

Programmieren I

Dokumentation mit javadoc



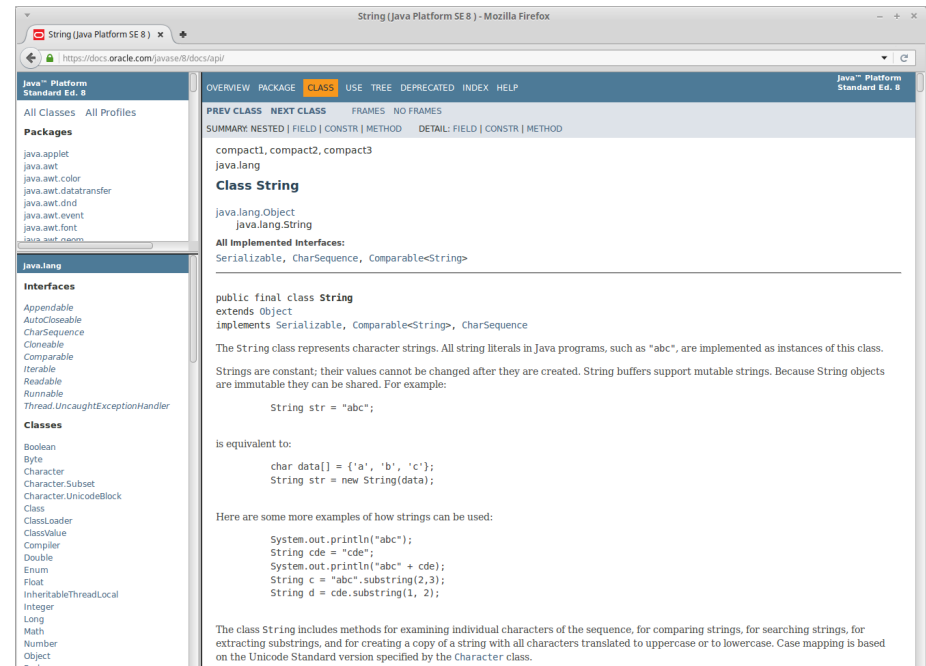
Heusch 10.4
Ratz 4.1.1

Institut für Automation und angewandte Informatik

```
final List<String> allResults = new ArrayList<String>();  
final Map<String, Integer> typeWordResultCount = new HashMap<String, Integer>();  
final Map<String, Integer> typePoints = new HashMap<String, Integer>();  
evaluation.put(type, typePoints);  
  
for (final Sheet sheet : this.sheets) {  
    final String sheetResult = sheet.getPlayerInput(type);  
    if (sheetResult.startsWith(start) && this.isValidWord(sheetResult, type)) {  
        validWordCountForType++;  
        allResults.add(sheetResult);  
    }  
}
```

Automatische Dokumentation

- Java bietet standardmäßig das Dokumentationssystem `javadoc`, das ganze Programmsysteme vollautomatisch dokumentieren kann
- Zur Dokumentation werden Kommentare verwendet;
`javadoc` kann nur dokumentieren, was auch kommentiert wurde!
- Anreicherung der Kommentare durch Platzhalter und Variablen



Kommentar-Typen

■ Einzeilige Kommentare (auch am Zeilenende)

```
// Ich bin ein Kommentar  
System.out.println("Hello World"); // Ausgabe einer Nachricht
```

■ Mehrzeilige Kommentare

```
/* String fractionString = Utils.inputString(System.in);  
 * int pos = fractionString.indexOf("/")  
 */  
  
/* Die Methode implementiert das Kürzen eines  
 * Bruchs mittels GGT-Algorithmus */
```

■ Kommentare zur automatischen Programmdokumentation

```
/** Programmdokumentation zur Klasse Fraction */  
public class Fraction {  
    /** Doku zur Variable numerator */  
    public int numerator;  
    /** Doku zur Methode add(Fraction) */  
    public Fraction add(Fraction f) { /*...*/ }  
}
```

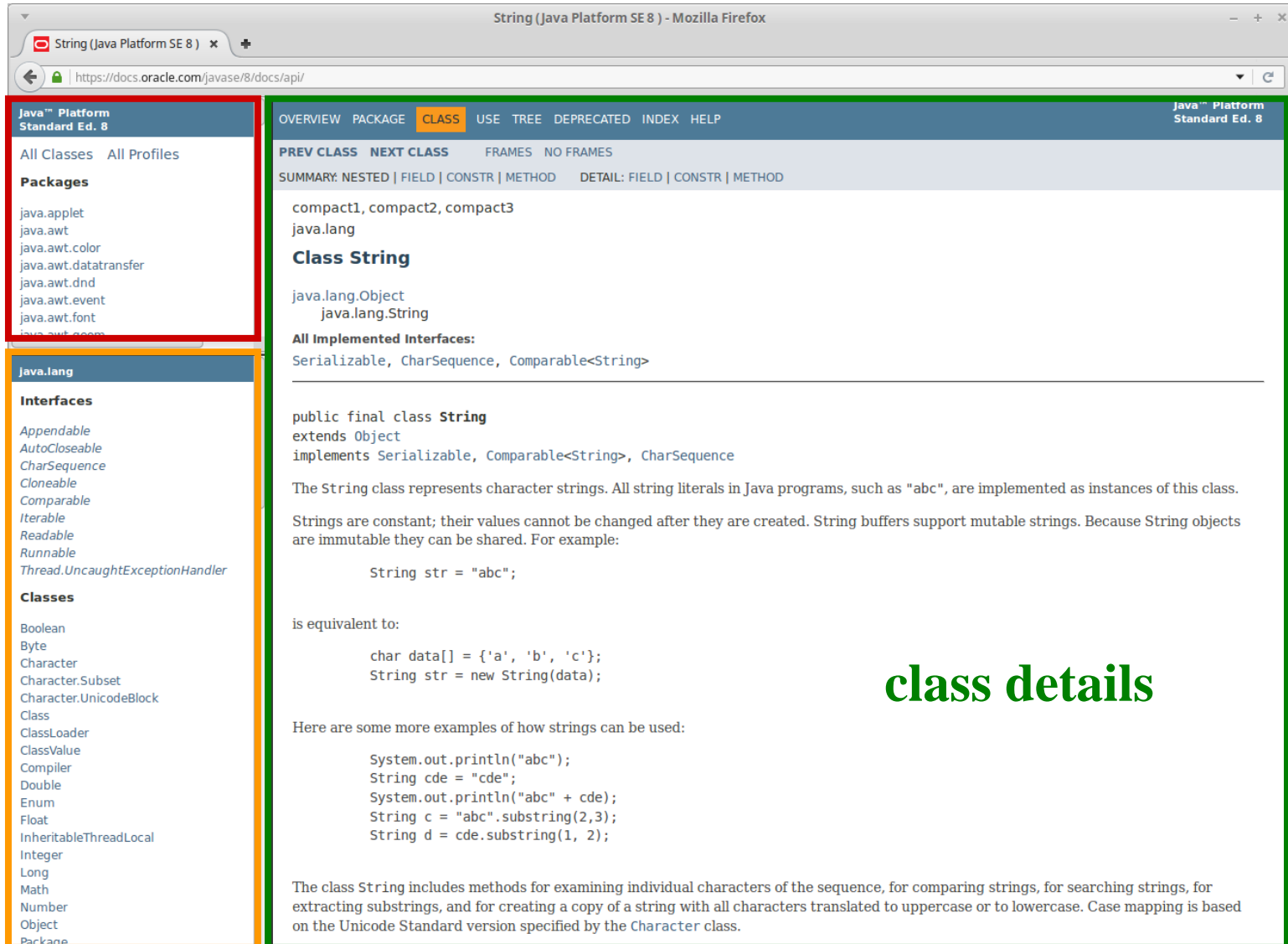
javadoc

- Aus Kommentaren, die mit `/**` beginnen und mit `*/` enden, erzeugt `javadoc` eine automatische Dokumentation im HTML-Format.
- `javadoc` erstellt standardmäßig nur für `public`- und `protected`-Elemente eine Dokumentation.
- Mit der `-private` Option von `javadoc` kann auch für `private`-Elemente eine Dokumentation erstellt werden.

