

KARSTEN SILZ

1 MAY 2020

[HTTPS://BPF.LI](https://bpf.li)



ECLIPSE OPENJ9: MEMORY DIET FOR YOUR JVM APPLICATIONS

DO YOU CARE HOW MUCH **MEMORY**
YOUR JVM APPLICATIONS USE?

AND CAN YOU **PICK** YOUR JVM?

ECLIPSE OPENJ9

LESS CONTAINER MEMORY

LOWER COST

IBM had an in-house JVM called "J9"

It powered IBM's Java products for many years

In 2017, IBM donated J9 to Eclipse as "OpenJ9"

OpenJ9 is drop-in replacement for Oracle HotSpot JDK...

...but not all HotSpot JVM options and not all tools work with OpenJ9!

Get OpenJ9 at [AdoptOpenJDK](https://adoptopenjdk.org/)

Prebuilt OpenJDK Binaries for Free!

Java™ is the world's leading programming language and platform. AdoptOpenJDK uses [infrastructure](#), [build](#) and [test](#) scripts to produce prebuilt binaries from [OpenJDK™](#) class libraries and a choice of either the [OpenJDK HotSpot](#) or [Eclipse OpenJ9](#) VM. All AdoptOpenJDK binaries and scripts are [open source licensed](#) and available for free.

Download for macOS x64

1. Choose a Version

- ☒ OpenJDK 8 (LTS)
- ☐ OpenJDK 11 (LTS)
- ☐ OpenJDK 14 (Latest)

2. Choose a JVM [Help Me Choose](#)

- ☒ HotSpot
- ☐ OpenJ9



[AdoptOpenJDK](#) Docker Images

- ▶ [Java 8](#)
- ▶ [Java 11](#)
- ▶ [Java 14](#)

BENCHMARKS, PLEASE!

Ran on my MacBook Pro

Benchmarks used Docker Compose

Java Container limited to 2 CPU cores & 2 GB RAM

Java options: `-Xmx1024m -Xms256m`

Additional OpenJ9 option `-Xtune:virtualized:`
lower CPU & memory usage, but also less throughput

Measured CPU & memory with `docker stats`

THESE ARE **NOT** SCIENTIFIC
BENCHMARKS!

THAT'S WHY **YOU** CAN RUN & TWEAK
THEM YOURSELVES FOR YOUR NEEDS!

