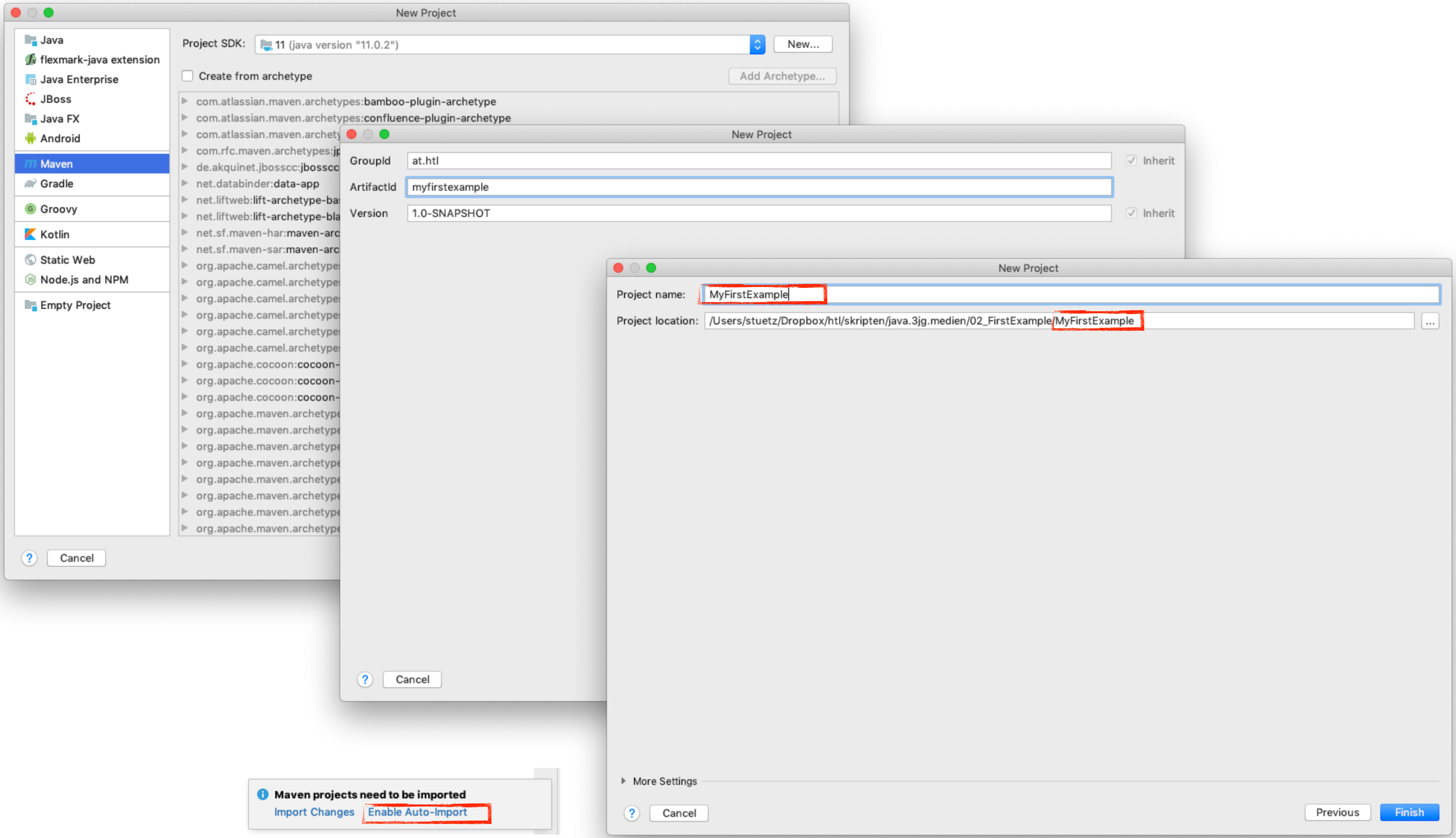
SEW3

IT-Medientechnik

My First Project

TWS


New Project

- Java
- flexmark-java extension
- Java Enterprise
- JBoss
- Java FX
- Android
- Maven
- Gradle
- Groovy
- Kotlin
- Static Web
- Node.js and NPM
- Empty Project

Project SDK: 11 (java version "11.0.2") New...

☐ Create from archetype Add Archetype...

```
com.atlassian.maven.archetypes:bamboo-plugin-archetype
com.atlassian.maven.archetypes:confluence-plugin-archetype
com.atlassian.maven.archety
com.rfc.maven.archetypes:jp
de.akquinet.jbosscc:jbosscc
net.databinder:data-app
net.liftweb:lift-archetype-ba
net.liftweb:lift-archetype-bla
net.sf.maven-har:maven-arc
net.sf.maven-sar:maven-arc
org.apache.camel.archetype
org.apache.camel.archetype
org.apache.camel.archetype
org.apache.camel.archetype
org.apache.camel.archetype
org.apache.camel.archetype
org.apache.cocoon:cocoon-
org.apache.cocoon:cocoon-
org.apache.cocoon:cocoon-
org.apache.maven.archetype
org.apache.maven.archetype
org.apache.maven.archetype
org.apache.maven.archetype
org.apache.maven.archetype
org.apache.maven.archetype
org.apache.maven.archetype
org.apache.maven.archetype
```

? Cancel

New Project

GroupId	at.htl	✓ Inherit
ArtifactId	myfirstexample	
Version	1.0-SNAPSHOT	✓ Inherit

? Cancel

New Project

Project name: MyFirstExample

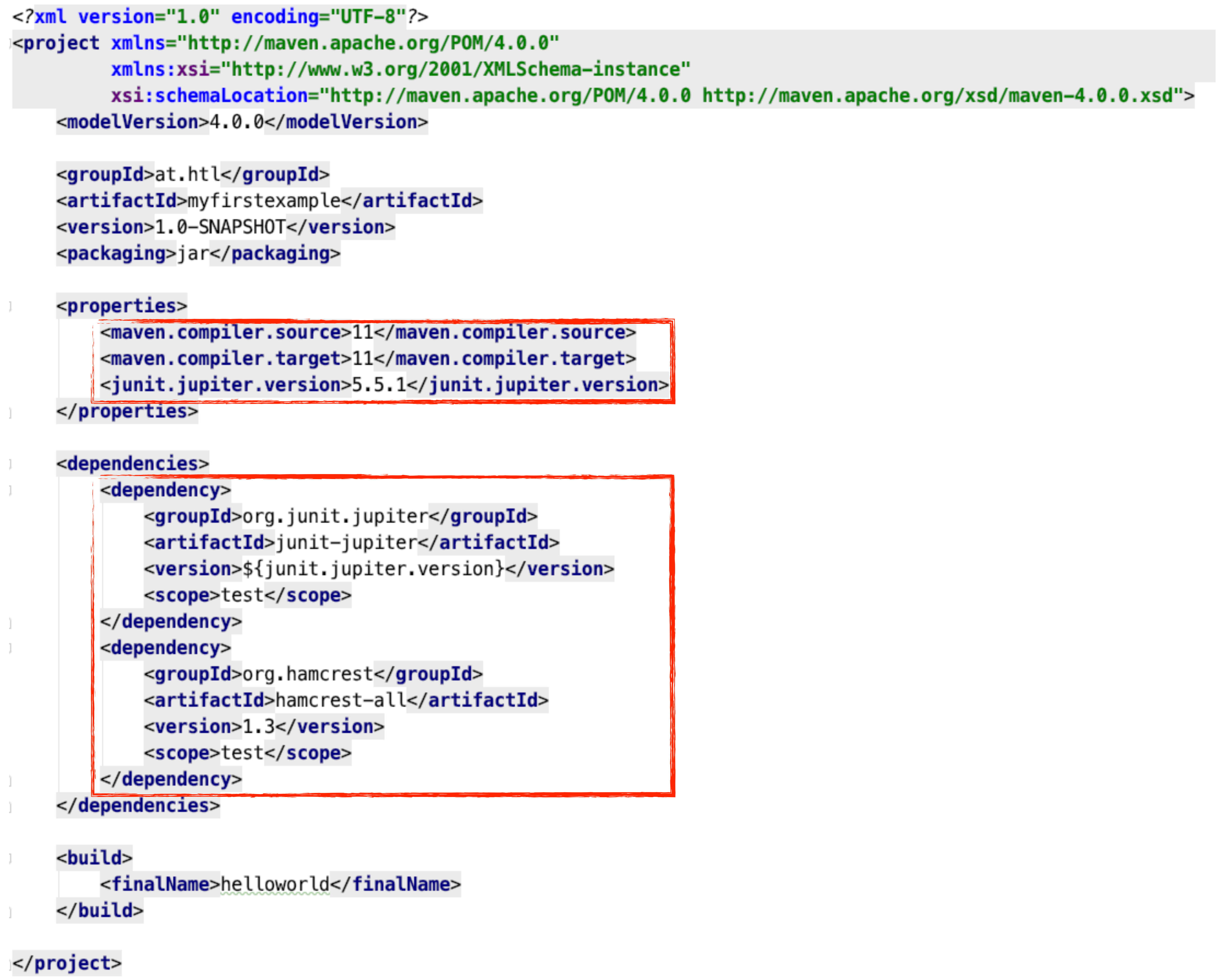
Project location: /Users/stuetz/Dropbox/htl/skripten/java.3jg.medien/02_FirstExample/MyFirstExample ...

More Settings

? Cancel Previous Finish

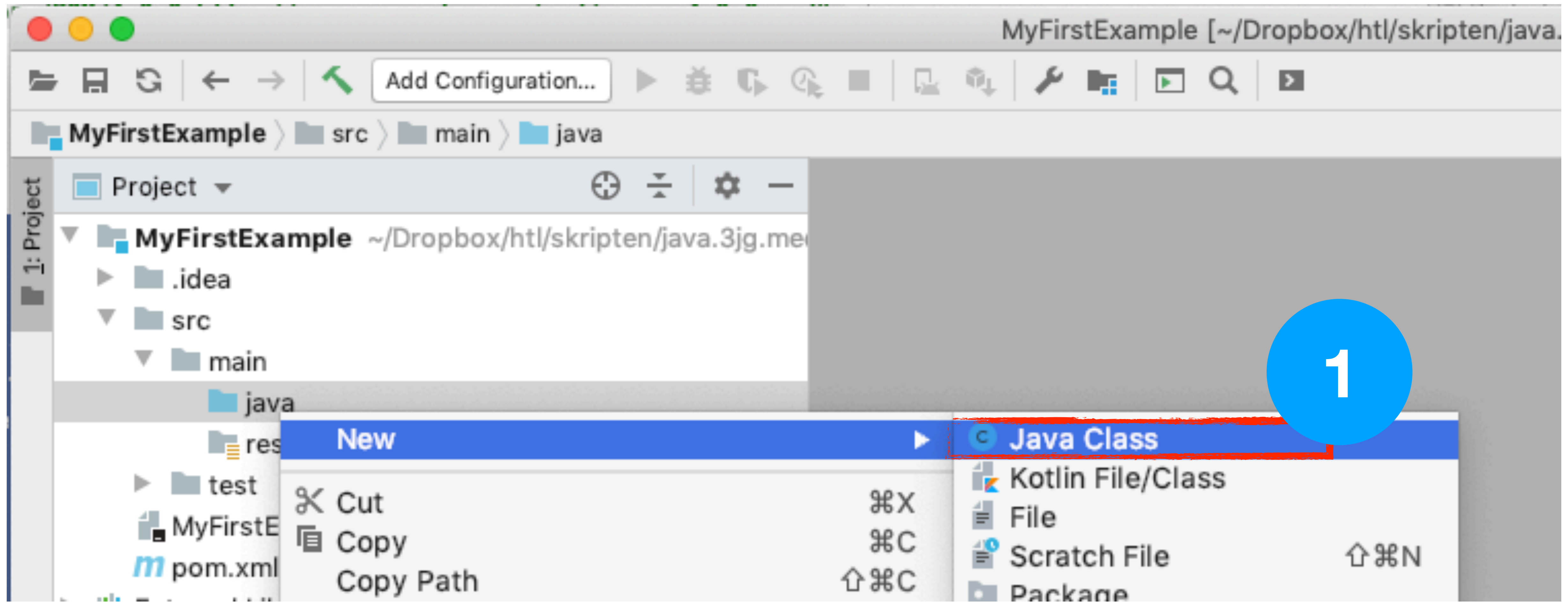
Maven projects need to be imported
Import Changes Enable Auto-Import

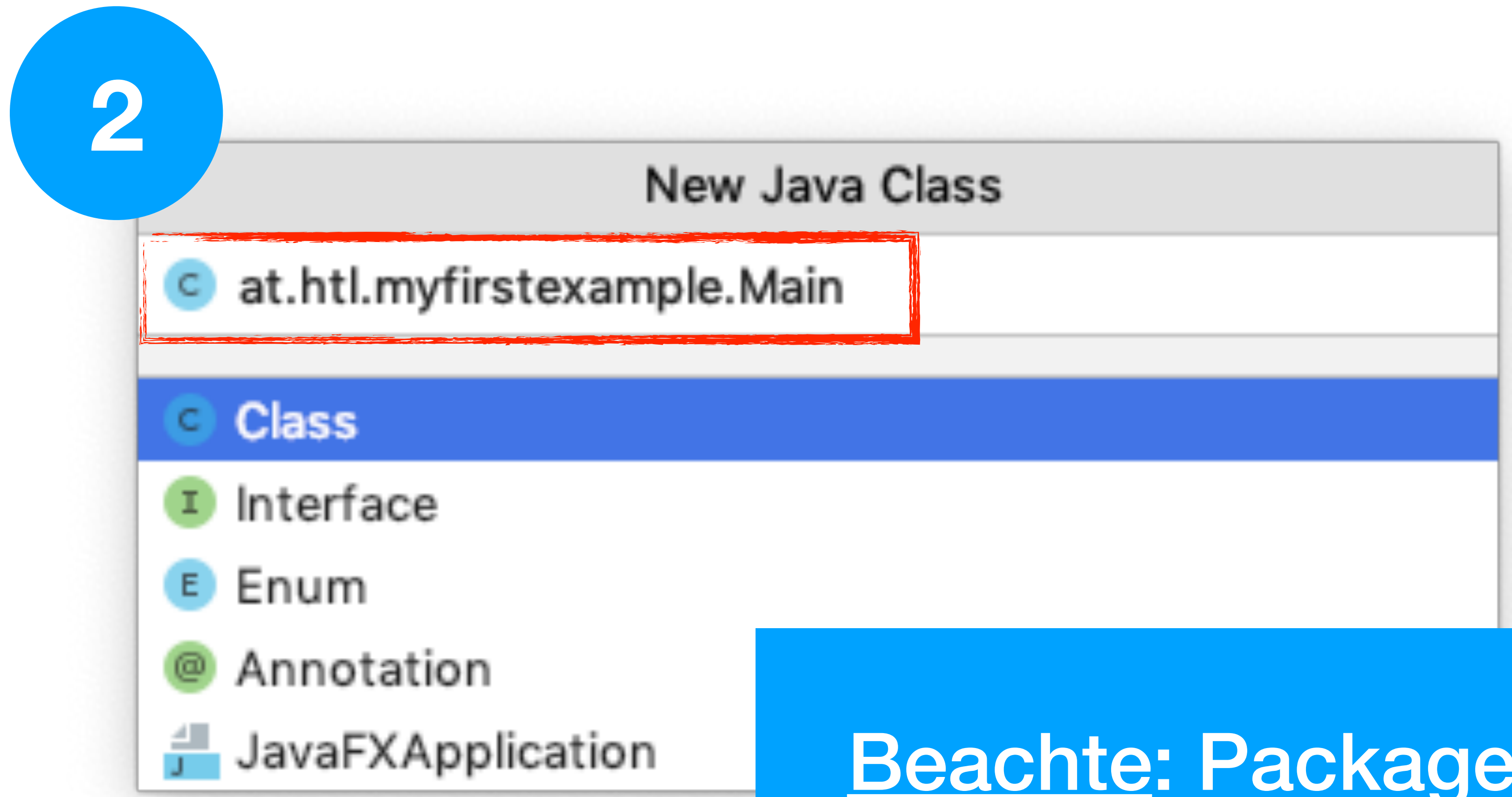
pom.xml



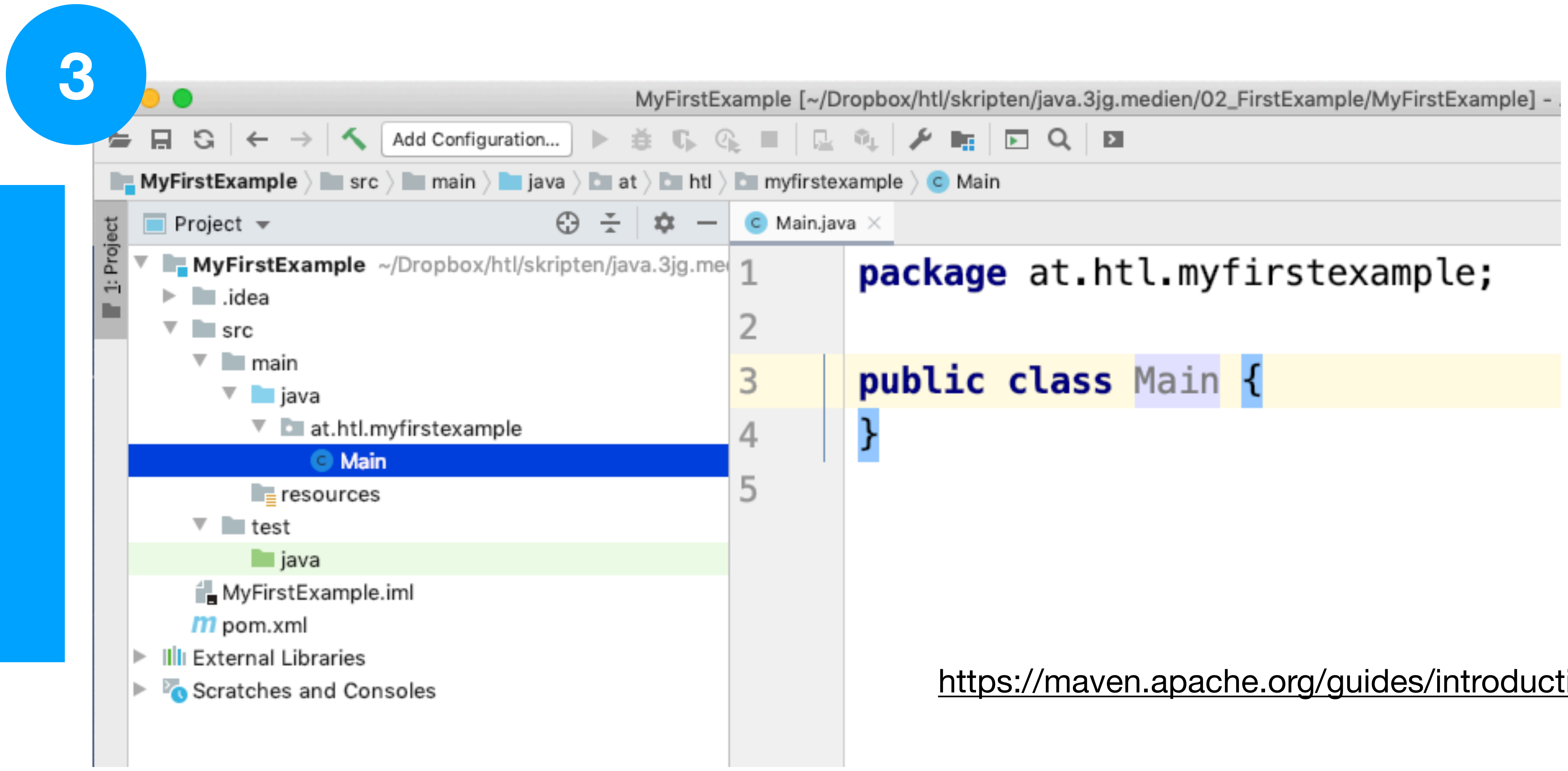
```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>at.htl</groupId>
    <artifactId>myfirstexample</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>jar</packaging>

    <properties>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
        <junit.jupiter.version>5.5.1</junit.jupiter.version>
    </properties>

    <dependencies>
        <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter</artifactId>
            <version>${junit.jupiter.version}</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.hamcrest</groupId>
            <artifactId>hamcrest-all</artifactId>
            <version>1.3</version>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <finalName>helloworld</finalName>
    </build>

</project>
```

