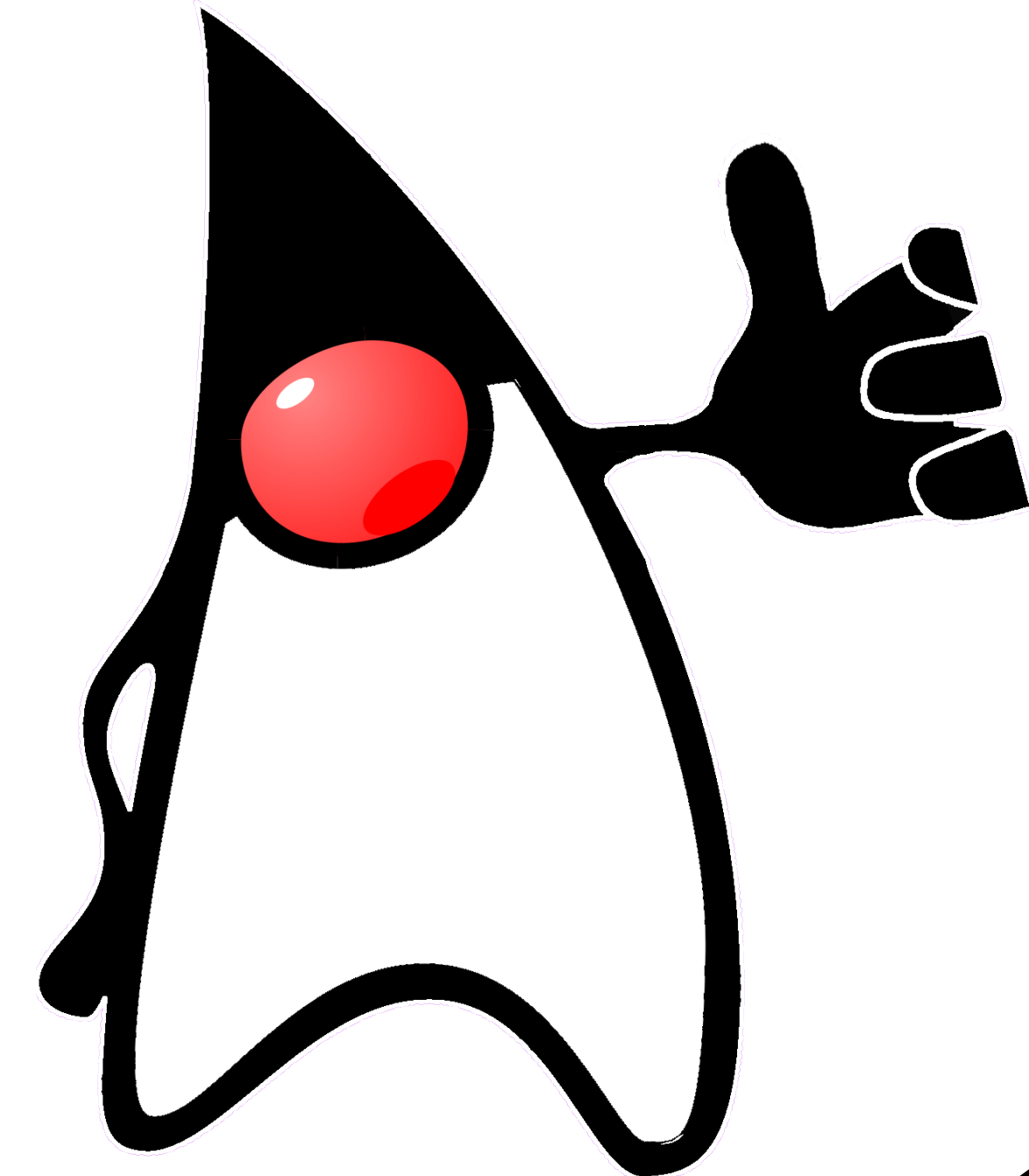# Java 9 - 14

Key Features and Enhancements

Chandra Guntur
Twitter: @CGuntur

@CGuntur

1

Java 9

#WorksFineWithJavaNine

Java 9

- Released: Sep. 2017

- => Java 8 was Mar. 2014

- 150 new features and enhancements

- One of the largest upgrades to Java

- https://openjdk.java.net/projects/jdk9/

- https://speakerdeck.com/cguntur/java-9-new-features

@CGuntur

Private Methods in Interfaces (1)