Javadoc-Output (z.B. <http://docs.oracle.com/javase/8/docs/api/>)

packages

interfaces and classes



The screenshot shows the Java Platform SE 8 API documentation for the `String` class. The left sidebar is divided into two sections: "packages" (highlighted in red) and "interfaces and classes" (highlighted in orange). The "packages" section lists various Java packages like `java.applet`, `java.awt`, etc. The "interfaces and classes" section lists interfaces like `Appendable`, `AutoCloseable`, etc., and classes like `Boolean`, `Byte`, etc. The main content area is titled "Class String" and includes the following information:

- Overview:** `compact1, compact2, compact3`, `java.lang`
- Class String:** `java.lang.Object`, `java.lang.String`
- All Implemented Interfaces:** `Serializable`, `CharSequence`, `Comparable<String>`
- Class Definition:**

```
public final class String
    extends Object
    implements Serializable, Comparable<String>, CharSequence
```
- Description:** The `String` class represents character strings. All string literals in Java programs, such as `"abc"`, are implemented as instances of this class. Strings are constant; their values cannot be changed after they are created. String buffers support mutable strings. Because `String` objects are immutable they can be shared. For example:

```
String str = "abc";
```
- Equivalent to:**

```
char data[] = {'a', 'b', 'c'};
String str = new String(data);
```
- Examples:** Here are some more examples of how strings can be used:

```
System.out.println("abc");
String cde = "cde";
System.out.println("abc" + cde);
String c = "abc".substring(2,3);
String d = cde.substring(1, 2);
```
- Summary:** The class `String` includes methods for examining individual characters of the sequence, for comparing strings, for searching strings, for extracting substrings, and for creating a copy of a string with all characters translated to uppercase or to lowercase. Case mapping is based on the Unicode Standard version specified by the `Character` class.

class details

Programmdokumentation

■ Embedded HTML

- Beliebiger HTML-Code kann in die Dokumentation integriert werden.

■ Beispiel:

```
/**
 * <pre>
 * System.out.println(new Date());
 * </pre>
 */
public static void main(String[] args) {
    System.out.println(new Date());
}
```

Beispiel: Klasse und Methode dokumentieren

```
/**
 * Hello-World-Programm in Java.
 * Dies ist ein Javadoc-Kommentar
 *
 * @author John Doe
 * @version 0.9
 * @since 0.1
 */
public class HelloWorld {
    /**
     * Hauptprogramm
     *
     * @param args Kommandozeilen-Parameter
     */
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

Die wichtigsten @-Elemente

- **@see** - erlaubt Referenz auf die Dokumentation anderer Klassen

```
@see Klassenname  
@see Klassenname#Methodenname
```

```
/** Beispiel  
 * @see java.lang.Object  
 */
```

- **@author** - Wer hat das verbockt?
- **@version** - Version der Methode/Programms/Klasse/...
- **@return** - Was ist die Bedeutung des Rückgabewertes

```
@return Beschreibung
```

- **@param** - Welche Parameter werden übergeben

```
@param Parametername Beschreibung
```

- **@throws**, **@exception** - Welche Ausnahmen sind möglich

```
@throws Ausnahme-Klassenname Beschreibung
```


Weitere @-Elemente

- @deprecated - veraltete Methoden markieren

```
@deprecated As of JDK 1.1, replaced by  
    {@link #setBounds(int,int,int,int)} */
```

- @since - wenn ein Feld oder eine Methode ab einer bestimmten Version verfügbar ist

```
@since 1.5
```

- @serial, @serialData, @serialField - für serialisierbare Attribute
- {@value} - zum Anzeigen des Wertes eines statischen Feldes

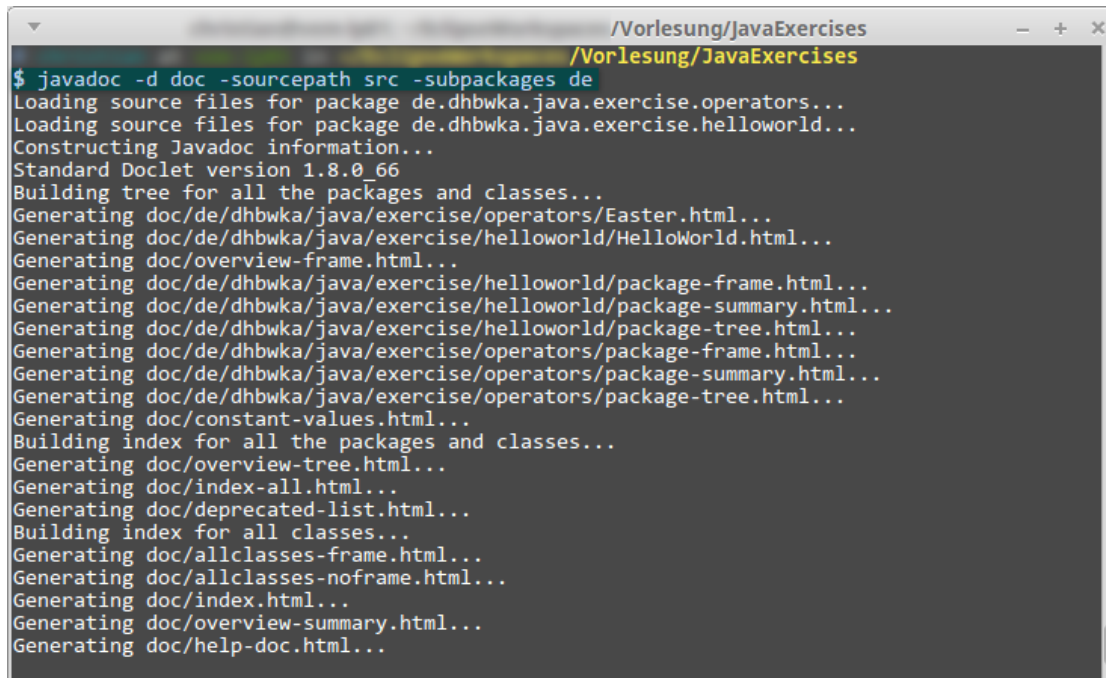
- Vollständige Beschreibung:

<http://docs.oracle.com/javase/8/docs/technotes/guides/javadoc/index.html>

Anwendung von javadoc (1) - Kommandozeile

- Generieren der Dokumentation mit dem Kommandozeilen-Tool `javadoc` (Bestandteil des JDK)

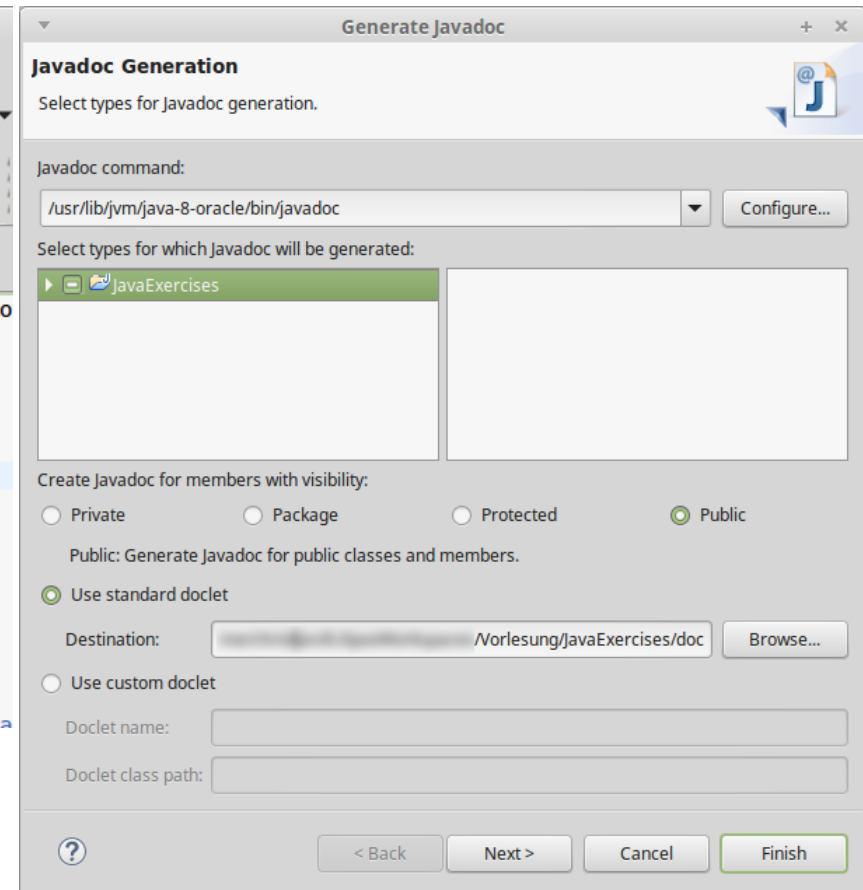
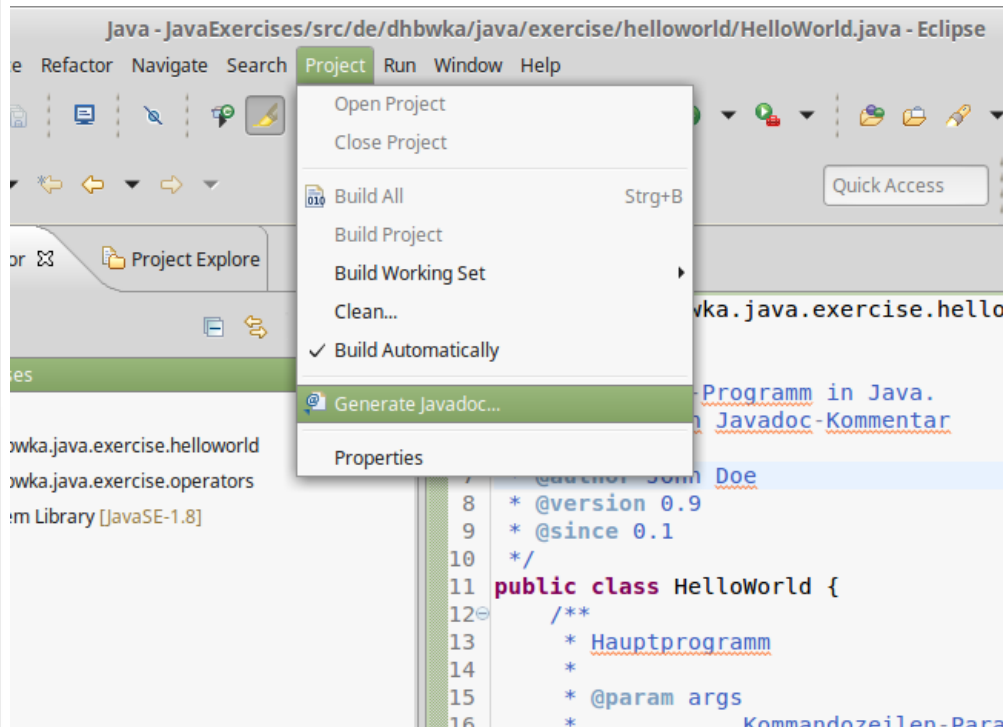
```
> javadoc *.java # im Sourceverzeichnis  
> javadoc -d doc -sourcepath src -subpackages de # im Projektverzeichnis
```