Create a new Class





<u>Beachte</u>: Package-Namen werden klein geschrieben, Klassennamen groß

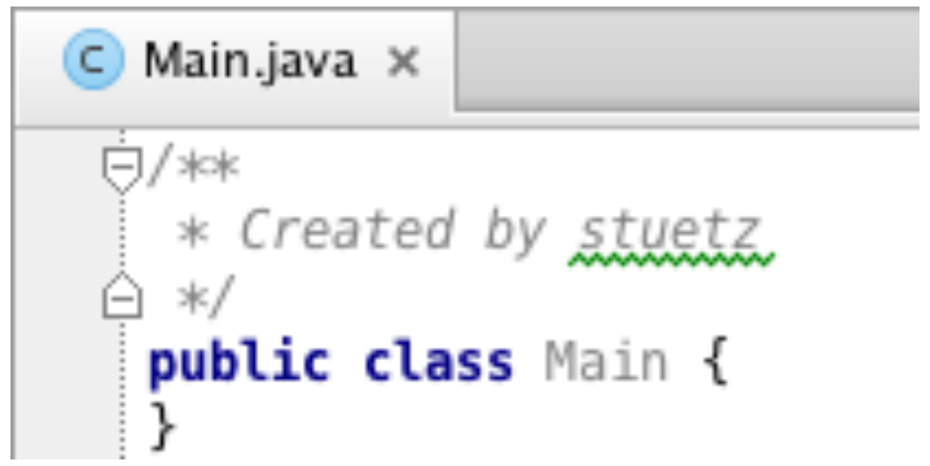

3
MyFirstExample [~/Dropbox/htl/skripten/java.3jg.medien/02_FirstExample/MyFirstExample] -
Add Configuration...
MyFirstExample 〉 src 〉 main 〉 java 〉 at 〉 htl 〉 myfirstexample 〉 Main
Project
Main.java
MyFirstExample ~/Dropbox/htl/skripten/java.3jg.me
.idea
src
main
java
at.htl.myfirstexample
Main
resources
test
java
MyFirstExample.iml
pom.xml
External Libraries
Scratches and Consoles

```java
package at.htl.myfirstexample;

public class Main {
}
```

Die Ordnerstruktur wird entsprechend dem ***maven standard directory layout*** automatisch angelegt

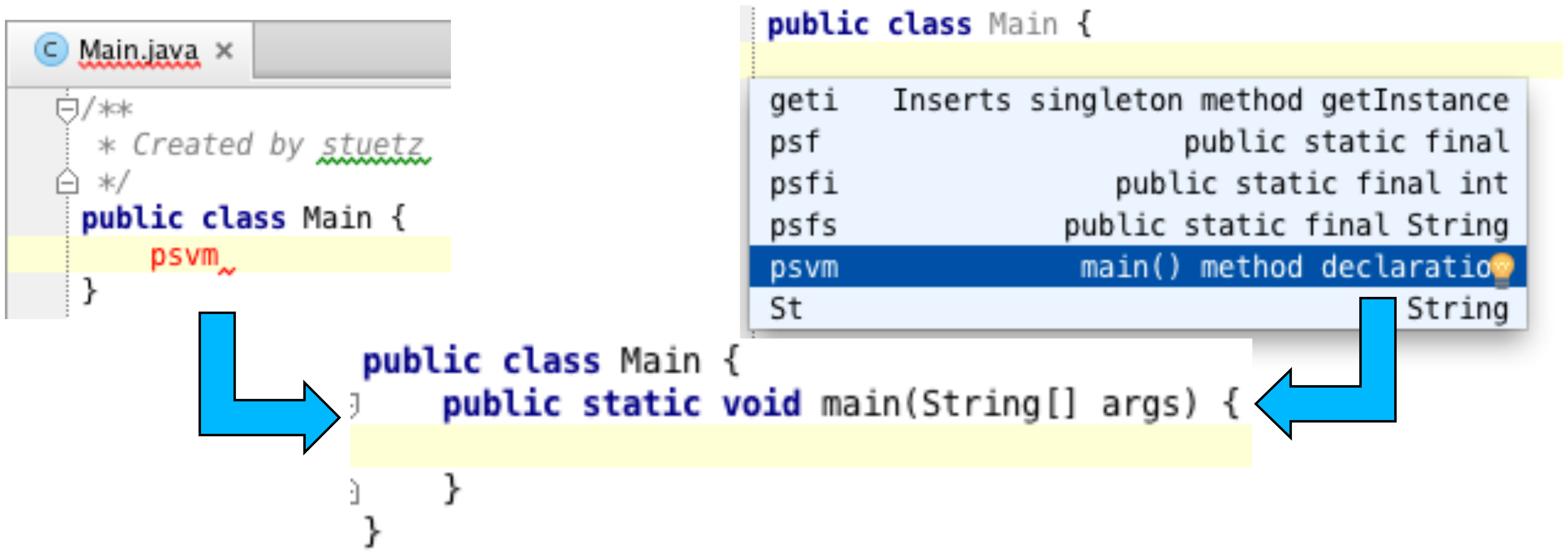
https://maven.apache.org/guides/introduction/introduction-to-the-standard-directory-layout.html

Generate main-Method



```
/**
 * Created by stuetz
 */
public class Main {
}
```

Anlegen des „Gerüsts“ für main() – Methode
psvm + <Tab>   oder   ⌥ + J bzw. <Alt> + J

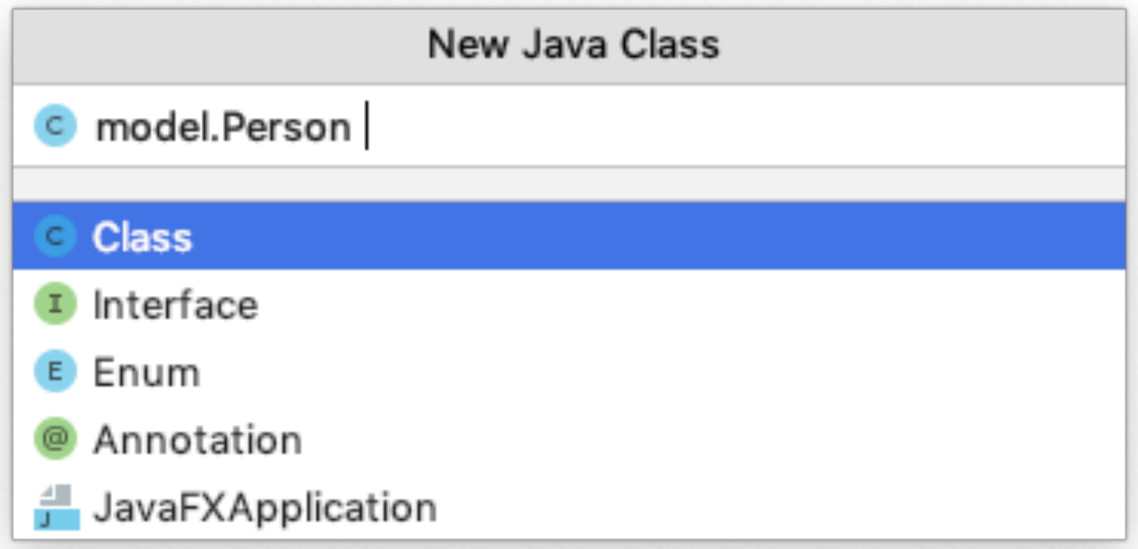


```
/**
 * Created by stuetz
 */
public class Main {
    psvm
}
```

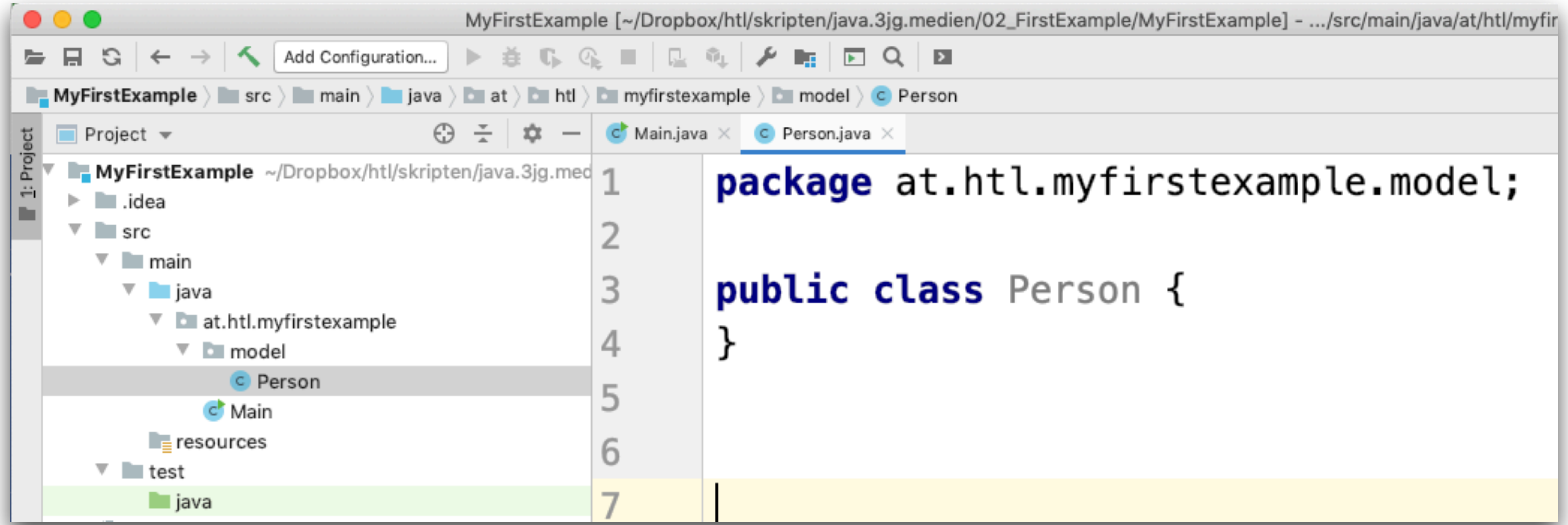
```
public class Main {

geti    Inserts singleton method getInstance
psf                       public static final
psfi                  public static final int
psfs               public static final String
psvm                 main() method declaration
St                                     String
```

```
public class Main {
    public static void main(String[] args) {

    }
}
```

Create a second Class

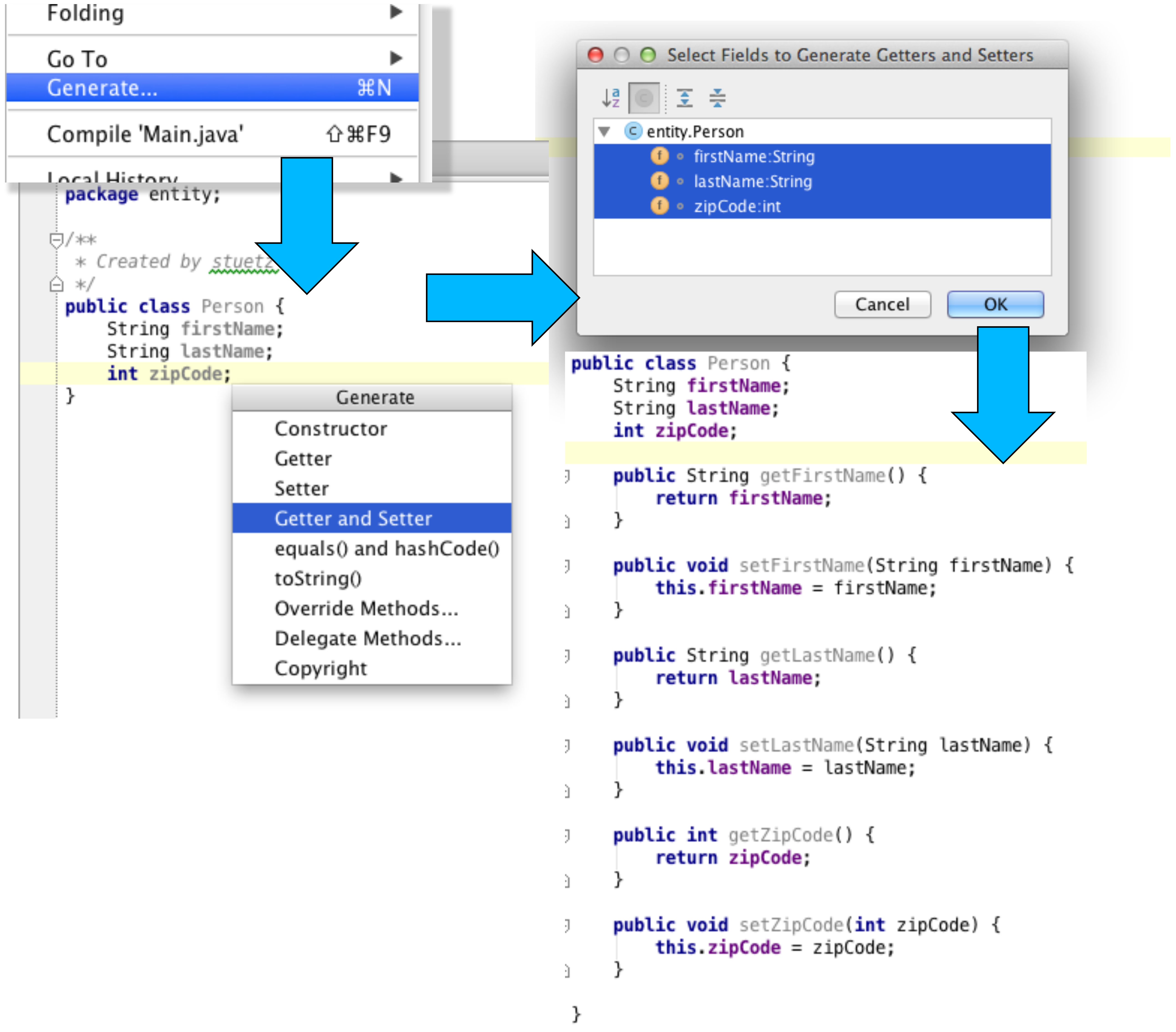


Beachte: Package-Namen werden klein geschrieben, Klassennamen groß



```java
package at.htl.myfirstexample.model;

public class Person {
}
```

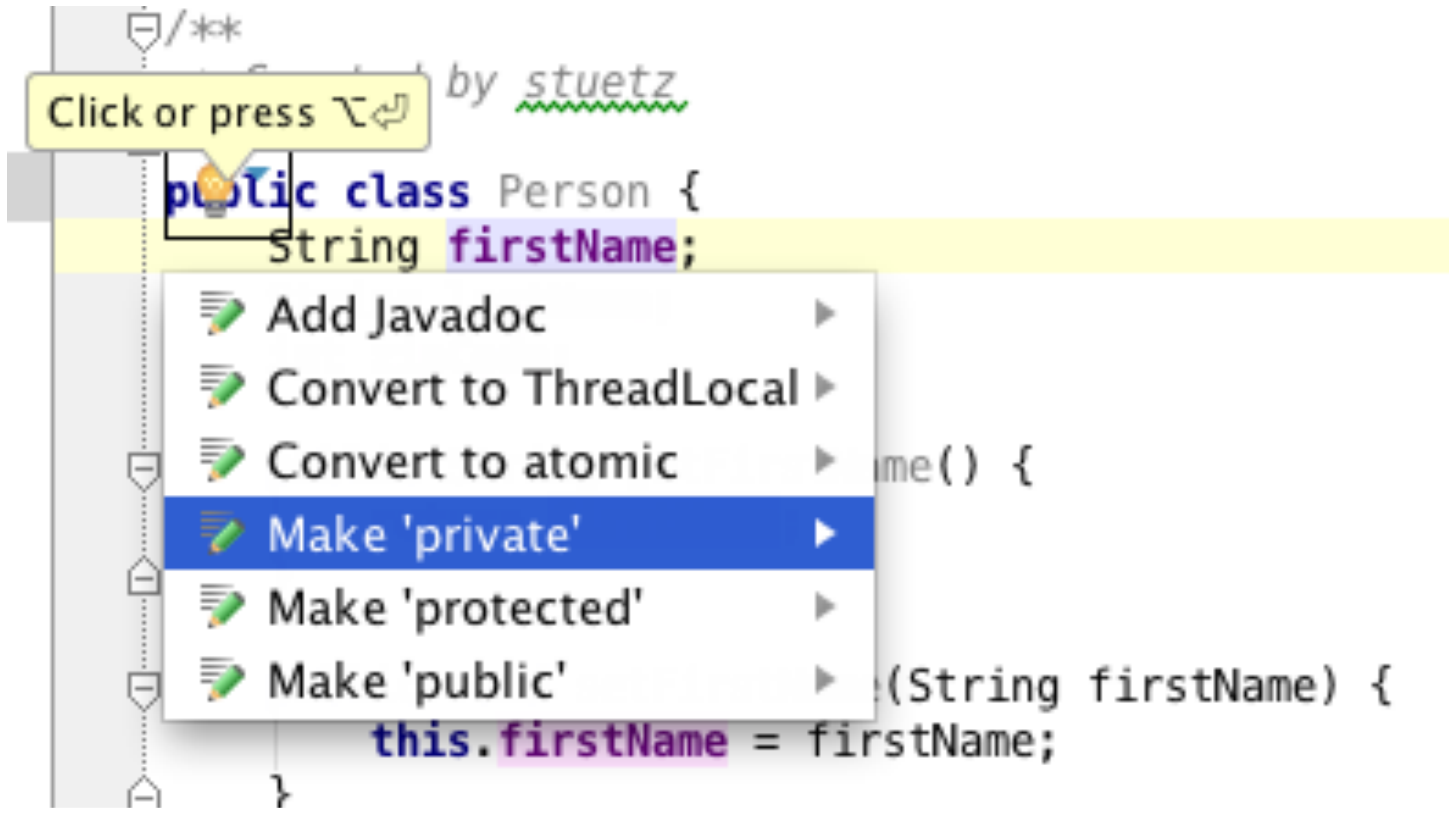
Funktion zum Generieren von Code ^N, <Alt> <Einf>