● To share common code between methods

● Methods with bodies: static and default

● private default & private abstract not valid combinations

● Originally in Java 8 experimental, dropped

● https://bugs.openjdk.java.net/browse/JDK-8071453

Writeup: https://cguntur.me/2017/09/02/java-9-features-private-interface-methods/

@CGuntur

4

Private Methods in Interfaces (2)

Java support in interfaces

	Java 7	Java 8	Java 9
constants			
abstract methods			
default methods			
public static methods			
private methods			
private static methods			

@CGuntur

Collections - Factory Methods

- Allow ease of declaration/instantiation

- Examples:

  - List.of("one", "two");

  - Map.of("key1", "val1", "key2, "val2");

- JEP 269 - Convenience Collection Factory Methods:
  http://openjdk.java.net/jeps/269

- https://blogs.oracle.com/java/collections-refueled

@CGuntur

G1GC - Default Garbage Collector

- Introduced in Java 7, targeted for Java 8

- JVM now uses a Metaspace instead of PermGen

- GC focusses on garbage-heavy regions

- JEP 248 - Make G1 the default collector:
  http://openjdk.java.net/jeps/248

- https://speakerdeck.com/cguntur/java-garbage-collection-basics

https://docs.oracle.com/javase/9/gctuning/garbage-first-garbage-collector.htm

@CGuntur

Version String Scheme

- Scheme in Java 9 is $MAJOR.$MINOR.$SECURITY_PATCH

- Replaces confusing mix of version and build numbers

- Deemed temporary, changes in future versions

- JEP 223 - Version-String Scheme:
  http://openjdk.java.net/jeps/223

- https://mreinhold.org/blog/forward-faster

@CGuntur

JShell - Java REPL

- Read-Eval-Print-Loop for a transcript evaluation

- Great teaching tool

- Originally targeted for Java 7

- Built on JShell API

- JEP 222 - jshell - The Java Shell:
  http://openjdk.java.net/jeps/222

http://cr.openjdk.java.net/~rfield/tutorial/JShellTutorial.html

🐦 **@CGuntur**

Java Platform Modularity System (1)

- Reduce the large and growing java package size

- Also aims to remove/deprecate vestigial packages

- Allows splitting the JDK in smaller units (modules)

- Root module called java.base

- Dependencies packaged as .jmod files

- Module path to replace classpath
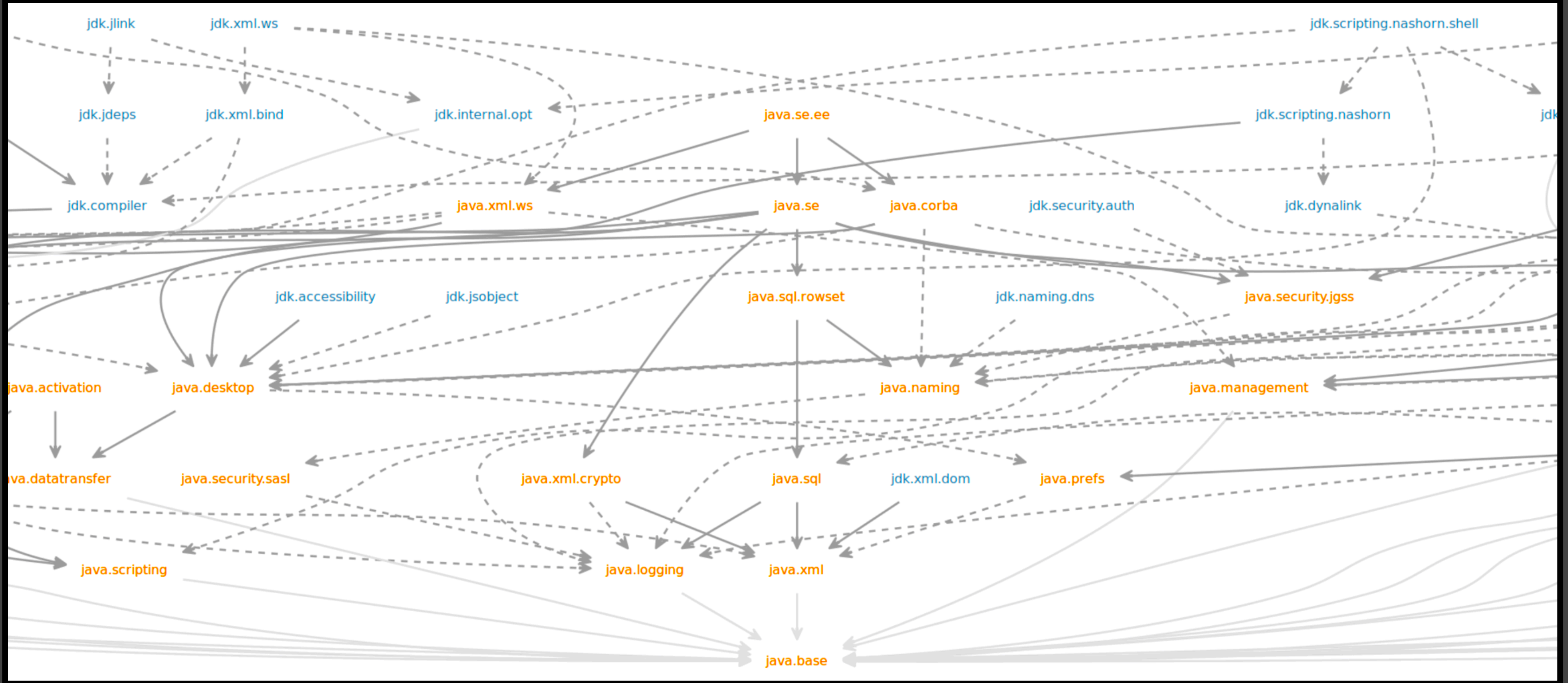
@CGuntur