```
/Vorlesung/JavaExercises  
$ javadoc -d doc -sourcepath src -subpackages de  
Loading source files for package de.dhbwka.java.exercise.operators...  
Loading source files for package de.dhbwka.java.exercise.helloworld...  
Constructing Javadoc information...  
Standard Doclet version 1.8.0_66  
Building tree for all the packages and classes...  
Generating doc/de/dhbwka/java/exercise/operators/Easter.html...  
Generating doc/de/dhbwka/java/exercise/helloworld/HelloWorld.html...  
Generating doc/overview-frame.html...  
Generating doc/de/dhbwka/java/exercise/helloworld/package-frame.html...  
Generating doc/de/dhbwka/java/exercise/helloworld/package-summary.html...  
Generating doc/de/dhbwka/java/exercise/helloworld/package-tree.html...  
Generating doc/de/dhbwka/java/exercise/operators/package-frame.html...  
Generating doc/de/dhbwka/java/exercise/operators/package-summary.html...  
Generating doc/de/dhbwka/java/exercise/operators/package-tree.html...  
Generating doc/constant-values.html...  
Building index for all the packages and classes...  
Generating doc/overview-tree.html...  
Generating doc/index-all.html...  
Generating doc/deprecated-list.html...  
Building index for all classes...  
Generating doc/allclasses-frame.html...  
Generating doc/allclasses-noframe.html...  
Generating doc/index.html...  
Generating doc/overview-summary.html...  
Generating doc/help-doc.html...
```

Anwendung von javadoc (2) - Eclipse

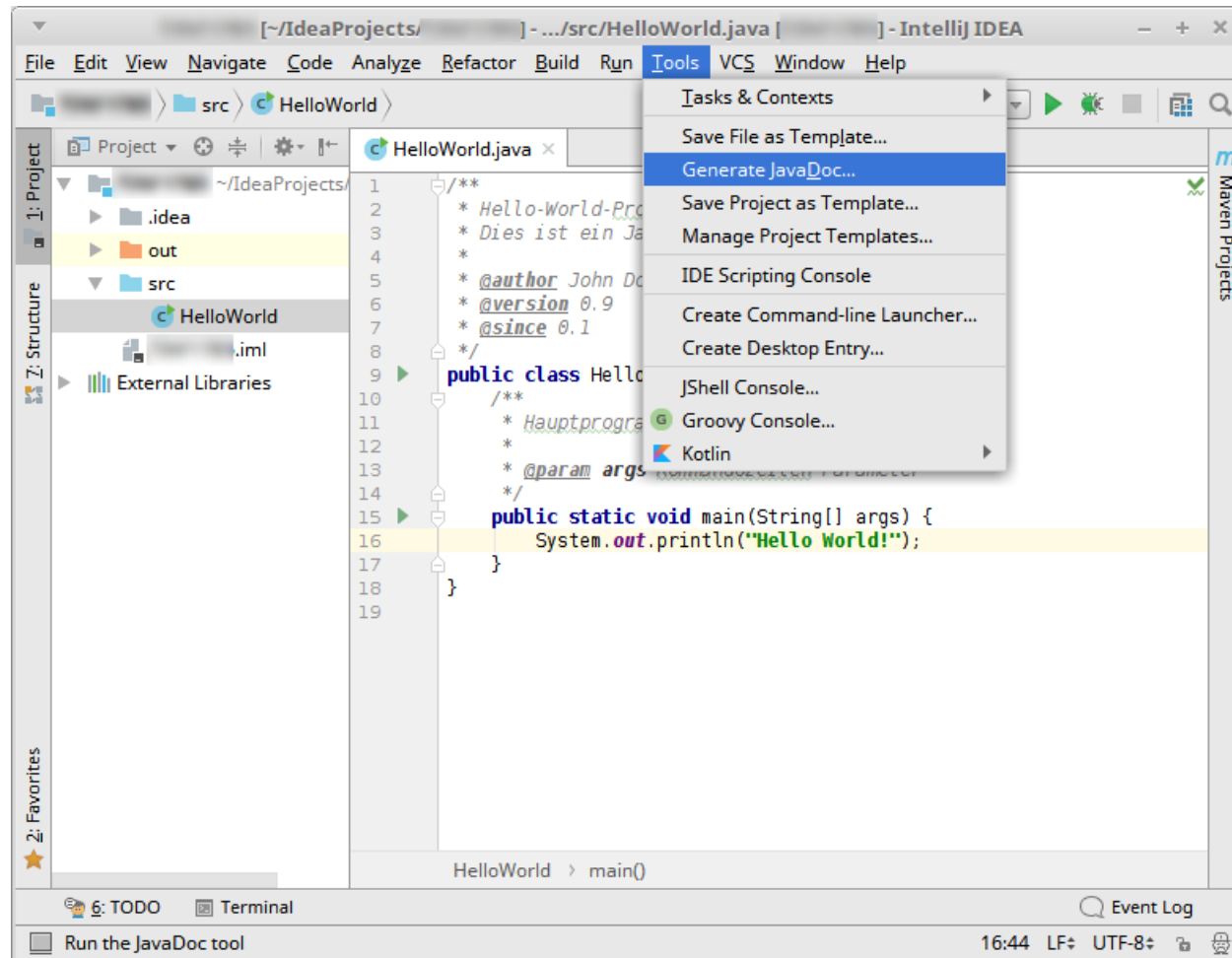
