

Programmieren I

Entwicklungsumgebungen Eclipse, IntelliJ IDEA und NetBeans

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);  
    }  
}
```

Was ist Programmieren

- Quellcode schreiben

Texteditor

- Der Maschine verständlich machen („kompilieren“)

javac

- Mit Standardbibliotheken verknüpfen („linken“) // nicht in Java

- Erzeugtes Programm ausführen und Fehler beheben („Debuggen“)




java

Vereinfachung: Ein Tool für alles!

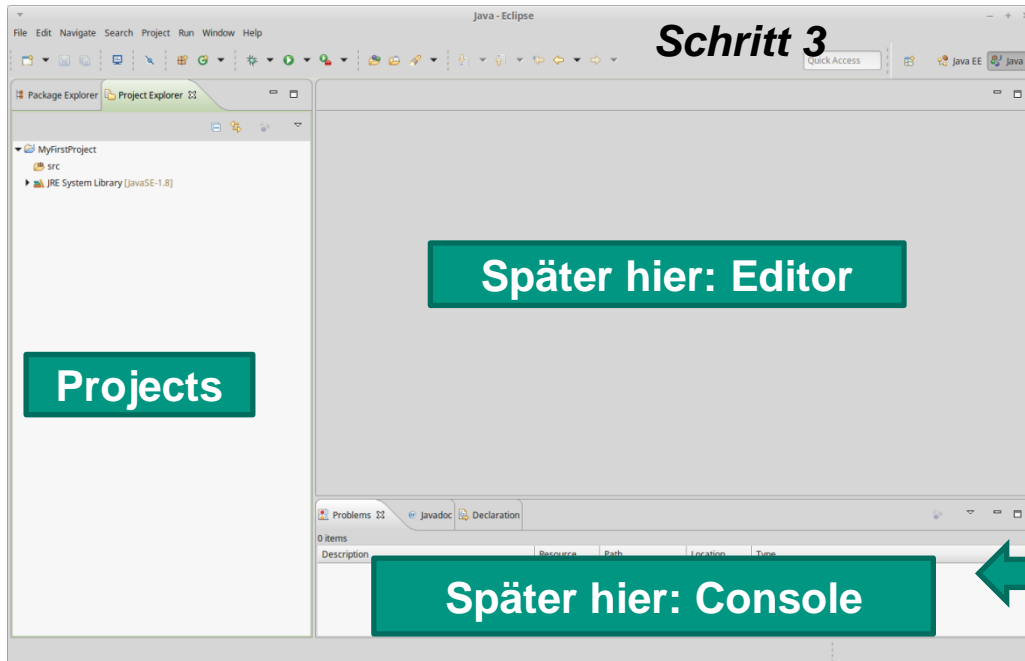
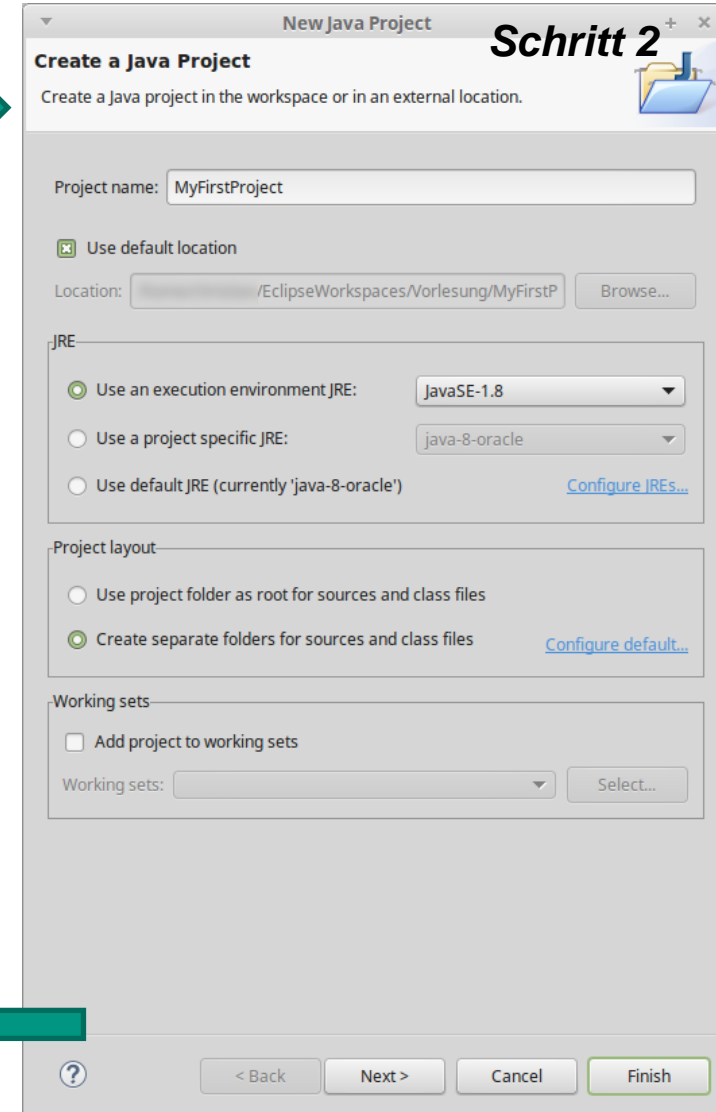
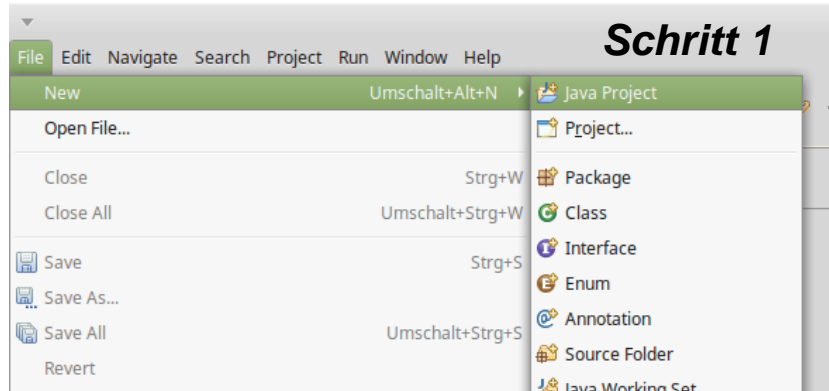
- Entwicklungsumgebungen bieten typischerweise Werkzeuge zum
 - Quellcode editieren
 - Code übersetzen
 - Linken
 - Ausführen
 - Debuggen

- Auch meist vorhanden:
 - Syntax-Highlighting
 - „Live“-Anzeige von Syntax-Fehlern
 - Unterstützung in Form verschiedener Code-Generatoren
 - Integration in verschiedene Tools (z.B. Versionskontrollsysteme)