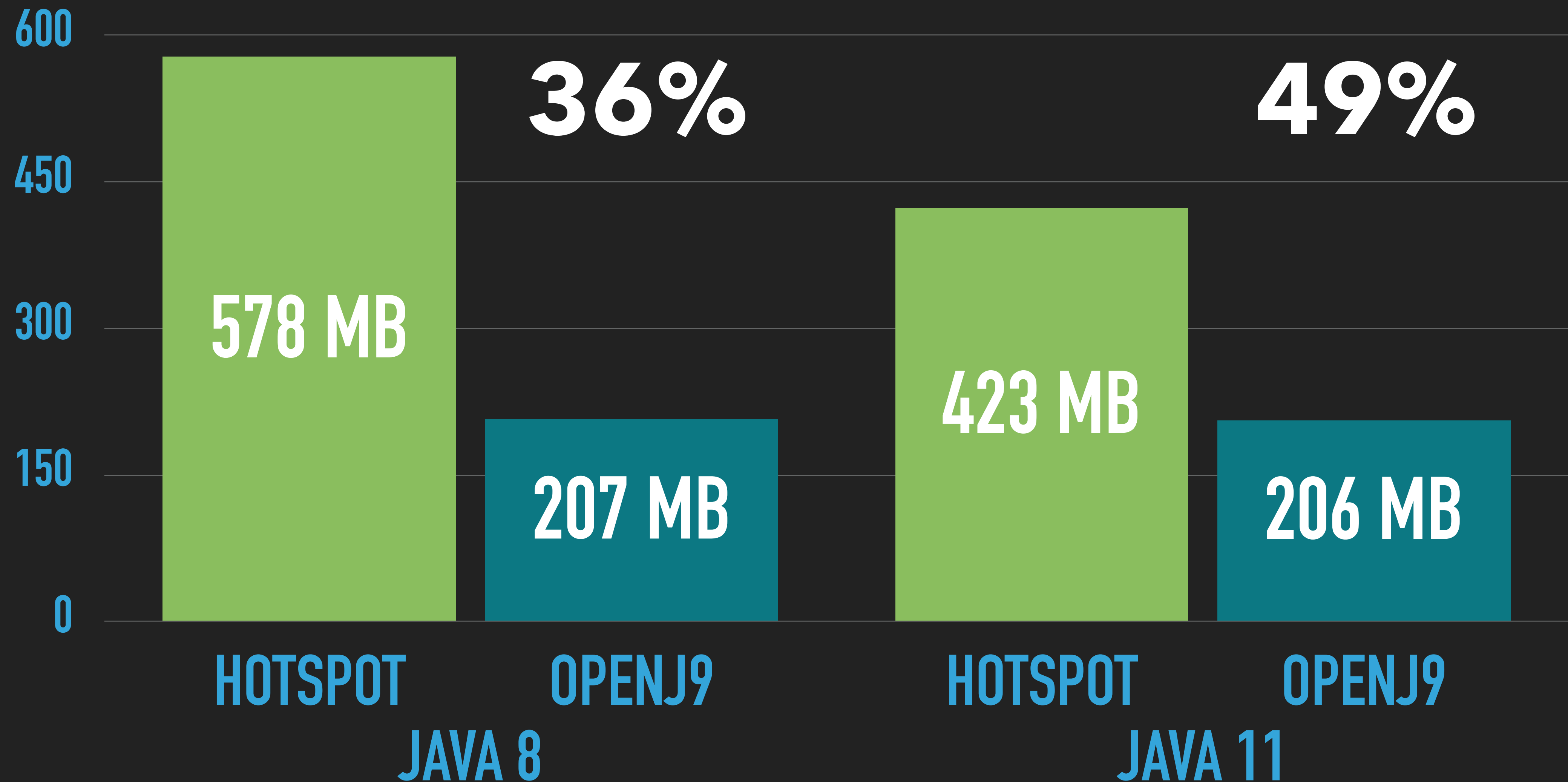
BENCHMARK 1: WEB APPLICATION

Generated with JHipster

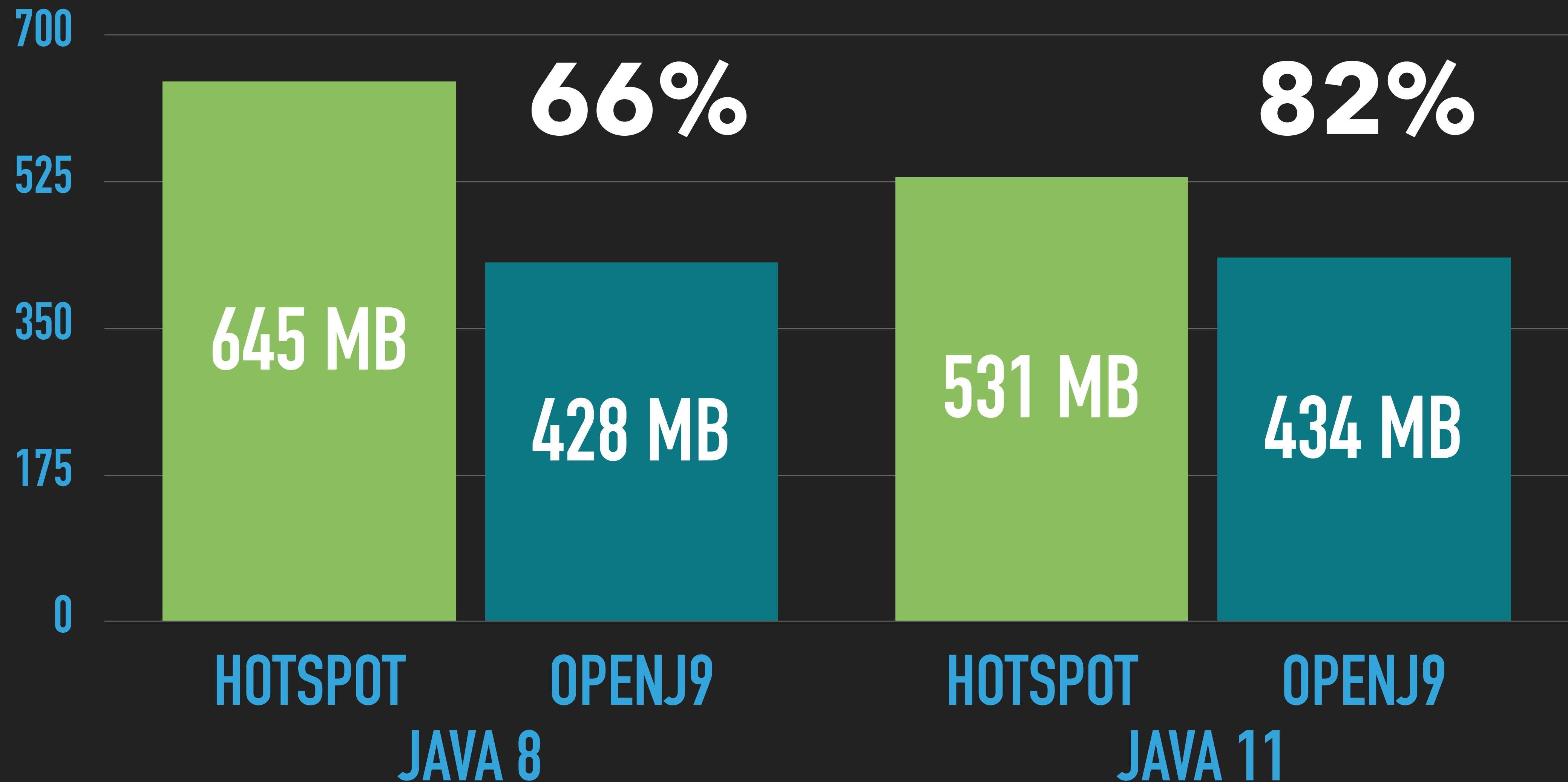
- ▶ Spring Boot 2.2
- ▶ Spring Data JPA with PostgreSQL
- ▶ Angular 9
- ▶ 5 Entity screens, 6 Admin screens

