Discovering Modern Java

What you need to know about Java 9, 10, 11 and beyond

Henri Tremblay
Managing Director, Head of TS Canada
TradingScreen
@henri_tremblay

Henri Tremblay

- More or less made possible class mocking and proxying
- Coined the term “partial mocking”

Henri Tremblay

EASYMOCK

OBJENESIS

EHCACHE

peRf UserGroup Paris

MONTREAL JAVA USER GROUP

Oracle Groundbreaker Ambassador

- More or less made possible class mocking and proxying
- Coined the term “partial mocking”

Good old

7

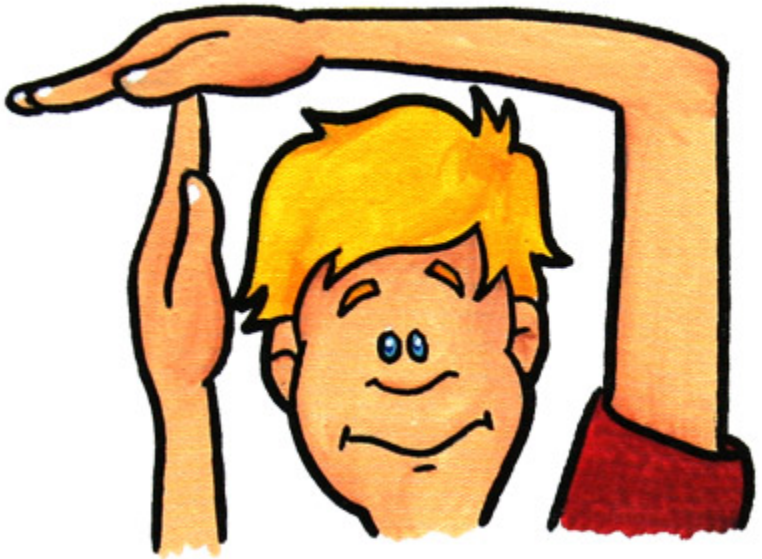
New & shinny

16

POLL

- Which version of Java are you currently using?
  - Java 6 or less
  - Java 7
  - Java 8
  - Java 9-10
  - Java 11
  - Above