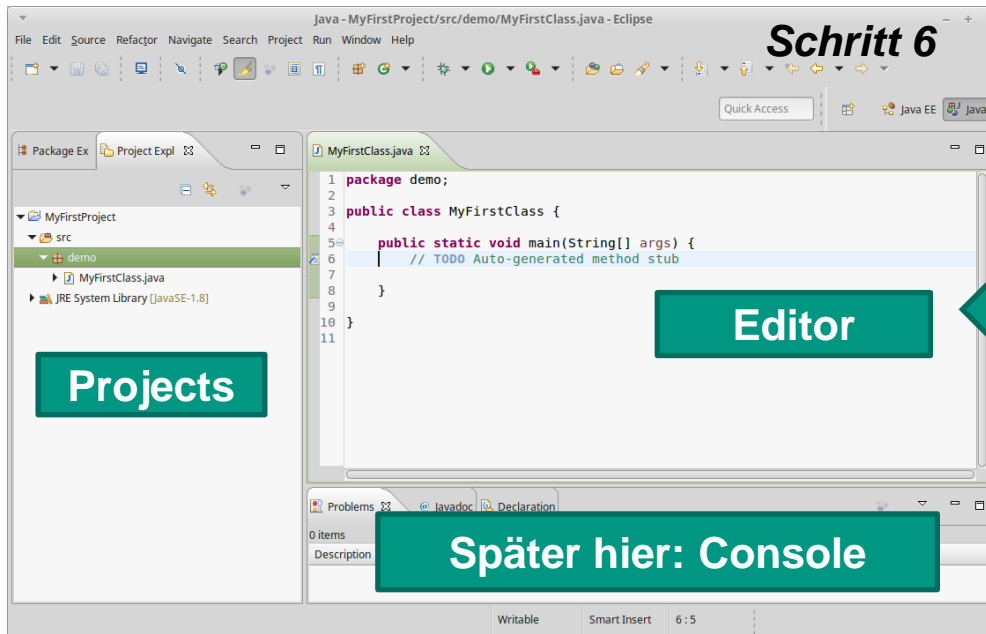
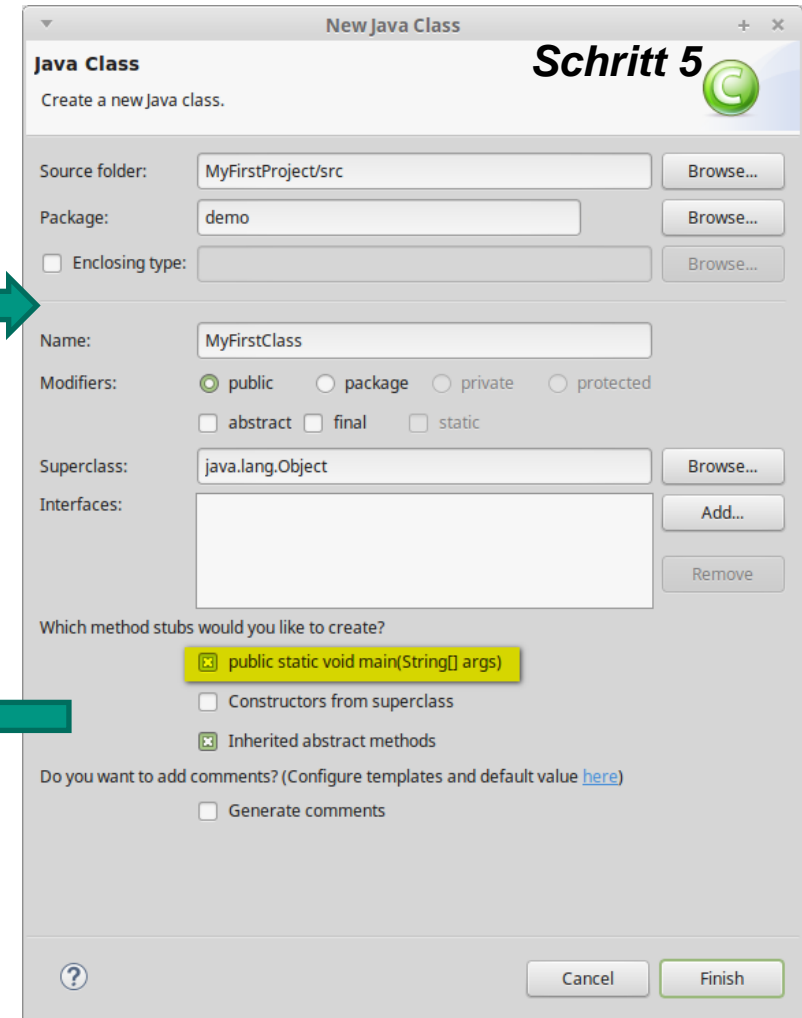
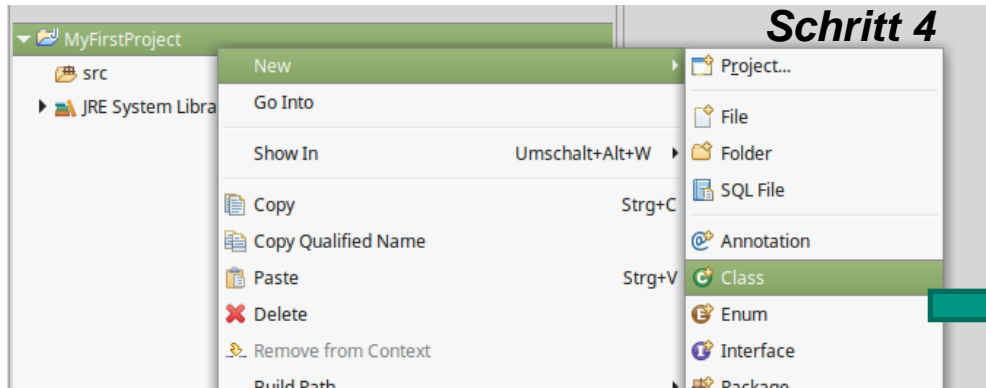
Entwicklungsumgebungen für Java

- Java Development Kit (JDK)
 - openJDK, bspw. von Adoptium: <https://adoptium.net>
- Englisch: *Integrated Development Environment* → IDE
- Eclipse (aktuell: Version 2024-12 / 4.34)  eclipse
<http://www.eclipse.org/>
- IntelliJ IDEA (aktuell: Version 2024.3.1.1)  IntelliJ IDEA