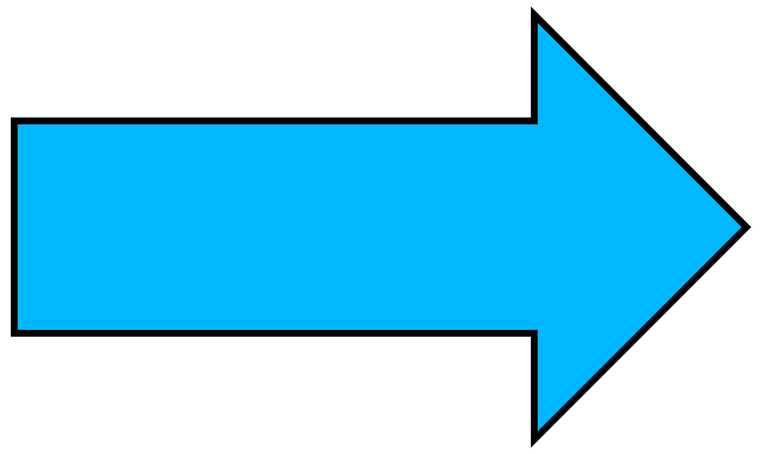
```java
package entity;

/**
 * Created by stuetz
 */
public class Person {
    String firstName;
    String lastName;
    int zipCode;
}
```

```java
public class Person {
    String firstName;
    String lastName;
    int zipCode;

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public int getZipCode() {
        return zipCode;
    }

    public void setZipCode(int zipCode) {
        this.zipCode = zipCode;
    }

}
```

IntelliSense ist natürlich auch vorhanden



```
/**
 * ... by stuetz
public class Person {
    String firstName;
                                   ...me() {

                                   ...(String firstName) {
        this.firstName = firstName;
    }
```

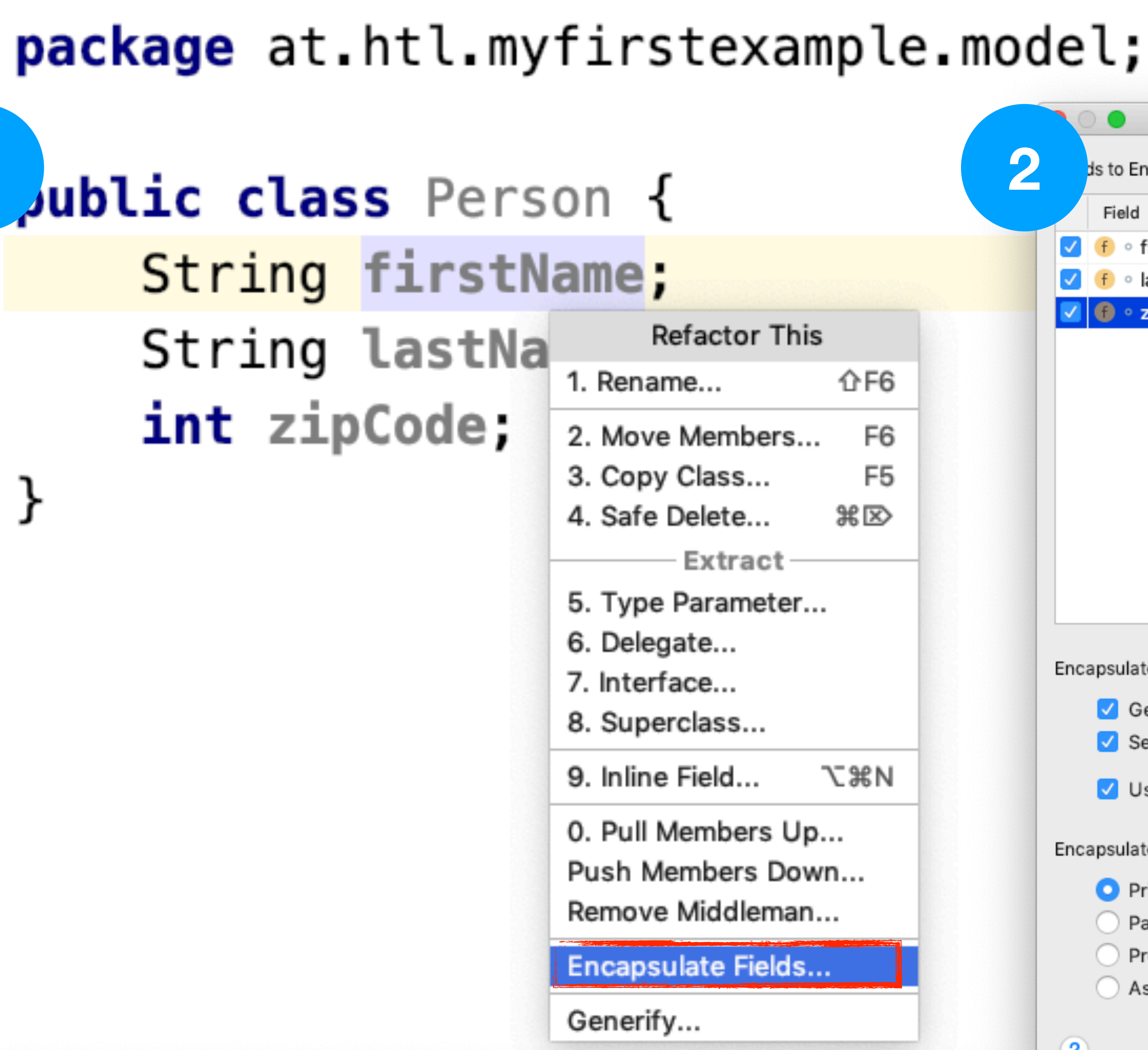


```
private String firstName;
```

Refactoring

empfohlene Vorgehensweise

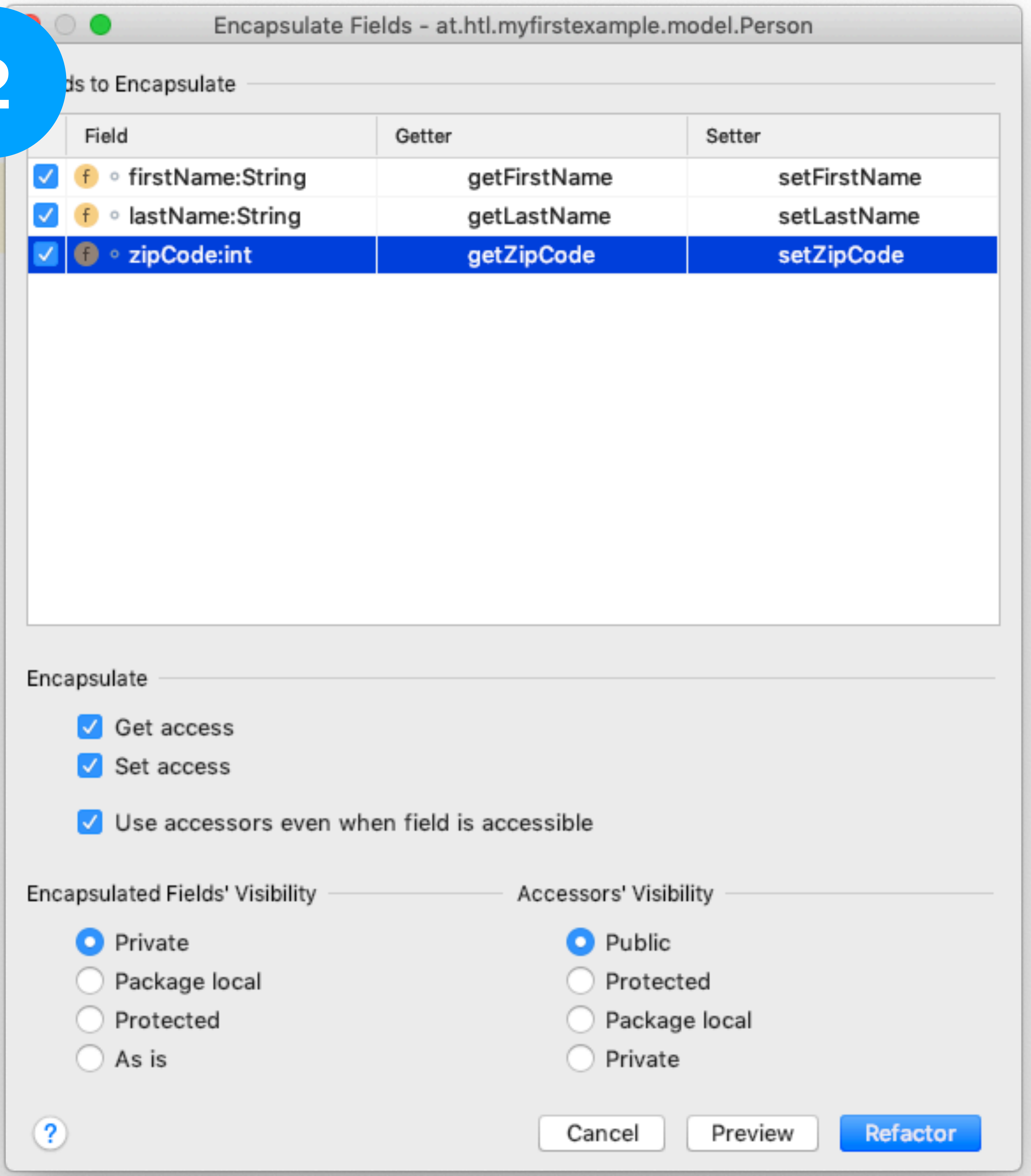
1



```java
package at.htl.myfirstexample.model;

ublic class Person {
    String firstName;
    String lastNa
    int zipCode;
}
```

Refactor This via ^T (Shift+Ctrl+Alt+T for Win/Linux)

2



Encapsulate Fields - at.htl.myfirstexample.model.Person

ds to Encapsulate

	Field	Getter	Setter
✓	firstName:String	getFirstName	setFirstName
✓	lastName:String	getLastName	setLastName
✓	zipCode:int	getZipCode	setZipCode

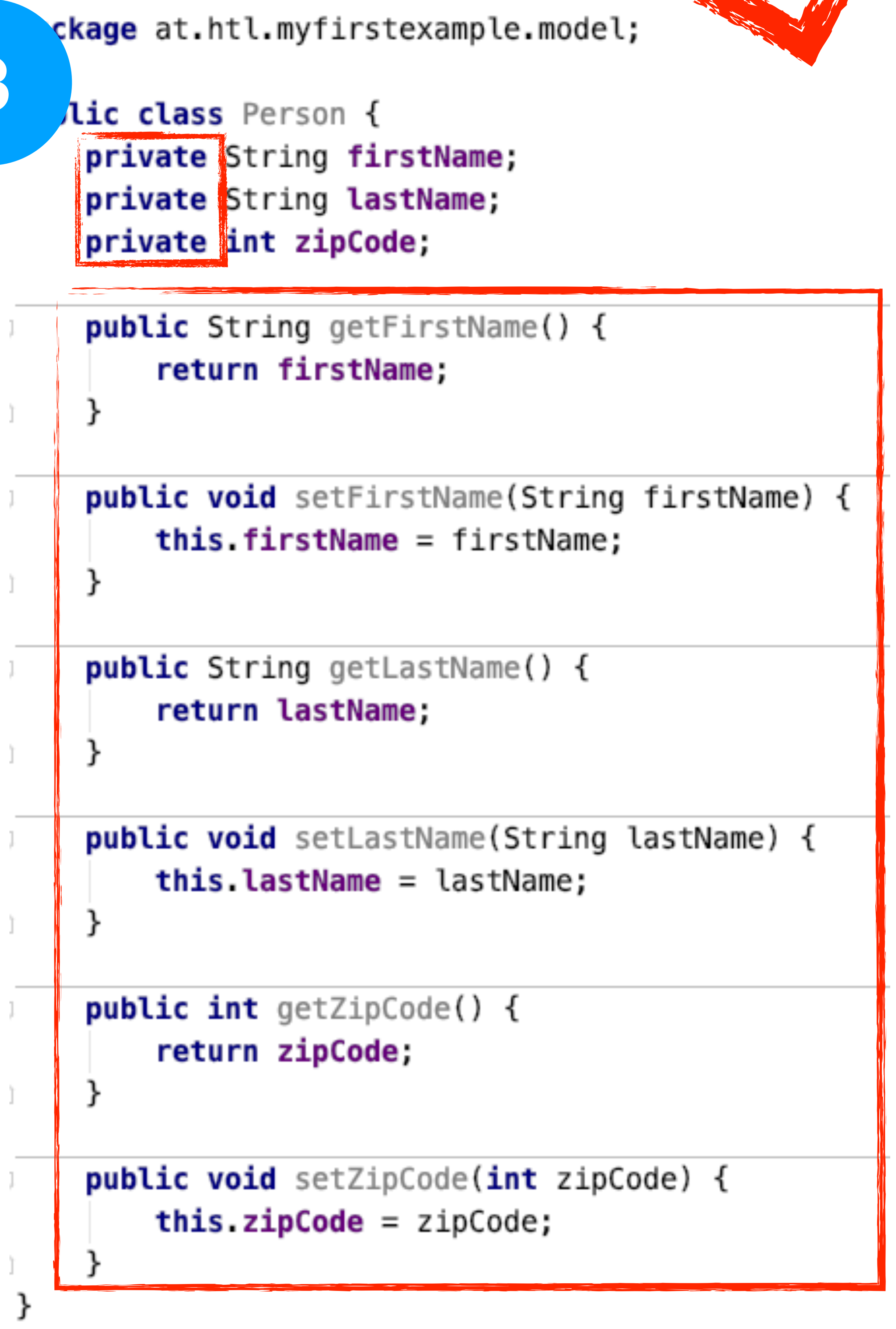
Encapsulate

- ✓ Get access
- ✓ Set access
- ✓ Use accessors even when field is accessible

Encapsulated Fields' Visibility: [● Private] ○ Package local ○ Protected ○ As is

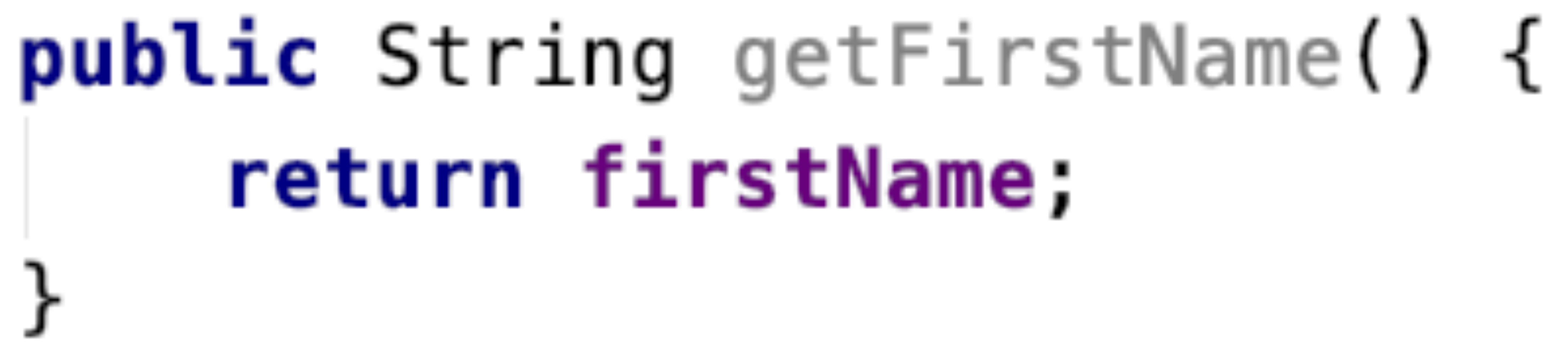
Accessors' Visibility: [● Public] ○ Protected ○ Package local ○ Private

3



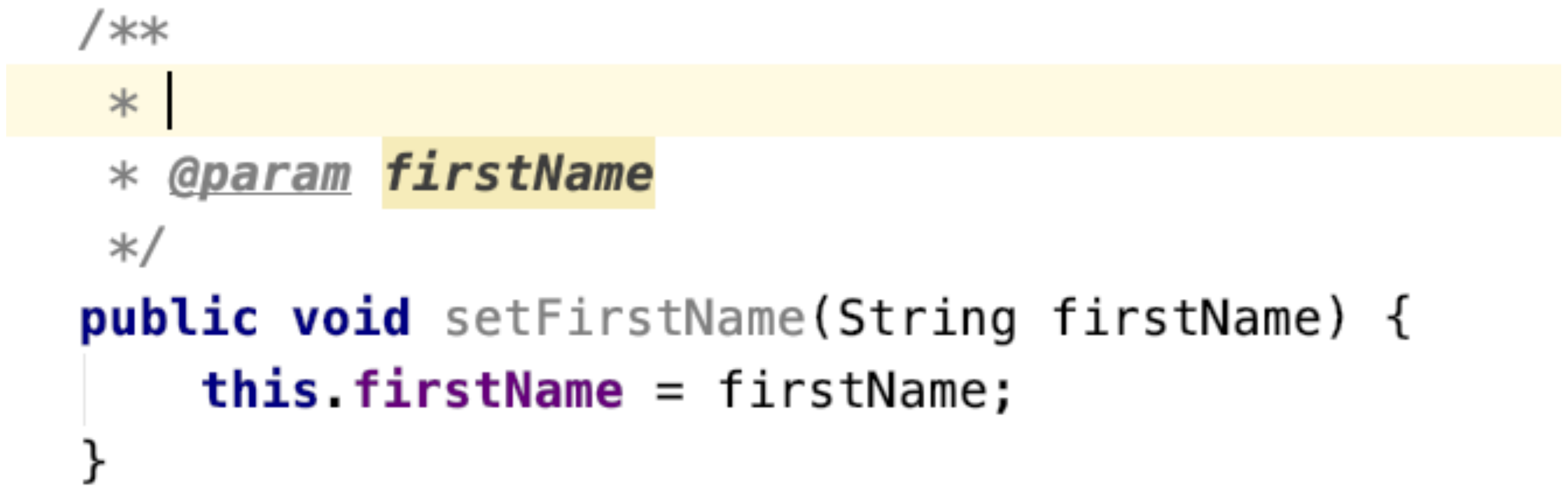
```java
ckage at.htl.myfirstexample.model;

lic class Person {
  private String firstName;
  private String lastName;
  private int zipCode;

  public String getFirstName() {
      return firstName;
  }

  public void setFirstName(String firstName) {
      this.firstName = firstName;
  }

  public String getLastName() {
      return lastName;
  }

  public void setLastName(String lastName) {
      this.lastName = lastName;
  }

  public int getZipCode() {
      return zipCode;
  }

  public void setZipCode(int zipCode) {
      this.zipCode = zipCode;
  }
}
```