- Entwicklungsumgebungen stellen Funktionalität zur Generierung von Javadoc bereit



Project > Generate Javadoc...



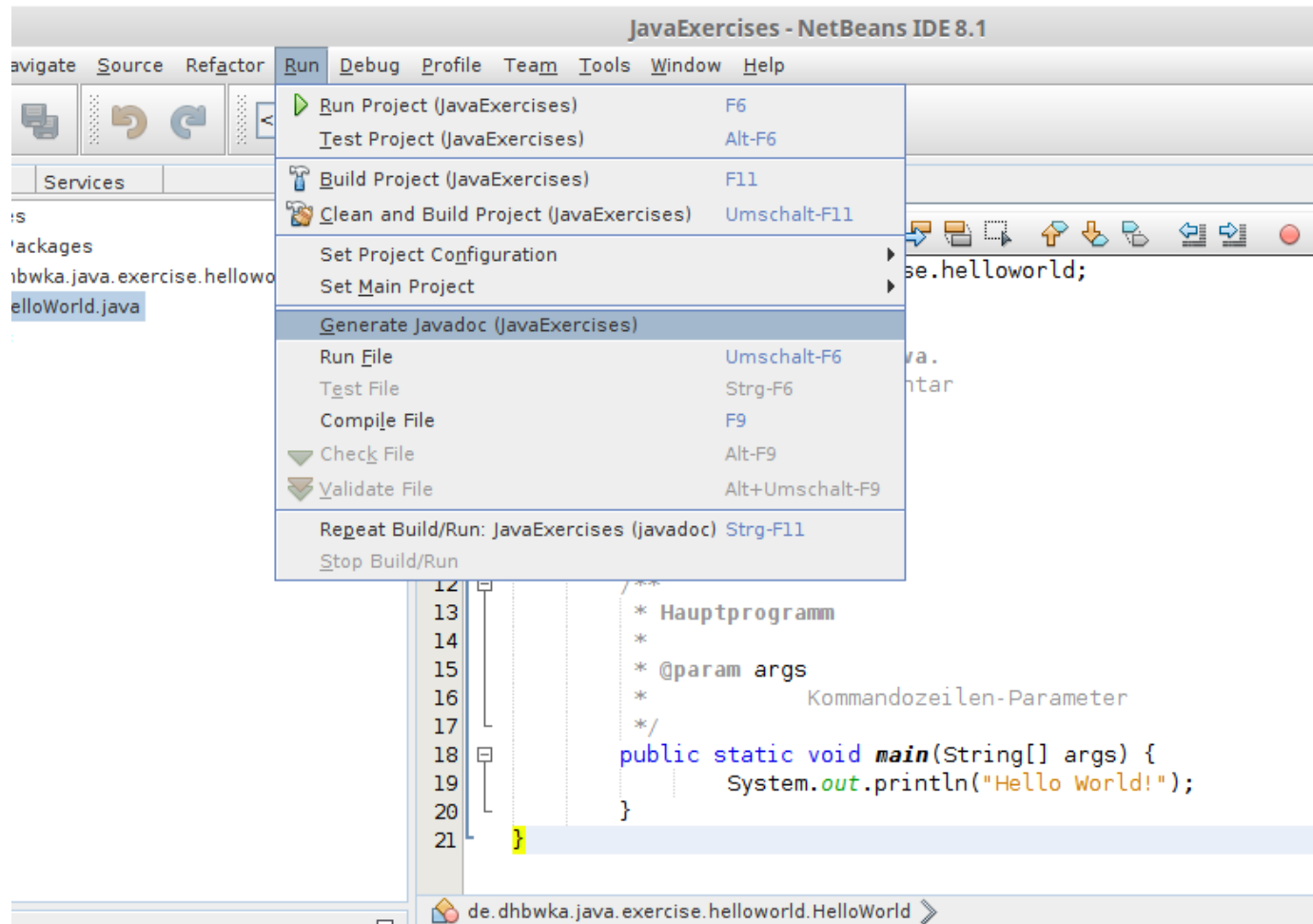
Anwendung von javadoc (3) – IntelliJ IDEA



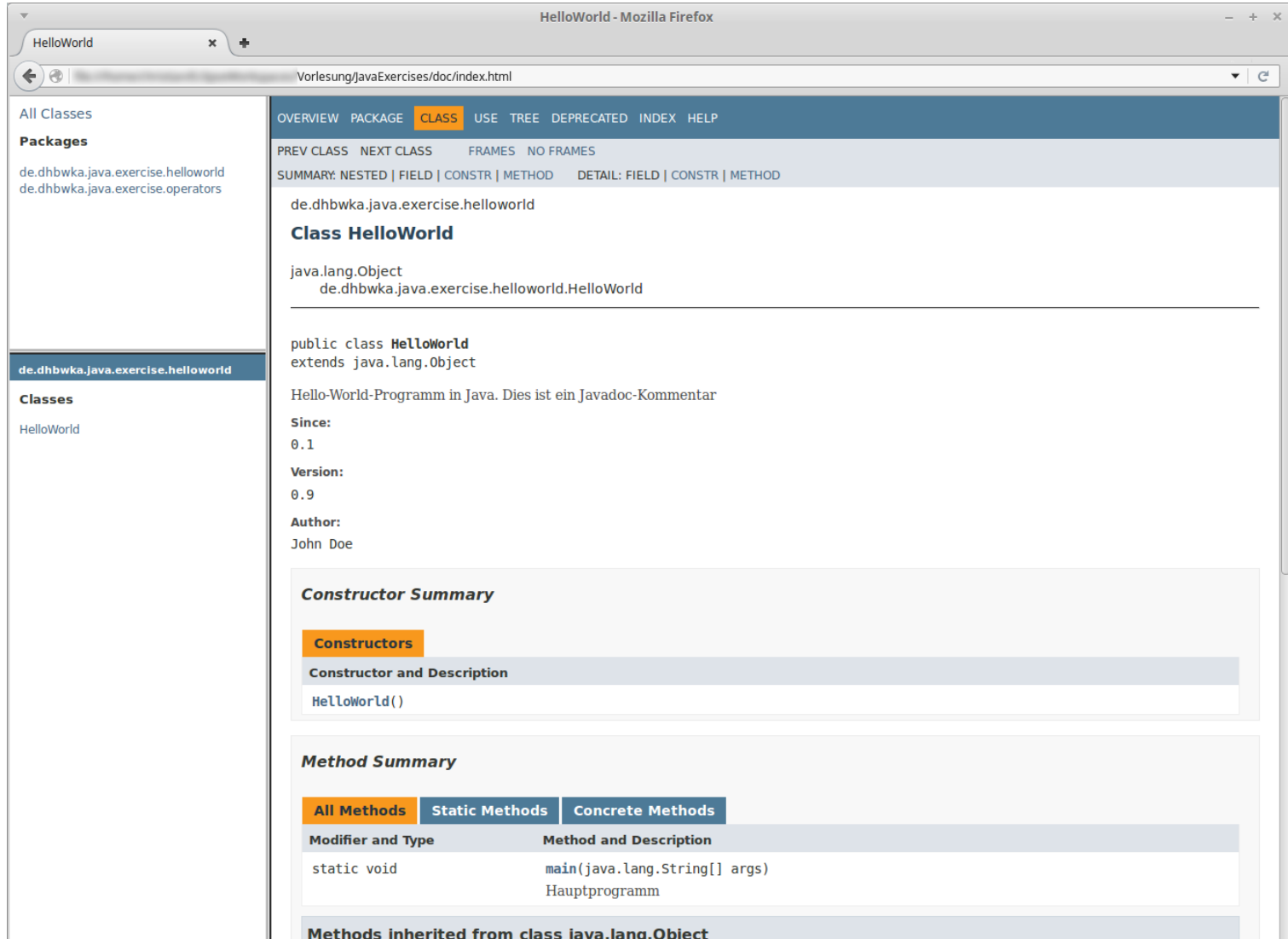
Tools > Generate JavaDoc



Anwendung von javadoc (4) - NetBeans



Ergebnis



The screenshot shows a web browser window titled "HelloWorld - Mozilla Firefox" displaying a JavaDoc page for the `HelloWorld` class. The page is organized into a sidebar on the left and a main content area on the right.

Sidebar:

- All Classes**
- Packages**
 - `de.dhbwka.java.exercise.helloworld`
 - `de.dhbwka.java.exercise.operators`
- Classes**
 - `HelloWorld`

Main Content Area:

Navigation tabs: OVERVIEW, PACKAGE, **CLASS**, USE, TREE, DEPRECATED, INDEX, HELP

Navigation links: PREV CLASS, NEXT CLASS, FRAMES, NO FRAMES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

de.dhbwka.java.exercise.helloworld

Class HelloWorld

java.lang.Object
de.dhbwka.java.exercise.helloworld.HelloWorld

public class **HelloWorld**
extends java.lang.Object

Hello-World-Programm in Java. Dies ist ein Javadoc-Kommentar

Since:
0.1

Version:
0.9

Author:
John Doe

Constructor Summary

Constructors

Constructor and Description
<code>HelloWorld()</code>

Method Summary

All Methods	Static Methods	Concrete Methods
Modifier and Type	Method and Description	
static void	<code>main(java.lang.String[] args)</code> Hauptprogramm	

Methods inherited from class java.lang.Object

Anhang – Offline-Konfiguration für IDE