<https://www.jetbrains.com/idea/>
- Apache NetBeans (aktuell: Version 24)  Apache NetBeans
<https://netbeans.apache.org>

Neues Projekt (1) – Eclipse I

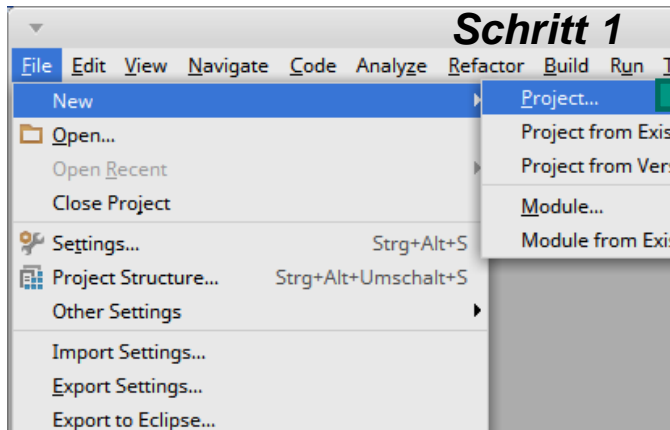


Neues Projekt (2) – Eclipse II

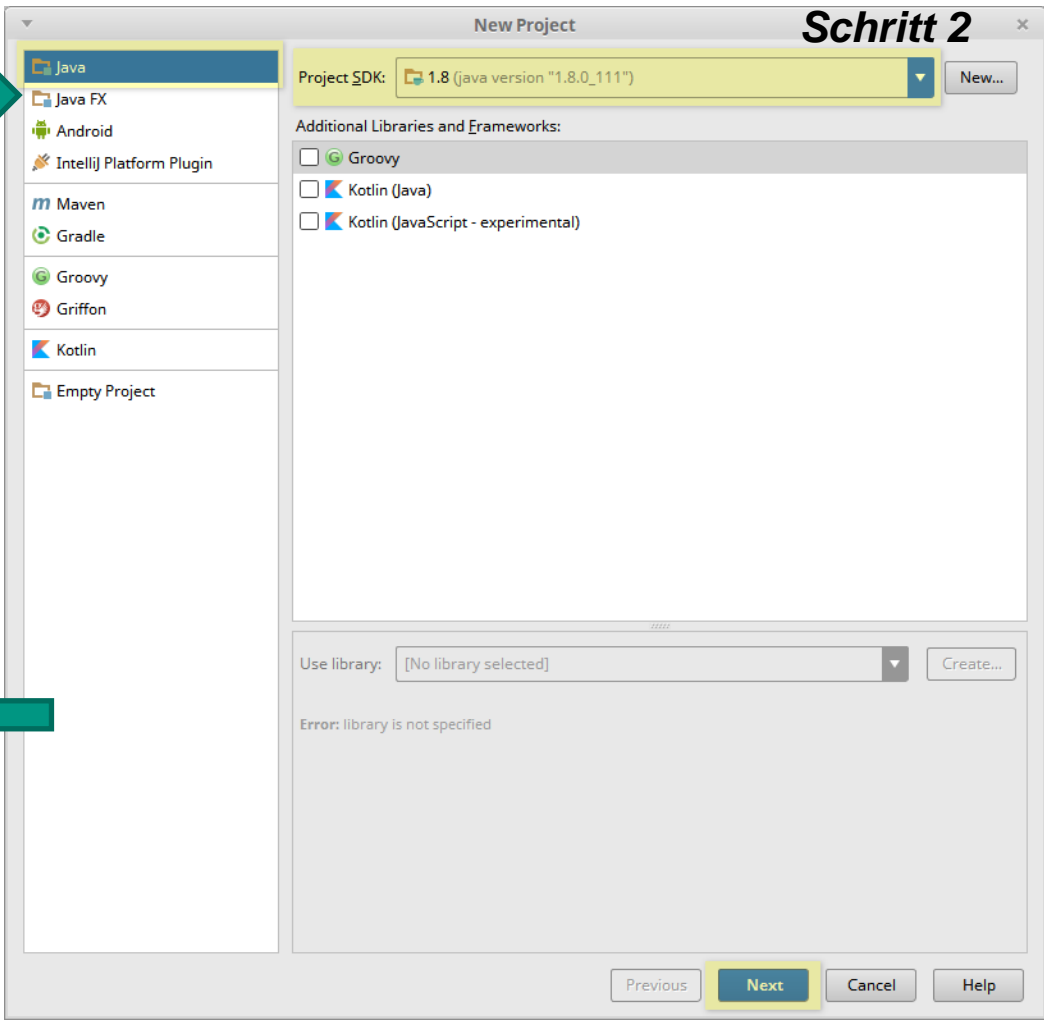


Neues Projekt (3) – IntelliJ IDEA I


Schritt 1



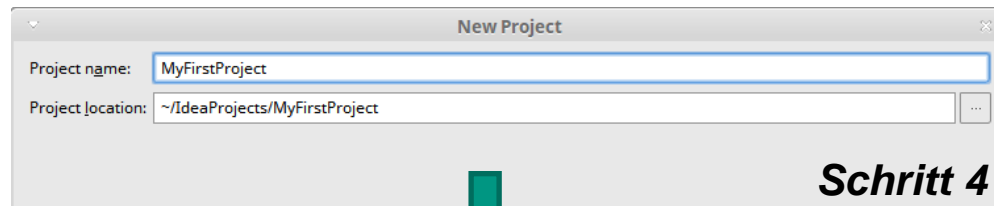
Schritt 2



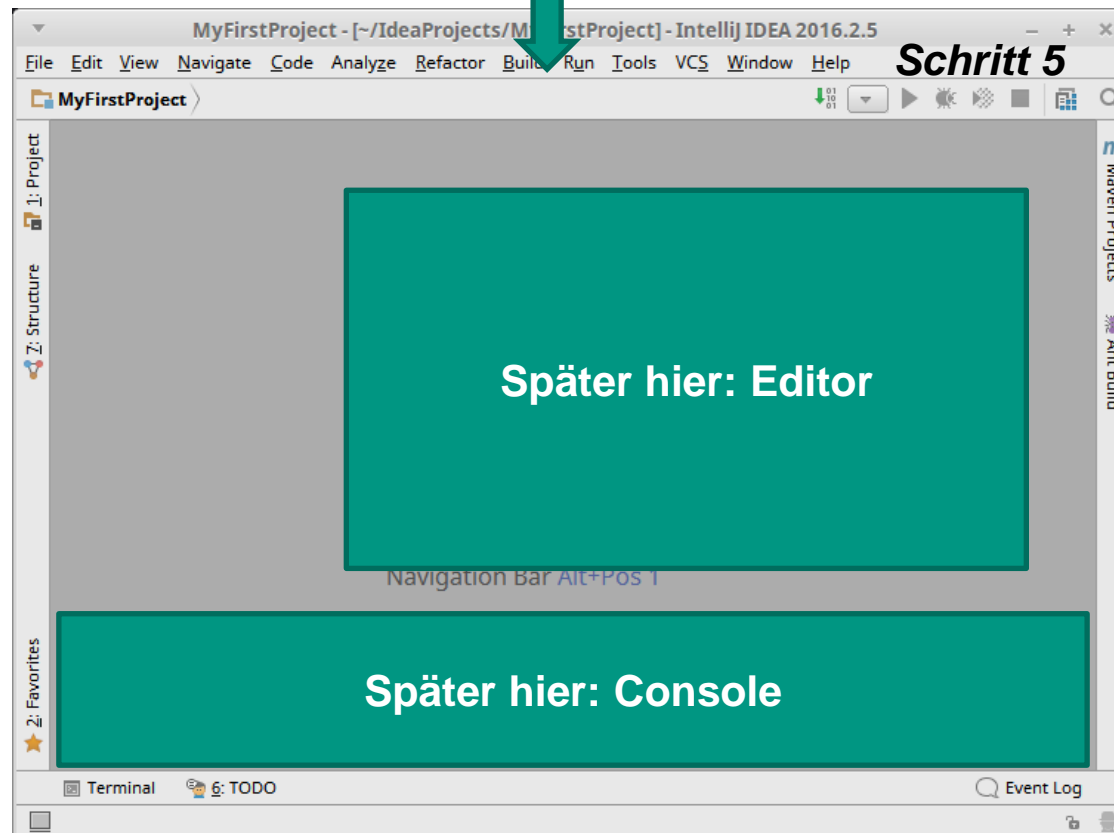
Schritt 3



