

# Java 9 - 16

What happened the last 7 years

Matthias

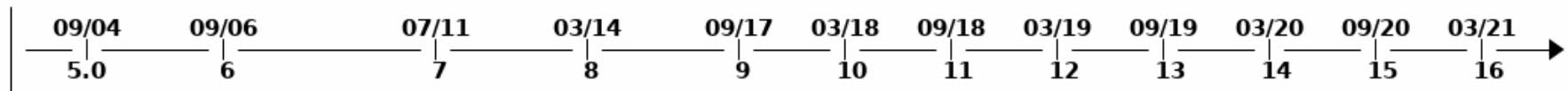
# Disclaimer

- Part I (tried to focus on language Features & API)
  - maybe following parts with
    - java "internal" changes (e.g. Garbage Collector, Performance, tooling)
    - deprecations
- no guarantee on completeness

# Sources

- [OpenJDK](#)
- [Marcobehler.com](#)
- [.Lost in Coding](#)
- [Advanced Web Machinery](#)

# Java history



# Why Java 8

- rich feature set
  - Stream API (Collections)
  - Lambda Functions & Functional Interfaces
  - Optionals
  - New Date Time API
- changed licensing which lead to confusion (Java Distributions)
- initial problems with commonly used build tools
- is a LTS version

# Java 9

*OpenJDK 9*

# Collection factory methods

```
List<String> list = List.of("one", "two", "three");  
Set<String> set = Set.of("one", "two", "three");  
Map<String, String> map = Map.of("foo", "one", "bar", "two");
```

## Streams

- takeWhile, dropWhile, **iterate** with condition

```
Stream<String> stream =  
    Stream.iterate("", s -> s + "s")  
        .takeWhile(s -> s.length() < 10);
```

## Optionals: ifPresentOrElse()

```
user.ifPresentOrElse(this::displayAccount, this::displayLogin);
```

# JShell (Java Playground)

```
% jshell
| Welcome to JShell -- Version 16.0.1
| For an introduction type: /help intro

jshell> int x = 10
x ==> 10

jshell> System.out.println(x > 5 ? "big number" : "small number")
big number
```

## Exit with

```
jshell> /exit
| Goodbye
```

(You are welcome ;-)



# misc

- TLS1.3 Support
- Java Platform module System
- Flow API (reactive Streams)
- MultiResolutionImage
- Class.getPackageName()

# Java 10

*OpenJDK 10*

# Local-variable type Inference: `var`-keyword

```
// Pre-Java 10  
String myName = "Volker";  
  
// With Java 10  
var myName = "Volker"
```

# Java 11

*OpenJDK 11*

# new methods on `String`

- `repeat()`, `isBlank()`, `strip()`, `lines()`

```
"".isBlank();           // -> true
"Each\nline\ncounts".lines().toList(); // -> Stream of lines
"  we messed it up  ".strip();         // -> "we messed it up"
"Hello".repeat(3)        // -> "HelloHelloHello"
```

## `var` supported by lambda parameters

```
(var firstName, var lastName) -> firstName + lastName
```

# HttpClient (Preview >= 9)

```
``Java import java.io.IOException; import java.net.http.HttpClient;  
import java.net.http.HttpRequest; import java.net.URI; import  
java.net.http.HttpResponse;
```

```
public class HttpClientExample {
```

```
public static void main(String[] args) throws IOException, InterruptedException {  
    HttpClient httpClient = HttpClient.newBuilder().build();  
  
    HttpRequest request =  
        HttpRequest.newBuilder()  
            .uri(URI.create("https://www.google.com/"))  
            .GET()  
            .build();  
  
    HttpResponse<String> response =  
        httpClient.send(request, HttpResponse.BodyHandlers.ofString());  
  
    System.out.println(response.statusCode());  
    System.out.println(response.body());  
}
```

```
}
```

```
</div><!-- .element style="font-size: 0.8em;" -->
```

# Java 12

*OpenJDK 12*

# new methods on `String`

- `indent()`, `transform()`

## CompactNumberFormat class

- formatting numbers for a Local
  - US: 1000 -> 1K, 1000000 -> 1M
  - IN: 1000 -> 1 000 000, 500000000 -> 5 00 00 00 00
  - DE: 1000 -> 1.000, 1000000 -> 1 Mio

## Unicode 11 support



# Java 13

*OpenJDK 13*

Unicode 12.1 support

# Java 14

*OpenJDK 14*

# Switch Expression (Preview: 12, 13)

- new (including new `yield` keyword)
  - no fall-through
  - single statement and code block
  - supporting value return (with `yield`)

```
int numLetters = switch (day) {  
    case MONDAY, FRIDAY, SUNDAY -> 6;  
    case TUESDAY                 -> 7;  
    default                     -> {  
        String s = day.toString();  
        int result = s.length();  
        yield result;  
    }  
};
```

# Java 15

*OpenJDK 15*

# Helpfull NullPointerException message

```
```Java public class NullPointerException {
```

```
public static void main(String[] args) {  
  
    System.out.println("NullPointerException message:");  
  
    MyClass myClass = null;  
    try {  
        System.out.println(myClass.myName.length());  
    } catch (Exception e){  
        e.printStackTrace();  
    }  
  
    myClass = new MyClass();  
    try {  
        System.out.println(myClass.myName.length());  
    } catch (Exception e){  
        e.printStackTrace();  
    }  
}  
  
private static class MyClass{  
    protected String myName;  
    protected MyClass myName;
```

```
}
```

```
</div><!-- .element style="font-size: 0.4em;" -->
```

```
* old  
<div>
```

NullPointerException message: java.lang.NullPointerException: null at  
de.mubn.java15.NullPointerExample.main(NullPointerExample.java:11)  
java.lang.NullPointerException: null at  
de.mubn.java15.NullPointerExample.main(NullPointerExample.java:20)

```
</div><!-- .element style="font-size: 0.8em;" -->  
  
* new  
<div>
```

NullPointerException message: java.lang.NullPointerException: Cannot  
read field "myName" because "myClass" is null at  
de.mubn.java15.NullPointerExample.main(NullPointerExample.java:11)  
java.lang.NullPointerException: Cannot invoke "String.length()" because  
"myClass.myName" is null at  
de.mubn.java15.NullPointerExample.main(NullPointerExample.java:20)

```
</div><!-- .element style="font-size: 0.8em;" -->
```

# Text Blocks (Preview: 13, 14)

- old

```
```Java String html = "\n" + "\n" + "
```

```
Hello, world
```

```
\n" + "\n" + ""`; ``
```

- new

```
```Java String html = ""
```

```
Hello, world
```

```
""; ``
```

# Java 16

*OpenJDK 16*



# Pattern Matching for instanceof

```
if (x instanceof String s) {  
    String a = s;  
}
```

## Record Type

- immutable data Class
- implicit methods -> avoiding boiler plate code
  - getters, constructor, equals, hashCode, toString

```
record Point(int x, int y){ }  
  
Point myPoint = new Point(10, 15);  
  
myPoint.x();           // -> 10  
System.out.println(myPoint); // -> "Point[x=10, y=15]"
```

# Sealed Classes (still PREVIEW)

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