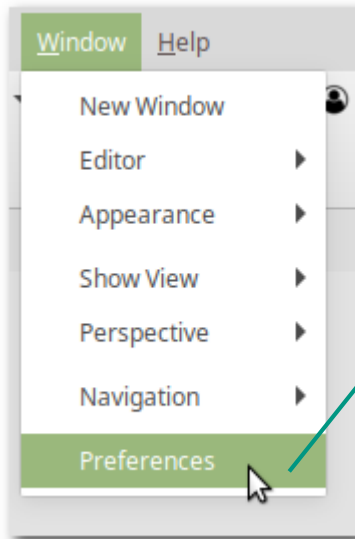
- Normalerweise Javadoc für Standard-Klassen von IDE automatisch über das Internet verfügbar
 - In Prüfungssituation ungünstig → kein Internet erlaubt
 - Möglichkeit der Offline-Konfiguration
 - 2 Möglichkeiten für Offline-Verfügbarkeit von Javadoc
 - Variante 1: Java-Quellcode (src.zip) verknüpfen
 - Variante 2: Javadoc-Archiv verknüpfen
- eine von beiden Varianten genügt für Javadoc,
für Debugging aber ist Variante 1 zu bevorzugen

Anhang – Variante 1: Quellcode-Archiv-Pfad

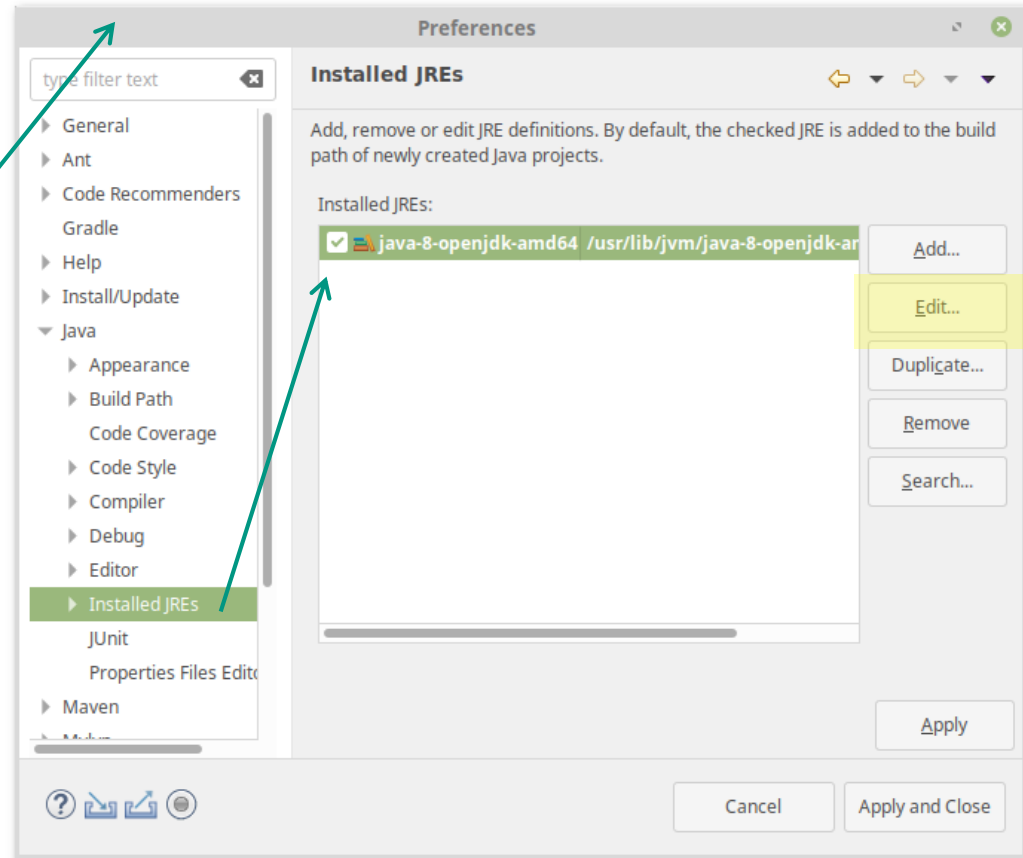
■ Variante 1: Source-Code-ZIP des JDKs

- Bei installiertem JDK im Installationsverzeichnis dabei, bspw:
 - Gängige Pfade für JDK-Verzeichnis (am Beispiel Java 8 Update 192):
 - **Windows:**
C:\Program Files\Java\jdk1.8.0_192\
C:\Program Files (x86)\Java\jdk1.8.0_192\
 - **Linux:**
/usr/lib/jvm/java-1.8.0-openjdk-amd64/ (*openJDK*)
/usr/lib/jvm/java-8-oracle/ (*Oracle JDK*)
 - **MacOS:**
/Library/Java/JavaVirtualMachines/jdk1.8.0_192.jdk/Contents/Home
 - Source-Code-Archiv liegt dann entweder unter
 - \$JDK_DIR/src.zip (bis Java 8)
 - \$JDK_DIR/lib/src.zip (ab Java 9)
- Javadoc der Standard-Klassen ist Teil des Quellcodes
→ kann von der IDE ausgewertet werden

Anhang – Variante 1 in Eclipse (1)

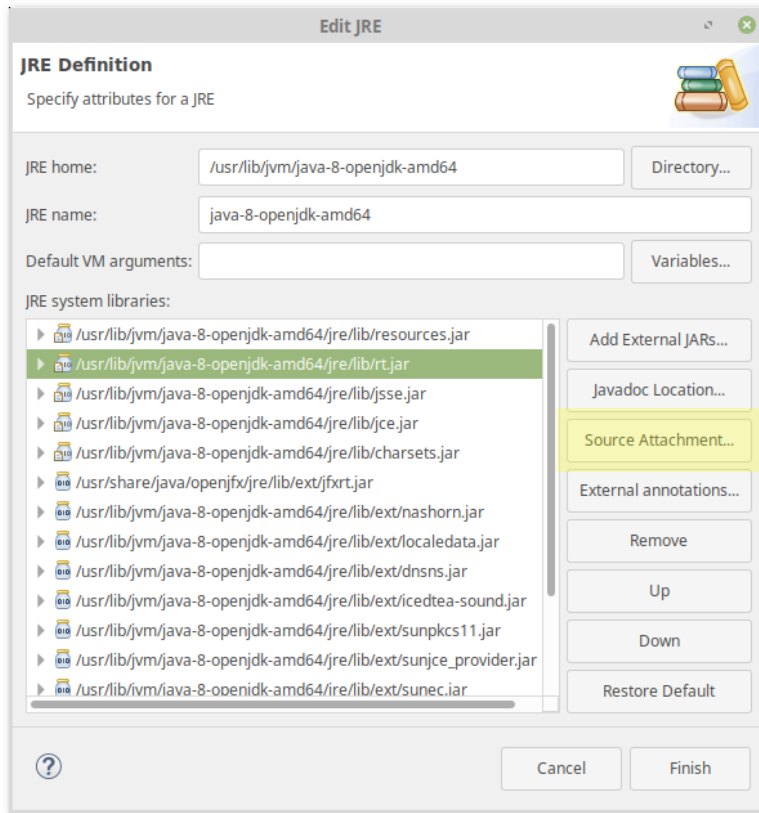


Window > Preferences



Installed JREs > JRE/JDK auswählen > Edit

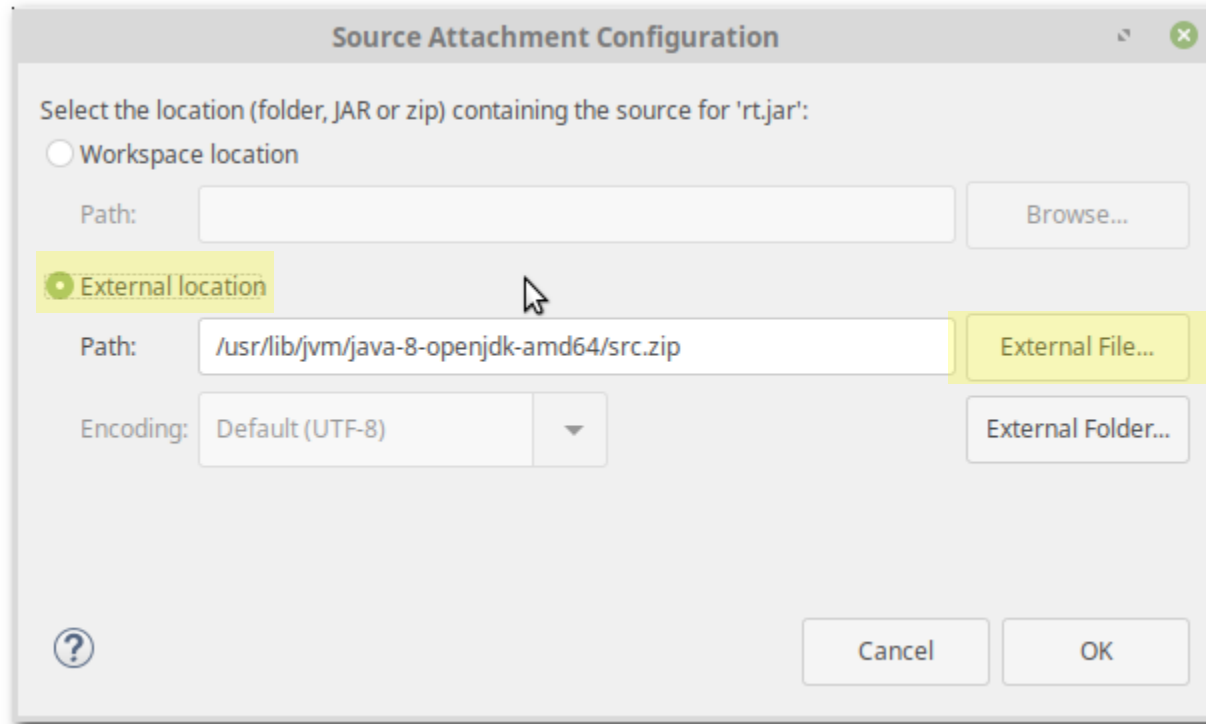
Anhang – Variante 1 in Eclipse (2)



`rt.jar`: bis Java 8
`jrt-fs.jar` ab Java 9