Neues Projekt (4) – IntelliJ IDEA II



Schritt 4

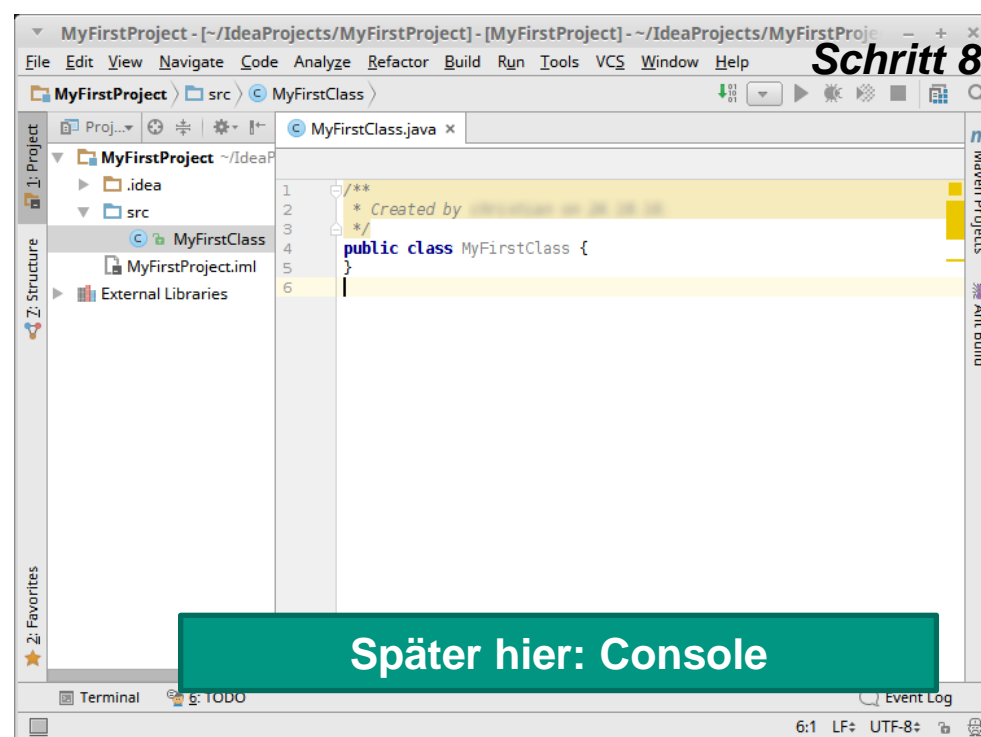
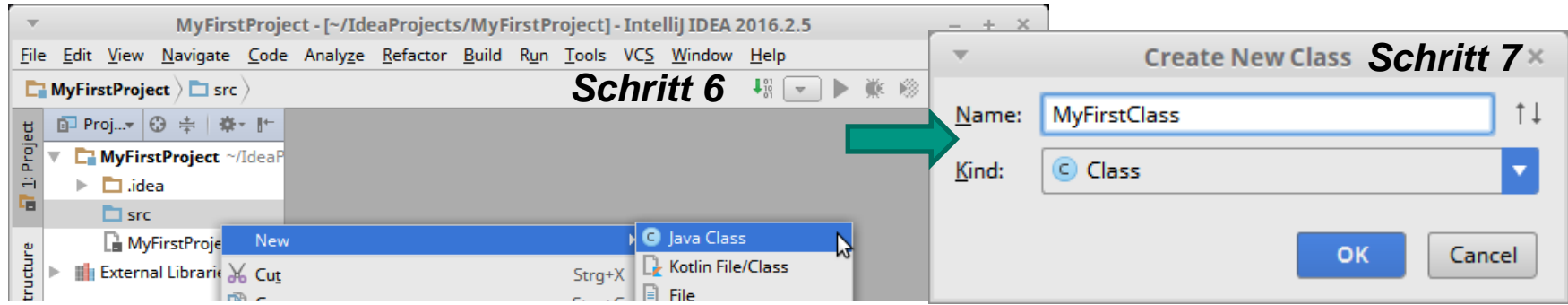


Schritt 5

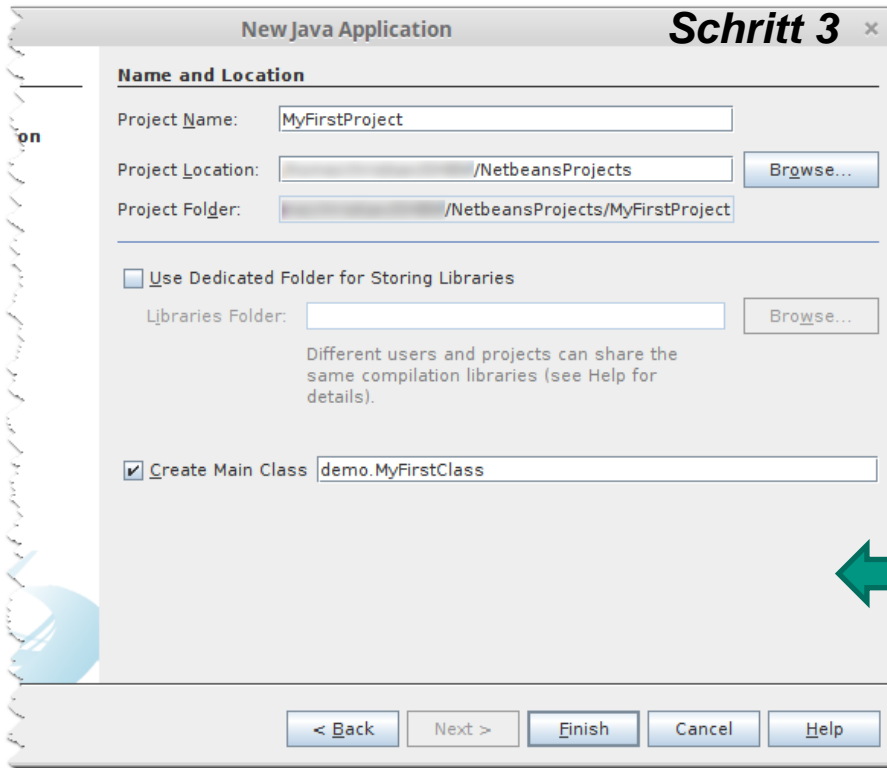
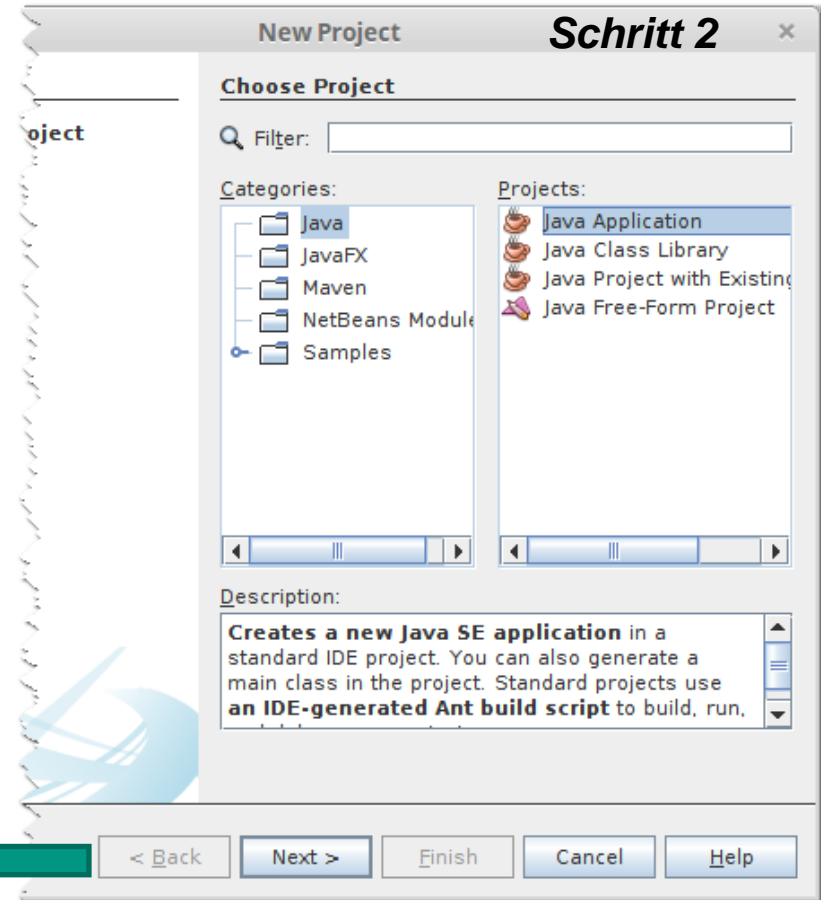
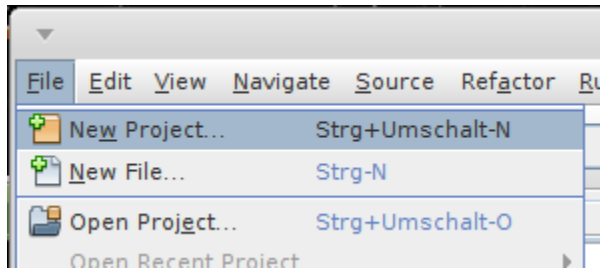
Später hier: Editor

Später hier: Console

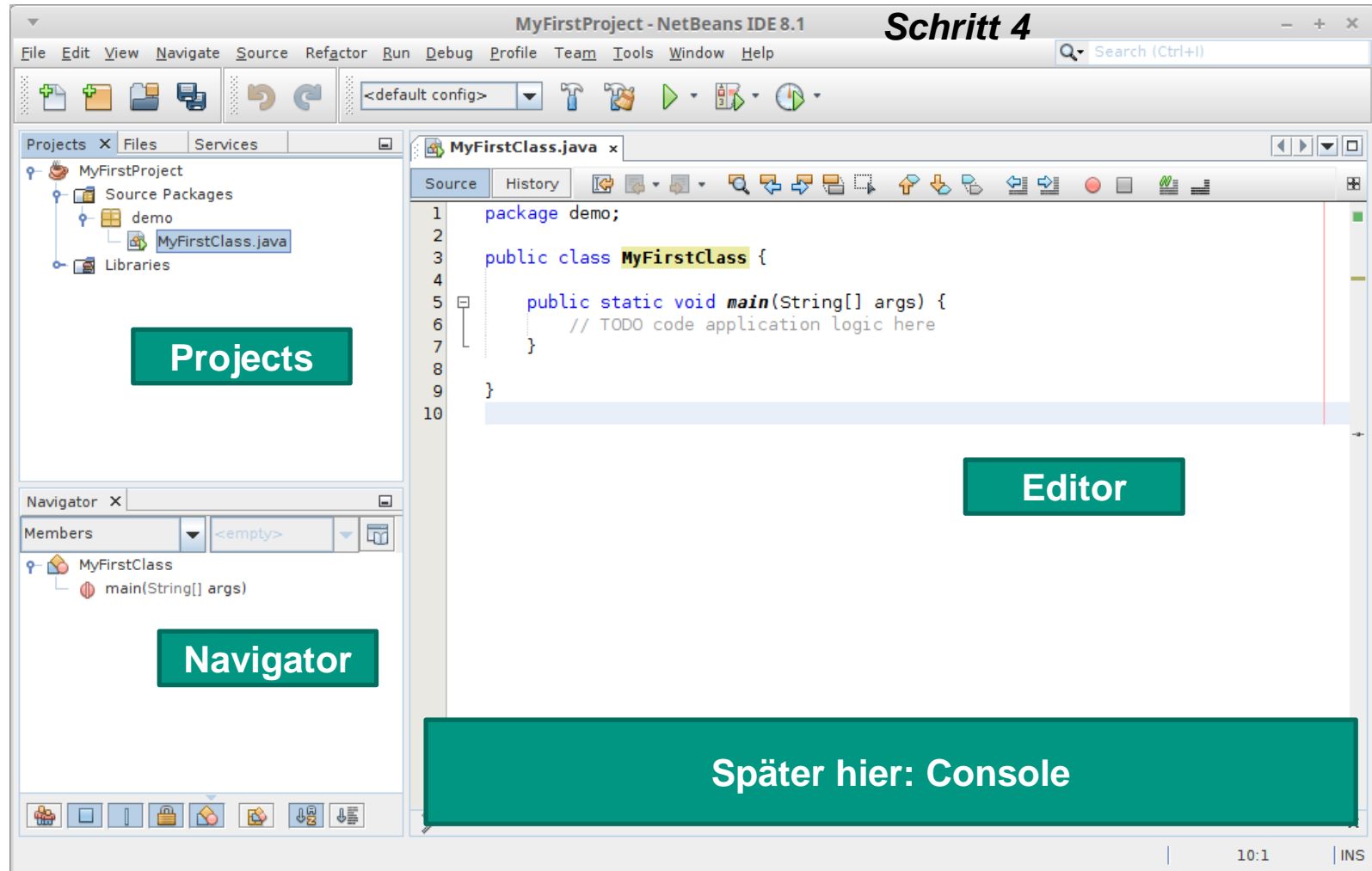
Neues Projekt (5) – IntelliJ IDEA III



Neues Projekt (6) – NetBeans I

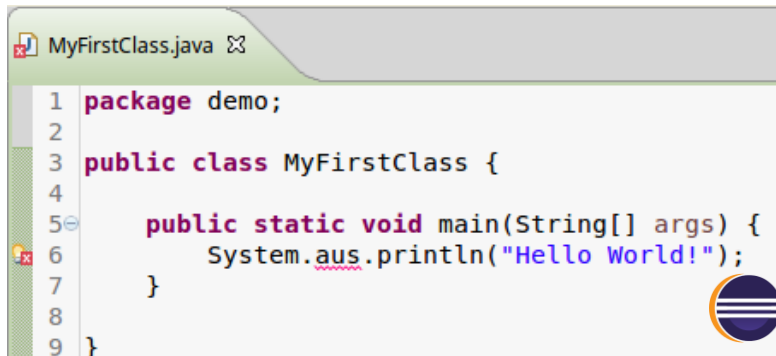


