Java in 2019

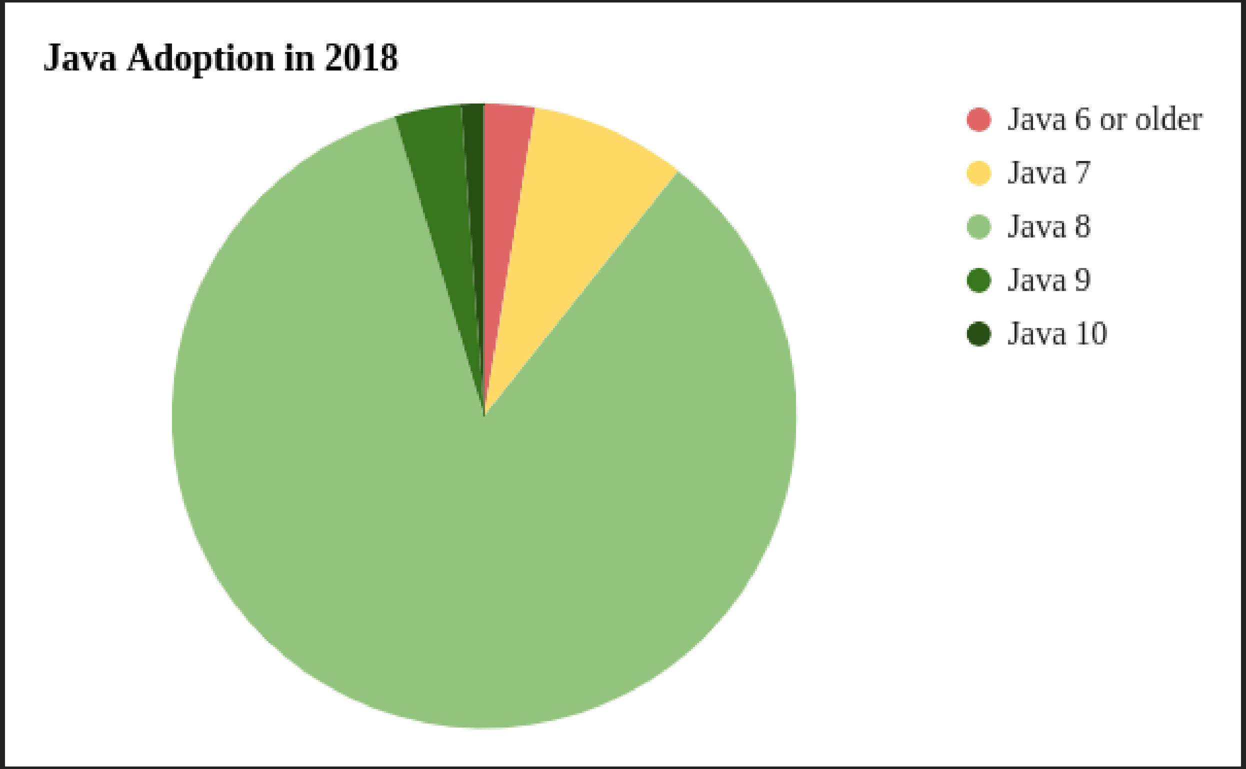
Where we’ve been and where we’re going

Paweł Żalejko
@pzalejko

🚀 Topics for today 🚀

1. Recent changes
2. Different distributions and support
3. Tough decisions ahead
4. Possible choices

The State of Java in 2018



https://www.baeldung.com/java-in-2018

84.7%

Java 8

January 2019

End of Public Updates for Oracle JDK 8

1 day left!

https://www.oracle.com/technetwork/java/javase/overview/index.html

JAVA 8 (LTS).

Release - March 2014
End:
- January
- December
- Jun 20
- Septembe

JAVA 10.