Started & used application 3 times

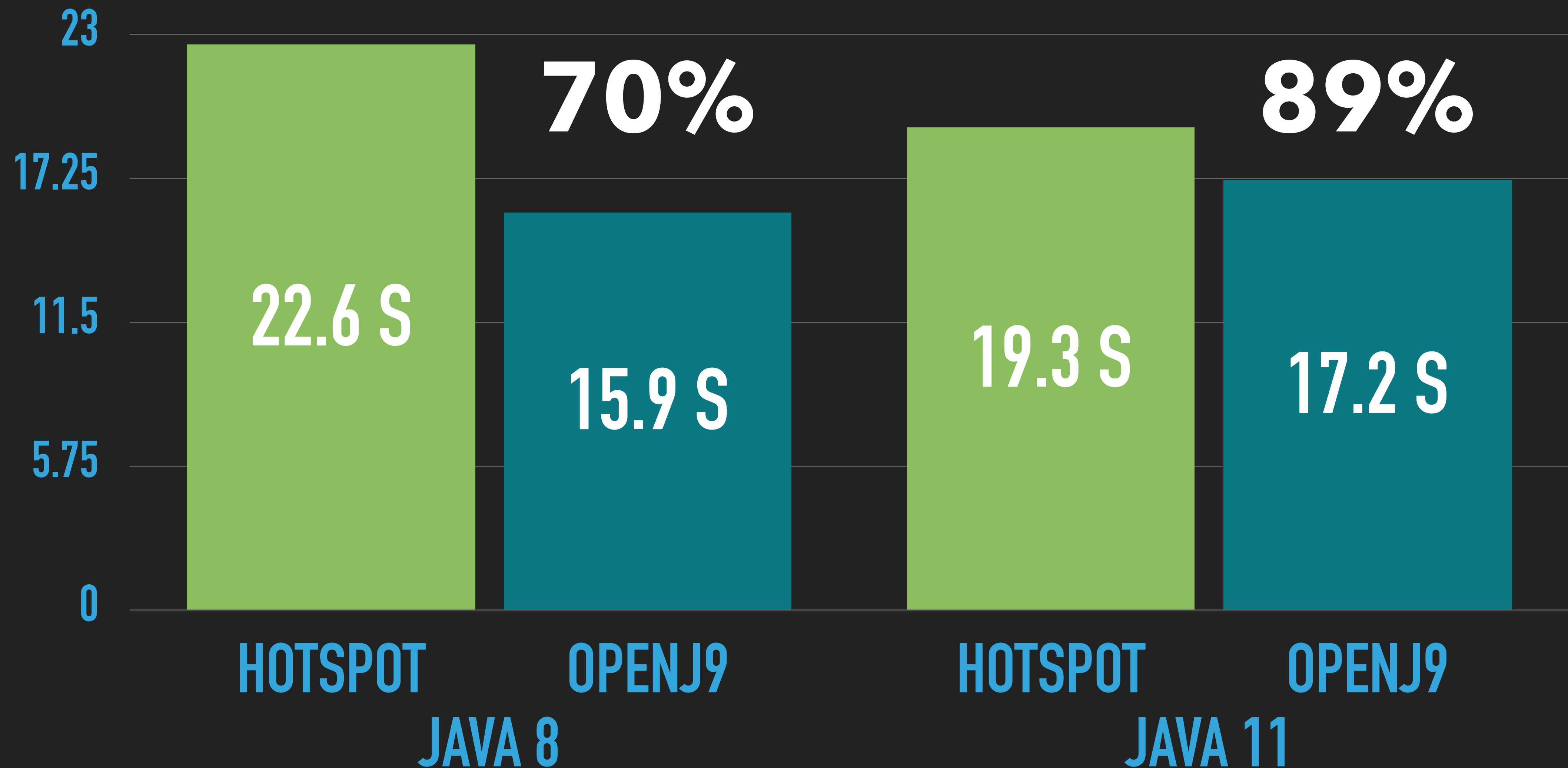
MEMORY CONSUMPTION AFTER START-UP



MEMORY CONSUMPTION AFTER USE



START-UP TIME (SELF-REPORTED, IN SECONDS)