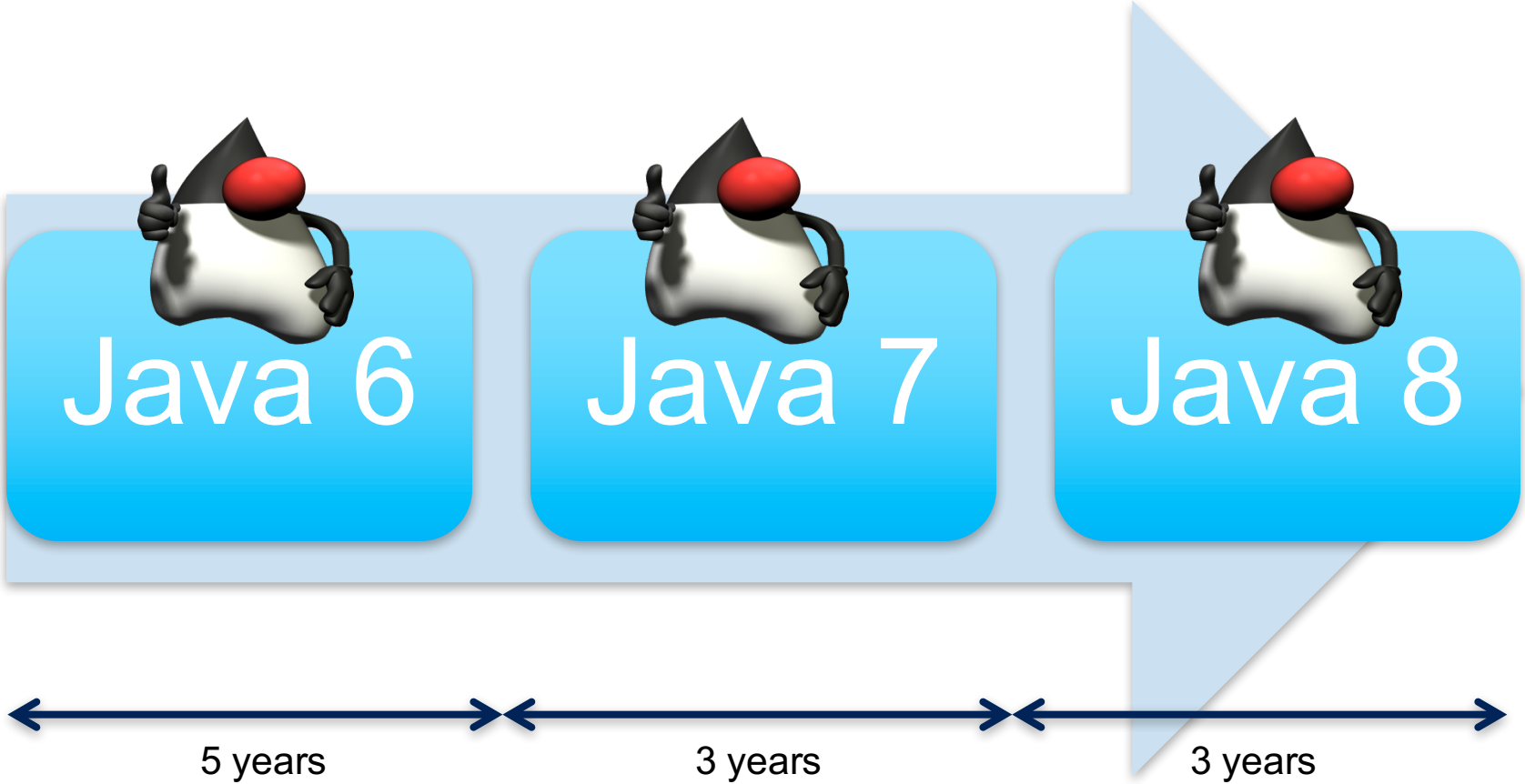
Breaks



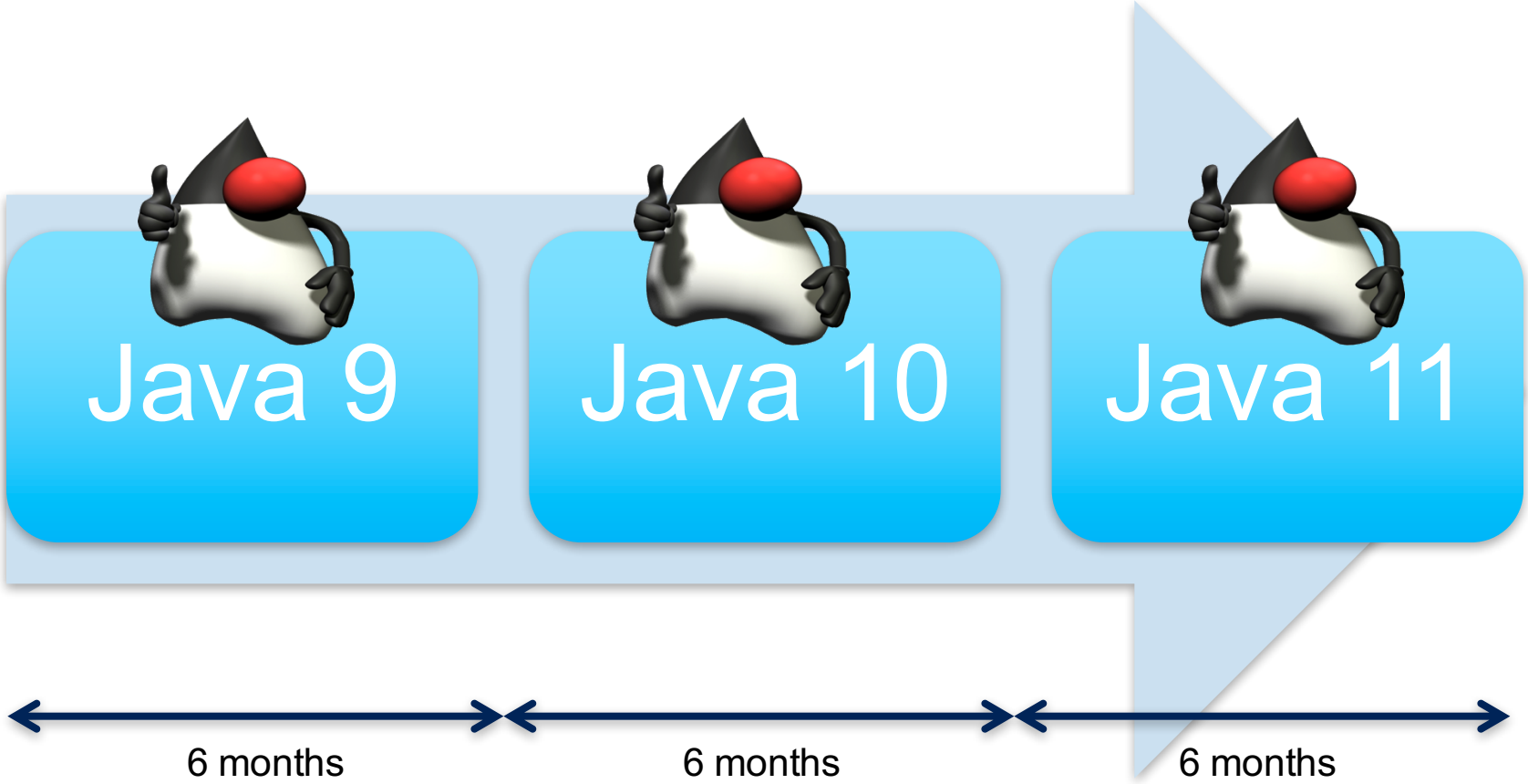
Questions

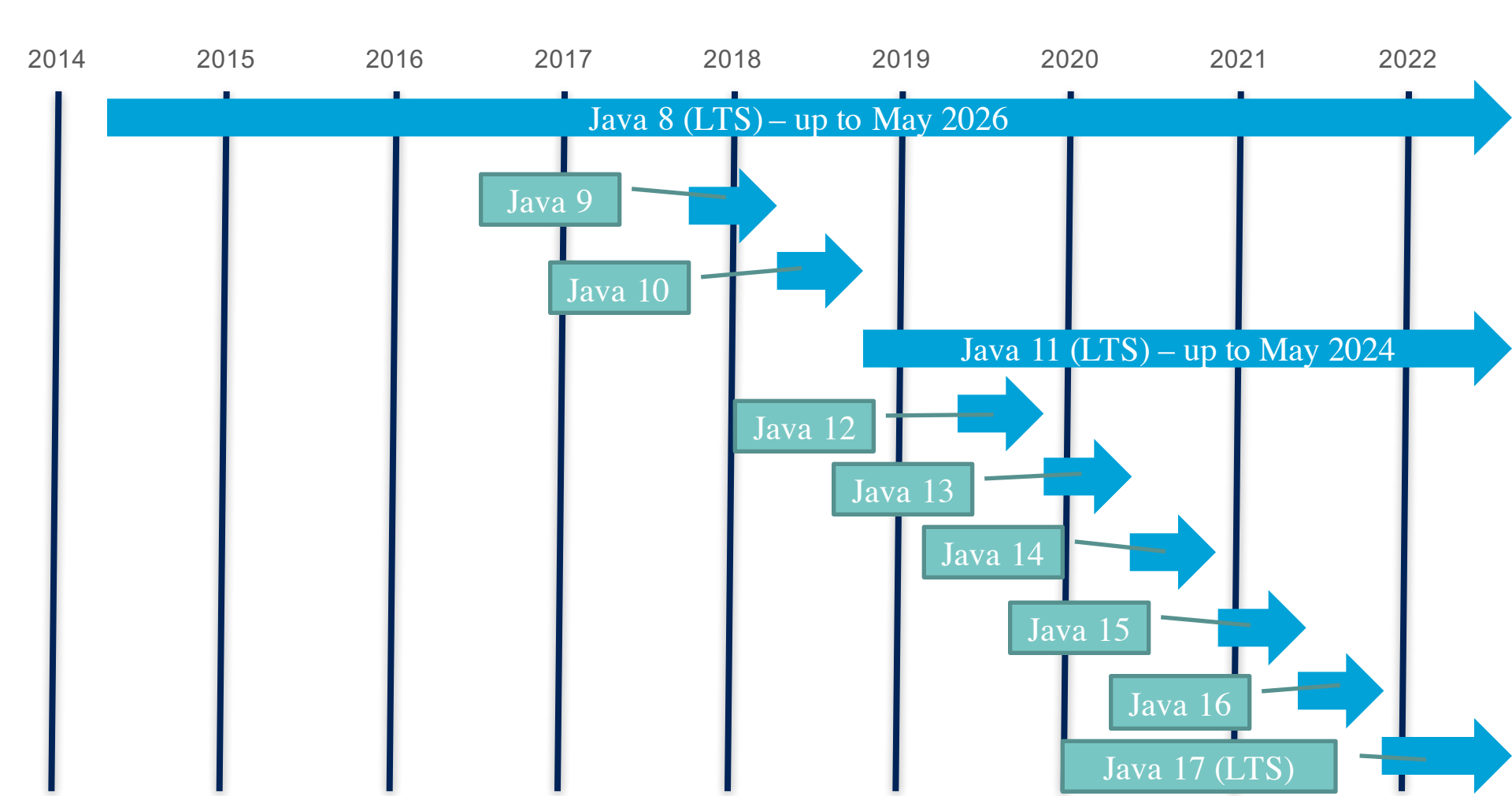
Java Delivery Process

Old delivery process



New delivery process




2014
2015
2016
2017
2018
2019
2020
2021
2022
Java 8 (LTS) – up to May 2026
Java 9
Java 10
Java 11 (LTS) – up to May 2024
Java 12
Java 13
Java 14
Java 15
Java 16
Java 17 (LTS)

All these JVMs

- **OracleJDK**: Free for development, supported for 6 months
- **Oracle OpenJDK:** Free forever, supported for 6 months
- **Other OpenJDK:** Free forever, LTS supported after 6 months
  - Built by **AdoptOpenJDK**, Azul, IBM, Amazon, Bellsoft…

- For more details see
  - https://medium.com/@javachampions/java-is-still-free-2-0-0-6b9aa8d6d244

- **Coin:** Syntactic sugar in Java 7
- **Valhalla:** Value types
- **Loom:** New type of threads called Fiber
- **Panama:** Improved IO
- **Amber:** Developer productivity

https://openjdk.java.net/projects/

- **JCP:** Java Community Process
  - Responsible of Java governance
- **JLS:** Java Language Specification
  - The is a specification for the Java Language
- **JSR:** Java Specification Request
  - Big piece of Java change submitted to JCP
- **JEP:** Java Enhancement Proposal
  - Little pieces of Java change