Release - March 2018
End: September 2018

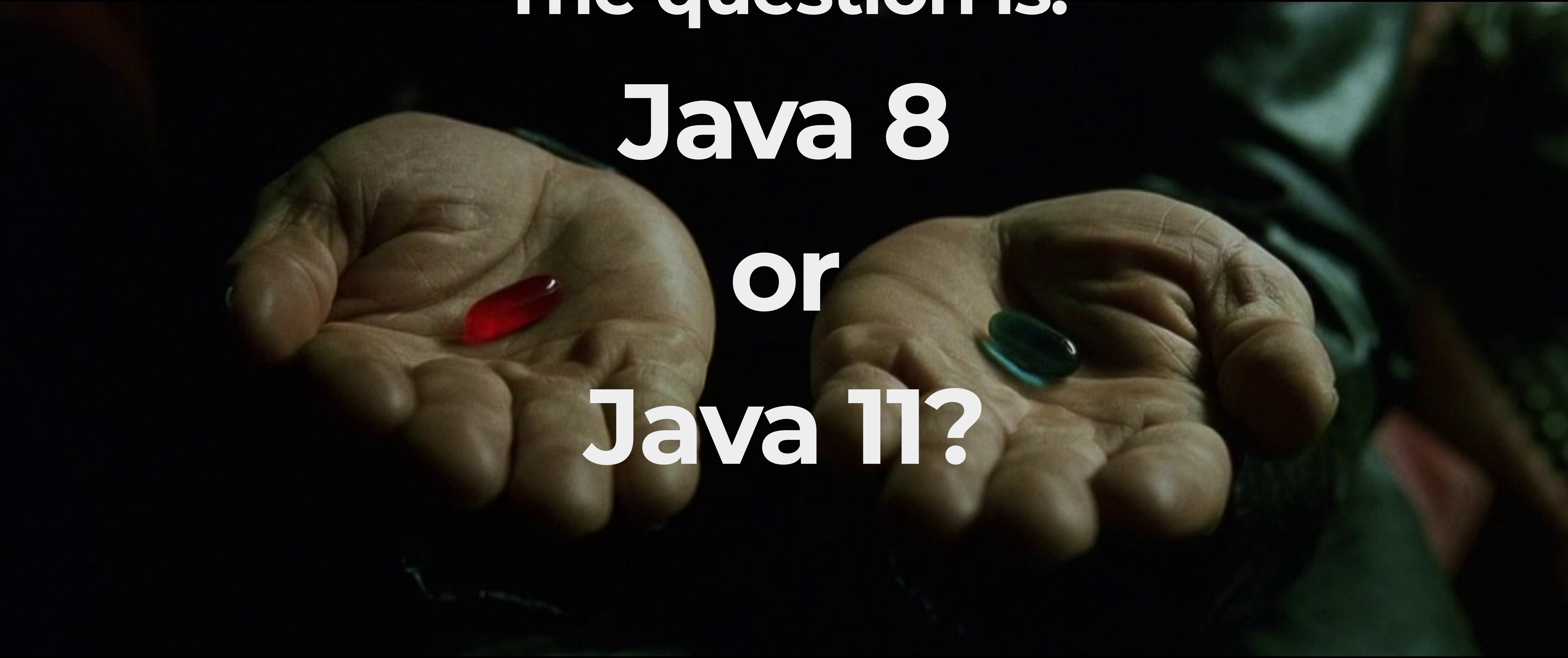
JAVA 9.

Release - September 2017
End: March 2018

11 (LTS).

eptember 2019
End:
- March 2019*
- September 2023
- September 2022+

There are many providers



The question is:

Java 8

or

Java 11?

Java 11

双汇集团
SHUANGHUI GROUP
GIFOVEA

Lots of changes

- 55 New Features in JDK 9
- 109 New Features in JDK 10
- 90 New Features in JDK 11

Many new methods in well-known classes

- Stream API
- Collection clases
- I/O classes
- String
- Optional
- and more...

Let's start with small things:

```java
Map<String, Integer> map = Map.of("a", 1, "b", 2);

Optional<String> value = ...;
Consumer<String> consumer = ...;
Runnable runnable = ...;
value.ifPresentOrElse(consumer, runnable);

List<String> strings = List.of("", "a", " ", "b", "  ", "c");
List<String> result = strings.stream()
        .filter(Predicate.not(String::isBlank))
        .collect(Collectors.toList());

result.stream()
        .takeWhile(s -> !s.equals("c"))
        .forEach(System.out::println);
```

Private method in interface:

```java
public interface Sample {

    default void sayHello(String message) {
        printMessage(message);
    }

    private void printMessage(String msg) {
        System.out.println(msg);
    }
}
```

A new Process API:

```java
public class Sample {
    public static void main(String[] args) {
        printInfo(ProcessHandle.current());
        ProcessHandle.allProcesses().forEach(Sample::printInfo);
    }
    private static void printInfo(ProcessHandle processHandle) {
        System.out.println(processHandle.pid());
        System.out.println(processHandle.info().user());
        System.out.println(processHandle.info().command());
        System.out.println(processHandle.info().commandLine());
    }
}
```

Local variable type inference:

```java
public static void main(String[] args) {
    var var = "var everywhere ;)";
    var stream = Stream.of("a", "b", sample, "c");
    var listA = new ArrayList<String>(); // as ArrayList<String>
    // List<String> listB = new ArrayList<>();
    var listB = new ArrayList<>(); // as ArrayList<Object>
    var tmp = ()->"foo"; // will not compile!

    List<String> items = stream
            .filter((@Nonnull var f) -> f.equals(sample))
            .collect(Collectors.toList());

    for (var item : items) { System.out.println(item); }
}
```