Kommentare



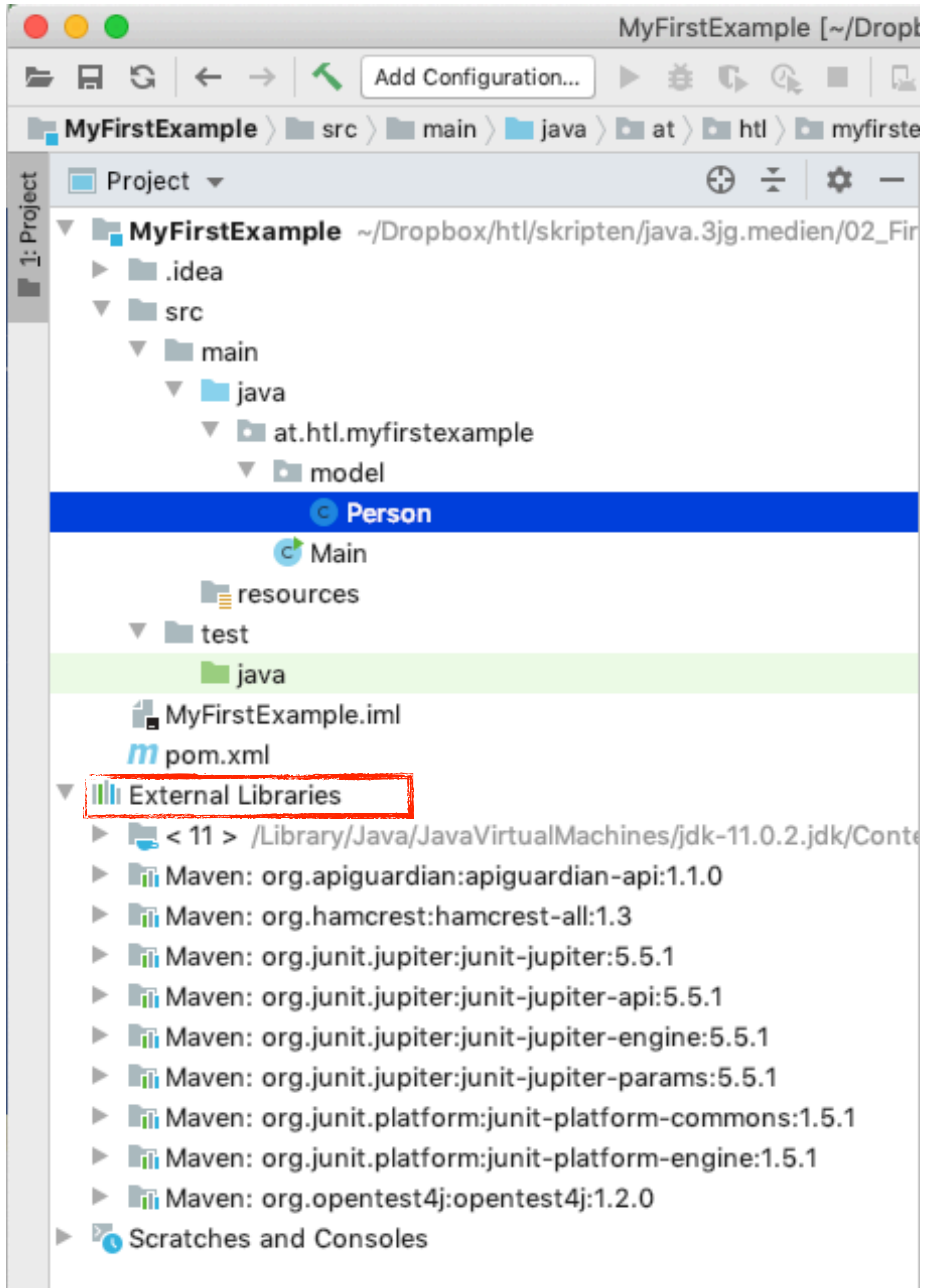
```
public String getFirstName() {
    return firstName;
}
```

/** + <Return> erstellt Kommentarblock



```
/**
 * 
 * @param firstName
 */
public void setFirstName(String firstName) {
    this.firstName = firstName;
}
```

Projektübersicht in IntelliJ IDEA Ultimate


MyFirstExample [~/Dropb

Add Configuration...

MyFirstExample 〉 src 〉 main 〉 java 〉 at 〉 htl 〉 myfirste

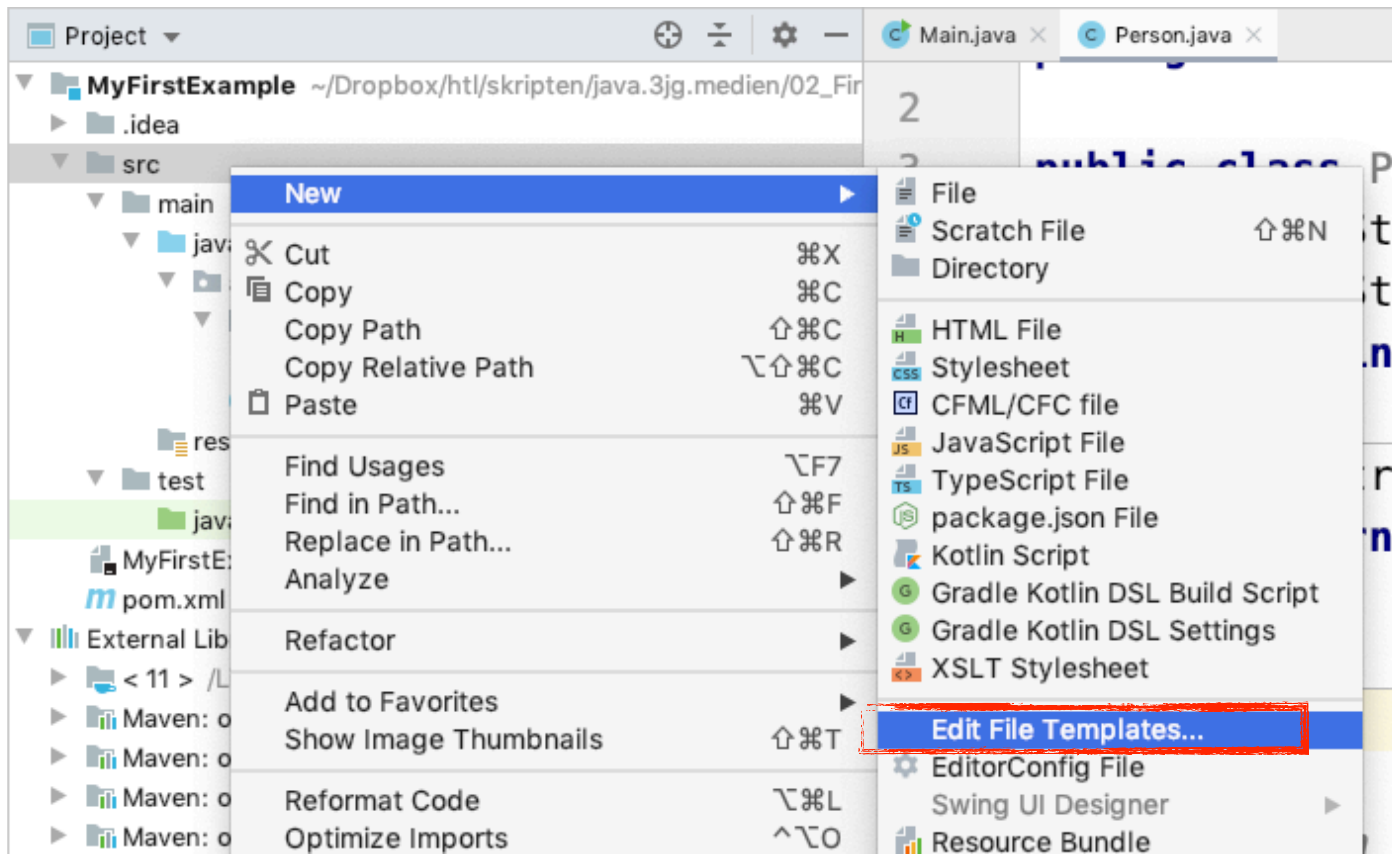
1: Project

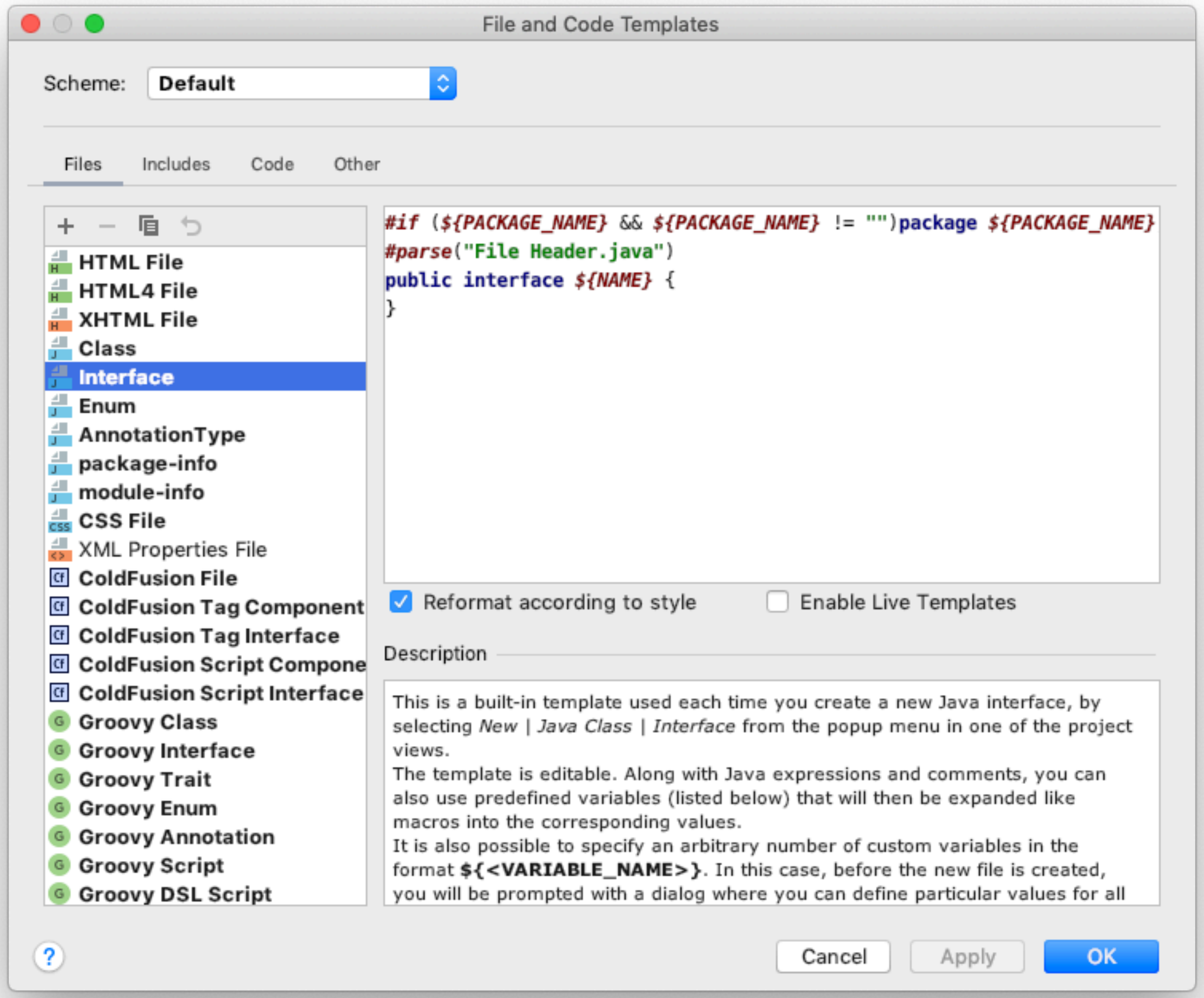
Project

```
▼ MyFirstExample  ~/Dropbox/htl/skripten/java.3jg.medien/02_Fir
  ▶ .idea
  ▼ src
    ▼ main
      ▼ java
        ▼ at.htl.myfirstexample
          ▼ model
              Person
            Main
        resources
    ▼ test
        java
    MyFirstExample.iml
    pom.xml
▼ External Libraries
  ▶ < 11 >  /Library/Java/JavaVirtualMachines/jdk-11.0.2.jdk/Conte
  ▶ Maven: org.apiguardian:apiguardian-api:1.1.0
  ▶ Maven: org.hamcrest:hamcrest-all:1.3
  ▶ Maven: org.junit.jupiter:junit-jupiter:5.5.1
  ▶ Maven: org.junit.jupiter:junit-jupiter-api:5.5.1
  ▶ Maven: org.junit.jupiter:junit-jupiter-engine:5.5.1
  ▶ Maven: org.junit.jupiter:junit-jupiter-params:5.5.1
  ▶ Maven: org.junit.platform:junit-platform-commons:1.5.1
  ▶ Maven: org.junit.platform:junit-platform-engine:1.5.1
  ▶ Maven: org.opentest4j:opentest4j:1.2.0
▶ Scratches and Consoles
```

Die „External Libraries“ sind oft ganz hilfreich, um kontrollieren zu können ob der maven-Import erfolgreich war

File und Code Templates anpassen





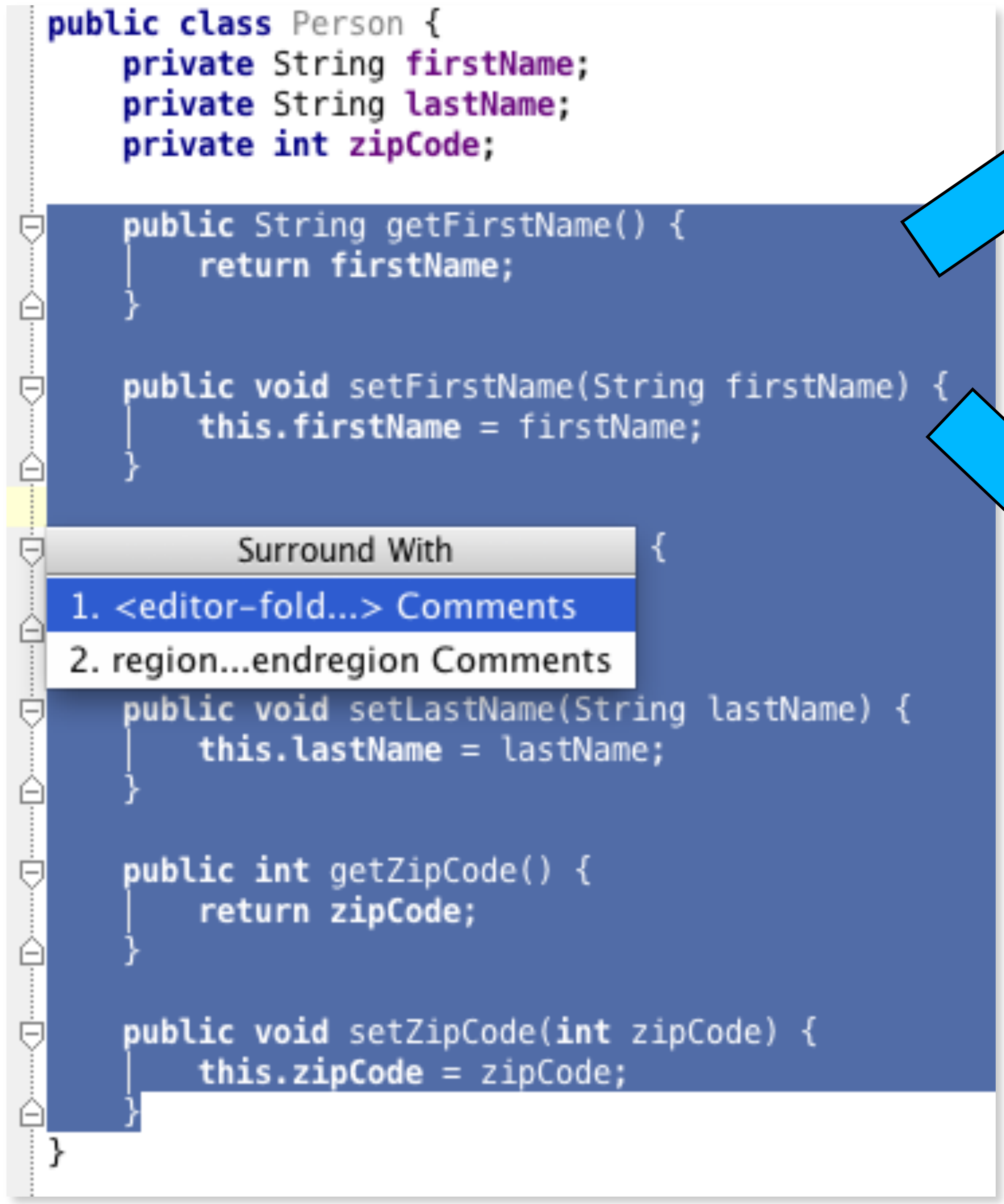
```
#if (${PACKAGE_NAME} && ${PACKAGE_NAME} != "")package ${PACKAGE_NAME}
#parse("File Header.java")
public interface ${NAME} {
}
```