Neues Projekt (7) – NetBeans II

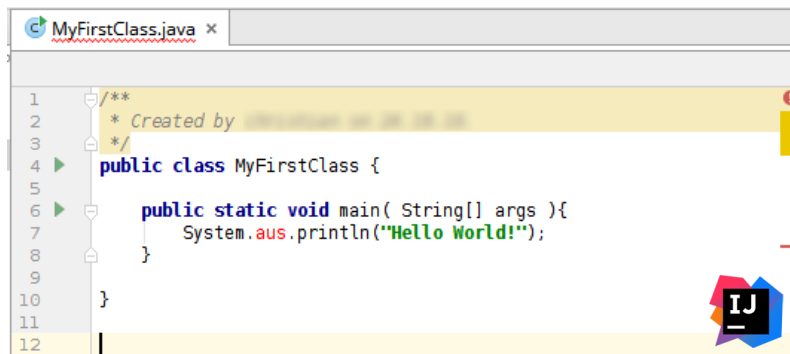


Projekt compilieren

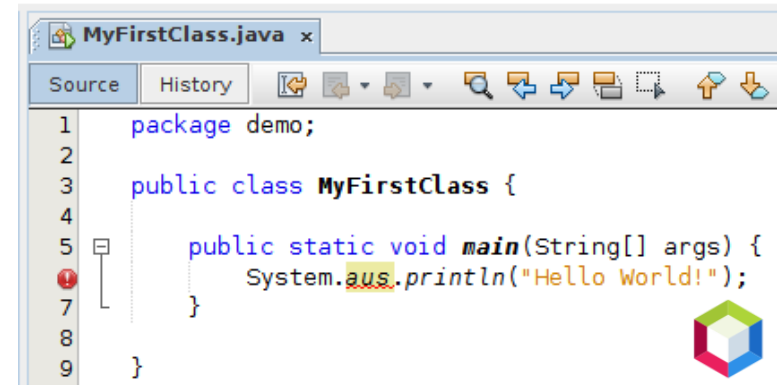
- IDEs übersetzen Java-Dateien beim Speichern in der Regel automatisch und zeigen ggf. Fehler an



```
1 package demo;
2
3 public class MyFirstClass {
4
5     public static void main(String[] args) {
6         System.out.println("Hello World!");
7     }
8
9 }
```



```
1 /**
2  * Created by ...