A new HTTP client:

```
public static void main(String[] args) {
    HttpClient client = HttpClient.newBuilder()
            .version(HttpClient.Version.HTTP_2) // default
            .build();
    HttpRequest request = HttpRequest.newBuilder()
            .uri(URI.create("http://openjdk.java.net/"))
            .GET()   // default
            .build();
    client.sendAsync(request, BodyHandlers.ofString())
            .thenApply(HttpResponse::body)
            .thenAccept(System.out::println)
            .join();
    }
```

GC and memory:

- JEP 307: Parallel Full GC for G1
- JEP 310: Application Class-Data Sharing
- JEP 318: Epsilon: A No-Op GC
- JEP 333: ZGC: A Scalable Low-Latency GC
- JEP 189: Shenandoah: A Low-Pause-Time GC **

Docker

- *namespaces* - isolates from other processes
- *cgroups* - limits the resource consumption

The problem ...

```
// Java8: it will not work as expected
docker container run -it -m=512M --cpus 2 ...
```

Java < 10

```
docker container run -it -m=512M --entrypoint bash openjdk:8
 #docker-java-home/bin/java -XX:+PrintFlagsFinal \
  -version | grep MaxHeapSize

uintx MaxHeapSize  := 4181721088   {product}
```

Java 10+

```
docker container run -it -m=512M --entrypoint bash openjdk:11
 #docker-java-home/bin/java -XX:+PrintFlagsFinal \
  -version | grep MaxHeapSize

size_t MaxHeapSize   = 132120576  {product} {ergonomic}
```

Docker, CPU and Java

- --cpus 2 / --cpu-period / --cpu-quota
- --cpu-shares 1024
- --cpuset-cpus="1,2,3"
- Container Awareness API

source1 source2

and much more:

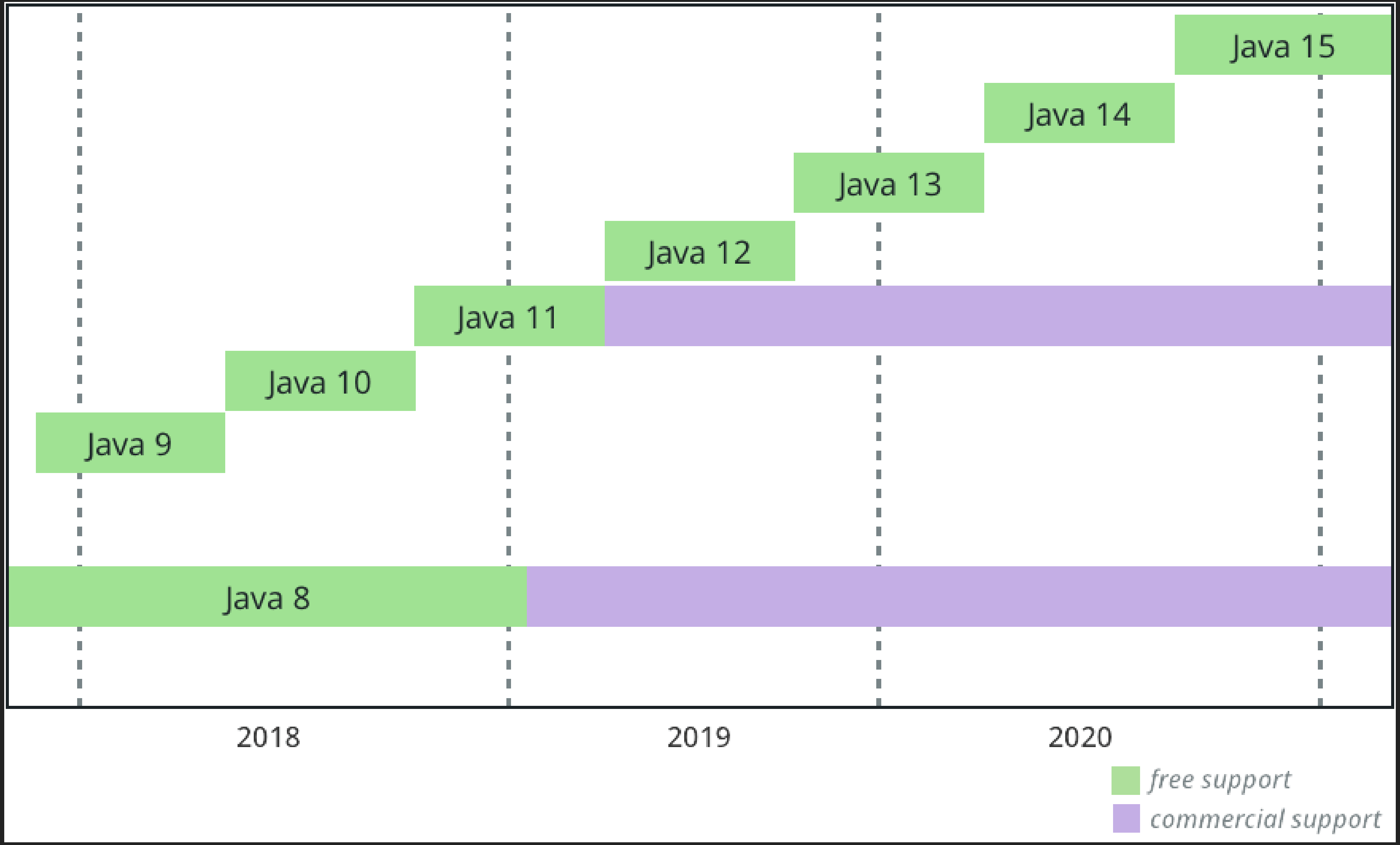
- Jigsaw
- JEP 282: jlink
- JEP 222: jshell
- JEP 332: Transport Layer Security (TLS) 1.3
- but..