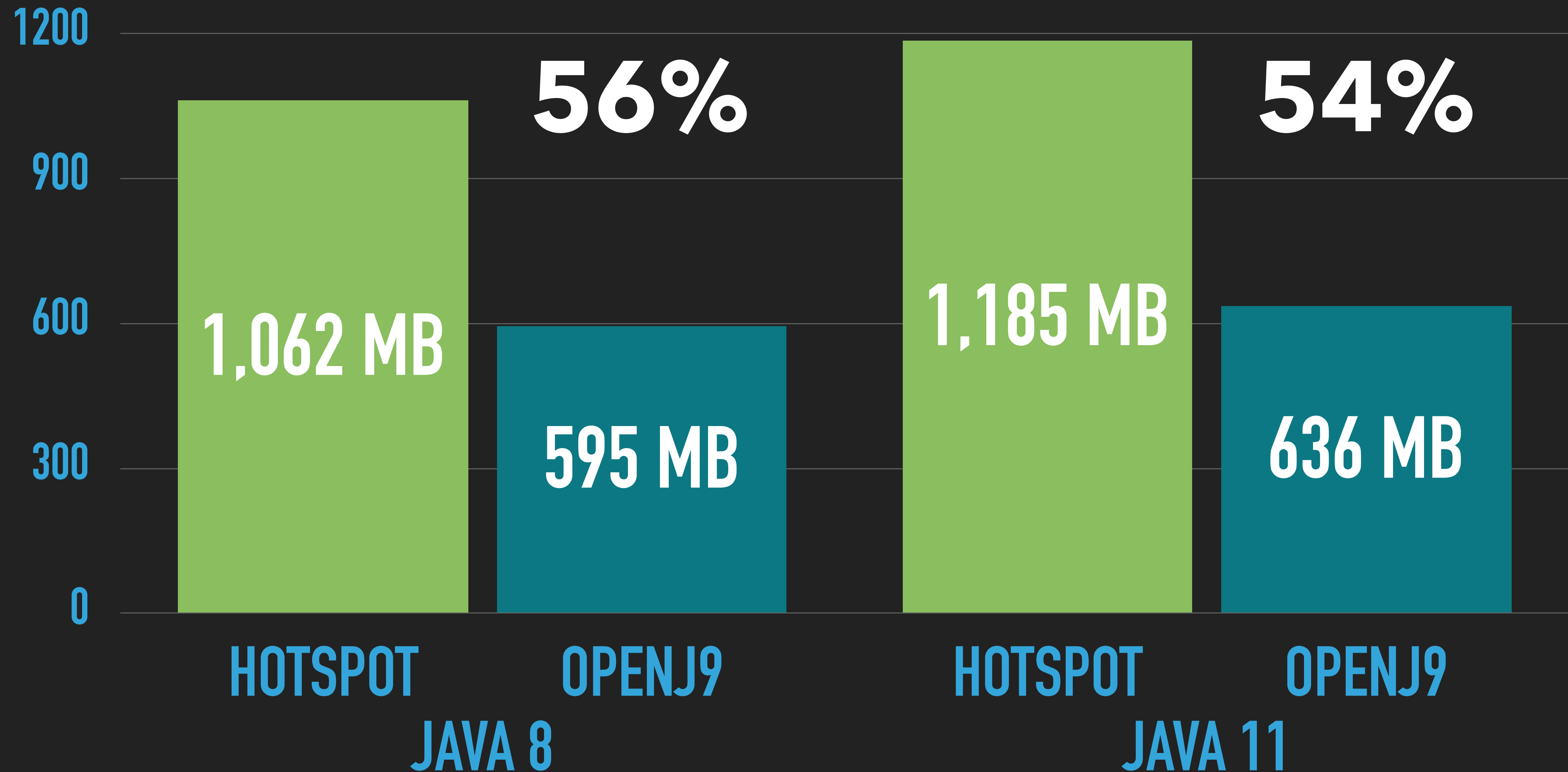
BENCHMARK 2: NUMBER CRUNCHING

Used open source benchmark [Renaissance Suite](#)