Trends

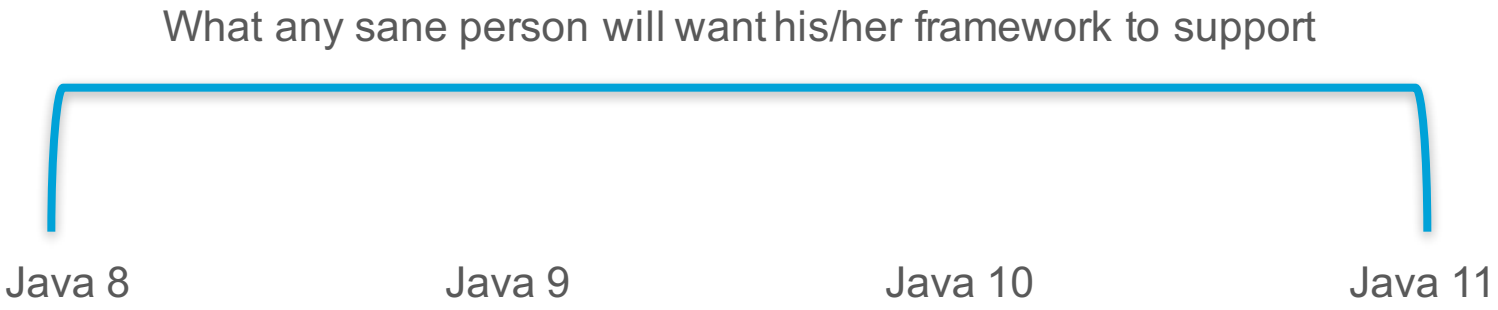
`@Deprecated`

is back

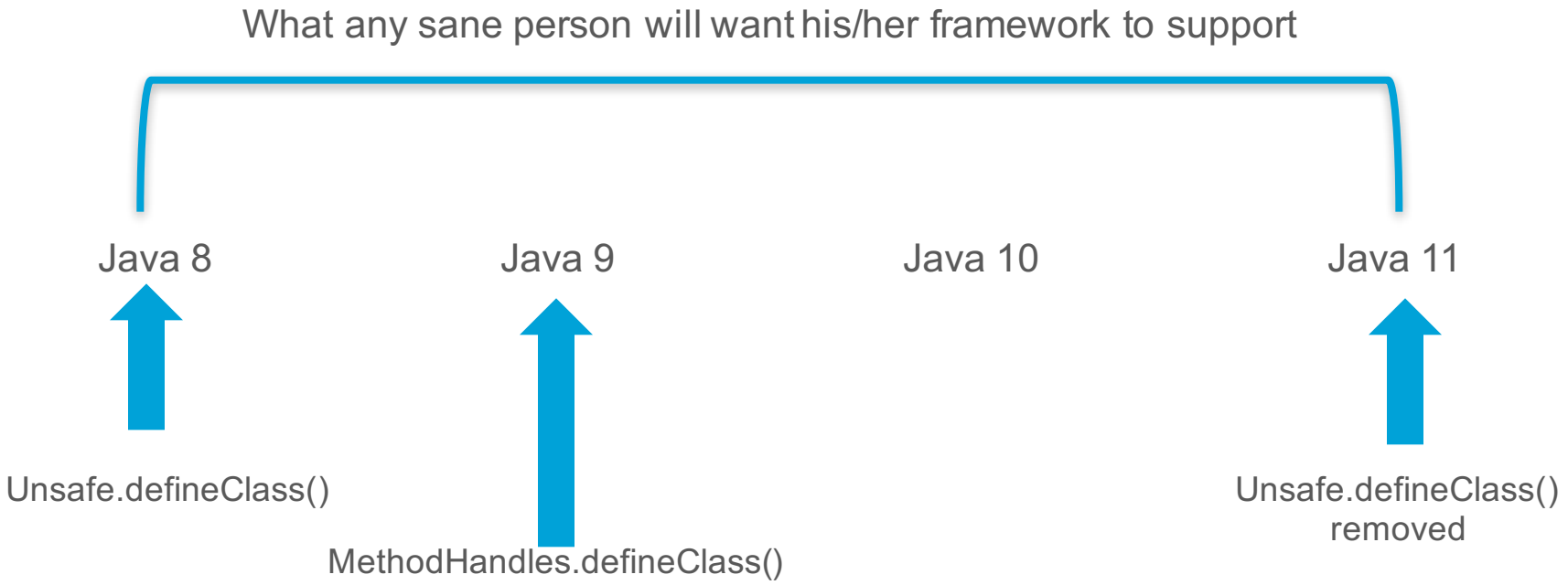
Stronger than ever

```
/**
 * Counts the number of stack frames in this thread. The thread must
 * be suspended.
 *
 * @return     the number of stack frames in this thread.
 * @throws     IllegalThreadStateException  if this thread is not
 *             suspended.
 * @deprecated The definition of this call depends on {@link #suspend},
 *             which is deprecated.  Further, the results of this call
 *             were never well-defined.
 *             This method is subject to removal in a future version of Java SE.
 * @see        StackWalker
 */
@Deprecated(since="1.2", forRemoval=true)
public native int countStackFrames();
```