☑ Reformat according to style ☐ Enable Live Templates

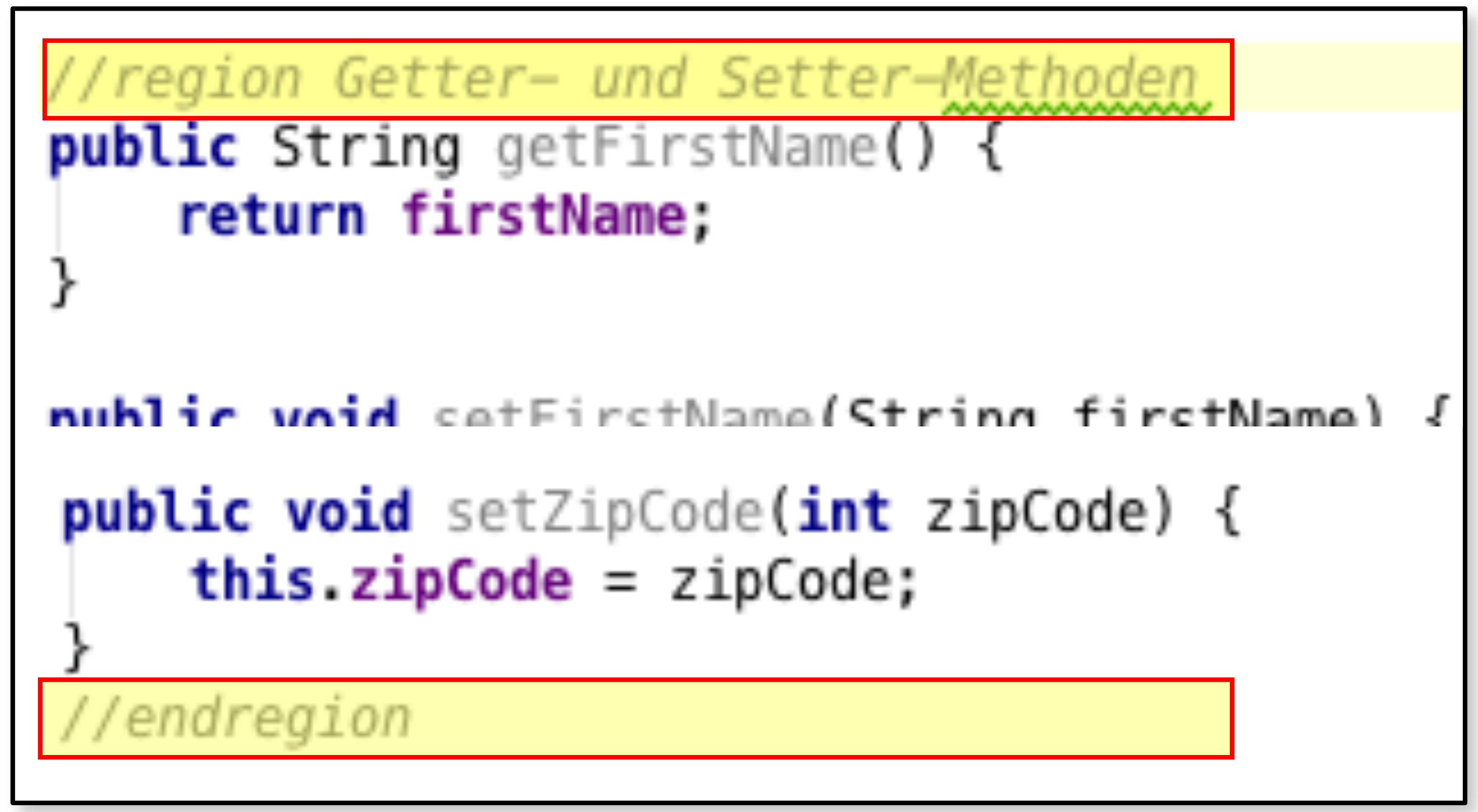
This is a built-in template used each time you create a new Java interface, by selecting *New* | *Java Class* | *Interface* from the popup menu in one of the project views.
The template is editable. Along with Java expressions and comments, you can also use predefined variables (listed below) that will then be expanded like macros into the corresponding values.
It is also possible to specify an arbitrary number of custom variables in the format **${<VARIABLE_NAME>}**. In this case, before the new file is created, you will be prompted with a dialog where you can define particular values for all

2 Varianten bei Regions

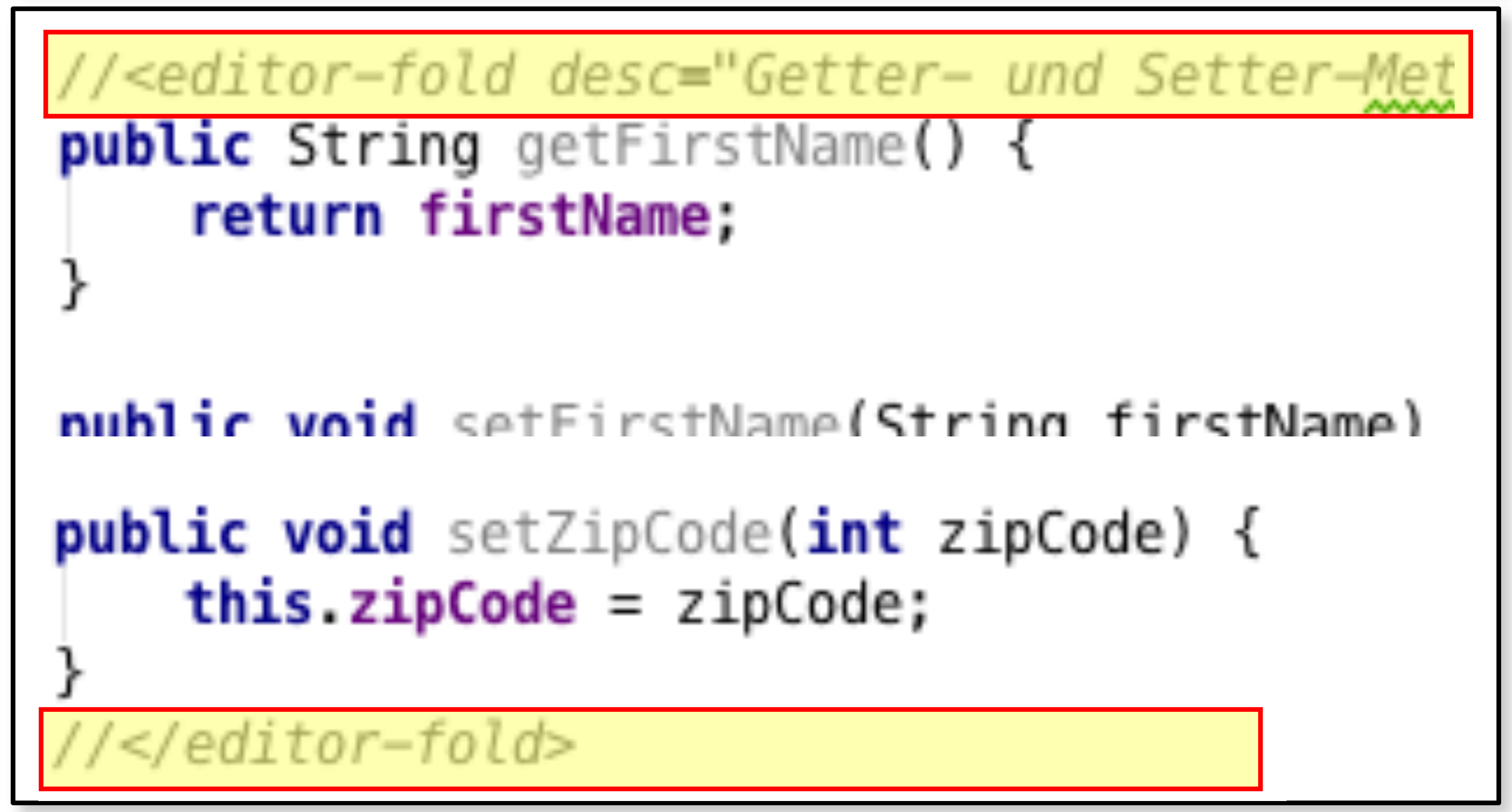
⌘⌥T ... Surround with



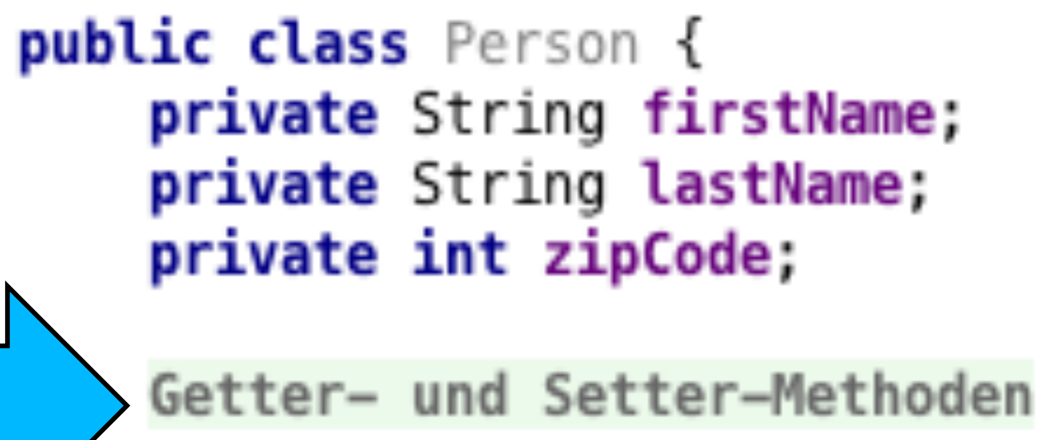
```
public class Person {
    private String firstName;
    private String lastName;
    private int zipCode;

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

                                   {

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public int getZipCode() {
        return zipCode;
    }

    public void setZipCode(int zipCode) {
        this.zipCode = zipCode;
    }
}
```



```
//region Getter- und Setter-Methoden
public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName) {

public void setZipCode(int zipCode) {
    this.zipCode = zipCode;
}
//endregion
```

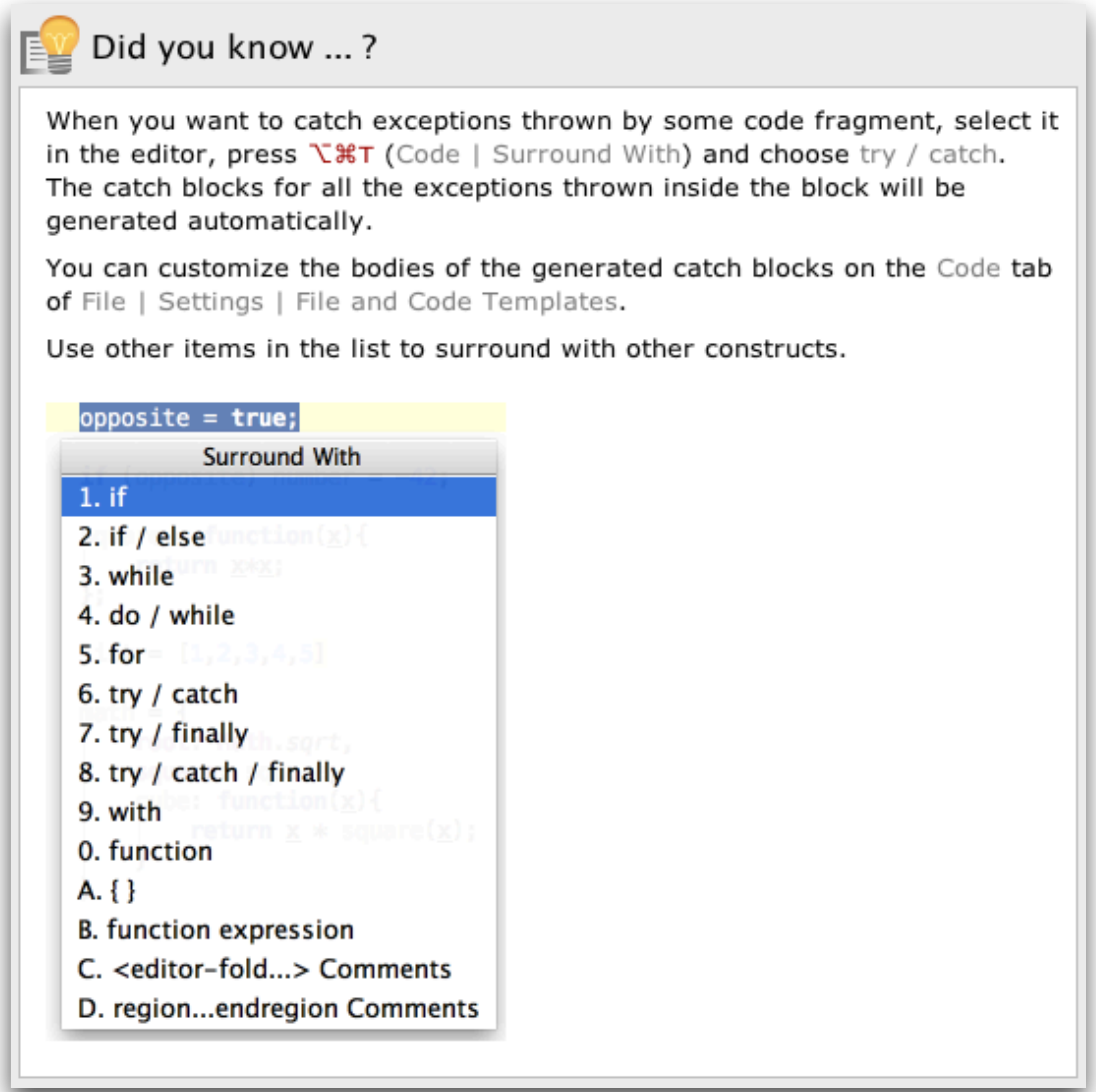


```
//<editor-fold desc="Getter- und Setter-Met
public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName)

public void setZipCode(int zipCode) {
    this.zipCode = zipCode;
}
//</editor-fold>
```



```
public class Person {
    private String firstName;
    private String lastName;
    private int zipCode;

    Getter- und Setter-Methoden
```

⌥⌘T für weitere „Surrounds“



Did you know ... ?

When you want to catch exceptions thrown by some code fragment, select it in the editor, press ⌥⌘T (Code | Surround With) and choose try / catch. The catch blocks for all the exceptions thrown inside the block will be generated automatically.

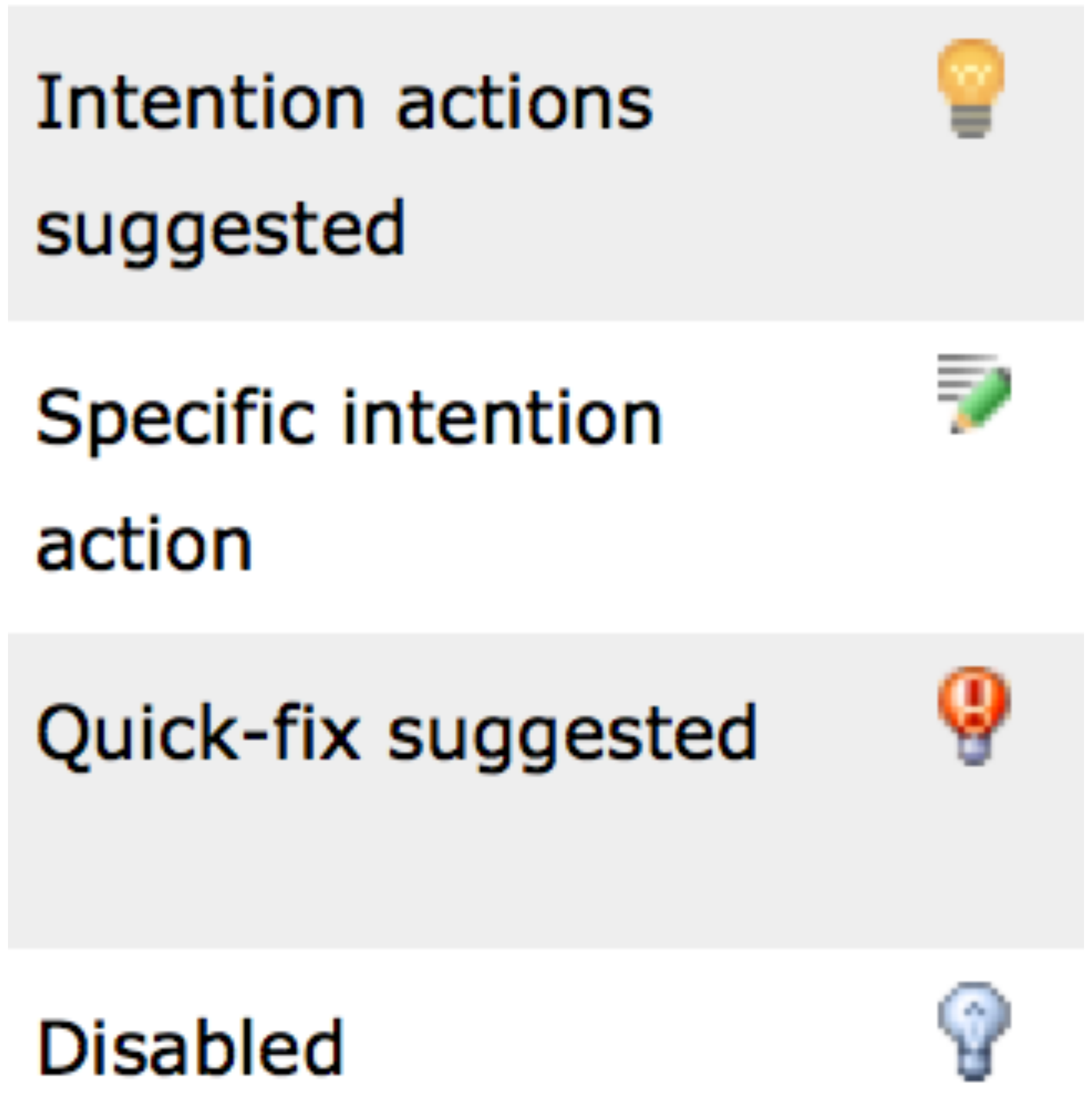
You can customize the bodies of the generated catch blocks on the Code tab of File | Settings | File and Code Templates.