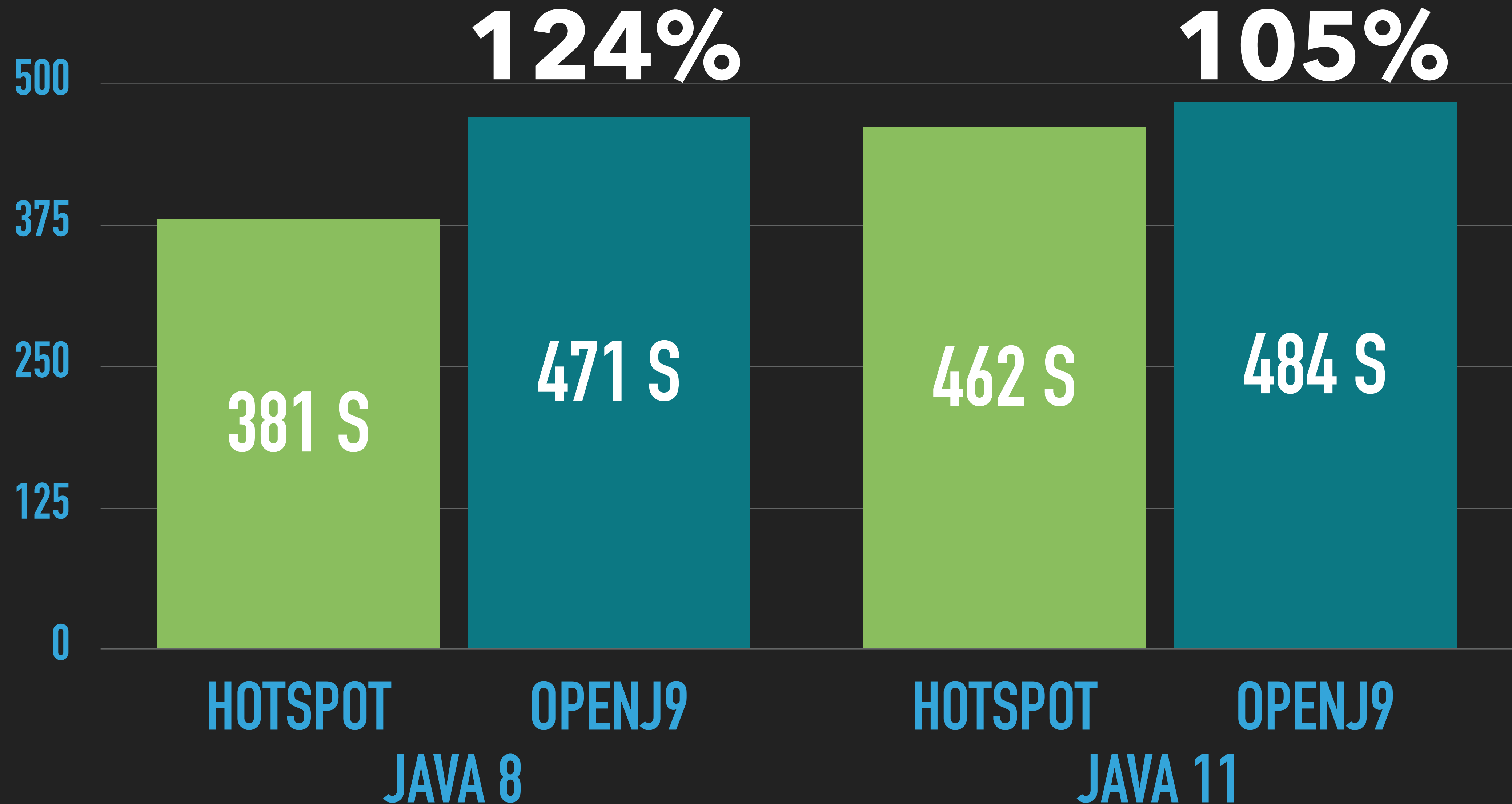
- ▶ Pure Java/Scala code
- ▶ No external database, no web requests

Picked 7 benchmarks, ran them 5 times each

AVERAGE MEMORY USAGE



TOTAL CPU TIME (TIME COMMAND, IN SECONDS)



OPENJ9 USES JUST 40–80%
OF HOTSPOT'S MEMORY

OPENJ9 STARTS FASTER, BUT IS
SLOWER IN NUMBER CRUNCHING

**WHEN CAN YOU NOT
USE OPENJ9?**

Your **tool chain** requires another JVM

Your **licensing model** doesn't allow for OpenJ9

OpenJ9 is **too slow**

ECLIPSE OPENJ9

LESS CONTAINER MEMORY

LOWER COST

SLIDES & SOURCE CODE



[HTTPS://BPF.LI/POJM](https://bpf.li/pojm)

APPENDIX

**HOW TO START WITH
OPENJ9?**

See [this Eclipse help page](#) to get started

- ▶ See [this page](#) and [this one](#) for command line options
- ▶ Read about [garbage collection](#)
- ▶ Read about [diagnostics tools](#)

Tips & Tricks

- ▶ Use `-Xtune:virtualized` when running in Docker
- ▶ [Class Data Sharing](#) and the [Ahead-of-time \(AOT\) compiler](#) will reduce start-up time (also helps local development)

For Java 8 (only?), you can also get OpenJ9 as “IBM Java”

- ▶ For desktop (but not macOS)
- ▶ Docker
- ▶ On **mainframes**, you have to use IBM Java 8 because OpenJDK 8 doesn't have a JIT compiler there

Stability

- ▶ In fall of 2018, OpenJ9 8 on my Mac **crashed** a lot
- ▶ Had **no problems** on my Mac with OpenJ9 11 since 2019