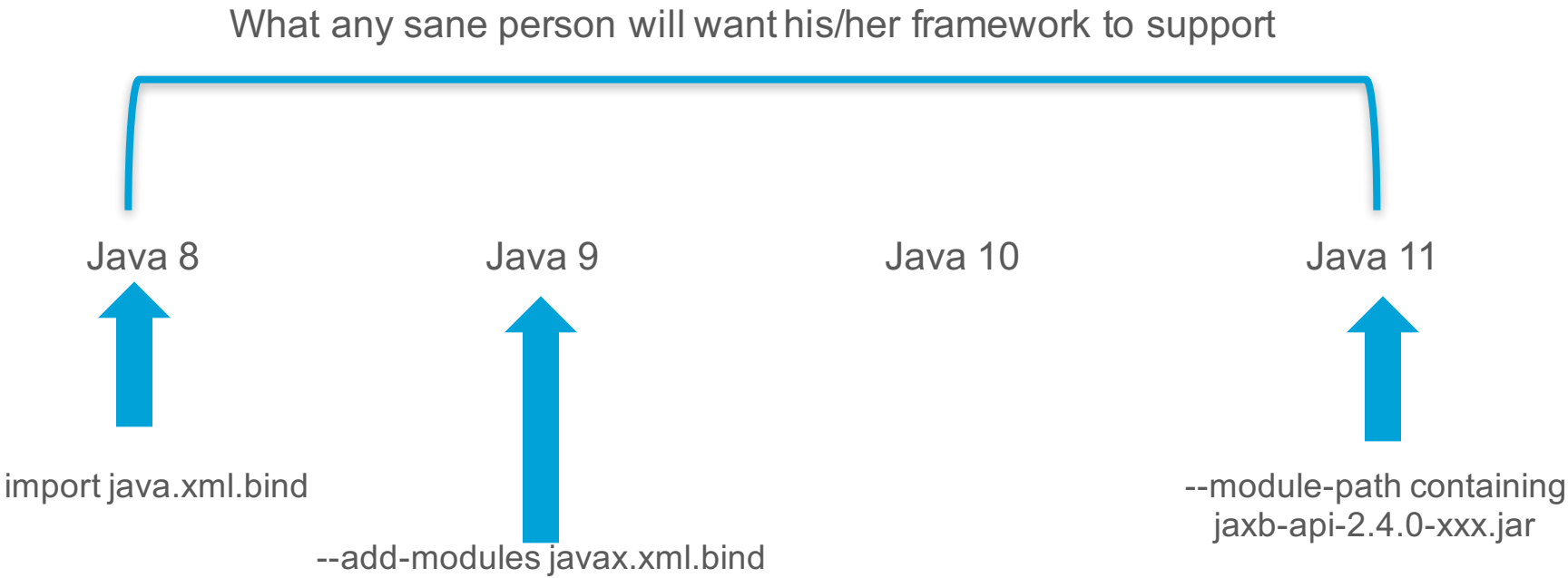
Prepare to do some stretching



Prepare to do some stretching



Prepare to do some stretching



GC

Garbage collectors for all

Java < 8

- Serial
- Parallel
- ParallelOld
- CMS
- iCMS

Java 8

- G1

Java 9

- Removed iCMS
- G1 (Default GC)

Java 11

- Epsilon
- Z (Experimental)

Java 12

- Shenandoah (Experimental)

Java 14

- Remove CMS

Java 15

- Shenandoah
- Z

Java 7

(2011, first from Oracle)

JAVA 7 BROUGHT A LOT OF SYNTACTIC SUGAR

(plus invokeDynamic, forkJoin, better file IO)


```
List<String> list = new ArrayList<>();
```

Before

```
InputStream in = new FileInputStream("allo.txt");
try {
   // … do stuff
} finally {
   try { in.close(); } catch(IOException e) {}
}
```

After

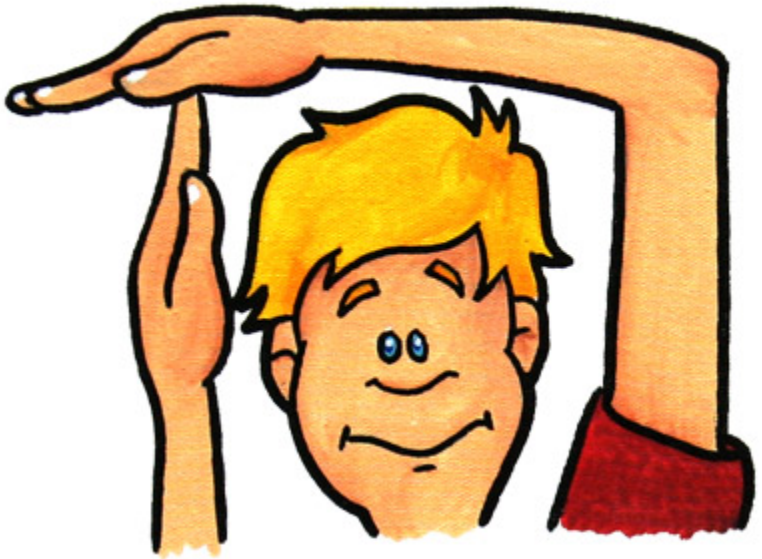
```
try(InputStream in = new FileInputStream("allo.txt")) {
   // … do stuff
}
```

Real code you should do

```
InputStream in = new FileInputStream("allo.txt");
try {
   // … do stuff
   in.close();
} catch(IOException e) {
   try {
      in.close();
   } catch(IOException e1) {
      e.addSuppressed(e1);
   }
   throw e;
}
```

DEMO

5 minutes break



Java 8

(2014)

JAVA 8: LAMBDA!

(and also a new date API, default methods, metaspace, Nashorn, JavaFX and CompletableFuture)

```
LocalDateTime now = LocalDateTime.now();
String thatSpecialDay = now
    .withDayOfMonth(1)
    .atZone(ZoneId.of("Europe/Paris"))
    .plus(Duration.ofDays(5))
    .format(DateTimeFormatter.ISO_ZONED_DATE_TIME);

System.out.println(thatSpecialDay);
```

Output

2016-09-06T17:45:22.488+01:00[Europe/Paris]

Lambda: 11th letter of the Greek alphabet

(also written as λ-calculus) is a formal system in mathematical logic for expressing computation based on function abstraction and application using variable binding and substitution. It is a universal model of computation that can be used to simulate any single-taped Turing machine and was first introduced by mathematician Alonzo Church in the 1930s as part of an investigation into the foundations of mathematics.