Use other items in the list to surround with other constructs.

```
opposite = true;
```

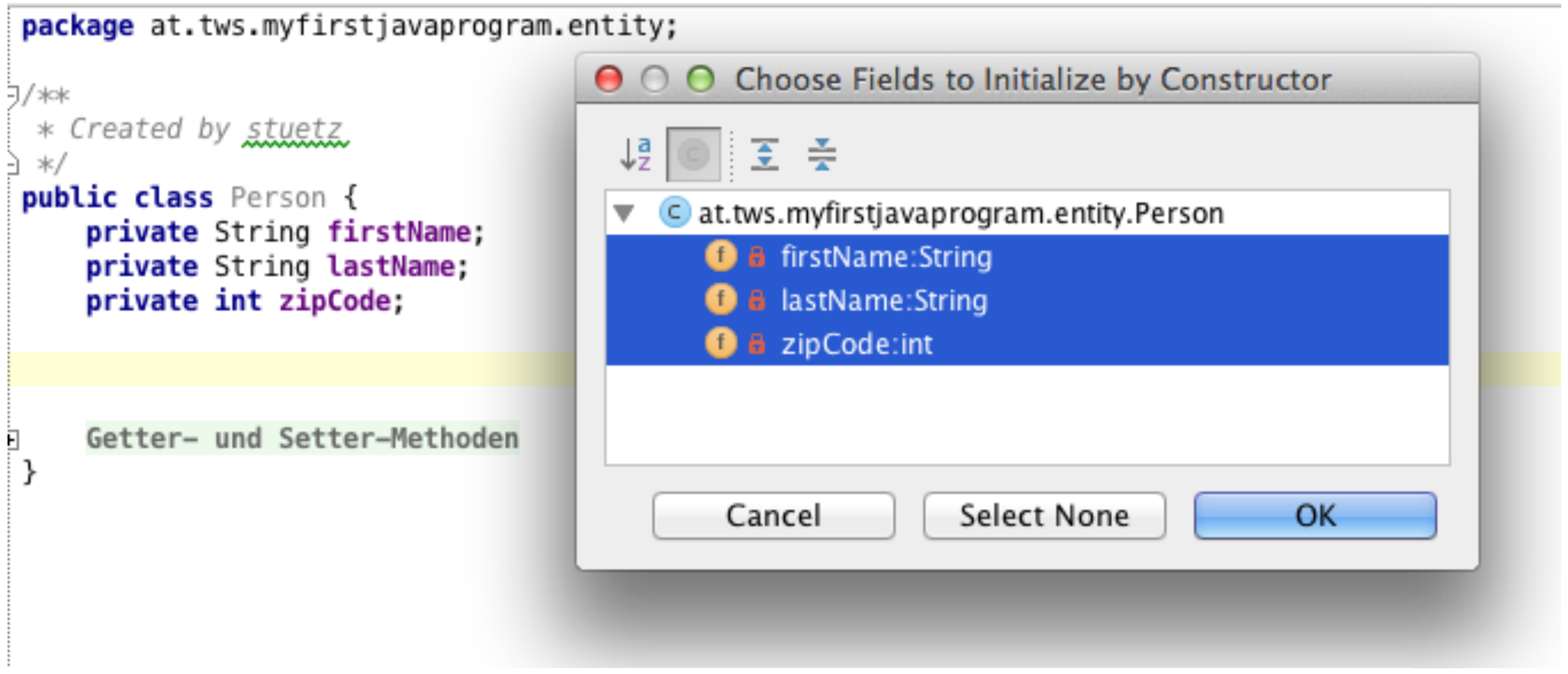
Intention Actions

- Im Kontext des Cursors werden Aktionen angeboten
- Shortcut <Alt> <Enter>
- Wird zB beim Generieren von Unit-Tests und beim Refactoring verwendet



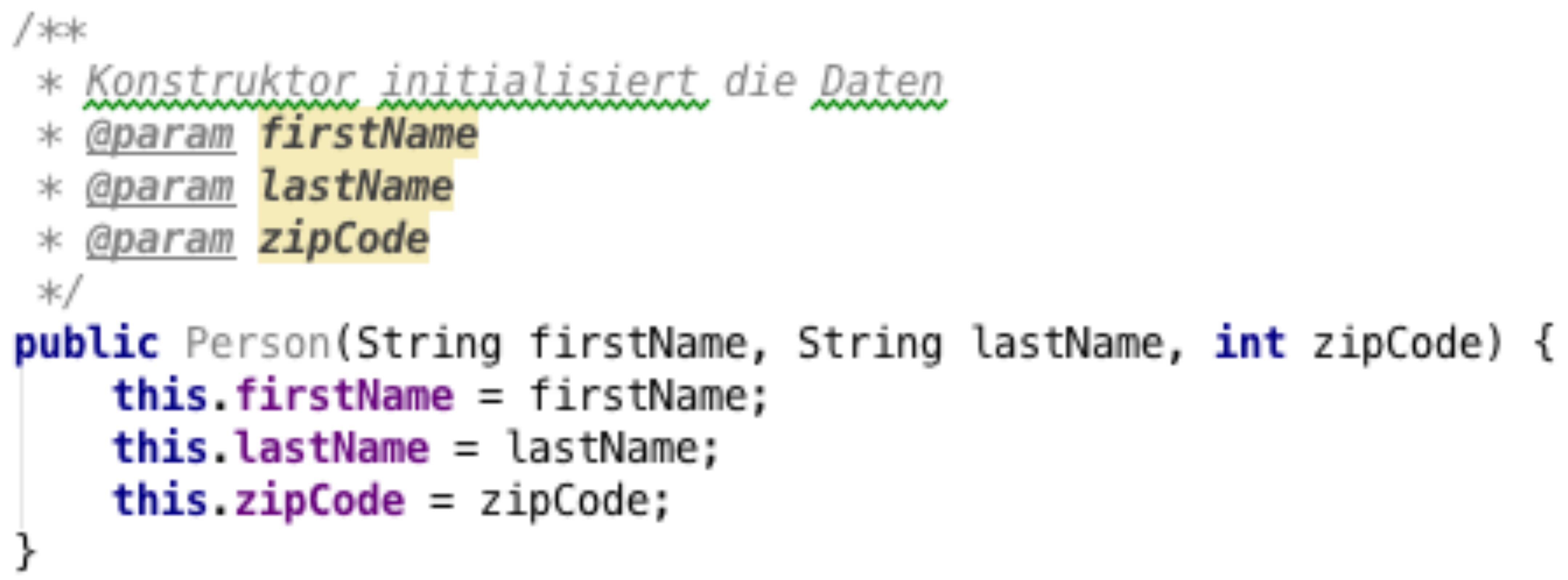
Intention actions suggested	💡
Specific intention action	✎
Quick-fix suggested	❗
Disabled	💡

Konstruktor wie gehabt mit ^N und /**⏎



```
package at.tws.myfirstjavaprogram.entity;

/**
 * Created by stuetz
 */
public class Person {
    private String firstName;
    private String lastName;
    private int zipCode;


    Getter- und Setter-Methoden
}
```

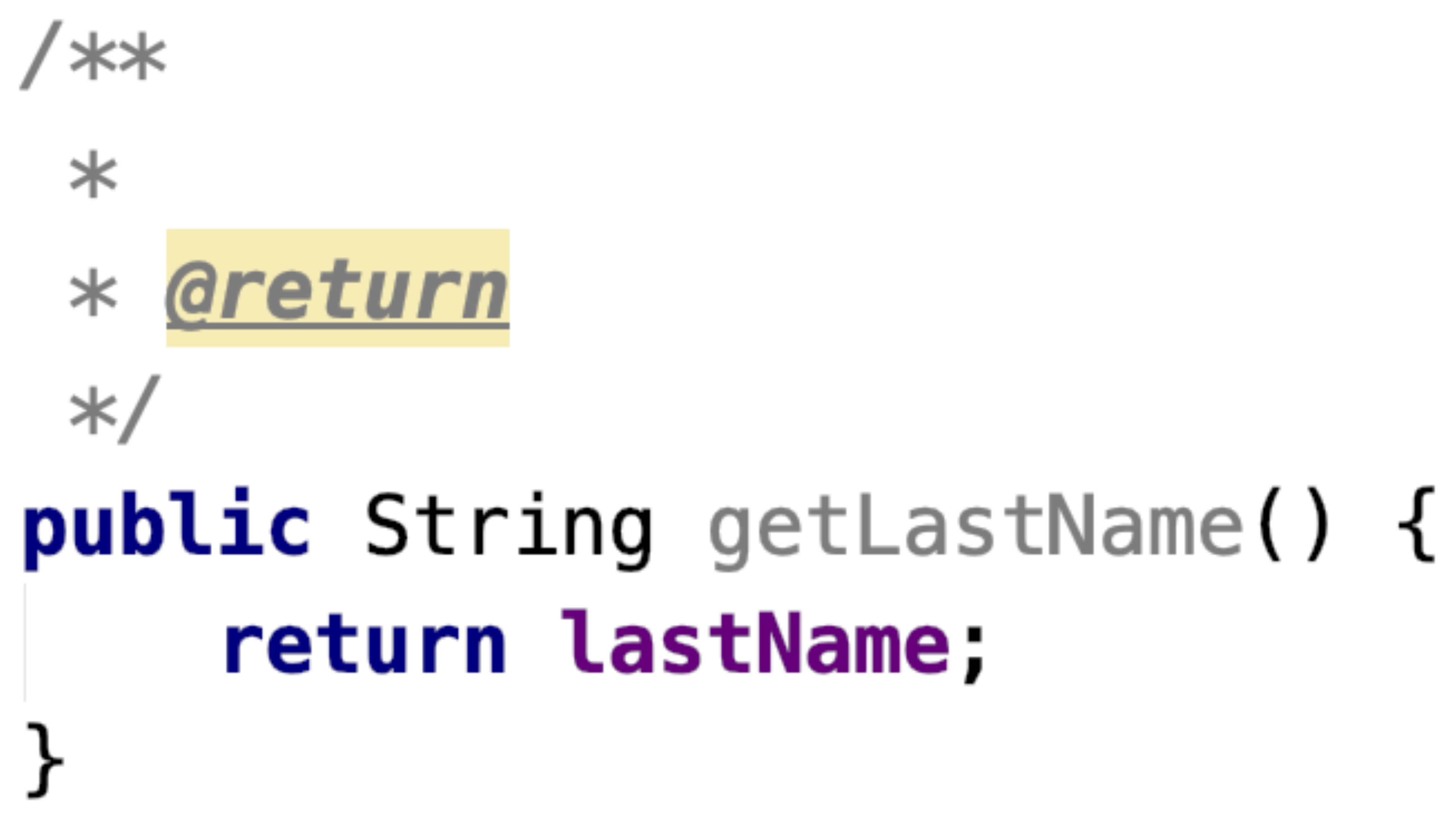
Choose Fields to Initialize by Constructor
at.tws.myfirstjavaprogram.entity.Person
firstName:String
lastName:String
zipCode:int
Cancel
Select None
OK



```
/**
 * Konstruktor initialisiert die Daten
 * @param firstName
 * @param lastName
 * @param zipCode
 */
public Person(String firstName, String lastName, int zipCode) {
    this.firstName = firstName;
    this.lastName = lastName;
    this.zipCode = zipCode;
}
```

Getter/Setter ist normale Methode

- Unterstützung beim Anlegen durch IntelliJ IDEA
- Natürlich sind auch berechnete Getter möglich

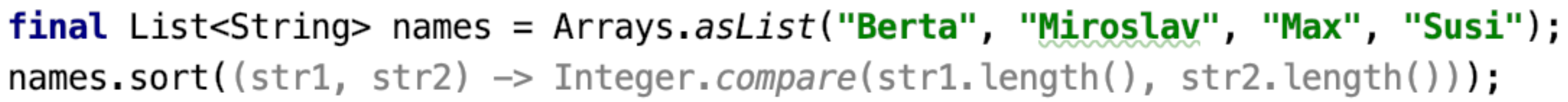


```java
/**
 *
 * @return
 */
public String getLastName() {
    return lastName;
}
```

Collections sortieren

- `Comparable` wie in C#, aber ohne ‚`I`‘
- `compareTo` verhält sich gleich

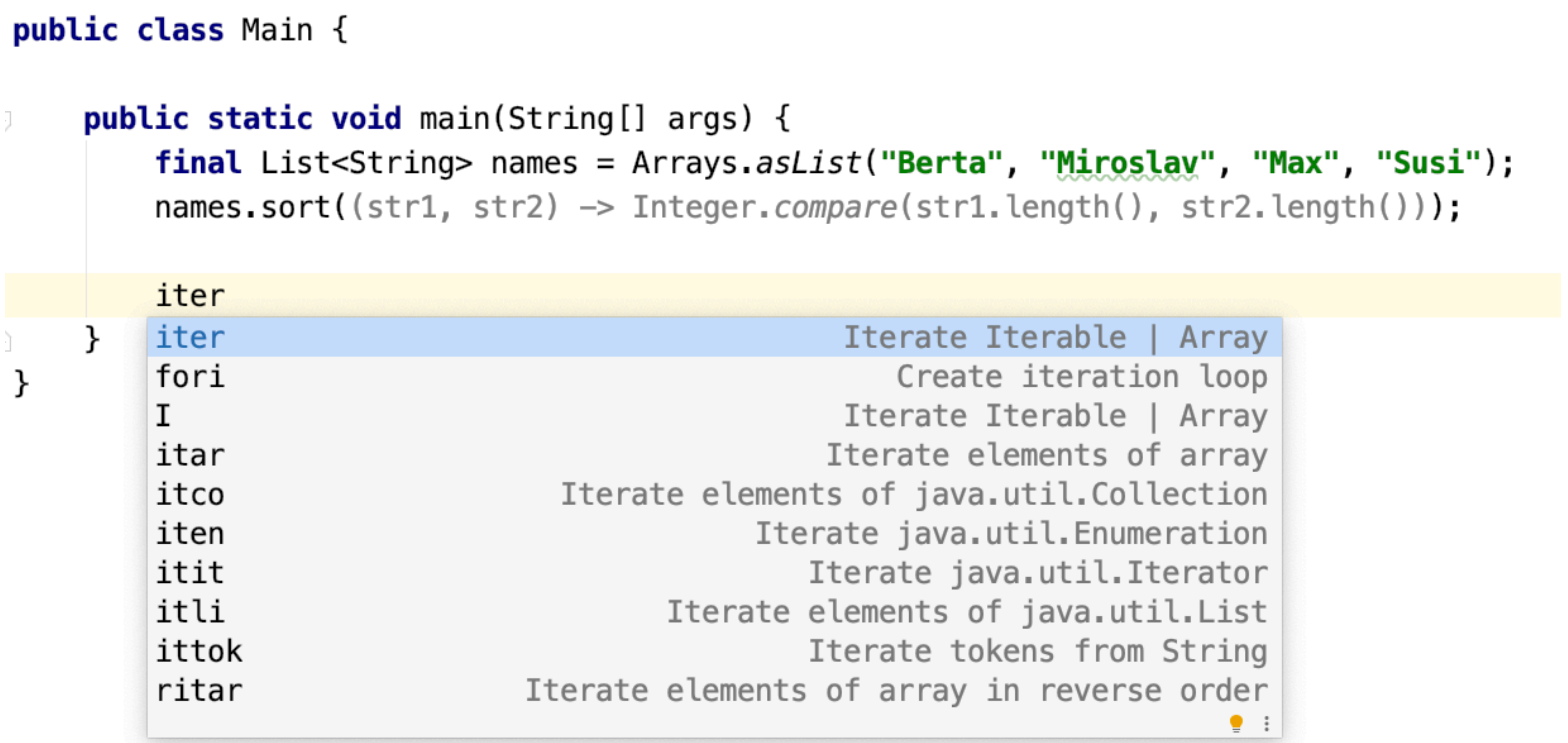
- Collections
    - Interface `List` statt `IList`
    - Generics wie gehabt
    - statische Methode `Collections.sort();`
    - oder ab Java 8: `<list>.sort(...)`



```java
final List<String> names = Arrays.asList("Berta", "Miroslav", "Max", "Susi");
names.sort((str1, str2) -> Integer.compare(str1.length(), str2.length()));
```

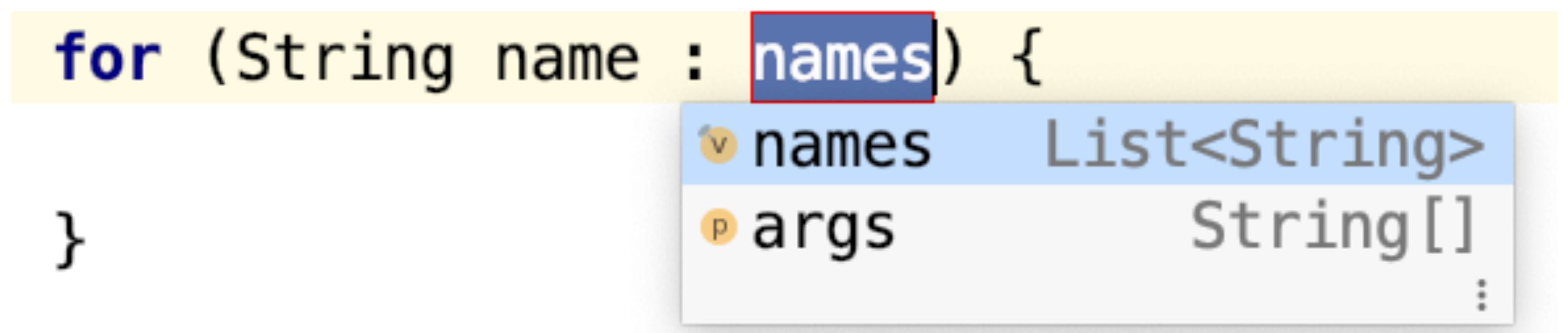
Live Templates

⌘J



```
public class Main {

    public static void main(String[] args) {
        final List<String> names = Arrays.asList("Berta", "Miroslav", "Max", "Susi");
        names.sort((str1, str2) -> Integer.compare(str1.length(), str2.length()));

        iter
    }
}
```

```
iter                                  Iterate Iterable | Array
fori                                       Create iteration loop
I                                     Iterate Iterable | Array
itar                                    Iterate elements of array
itco                Iterate elements of java.util.Collection
iten                            Iterate java.util.Enumeration
itit                               Iterate java.util.Iterator
itli                      Iterate elements of java.util.List
ittok                              Iterate tokens from String
ritar              Iterate elements of array in reverse order
```



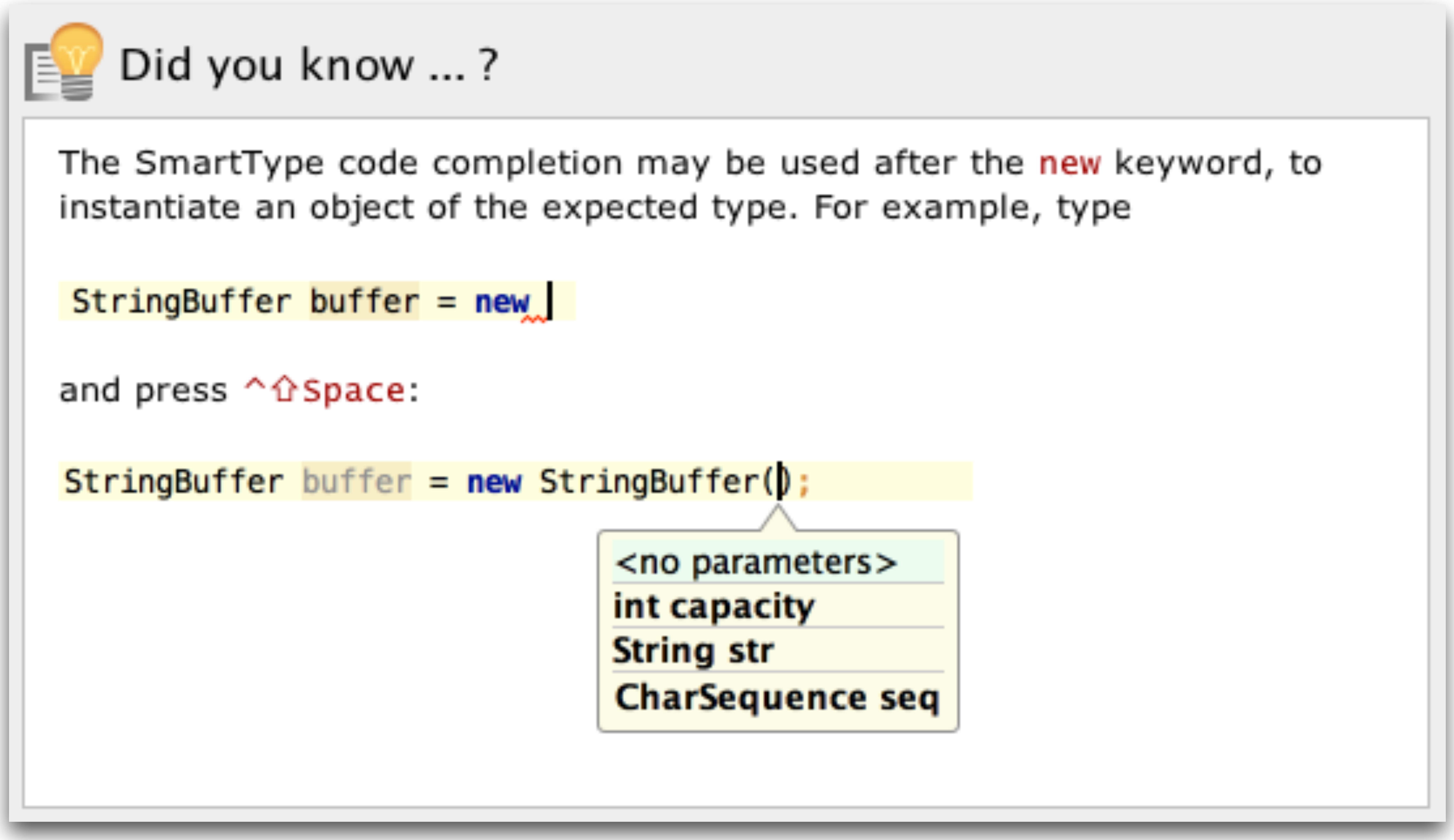
```
for (String name : names) {

}
```

```
names      List<String>
args         String[]
```

```
iter ... Iterate (for each..in)
itit ... java.util.Iterator
itli ... Iterate over a List
itar ... Iterate elements of array
ritar ... Iterate elements of array in reverse order
```

Insert Live Template via ⌘J (Ctrl+J for Win/Linux)

SmartType Code Completion ^⌥Space



Did you know ... ?

