

# **Java New Feature Overview**

**From Java 8 to Java 11 to Java 17**

**CF, 21.03.2021**

# Process: JEP

## Java Enhancement Proposal

- JEP 0: Index
  - <http://openjdk.java.net/jeps/0>
  - Nice list to check many changes.
- Java 11-17: JEP 325, 334, 338 - 398
- Mark Reinhold

# Java Support Roadmap

## LTS vs MTS

- LTS: Long Term Support (Java 8, Java 11, Java 17)
  - Java 8: 03.2014 - 03.2022, extended support: 12.2030.
  - Java 11: 09.2018 - 09.2023, extended support: 09.2026
  - Java 17: 09.2021 - 09.2026, extended support: 09.2029
- MTS: Medium Term Support (9-10, 12-16)
- New Features appear in MTS versions as “preview”.
- <https://www.oracle.com/java/technologies/java-se-support-roadmap.html>

# JDK

## OracleJDK versus OpenJDK

- Oracle: Commercial use requires License
- OpenJDK: OpenSource, no support.
- Azul: Fast VM.

# Java 8

# Java 8

## References

- <http://openjdk.java.net/jeps/0>
- <https://www.baeldung.com/java-11-new-features>

# Java 8

- Reminder for some features of Java 8
- `try-with-resource` **and** `AutoCloseable`
  - <https://docs.oracle.com/javase/tutorial/essential/exceptions/tryResourceClose.html>
-

# Java 11



# Java 11

## References

- <http://openjdk.java.net/jeps/0>
- <https://www.baeldung.com/java-11-new-features>

# Java 11

- String methods
  - `isBlank`, `strip`, `stripLeading`, `stripTrailing`, `repeat`.
- File methods (`java.nio.Files`)
  - `readString` and `writeString` as static methods to `Files`.
- Conversion between List and Array:
  - `List stringList = Arrays.asList("a","b","c")`
  - `String[] array = stringList.toArray(String[]::new)`

# Java 11

## Local Variable Syntax for Lambda Expressions

- Local Variable Syntax: You may use the word `var` and the type is inferred.
- Type Inference

```
var str = "Java 11";  
var list = new ArrayList<String>();
```

- Local Variable Syntax may be used in Lambda expressions

```
(var a, var b) -> a + b
```

- May be used to specify an annotation within a Lambda expression

# Java 11

## Local Variable Syntax for Lambda Expressions

- May be used to specify an annotation within a Lambda expression

```
@NonNull var x = new Foo();
```

or

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

# Execution: Implementing against an Interface

```
Shape shape = new Circle();  
shape.rotate(-90);
```

# Java 11

## Nest-Based Access Control

- <http://openjdk.java.net/jeps/181>
- <https://www.baeldung.com/java-nest-based-access-control>
- The change affects the internal way of calling private methods within an “nest” (outer class hosting inner classes) and adds functionality for reflection (getNestHost, getNestMember)
- See code example.

# Java 12

# Java 12

- Teeing Collector for Java Streams

```
Collector<T, ?, R> teeing(  
    Collector<? super T, ?, R1> downstream1,  
    Collector<? super T, ?, R2> downstream2,  
    BiFunction<? super R1, ? super R2, R> merger)
```

## Usage:

```
@Test  
public void givenSetOfNumbers_thenCalculateAverage() {  
    double mean = Stream.of(1, 2, 3, 4, 5)  
        .collect(Collectors.teeing(Collectors.summingDouble(i -> i),  
            Collectors.counting(), (sum, count) -> sum / count));  
    assertEquals(3.0, mean);  
}
```



# Java 12

- Switch Expressions (JEP 354) (will be continued in Java 13)
- See below.

# Java 13

# Java 13

## References

- <http://openjdk.java.net/jeps/0>
-

# Java 13

## Switch Expressions (JEP 354) / Keyword yield

- There is a new keyword: yield
- The switch statement can be considered as a function that returns a value.
- Instead of return, yield is used.

# Java 13

## Text Blocks

- Improve the inline creation of String that newline and double quote (“).
- Useful for inline creation of JSON.
- Maybe useful in specifying input data for Unit Tests.

# Java 14

# Java 14

## References

- <https://www.baeldung.com/java-14-new-features>
-

# Java 14

## instanceOf Pattern Matching (JEP 305) (PREVIEW)

- Automatically casts an object inside an `if(object instanceof ...) { }` block to the respective type.
- Simplifies syntax. Still explicit (compare this with Kotlin).
- Instead of
- write



# Java 14

## record (JEP 359) (PREVIEW)

- Records are specifically to streamline the definition of POJOs / DTO.

```
public record User(int id, String password) { };
```

- Defines a class with fields id and password with
  - getters, setters, toString(), equals(), hashCode()
- See [code example](#).

# Java 15

# Java 15

## Sealed Class (JEP 360, 397)

- Will be continued in Java 16.

# Java 16

# Java 16

## References

- <https://www.techgeeknext.com/java/java16-features>
- <https://www.infoworld.com/article/3569150/jdk-16-the-new-features-in-java-16.html>
- <https://docs.oracle.com/en/java/javase/16/language/java-language-changes.html>
-

# Java 16

## Sealed Class (JEP 360, 397)

- Fine grained control over inheritance.
- A class A can specify explicitly which outer classes are allowed to use  
`extends A`

```
public sealed class Shape permits Circle, Square, Rectangle {  
  
}
```

- Classes extending the sealed class need to be in the same module/package.

# Java 16

## Sealed Class (JEP 360, 397)

- <https://docs.oracle.com/en/java/javase/16/language/sealed-classes-and-interfaces.html>
- <https://cr.openjdk.java.net/~briangoetz/amber/datum.html>

# Java 17



# Java 17

- <https://www.techgeeknext.com/java/java17-features>
-

**Outlook (beyond 17)**

# Outlook

- JEP 402: Unify the Basic Primitives with Objects (Preview)
  - <http://openjdk.java.net/jeps/402>
- JEP 169: Value Objects
  - <http://openjdk.java.net/jeps/169>

# Draft: Concise Method Bodies

- <http://openjdk.java.net/jeps/8209434>
-