3  */
4 public class MyFirstClass {
5
6     public static void main( String[] args ){
7         System.out.println("Hello World!");
8     }
9
10 }
11
12
```



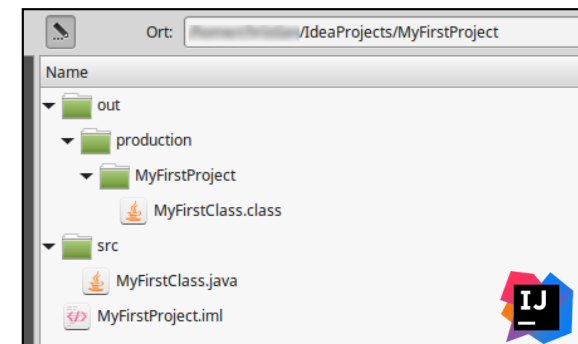
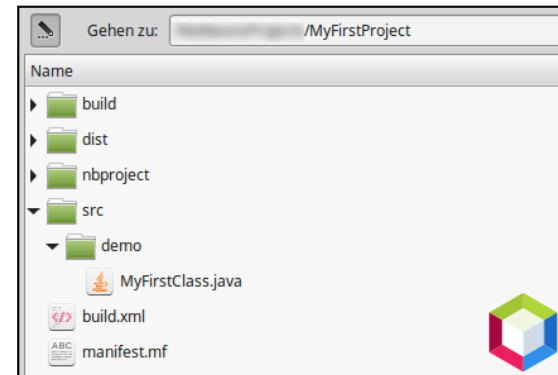
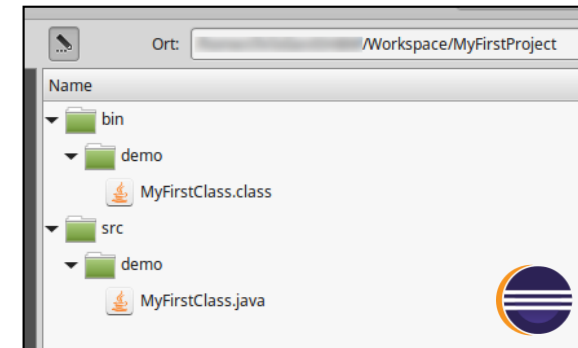
```
1 package demo;
2
3 public class MyFirstClass {
4
5     public static void main(String[] args) {
6         System.out.println("Hello World!");
7     }
8
9 }
```

Dateien und Dateibaum im Projektverzeichnis

- **src** („Source“)
 - Quellcode / `.java`-Dateien

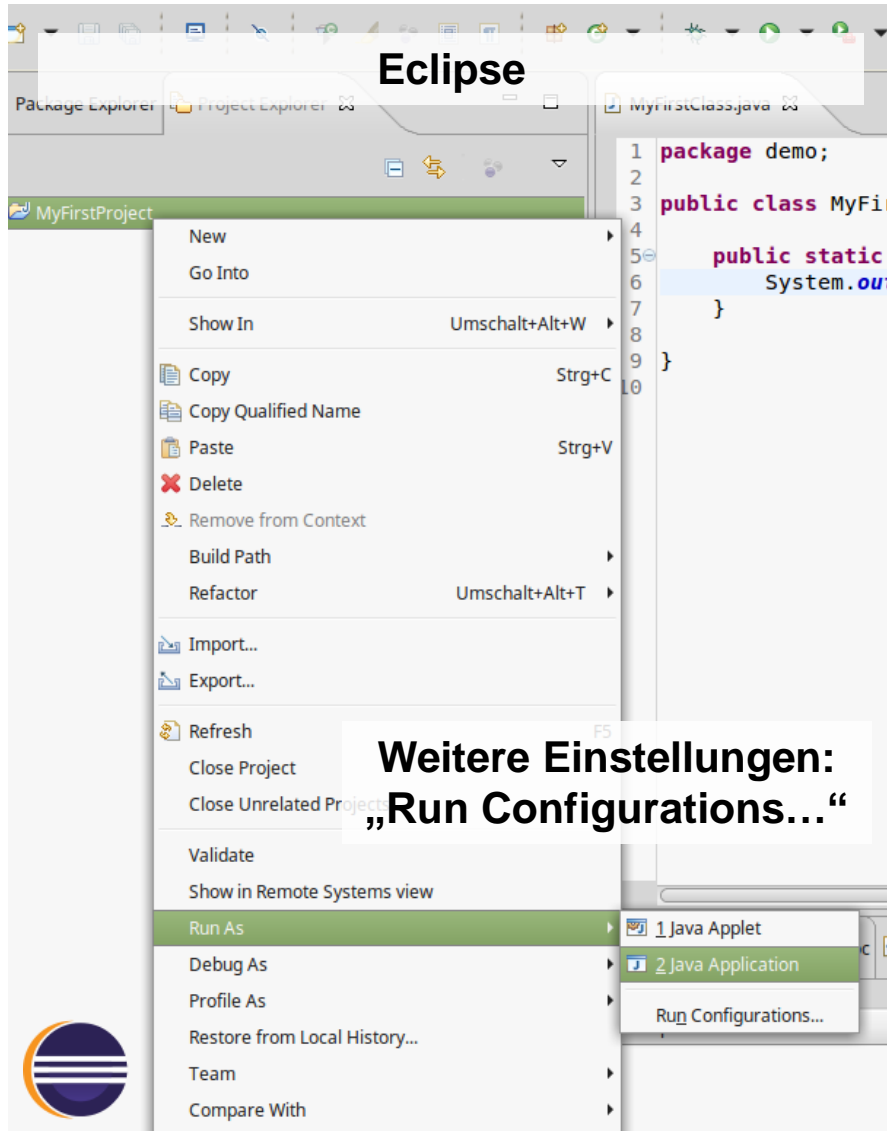
- **bin** (Eclipse)
 - build** (NetBeans)
 - out** (IntelliJ IDEA)
 - Bytecode / `.class`-Dateien

- NetBeans-Spezialität:
 - dist** („Distribution“)
 - `.jar`-Archiv / Dokumentation



Programm ausführen (1)

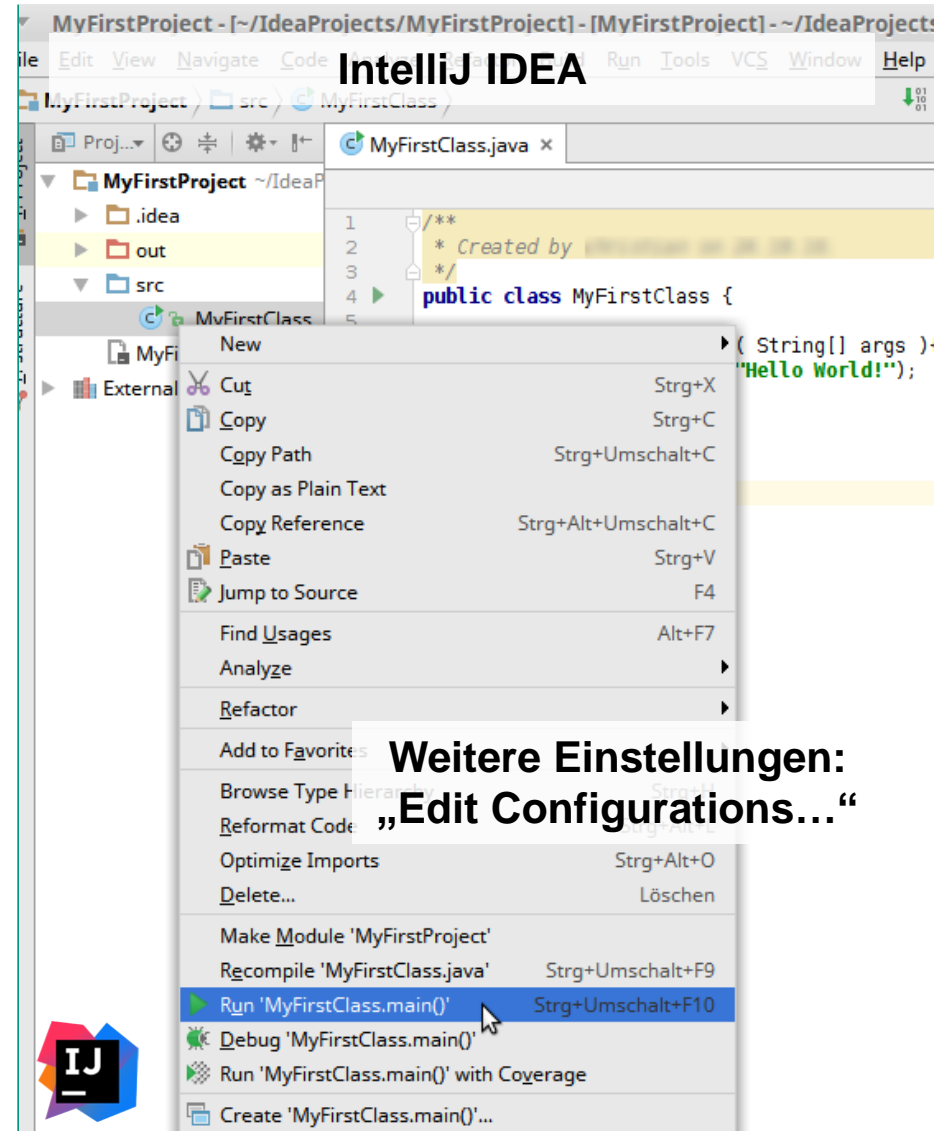
Eclipse



The screenshot shows the Eclipse IDE with a project named 'MyFirstProject'. A context menu is open over the 'MyFirstClass.java' file. The menu options include 'New', 'Go Into', 'Show In', 'Copy', 'Copy Qualified Name', 'Paste', 'Delete', 'Remove from Context', 'Build Path', 'Refactor', 'Import...', 'Export...', 'Refresh', 'Close Project', 