

Open source on GitHub

https://github.com/oracle/graal


Oracle Labs GraalVM: Downlo
Yudi
www.oracle.com/technetwork/oracle-labs/program-languages/downloads/index.html
ORACLE
Menu
Account
Oracle Technology Network / Oracle Labs / Programming Languages and Runtimes / Downloads
Parallel Graph AnalytiX
Programming Languages and Runtimes
Souffle
Datasets
Overview
Java
Polyglot
Downloads
Learn More

Oracle Labs GraalVM

Thank you for downloading this release of the Oracle Labs GraalVM. With this release, one can execute Java applications with Graal, as well as applications written in JavaScript, Ruby, R, and Python, with our Polyglot language engines.

You must accept the OTN License Agreement to download this software.
○ Accept License Agreement | ○ Decline License Agreement

GraalVM based on JDK8, preview for Linux (1.0.0 RC1)
GraalVM based on JDK8, preview for macOS (1.0.0 RC1)

About this OTN Release

Oracle Labs GraalVM is a research artifact from Oracle Labs, whereas the current OTN release is a technology preview version of it. Henceforth, this release is intended for information purpose only, and may not be incorporated into any contract. This is not a commitment to deliver any material, code, or functionality to Oracle products, and thus should not be relied upon in making any purchase decisions. The development, release and timing of any features or functionality described for products of Oracle remains at the sole discretion of Oracle.

WARNING: This release contains older versions of the JRE and JDK that are provided to help developers debug issues in older systems. They are not updated with the latest security patches and are not recommended for use in production.

Enterprise Release

Search for “OTN Graal”

“Take it easy and avoid excitement.”

ORACLE®