This is a function:

$$\mathrm{square\_sum}(x, y) = x^2 + y^2$$

This is a lambda:

$$(x, y) \mapsto x^2 + y^2$$

DEMO

DEMO

```
public interface List<E> extends Collection<E> {
   default void replaceAll(UnaryOperator<E> operator) {
      // …
   }
}
```

```
public interface IntStream {
    static IntStream empty() {
        return …;
    }
}
```

```
IntStream stream = IntStream.empty();
```

DEMO

Method references

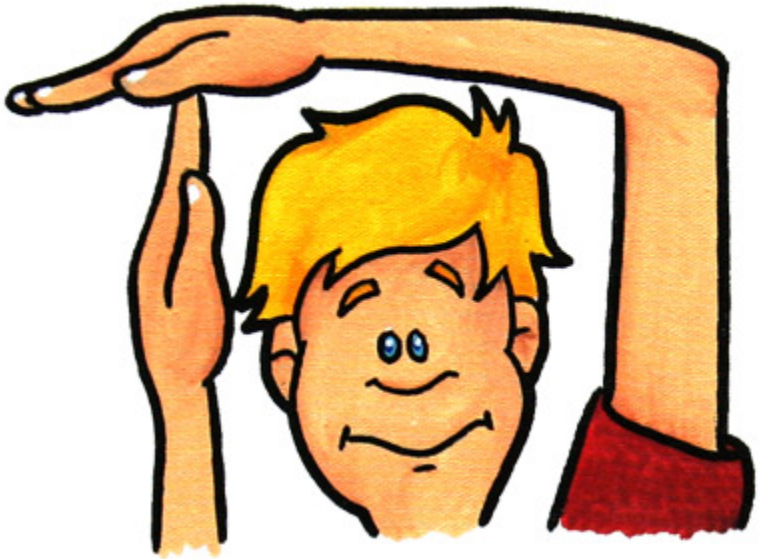
```
public static class Passenger {
    public void inboard(Train train) {
        System.out.println("Inboard " + train);
    }
}

public static class Train {
    public Train () { }
    public Train (int serial) { }

    public static void paintBlue(Train train) {
        System.out.println("Painted blue " + train);
    }

    public void repair() {
        System.out.println( "Repaired " + this);
    }
}

List<Train> trains = Arrays.asList(train);
```

```
// static method
trains.forEach(Train::paintBlue);

// instance method
trains.forEach(Train::repair);

// instance method taking this in param
Passenger p = new Passenger();
trains.forEach(p::inboard);
trains.forEach(System.out::println);      // useful!

// constructor
public static Train create(Supplier<Train> supplier) {
    return supplier.get();
}
Train train = Train.create(Train::new);

public static Train create(IntFunction<Train> func) {
    return func.apply(42);
}
Train train = Train.create(Train::new);
```

NullPointerException on method reference 08

DEMO

5 minutes break



Java 9

(2017)

JAVA 9: MODULES!

(and G1 by default, var handles, new http client, jshell, jcmd, immutable collections, unified JVM logging)

Modern mustache instantiation

```
Map<String, String> map = new HashMap<>() {{
    put("key", "value");
}};
```

```
Map<String, String> map =
    Map.of("key", "value");
```

Slightly longer map instantiation

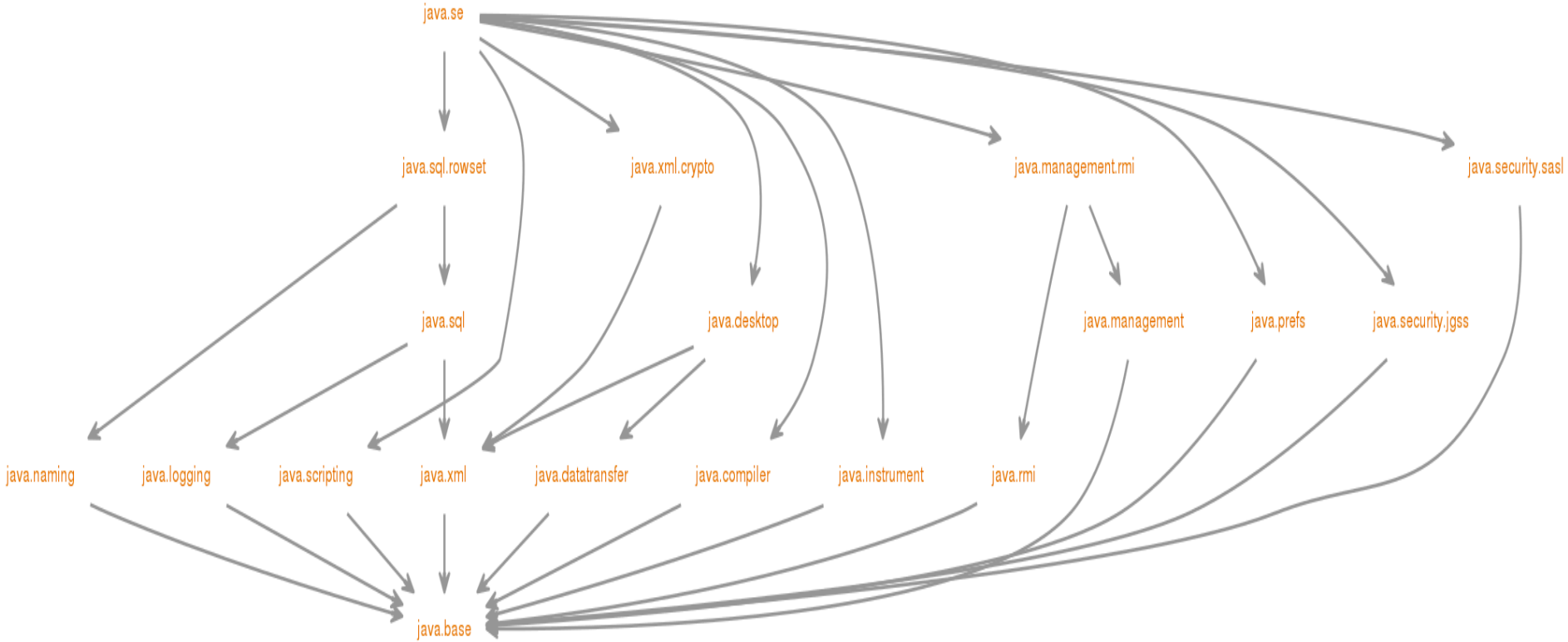
```
private Map<String, String> map = new HashMap<>();
{
  map.put("key", "value");
}
```