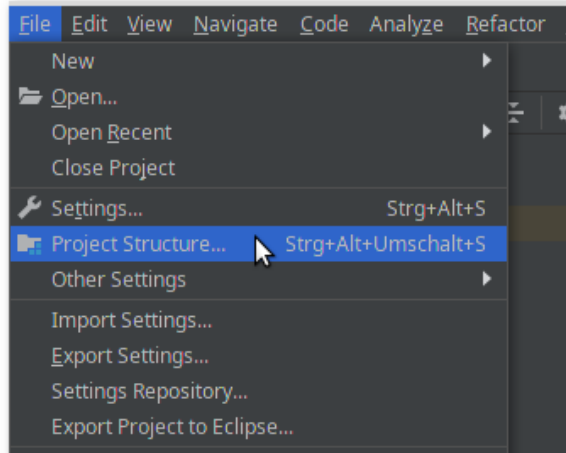
In der Liste `rt.jar/jrt-fs.jar` auswählen > Source Attachment

Anhang – Variante 1 in Eclipse (3)

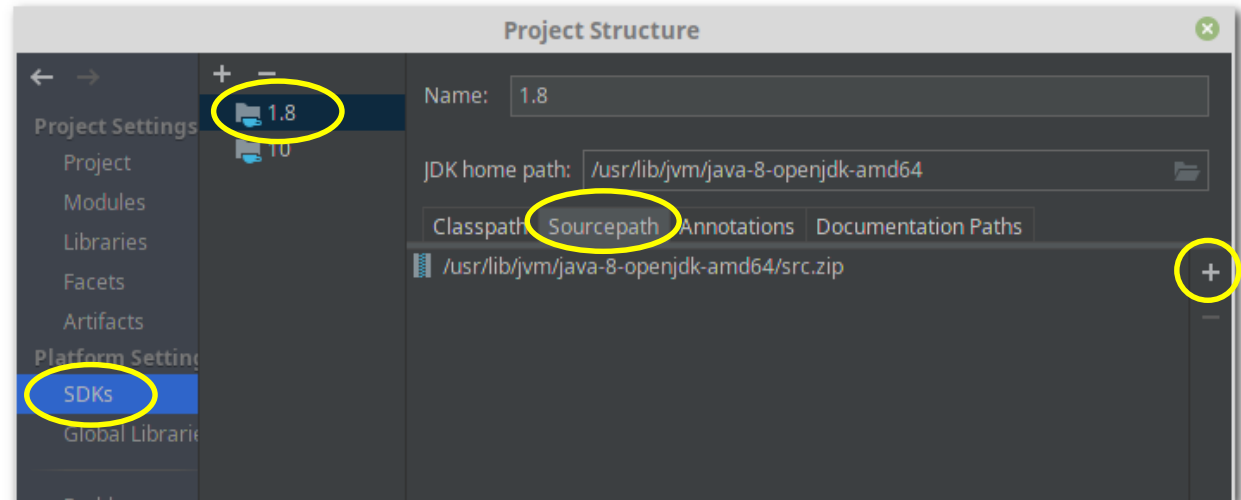


External location > External File... > src.zip-Datei auswählen

Anhang – Variante 1 in IntelliJ



File > Project Structure...



SDKs > 1.8 > Sourcepath-Tab

Normalerweise ist src.zip bereits automatisch verknüpft. Sollte das nicht der Fall sein, kann es mit „+“ (rechter Rand) entsprechend hinzugefügt werden.

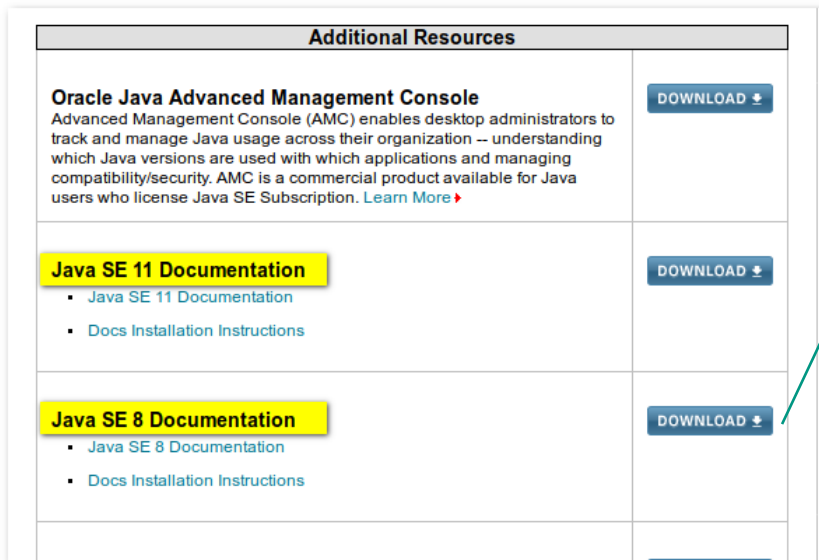
Anhang – Variante 2: Javadoc-Archiv

■ Javadoc-ZIP muss heruntergeladen werden

- <http://java.oracle.com> > Top Downloads > JavaSE

(aktueller Direktlink: <https://www.oracle.com/technetwork/java/javase/downloads/index.html>)

- Unter „Additional Resources“ gibt es dann die Downloads für die aktuell unterstützten Versionen Java 8 und 11:

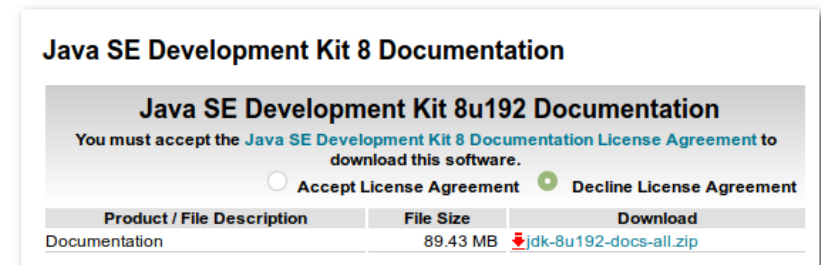


Additional Resources

Oracle Java Advanced Management Console
Advanced Management Console (AMC) enables desktop administrators to track and manage Java usage across their organization – understanding which Java versions are used with which applications and managing compatibility/security. AMC is a commercial product available for Java users who license Java SE Subscription. [Learn More](#) ▶

Java SE 11 Documentation
• [Java SE 11 Documentation](#)
• [Docs Installation Instructions](#)

Java SE 8 Documentation
• [Java SE 8 Documentation](#)
• [Docs Installation Instructions](#)