The SmartType code completion may be used after the new keyword, to instantiate an object of the expected type. For example, type

```
StringBuffer buffer = new 
```

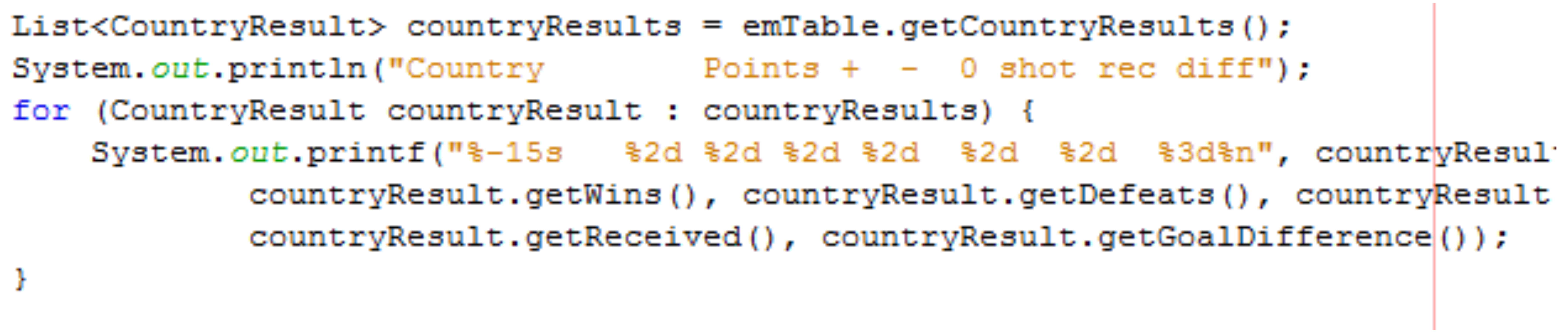
and press ^⇧Space:

```
StringBuffer buffer = new StringBuffer();
```

Spezielles

- Syntax für foreach

- Ausgabe formatiert

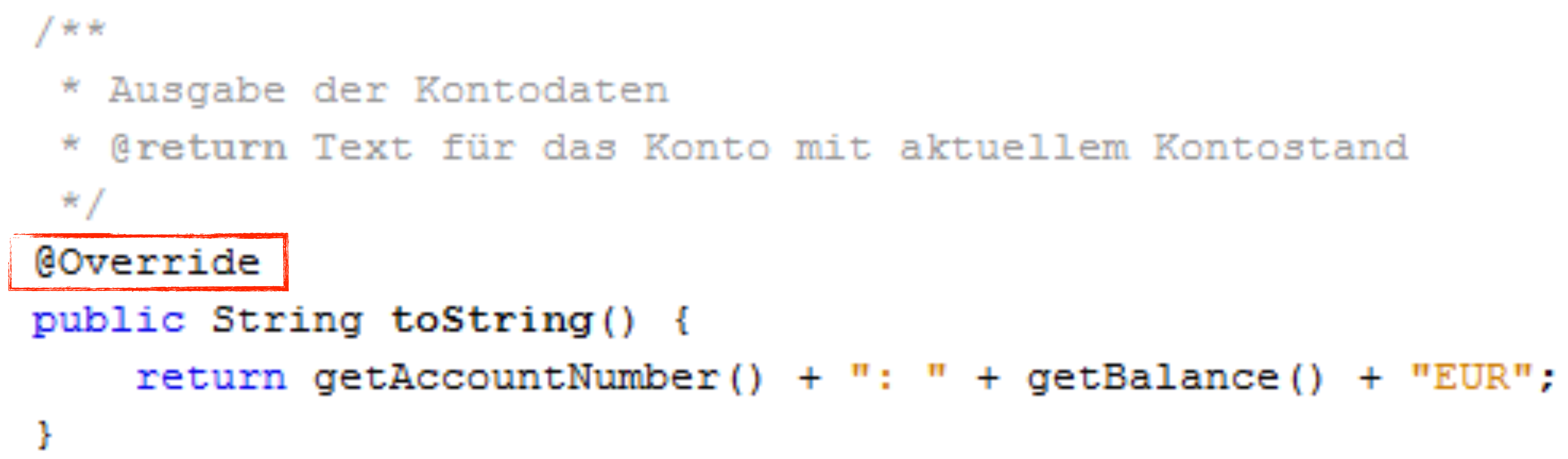


```
List<CountryResult> countryResults = emTable.getCountryResults();
System.out.println("Country         Points +  -  0 shot rec diff");
for (CountryResult countryResult : countryResults) {
    System.out.printf("%-15s   %2d %2d %2d %2d  %2d  %2d  %3d%n", countryResul
            countryResult.getWins(), countryResult.getDefeats(), countryResult
            countryResult.getReceived(), countryResult.getGoalDifference());
}
```

- Codezeile vervollständigen mit

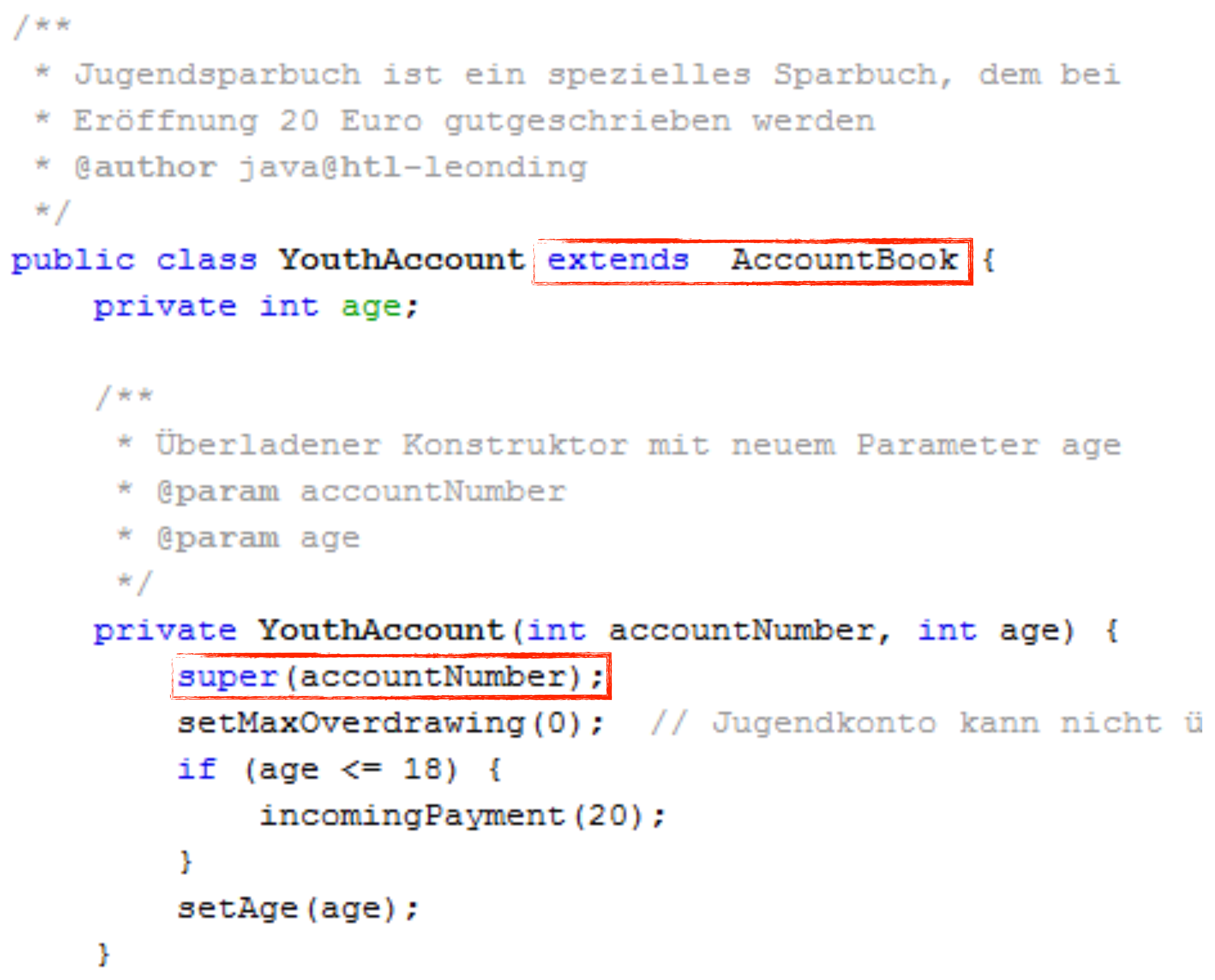
  <Shift>⌘ ↵   bzw.   <Ctrl><Shift> <Enter>

Methode überschreiben



```
/**
 * Ausgabe der Kontodaten
 * @return Text für das Konto mit aktuellem Kontostand
 */
@Override
public String toString() {
    return getAccountNumber() + ": " + getBalance() + "EUR";
}
```

Abgeleitete Klasse

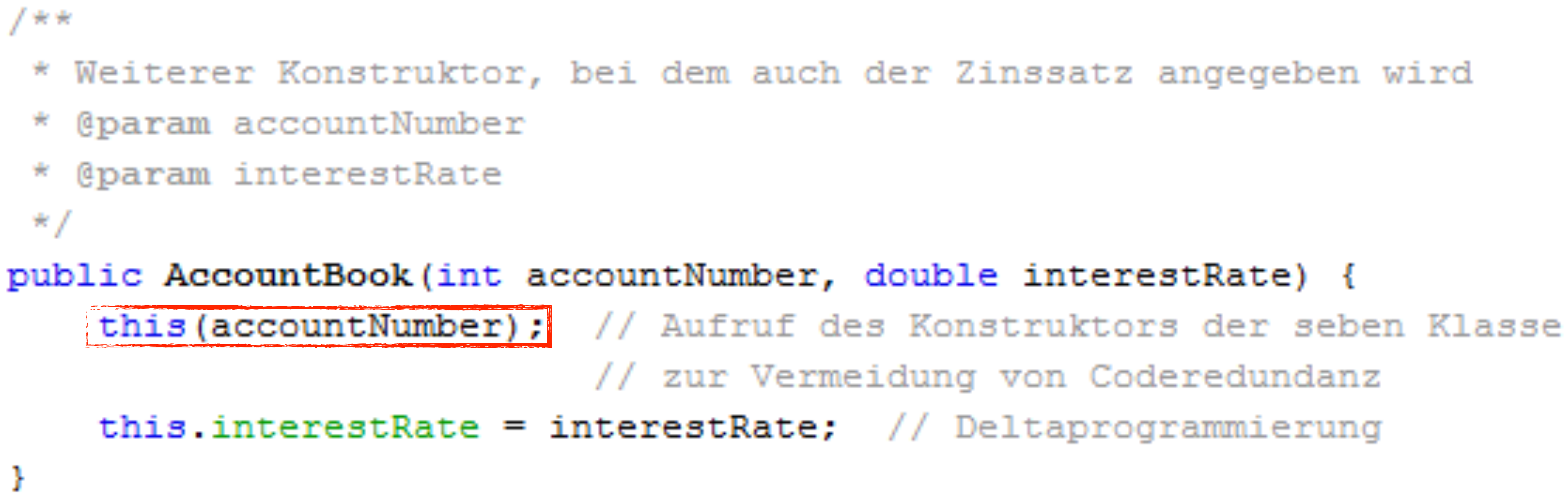


```java
/**
 * Jugendsparbuch ist ein spezielles Sparbuch, dem bei
 * Eröffnung 20 Euro gutgeschrieben werden
 * @author java@htl-leonding
 */
public class YouthAccount extends  AccountBook {
    private int age;

    /**
     * Überladener Konstruktor mit neuem Parameter age
     * @param accountNumber
     * @param age
     */
    private YouthAccount(int accountNumber, int age) {
        super(accountNumber);
        setMaxOverdrawing(0);  // Jugendkonto kann nicht ü
        if (age <= 18) {
            incomingPayment(20);
        }
        setAge(age);
    }
```

Überladener Konstruktor

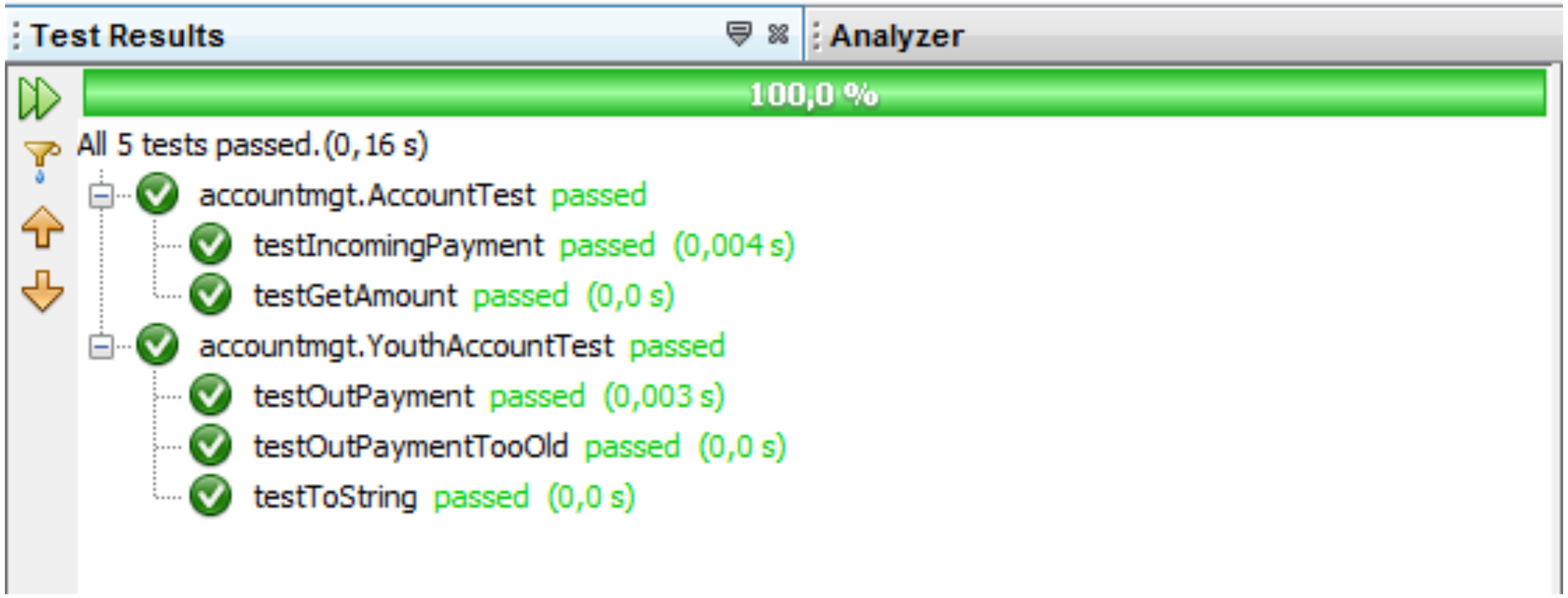
- Codeverdopplung vermeiden

- Zugriff auf anderen Konstruktor



```java
/**
 * Weiterer Konstruktor, bei dem auch der Zinssatz angegeben wird
 * @param accountNumber
 * @param interestRate
 */
public AccountBook(int accountNumber, double interestRate) {
    this(accountNumber);   // Aufruf des Konstruktors der seben Klasse
                           // zur Vermeidung von Coderedundanz
    this.interestRate = interestRate;  // Deltaprogrammierung
}
```

Klassen realisieren, die Tests bestehen

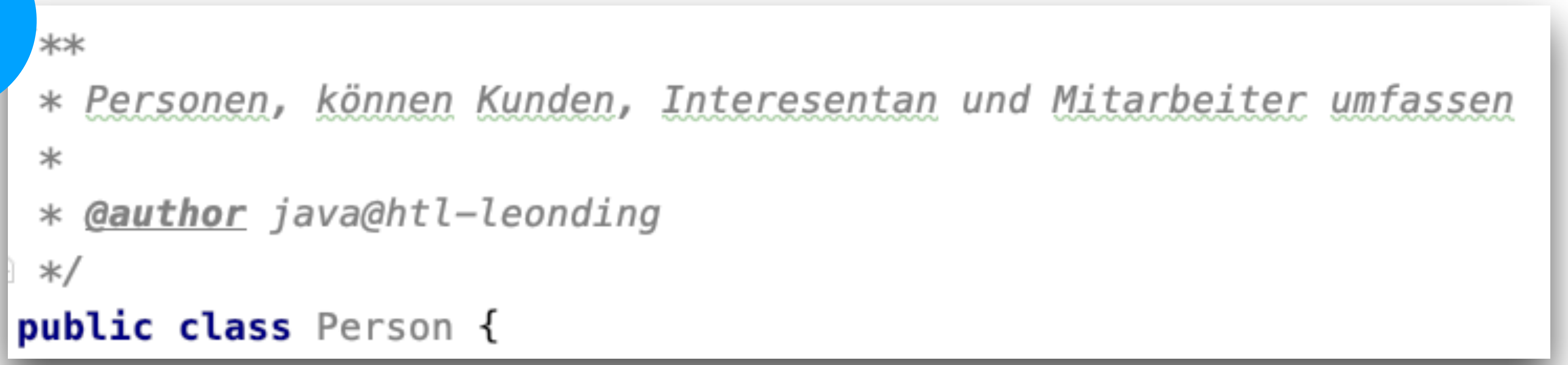


```
100,0 %
All 5 tests passed.(0,16 s)
accountmgt.AccountTest passed
    testIncomingPayment passed (0,004 s)
    testGetAmount passed (0,0 s)
accountmgt.YouthAccountTest passed
    testOutPayment passed (0,003 s)
    testOutPaymentTooOld passed (0,0 s)
    testToString passed (0,0 s)
```