Improved try-with-resource

```
ByteArrayOutputStream in = new ByteArrayOutputStream();

try(in) {

}
```

```
public interface Printer {
   private void print(String s) {
      System.out.println(s);
   }

   default void printAll(String[] list) {
      Arrays.stream(list).forEach(this::print);
   }
}
```

Java SE Modules



Credits: Oracle

DEMO

Java 10

JSR 383 (March 2018)

JEP 286: Local-Variable Type Inference
JEP 296: Consolidate the JDK Forest into a Single Repository
JEP 304: Garbage-Collector Interface
JEP 307: Parallel Full GC for G1
JEP 310: Application Class-Data Sharing
JEP 312: Thread-Local Handshakes
JEP 313: Remove the Native-Header Generation Tool (javah)
JEP 314: Additional Unicode Language-Tag Extensions
JEP 316: Heap Allocation on Alternative Memory Devices
JEP 317: Experimental Java-Based JIT Compiler
JEP 319: Root Certificates
JEP 322: Time-Based Release Versioning

DEMO

- Keywords
  - `while, if, abstract, public, default, class, enum (Java 5), _ (Java 9)`
- Restricted Keywords
  - open, module, to, with
- Literals
  - true, false, null
- Reserved identifier
  - var

DEMO

Java 11

JSR 384 (September 2018)

Cleanup!

JEP 181: Nest-Based Access Control
JEP 309: Dynamic Class-File Constants
JEP 315: Improve Aarch64 Intrinsics
JEP 318: Epsilon: A No-Op Garbage Collector
JEP 320: Remove the Java EE and CORBA Modules
JEP 321: HTTP Client (Standard)
JEP 323: Local-Variable Syntax for Lambda Parameters
JEP 324: Key Agreement with Curve25519 and Curve448
JEP 327: Unicode 10
JEP 328: Flight Recorder
JEP 329: ChaCha20 and Poly1305 Cryptographic Algorithms
JEP 330: Launch Single-File Source-Code Programs
JEP 331: Low-Overhead Heap Profiling
JEP 332: Transport Layer Security (TLS) 1.3
JEP 333: ZGC: A Scalable Low-Latency Garbage Collector (Experimental)
JEP 335: Deprecate the Nashorn JavaScript Engine
JEP 336: Deprecate the Pack200 Tools and API

DEMO

DEMO

Java 12

JSR 386 (March 2019)

JEP 189: Shenandoah: A Low-Pause-Time Garbage Collector (Experimental)

JEP 230: Microbenchmark Suite

JEP 325: Switch Expressions (Preview)

JEP 334: JVM Constants API

JEP 340: One AArch64 Port, Not Two

JEP 341: Default CDS Archives

JEP 344: Abortable Mixed Collections for G1

JEP 346: Promptly Return Unused Committed Memory from G1

Switch expressions

```
private boolean isWeekDay(DayOfWeek day) {
  boolean weekDay;

  switch(day) {
    case MONDAY:
    case TUESDAY:
    case WEDNESDAY:
    case THURSDAY:
    case FRIDAY:
      weekDay = true;
      break;
    case SATURDAY:
    case SUNDAY:
      weekDay = false;
    default:
      throw new IllegalStateException("A new day was added in my week: " + day);
  }
  return weekDay;
}
```

Switch expressions

```
private boolean isWeekDay(DayOfWeek day) {
  boolean weekDay;

  switch(day) {
    case MONDAY:
    case TUESDAY:
    case WEDNESDAY:
    case THURSDAY:
    case FRIDAY:
      weekDay = true;
      break;
    case SATURDAY:
    case SUNDAY:
      weekDay = false;
      break;
    default:
      throw new IllegalStateException("A new day was added in my week: " + day);
  }
  return weekDay;
}
```

Switch expressions

```
private boolean isWeekDay(DayOfWeek day) {
  boolean weekDay;

  switch(day) {
    case MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY -> weekDay = true;
    case SATURDAY, SUNDAY -> weekDay = false;
    default -> throw new IllegalStateException("A new day was added in my week: " + day);
  }

  return weekDay;
}
```