Java SE Development Kit 8 Documentation

Java SE Development Kit 8u192 Documentation

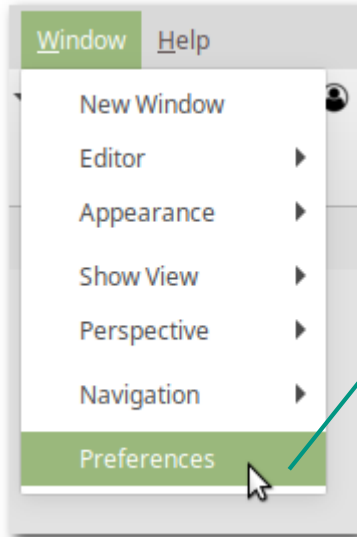
You must accept the [Java SE Development Kit 8 Documentation License Agreement](#) to download this software.

☐ Accept License Agreement ☒ Decline License Agreement

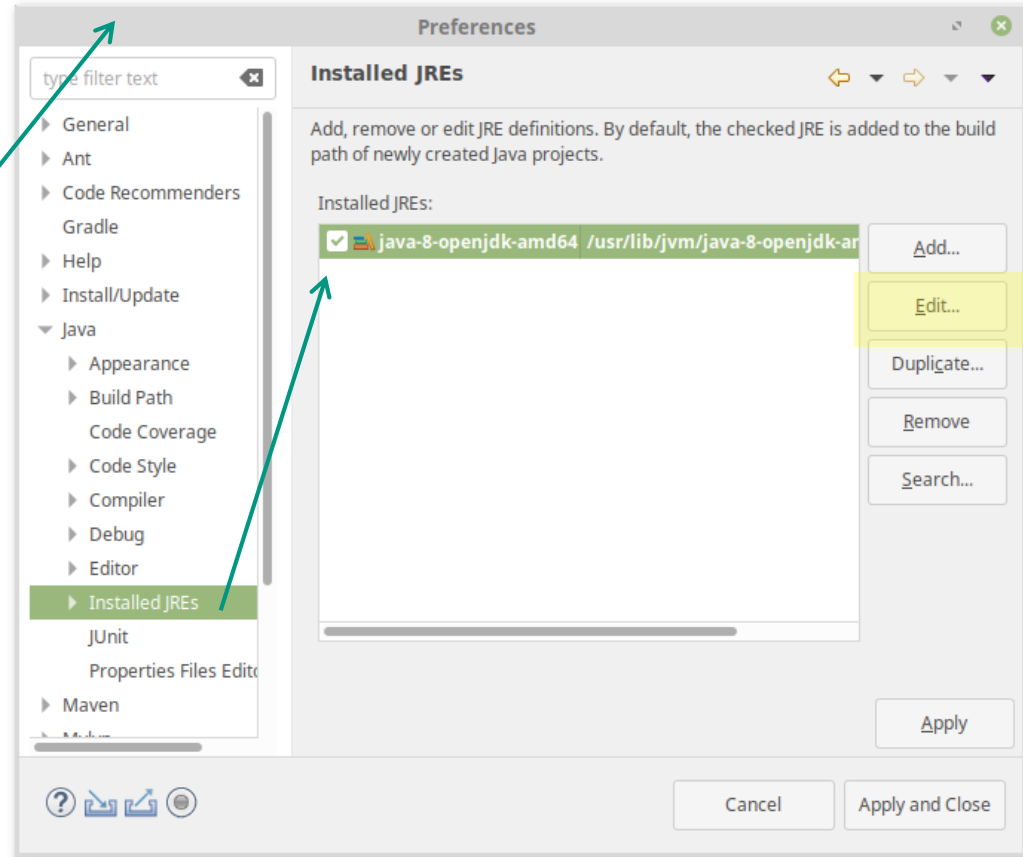
Product / File Description	File Size	Download
Documentation	89.43 MB	jdk-8u192-docs-all.zip

Zum Herunterladen muss analog zum JDK der entsprechenden Lizenz zugestimmt werden

Anhang – Variante 2 in Eclipse (1)

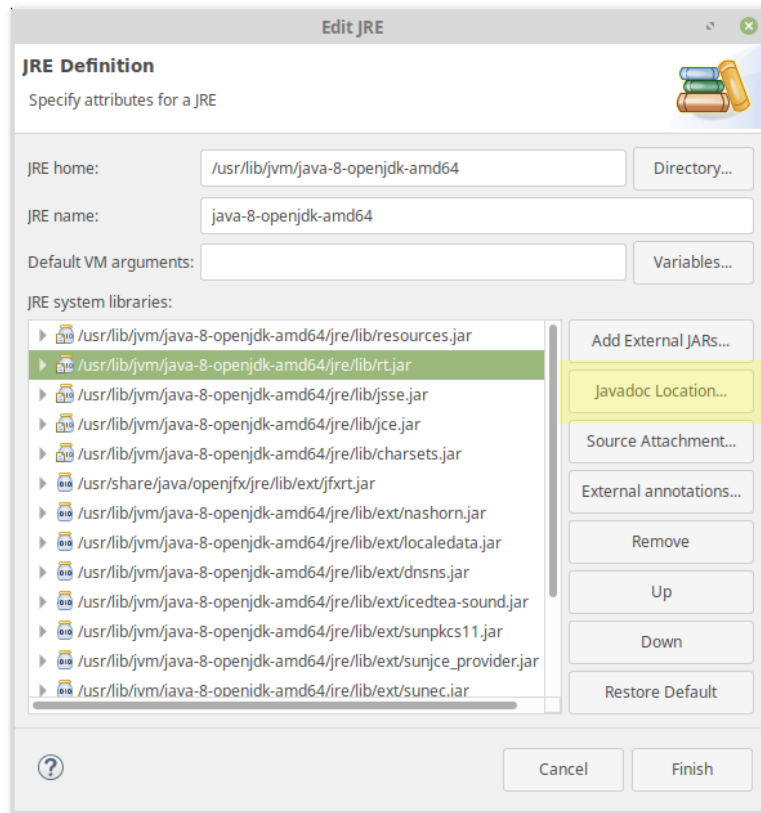


Window > Preferences



Installed JREs > JRE/JDK auswählen > Edit

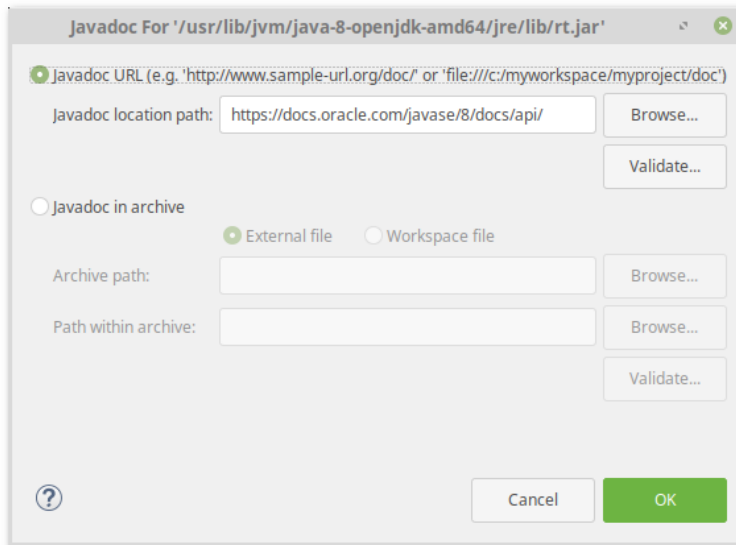
Anhang – Variante 2 in Eclipse (2)



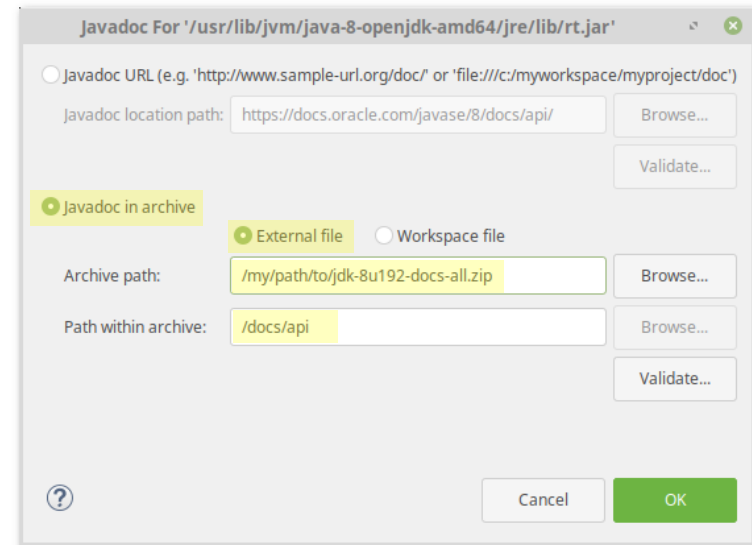
rt.jar: bis Java 8
jrt-fs.jar ab Java 9

In der Liste rt.jar/jrt-fs.jar auswählen > Javadoc Location...

Anhang – Variante 2 in Eclipse (3)

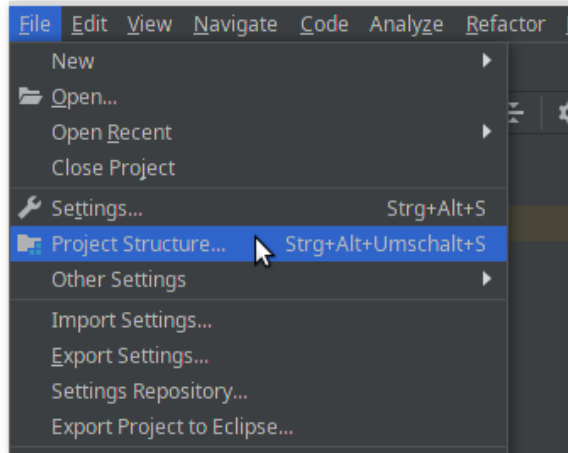


Standard-Einstellung: Online
direkt von docs.oracle.com

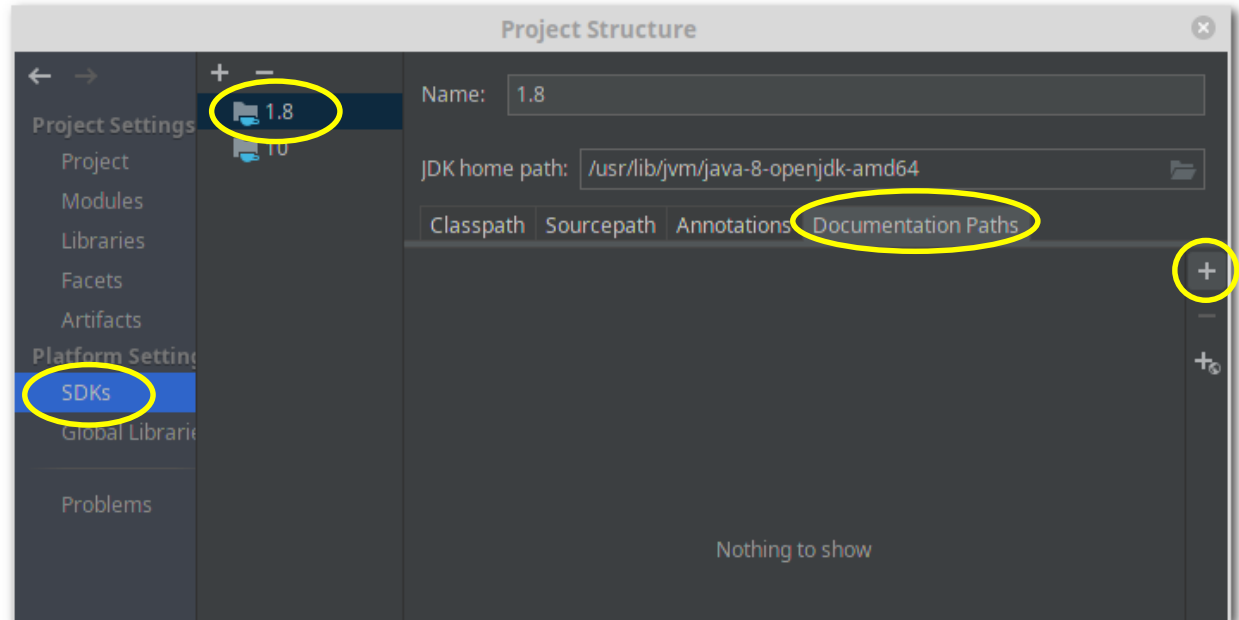


Offline-Variante die
heruntergeladenes Archiv nutzt

Anhang – Variante 2 in IntelliJ (1)

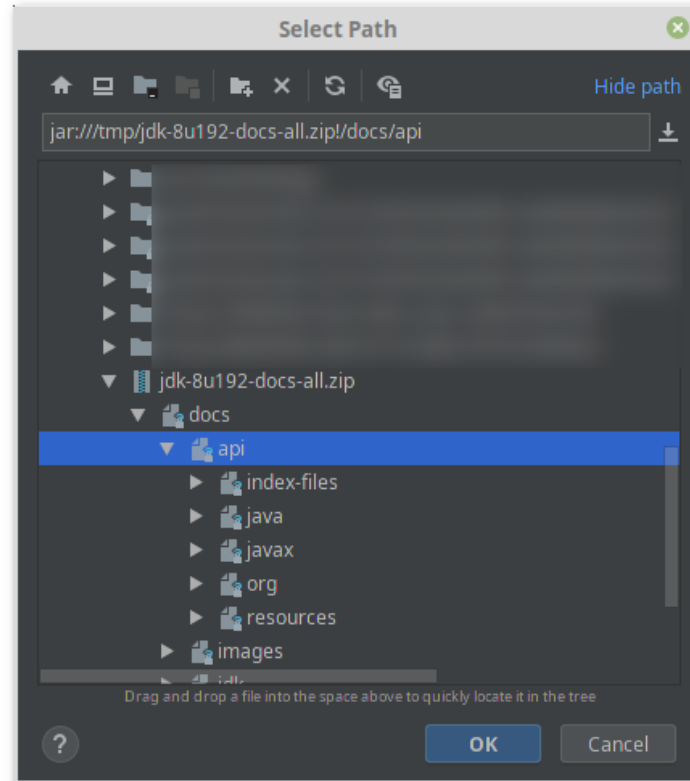


File > Project Structure...



SDKs > 1.8 > Documentation Paths Tab > „+“

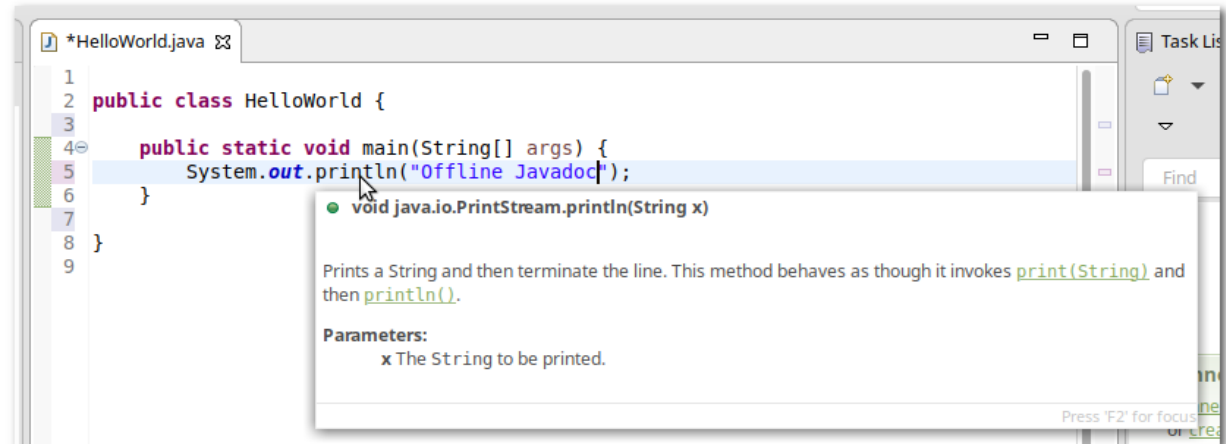
Anhang – Variante 2 in IntelliJ (2)



Zum Javadoc-ZIP navigieren und innerhalb
des Archivs dann docs/api auswählen

Anhang – Offline-Javadoc

- Bei beiden Varianten funktioniert nun Javadoc auch offline