Java 11

changes everything

New release train

- time-based releases
- "feature" release every six months
- LTS release every three years


Java 15
Java 14
Java 13
Java 12
Java 11
Java 10
Java 9
Java 8
2018
2019
2020
free support
commercial support

source

Java LTS

8, 11, 17

Oracle

- OpenJDK builds (GPLv2)
- Oracle JDK builds (commercial product!)

> *“From Java 11 forward, therefore, Oracle JDK builds and OpenJDK builds will be essentially identical. ...yet with some cosmetic and packaging differences*

https://blogs.oracle.com/java-platform-group/oracle-jdk-releases-for-java-11-and-later

Oracle LTS

- OpenJDK builds from Oracle are not LTS!
- Oracle JDK builds are LTS (commercial)

The Price

- Desktop pricing is $2.50 per user per month
- Processor pricing for use on Servers and/or Cloud deployments is $25.00 per month
- *"It also includes access to My Oracle Support (MOS) 24x7, support in 27 languages"*
- More info here
- Oracle Java SE Subscriptions program

Usage of Oracle JDK after March 2019

requires a commercial license

Purges in Java

- Removal of the Java EE and CORBA Modules
- Removal of Java Mission Control
- Removal of JavaFX from JDK 11
- Deprecate the Nashorn JavaScript Engine
- Removal of the Java Plugin and Java WebStart

source

What if I don't want Oracle JDK?

OpenJDK

https://openjdk.java.net/

Red Hat

- OpenJDK 8 - June 2023 (commercial support)
- OpenJDK 11 - October 2024 (commercial support)
- Shenandoah GC is available for JDK 8u and 11u
- JDK 11+ - both Shenandoah and ZGC
- you can download for Windows
- an "upstream first" policy

IBM

- Free IBM SDK for Java 8
- IBM will continue to update OpenJDK Java 8
- and they will do it for 4 years

Azul Systems

- Zulu: Free build of OpenJDK
- Builds of Zulu With OpenJFX
- Builds of Zulu for 64-bit Armv8
- Microsoft Azure uses Zulu Enterprise (LTS)
- Zulu Enterprise
    - Java 8 - 2026
    - Java 11 - 2027

Amazon Corretto

- In preview
- No-cost
- Corretto 8 - GA is planned for Q1 2019
- Corretto 8 - until at least June 2023
- Corretto 11 - GA during the first half of 2019
- Corretto 11 - until at least August 2024

source

OpenJDK - DIY!

- turned out to be not so hard
  - Boot JDK
  - source code (Mercurial or Git)
  - tools (automake, gcc-c++, ... , Docker)
- 16 GB RAM, i7-4700MQ, SSD -> 10 minutes
- My OpenJDK build;)

AdoptOpenJDK

- https://adoptopenjdk.net/
- community-driven project
- provides prebuilt OpenJDK binaries
- sponsored and supported by many companies
- Java 8 and Java 11
- JVM: HotSpot and OpenJ9

JVM

- Responsibilities:
  - Loading, verifying, and executing the byte code
  - Providing a runtime environment
  - Memory management and garbage collection
- specification
- 19 implementations!

JVM implementations

- HotSpot
- OpenJ9 (IBM J9 VM)
- GraalVM

Examples

★ Thank you ★

Q&A Time