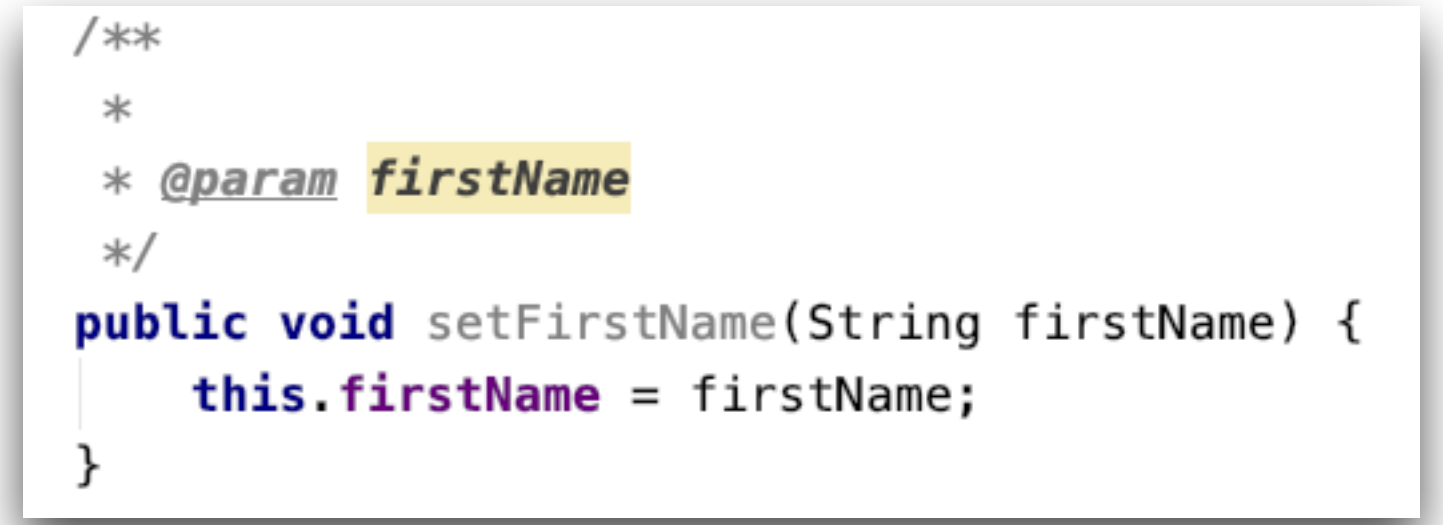
JavaDoc: Gerüst anlegen

- Über Methode, Klasse, … /** eingeben

1

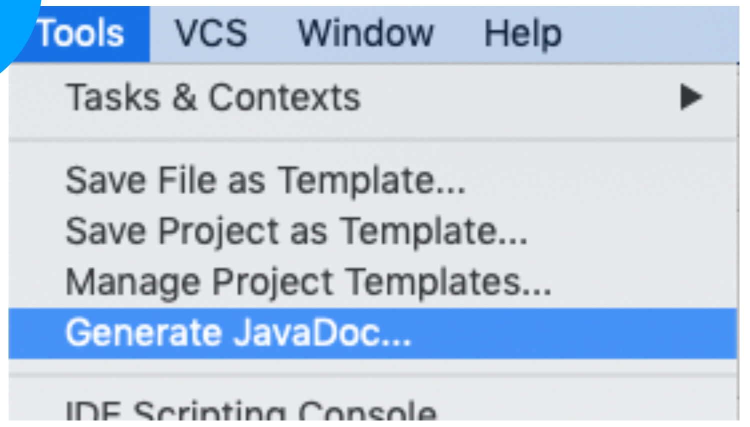


```
/**
 * Personen, können Kunden, Interesentan und Mitarbeiter umfassen
 *
 * @author java@htl-leonding
 */
public class Person {
```

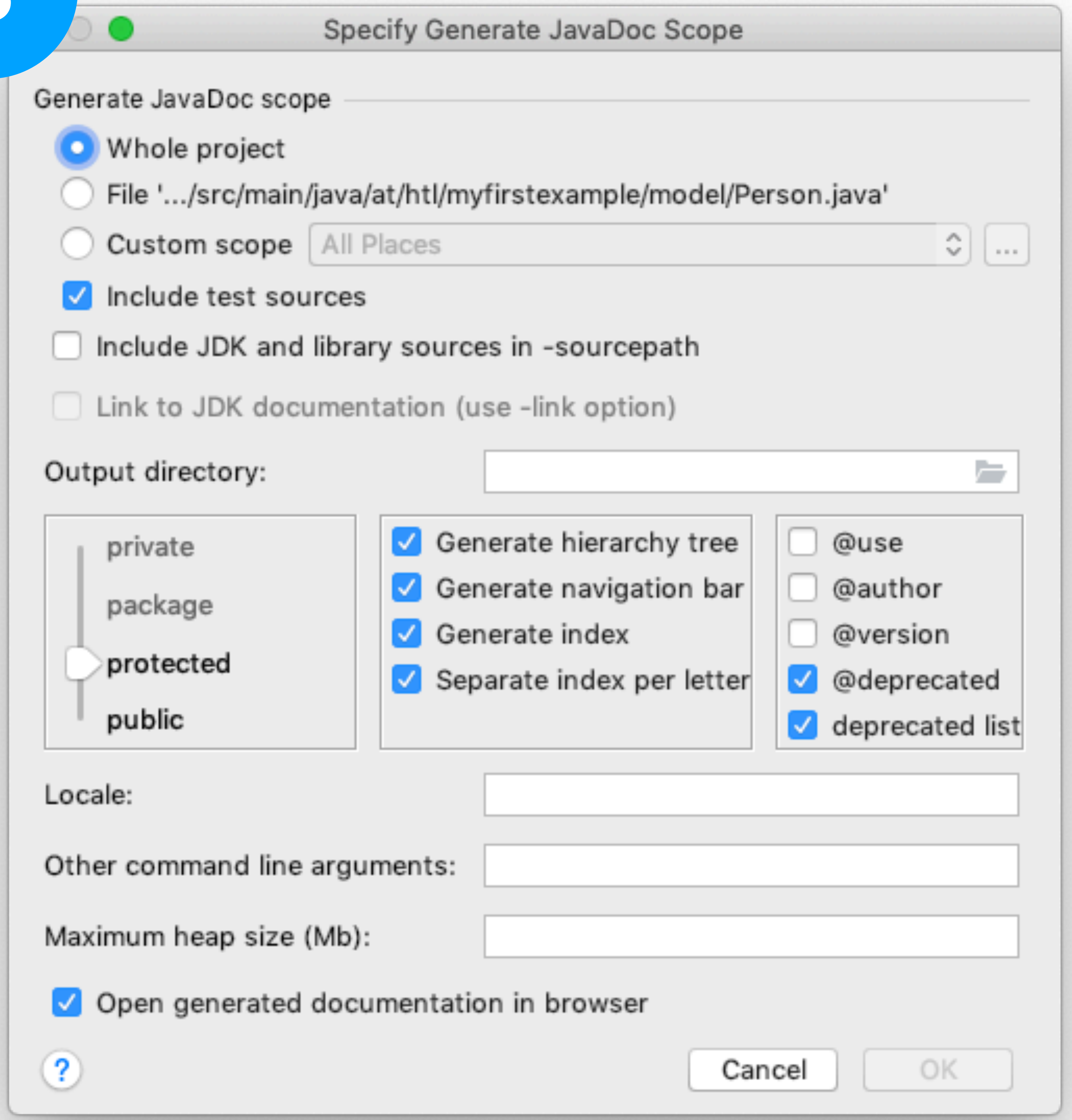


```
/**
 *
 * @param firstName
 */
public void setFirstName(String firstName) {
    this.firstName = firstName;
}
```

2



3



Specify Generate JavaDoc Scope

Generate JavaDoc scope

- [✓ Whole project]
- File '.../src/main/java/at/htl/myfirstexample/model/Person.java'
- Custom scope: All Places

- [✓] Include test sources
- [ ] Include JDK and library sources in -sourcepath
- [ ] Link to JDK documentation (use -link option)

Output directory:

private / package / [✓ protected] / public

- [✓] Generate hierarchy tree
- [✓] Generate navigation bar
- [✓] Generate index
- [✓] Separate index per letter

- [ ] @use
- [ ] @author
- [ ] @version
- [✓] @deprecated
- [✓] deprecated list

Locale:

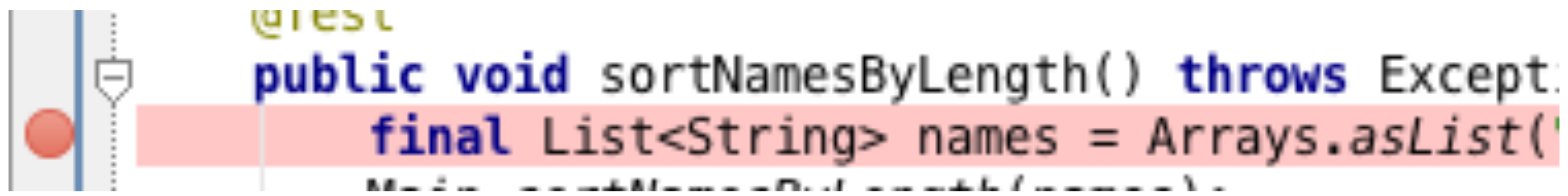
Other command line arguments:

Maximum heap size (Mb):

- [✓] Open generated documentation in browser

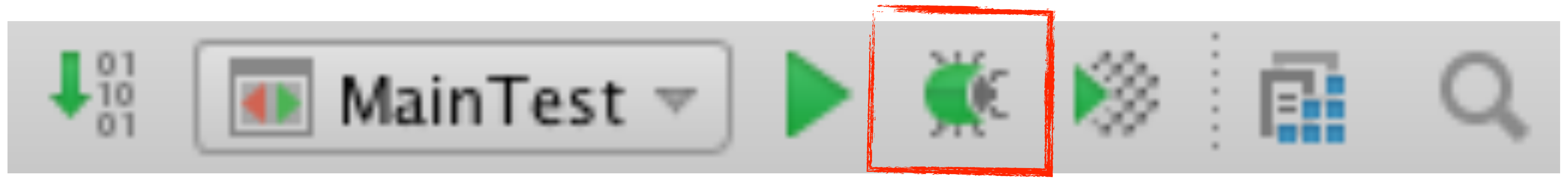
Unit-Test debuggen

- Breakpoint setzen

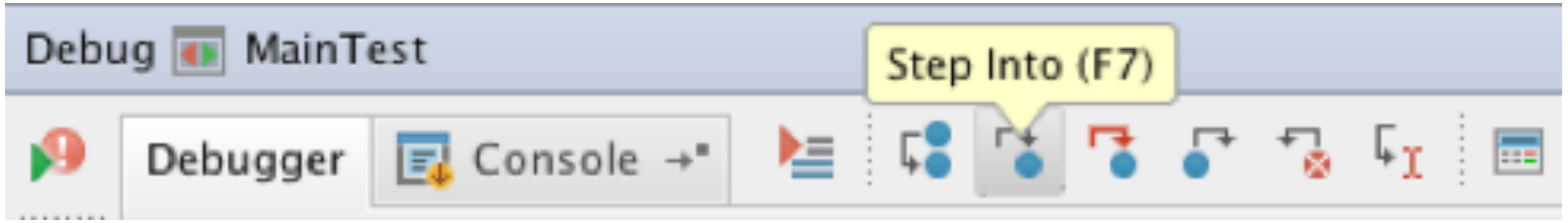


```
@Test
public void sortNamesByLength() throws Except
    final List<String> names = Arrays.asList(
```

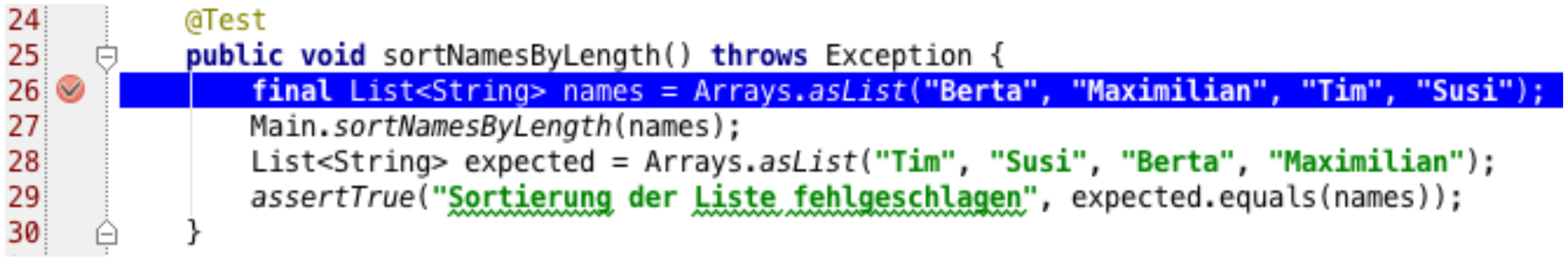
- Test in Debugmode starten



- Step into → F7
- Step over → F8

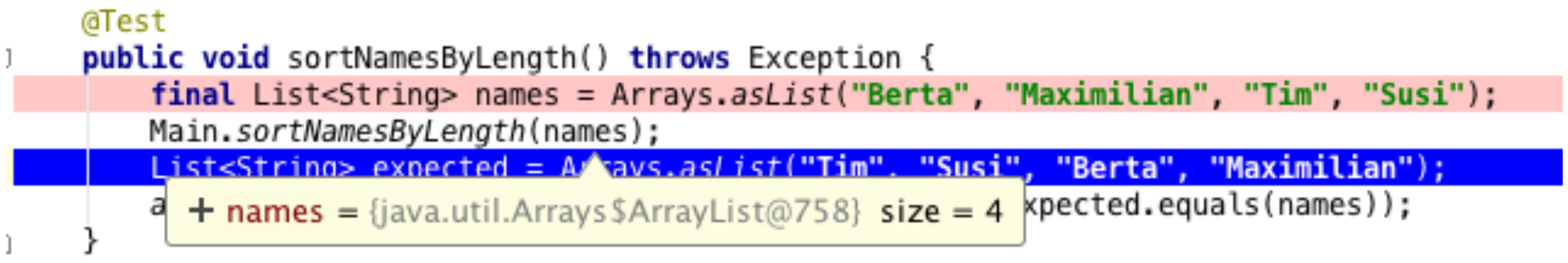


- Debuggen beenden Shift-F2 bzw. ⌘F2



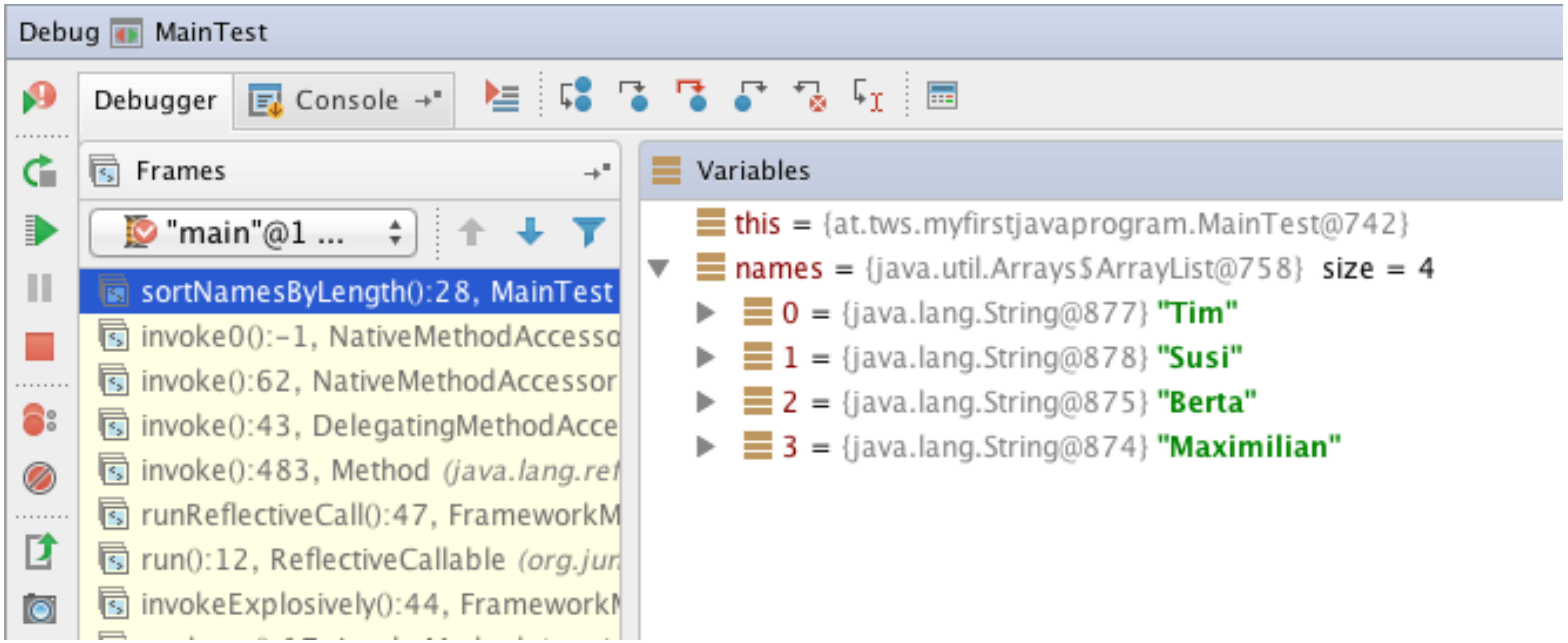
```
    @Test
    public void sortNamesByLength() throws Exception {
        final List<String> names = Arrays.asList("Berta", "Maximilian", "Tim", "Susi");
        Main.sortNamesByLength(names);
        List<String> expected = Arrays.asList("Tim", "Susi", "Berta", "Maximilian");
        assertTrue("Sortierung der Liste fehlgeschlagen", expected.equals(names));
    }
```

Variablen einsehen (Watchpoints)



```
@Test
public void sortNamesByLength() throws Exception {
    final List<String> names = Arrays.asList("Berta", "Maximilian", "Tim", "Susi");
    Main.sortNamesByLength(names);
    List<String> expected = Arrays.asList("Tim", "Susi", "Berta", "Maximilian");
    a                                                        xpected.equals(names));
}
```

```
+ names = {java.util.Arrays$ArrayList@758} size = 4
```



```
sortNamesByLength():28, MainTest
invoke0():-1, NativeMethodAccesso
invoke():62, NativeMethodAccessor
invoke():43, DelegatingMethodAcce
invoke():483, Method (java.lang.ref
runReflectiveCall():47, FrameworkM
run():12, ReflectiveCallable (org.jur
invokeExplosively():44, FrameworkM
```

```
this = {at.tws.myfirstjavaprogram.MainTest@742}
names = {java.util.Arrays$ArrayList@758} size = 4
    0 = {java.lang.String@877} "Tim"
    1 = {java.lang.String@878} "Susi"
    2 = {java.lang.String@875} "Berta"
    3 = {java.lang.String@874} "Maximilian"
```

Noch
Fragen?

HTL LEONDING

Schön, hier zu lernen