```

1 public class HelloWorld {
2
3
4 public static void main(String[] args) {
5     System.out.println("Offline Javadoc");
6 }
7 }
8
9

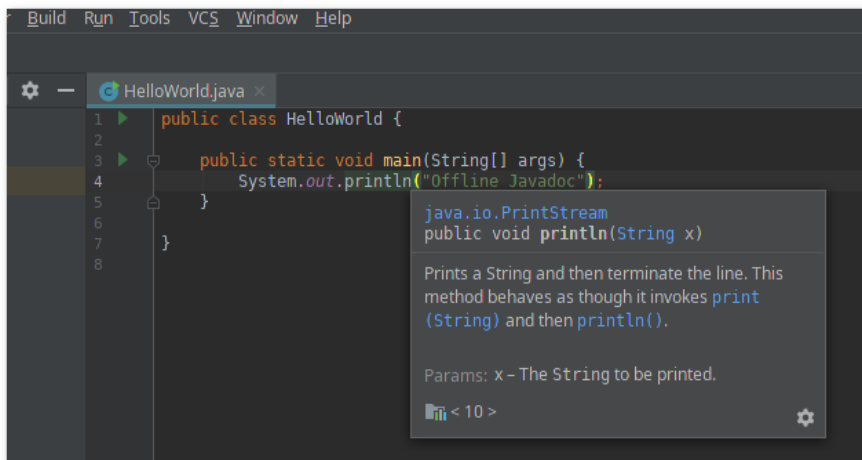
```

void java.io.PrintStream.println(String x)

Prints a String and then terminate the line. This method behaves as though it invokes `print(String)` and then `println()`.

Parameters:

- x The String to be printed.



```

1 public class HelloWorld {
2
3
4 public static void main(String[] args) {
5     System.out.println("Offline Javadoc");
6 }
7 }
8

```

java.io.PrintStream

public void println(String x)

Prints a String and then terminate the line. This method behaves as though it invokes `print(String)` and then `println()`.

Params: x - The String to be printed.