Switch expressions

```
private boolean isWeekDay(DayOfWeek day) {
  boolean weekDay = switch(day) {
    case MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY -> true;
    case SATURDAY, SUNDAY -> false;
    default -> throw new IllegalStateException("A new day was added in my week: " + day);
  };

  return weekDay;
}
```

Switch expressions

```
private boolean isWeekDay(DayOfWeek day) {
  return switch(day) {
    case MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY -> true;
    case SATURDAY, SUNDAY -> false;
    default -> throw new IllegalStateException("A new day was added in my week: " + day);
  };
}
```

Java 13

JSR 388 (September 2019)

JEP 350: Dynamic CDS Archives

JEP 351: ZGC: Uncommit Unused Memory

JEP 353: Reimplement the Legacy Socket API

JEP 354: Switch Expressions (Preview)

JEP 355: Text Blocks (Preview)

Switch expressions (still a preview)

```
private boolean isWeekDay(DayOfWeek day) {
  return switch(day) {
    case MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY -> true;
    case SATURDAY, SUNDAY -> false;
    default -> {
        if(onEarth) {
            yield false;
        }
        yield true;
    }
  };
}
```

```
String script = """
                function hello() {
                  print('"Hello, world\"');
                }

                hello();
                """;
```

Java 14

JSR 389 (March 2020)

Big one

JEP 305: Pattern Matching for instanceof (Preview)
JEP 343: Packaging Tool (Incubator)
JEP 345: NUMA-Aware Memory Allocation for G1
JEP 349: JFR Event Streaming
JEP 352: Non-Volatile Mapped Byte Buffers
JEP 358: Helpful NullPointerExceptions
JEP 359: Records (Preview)
JEP 361: Switch Expressions (Standard)
JEP 362: Deprecate the Solaris and SPARC Ports
JEP 363: Remove the Concurrent Mark Sweep (CMS) Garbage Collector
JEP 364: ZGC on macOS
JEP 365: ZGC on Windows
JEP 366: Deprecate the ParallelScavenge + SerialOld GC Combination
JEP 367: Remove the Pack200 Tools and API
JEP 368: Text Blocks (Second Preview)
JEP 370: Foreign-Memory Access API (Incubator)

DEMO

DEMO

DEMO

Java 15

JSR 390 (September 2020)

Previews

JEP 339: Edwards-Curve Digital Signature Algorithm (EdDSA)
JEP 360: Sealed Classes (Preview)
JEP 371: Hidden Classes
JEP 372: Remove the Nashorn JavaScript Engine
JEP 373: Reimplement the Legacy DatagramSocket API
JEP 374: Disable and Deprecate Biased Locking
JEP 375: Pattern Matching for instanceof (Second Preview)
JEP 377: ZGC: A Scalable Low-Latency Garbage Collector
JEP 378: Text Blocks
JEP 379: Shenandoah: A Low-Pause-Time Garbage Collector
JEP 381: Remove the Solaris and SPARC Ports
JEP 383: Foreign-Memory Access API (Second Incubator)
JEP 384: Records (Second Preview)
JEP 385: Deprecate RMI Activation for Removal

DEMO

DEMO

DEMO

Java 16

JSR 391 (March 2021)

Make the JDK developers life easier

JEP 338: Vector API (Incubator)
JEP 347: Enable C++14 Language Features
JEP 357: Migrate from Mercurial to Git
JEP 369: Migrate to GitHub
JEP 376: ZGC: Concurrent Thread-Stack Processing
JEP 380: Unix-Domain Socket Channels
JEP 386: Alpine Linux Port
JEP 387: Elastic Metaspace
JEP 388: Windows/AArch64 Port
JEP 389: Foreign Linker API (Incubator)
JEP 390: Warnings for Value-Based Classes
JEP 392: Packaging Tool
JEP 393: Foreign-Memory Access API (Third Incubator)
JEP 394: Pattern Matching for instanceof
JEP 395: Records
JEP 396: Strongly Encapsulate JDK Internals by Default
JEP 397: Sealed Classes (Second Preview)

Strongly encapsulate JDK internals by default

--illegal-access=permit ➔ Open to unnamed modules. 1 warning on the console

--illegal-access=warn ➔ Same. But tons of warnings

--illegal-access=debug ➔ Same, but stack trace of the warnings

--illegal-access=deny ➔ Forbidden unless –add-opens is used

DEMO

DEMO

Conclusion

Who has learned something today?

Links

- Java Champions – Java is still free
  - https://medium.com/@javachampions/java-is-still-free-2-0-0-6b9aa8d6d244
- Inside Java
  - https://inside.java/
- Adopt OpenJDK
  - https://adoptopenjdk.net/
- Zeroturnaround Module Cheat Sheet
  - http://files.zeroturnaround.com/pdf/RebelLabs-Java-9-modules-cheat-sheet.pdf
- Var styleguide
  - http://openjdk.java.net/projects/amber/LVTIstyle.html
- Source code
  - https://github.com/henri-tremblay/java91011beyond
- JDK
  - https://openjdk.java.net/projects/jdk/
- Sdkman
  - https://sdkman.io/

- Courses by
  - Cay S. Horstmann
  - Heinz M. Kabutz
  - Kirk C. Pepperdine

http://objenesis.org

http://montreal-jug.org

Questions?

Henri Tremblay
henri@tremblay.pro
http://blog.tremblay.pro
@henri_tremblay

EASYMOCK

http://easymock.org