Java Platform Modularity System (2)

- Java 9 itself is a modular system

- Restructures JDK and JRE runtime images - as modules

- No more rt.jar and tools.jar in libs

- Enables Compact Profiles:
  http://openjdk.java.net/jeps/161

- jlink responsible for assembly and optimizing modules

- jlink produces custom runtime images

@CGuntur

12

Java Platform Modularity System (3)

JPMS (JSR 376):
http://openjdk.java.net/projects/jigsaw/spec/

JEP 261 - Module System:
http://openjdk.java.net/jeps/261

JEP 200 - The Modular JDK:
http://openjdk.java.net/jeps/200

JEP 220 - Modular Run-Time Images:
http://openjdk.java.net/jeps/220

JEP 260 - Encapsulate Internal APIs:
http://openjdk.java.net/jeps/260

@CGuntur

Other Important enhancements

- java.util.Optional - new methods added

- java.util.stream.Stream - new methods added

- java.util.concurrent.* - reactive and futures

- Compact Strings - major fixes to compressed strings

- Deprecation - added forRemoval & since attributes

- Compiler - new flags to compile (--release)

@CGuntur

14

Java 10

#WorksWhenOnJavaTen

Java 10

- Released: Mar. 2018

- First release in the 6-month release model

- Fewer features and enhancements

- Marked as a feature-release, not LTS

- https://openjdk.java.net/projects/jdk/10/

@CGuntur

Local Variable Type Inference (1)

- Introduction of var as a variable type

- Examples:

  ☘ var list = new ArrayList<String>();// infers ArrayList<String>

  ☘ var stream = list.stream(); // infers Stream<String>

- Requires initialization of the variable

- JEP 286 - Local-variable Type Inference:
  http://openjdk.java.net/jeps/286

@CGuntur

Local Variable Type Inference (2)

- Incorrect Examples:

  ♣ var x; // cannot infer type for local

  ♣ var f = () -> { }; // cannot infer type for local

  ♣ var g = null; // cannot infer type for local

  ♣ var k = { 1 , 2 }; // cannot infer type for local

  ♣ (var x, var y) -> x.process(y) // not supported

https://openjdk.java.net/projects/amber/LVTIstyle.html

@CGuntur

18

Application Class-Data Sharing

- Improves startup and memory footprint

- Extends existing CDS (JDK 5) to application classes

- Allows loading to custom class loaders as well

- Use -XX:+UseAppCDS to enable Application CDS

- JEP 310 - Application Class-Data Sharing:
  http://openjdk.java.net/jeps/310

@CGuntur

Experimental Java-Based JIT Compiler

- Enables Graal Ahead-of-time JIT Compiler

- First step for Project Metropolis

- Extends the compiler via a JVM Compiler Interface

- Use with -XX:+UseJVMCICompiler to enable

- JEP 317 - Experimental Java-Based JIT Compiler:
  http://openjdk.java.net/jeps/317

Metropolis: http://mail.openjdk.java.net/pipermail/announce/2017-September/000233.html

@CGuntur

Time-Based Release Versioning

- New scheme: $FEATURE.$INTERIM.$UPDATE.$PATCH

- Also added a java.version.date System property

- Make it easy for a developer to figure out how old a release is

- JEP 322 - Time-Based Release Versioning:
  http://openjdk.java.net/jeps/322

@CGuntur

Other Features

- JEP-296: Consolidate the JDK Forest into a Single Repository

- JEP-304: Garbage-Collector Interface

- JEP-307: Parallel Full GC for G1

- 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-319: Root Certificates

@CGuntur

Java 11

#WorksLikeHeavenWithJavaEleven

Java 11

- Released: Sep. 2018

- Marked as Long Term Support release, not FR

- https://openjdk.java.net/projects/jdk/11/

@CGuntur

Local-Variable Syntax for Lambda Parameters

- Fixes issues with `var` usage in lambda parameters

- Great addition for annotating params in lambdas

- Examples:

  ✤ `(var x, var y) -> x.process(y)` `// Works in Java 11`

  ✤ `(@Nonnull var x, @Nullable var y) -> x.process(y)`

- JEP 323 - Local-Variable Syntax for Lambda Parameters:
  http://openjdk.java.net/jeps/323

@CGuntur

Launch Single-File Source-Code Programs

- Enables launching a single file Java program

- Automatically compiles and runs Java file

- Example: java HelloWorld.java (no need for javac)

- Great tool for students and new beginners

- JEP 330 - Launch Single-File Source-Code Programs:
http://openjdk.java.net/jeps/330

@CGuntur

Remove the Java EE and CORBA Modules

- Removes Java EE & CORBA modules deprecated in Java 9

- Source code deleted from OpenJDK repository

- Binaries do not include these deleted modules

- Related to the separation of Java SE and Jakarta EE

- JEP 320 - Remove the Java EE and CORBA Modules:
  http://openjdk.java.net/jeps/320

@CGuntur

Epsilon Garbage Collector

- **Experimental** No-Op memory allocator without any GC

- Very useful for:

  ✤ Performance testing

  ✤ Memory pressure testing

  ✤ Extremely short-lived jobs

  ✤ Other deeper JVM development

- JEP 318 - Epsilon: A No-Op Garbage Collector: http://openjdk.java.net/jeps/318

Z Garbage Collector

- Experimental Low latency scalable *garbage* collector

- Main features:

  ♣ Guaranteed pause times (10ms)

  ♣ Handles range of small (few MB) to very large memory (TBs)

  ♣ Foundation for future GC features and optimizations

  ♣ Concurrent, single-generation, region-based, NUMA-aware, compacting collector

- JEP 333 - ZGC: A Scalable Low-Latency Garbage Collector:
  http://openjdk.java.net/jeps/333

@CGuntur

Other Features (1)

- JEP-181: Nest-Based Access Control

- JEP-309: Dynamic Class-File Constants

- JEP-315: Improve Aarch64 Intrinsics

- JEP-321: HTTP Client (Standard)

- JEP-324: Key Agreement with Curve25519 and Curve448

- JEP-327: Unicode 10

@CGuntur

Other Features (2)

- JEP-328: Flight Recorder

- JEP-329: ChaCha20 and Poly1305 Cryptographic Algorithms

- JEP-331: Low-Overhead Heap Profiling

- JEP-332: Transport Layer Security (TLS) 1.3

- JEP-335: Deprecate the Nashorn JavaScript Engine

- JEP-336: Deprecate the Pack200 Tools and API

@CGuntur

Java 12

@CGuntur

Java 12

- Released: Mar. 2019

- Marked as a feature-release, not LTS

- https://openjdk.java.net/projects/jdk/12/

Switch Expressions (Preview)

- Preview for switch used as statement or expression

```java
public class Quarter {

    public static void main(String[] args) {

        Month month = Month.SEPTEMBER;

        int quarter = switch (month) {
            case JANUARY, FEBRUARY, MARCH      -> 1;
            case APRIL, MAY, JUNE              -> 2;
            case JULY, AUGUST, SEPTEMBER       -> 3;
            case OCTOBER, NOVEMBER, DECEMBER   -> 4;
        };

        System.out.println("--> Q" + quarter);
    }
}
```

--> Q3

- JEP 325 - Switch Expressions (Preview):
  http://openjdk.java.net/jeps/325

@CGuntur

34

Compact Number Formatting

```java
public class TwitterStats {

    public static void main(String[] args) {
        final Locale locale = new Locale(language: "en", country: "US");

        NumberFormat impressions = NumberFormat
                .getCompactNumberInstance(locale: locale, formatStyle: SHORT);
        impressions.setMaximumFractionDigits(1);


        NumberFormat followers = NumberFormat
                .getCompactNumberInstance(locale: locale, formatStyle: LONG);
        followers.setMaximumFractionDigits(2);

        System.out.println("~ "
                + impressions.format(number: 76092) + " impressions & "
                        + followers.format(number: 2011) + " followers");
    }
}
```

`~ 76.1K impressions & 2.01 thousand followers`

- JDK-8188147 - Compact Number Formatting support:
  https://bugs.java.com/bugdatabase/view_bug.do?bug_id=8188147

@CGuntur

Shenandoah Garbage Collector

- **Experimental** Tiny & consistent pause-time garbage collector

- Main features:
  - ✤ Adds an indirection pointer to every Java object
  - ✤ Concurrent marking and compacting
  - ✤ Optimized to never interrupt the running program
  - ✤ Pause Java threads only to scan thread stacks to find & update object graph

- JEP 189 - Shenandoah: A Low-Pause-Time Garbage Collector:
  http://openjdk.java.net/jeps/189

@CGuntur

36

Microbenchmark Suite

● Based on the Java Microbenchmark Harness (JMH)

● Eases the addition, build and running of new benchmarks

● Targeted for continuous performance testing

● JEP 230 - Microbenchmark Suite:
http://openjdk.java.net/jeps/230

@CGuntur

Other Features

- JEP-350: Java 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

Java 13

#AllGreenWithJavaThirteen

Java 13

- Released: Sep. 2019

- Marked as a feature-release, not LTS

- https://openjdk.java.net/projects/jdk/13/

Dynamic CDS Archives

- Extends AppCDS to archive at application exit

- Further speeds up startup times

- CDS and AppCDS were static, Dynamic CDS is an addition

- Use -XX:ArchiveClassesAtExit=my.jsa - to enable
  Use -XX:SharedArchiveFile=my.jsa - to load

- JEP 350 - Dynamic CDS Archives:
  http://openjdk.java.net/jeps/350

Switch Expressions (Preview)

- 2nd `Preview` for switch used as statement or expression

```java
public class Quarter {

    public static void main(String[] args) {

        Month month = Month.SEPTEMBER;

        int quarter = switch (month) {
            case JANUARY, FEBRUARY, MARCH:    yield 1;
            case APRIL, MAY, JUNE:            yield 2;
            case JULY, AUGUST, SEPTEMBER:     yield 3;
            case OCTOBER, NOVEMBER, DECEMBER: yield 4;
        };

        System.out.println("--> Q" + quarter);
    }
}
```

`--> Q3`

- JEP 354 - Switch Expressions (Preview):
  http://openjdk.java.net/jeps/354

@CGuntur

42

Text Blocks (Preview)

- Preview for multi-line text blocks

```java
@Test
void noTalksYet() {
    String expected = """
        <conference name="OracleCodeOne">
        </conference>
        """;

    String actual = XmlHelper.generateXml();

    assertEquals(expected: expected, actual: actual);
}
```

- JEP 355 - Text Blocks (Preview):
  http://openjdk.java.net/jeps/355

@CGuntur

43

Other Features

- JEP-351: ZGC: Uncommit Unused Memory

- JEP-353: Reimplement the Legacy Socket API

Java 14

#AllTestsGreenWithJavaFourteen

Java 14

- Released: Mar. 2020

- Marked as a feature-release, not LTS

- Many new and exciting features !!!

- https://openjdk.java.net/projects/jdk/14/

Pattern Matching for instanceof (Preview)

- Replaces the need to cast once instance is confirmed

- Example:

```java
if (obj instanceof String s) {
    // can use s here
} else {
    // can't use s here
}
```

- JEP 305 - Pattern matching for instanceof (Preview):
  http://openjdk.java.net/jeps/305

@CGuntur

47

Helpful NullPointerExceptions

- Offer helpful information about the premature termination of a program

- Example:

    a.b.c.i = 99;

- Results in:
  Exception in thread "main" java.lang.NullPointerException:
  Cannot read field "c" because "a.b" is null at Prog.main(Prog.java:5)

- JEP 358 - Helpful NullPointerExceptions:
  http://openjdk.java.net/jeps/358

@CGuntur

48

Algebraic Data Types - Records

- Introduction of algebraic data types into Java, with records

- Immutable data objects with built in constructors and accessors.

```
record Range(int lo, int hi) {
  public Range {
    if (lo > hi)  /* referring here to the implicit constructor parameters */
      throw new IllegalArgumentException(String.format("(%d,%d)", lo, hi));
  }
}
```

- JEP 359 - Records (Preview):
  https://openjdk.java.net/jeps/359

@CGuntur

Other Features

- JEP-351: ZGC: Uncommit Unused Memory

- JEP-353: Reimplement the Legacy Socket API

THANK YOU!