'Close Unrelated Projects', 'Validate', 'Show in Remote Systems view', 'Run As', 'Debug As', 'Profile As', 'Restore from Local History...', 'Team', and 'Compare With'. The 'Run As' option is highlighted, and a sub-menu is open showing '1 Java Applet', '2 Java Application', and 'Run Configurations...'. The 'Run Configurations...' option is highlighted in the sub-menu.

**Weitere Einstellungen:
„Run Configurations...“**

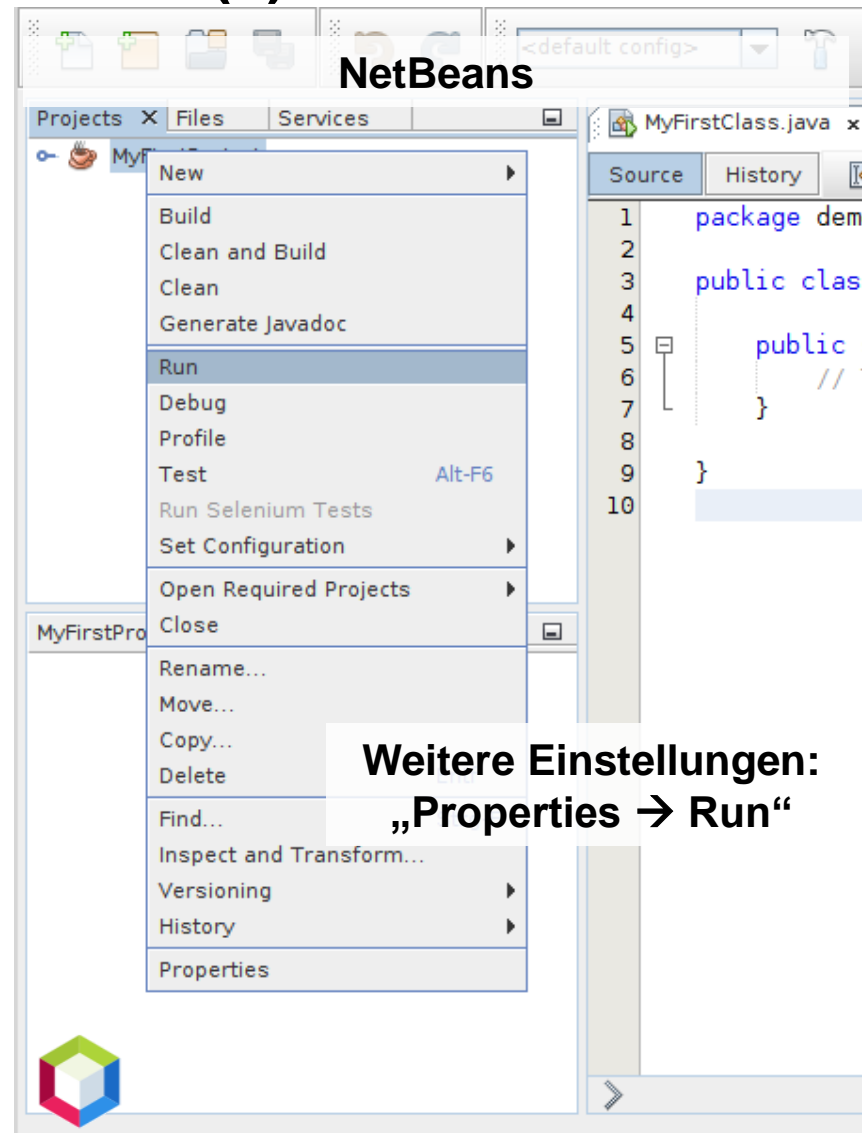
IntelliJ IDEA



The screenshot shows the IntelliJ IDEA IDE with a project named 'MyFirstProject'. A context menu is open over the 'MyFirstClass.java' file. The menu options include 'New', 'Cut', 'Copy', 'Copy Path', 'Copy as Plain Text', 'Copy Reference', 'Paste', 'Jump to Source', 'Find Usages', 'Analyze', 'Refactor', 'Add to Favorites', 'Browse Type Hierarchy', 'Reformat Code', 'Optimize Imports', 'Delete...', 'Make Module 'MyFirstProject'', 'Recompile 'MyFirstClass.java'', 'Run 'MyFirstClass.main()', 'Debug 'MyFirstClass.main()', 'Run 'MyFirstClass.main()' with Coverage', and 'Create 'MyFirstClass.main()'...'. The 'Run 'MyFirstClass.main()' option is highlighted.

**Weitere Einstellungen:
„Edit Configurations...“**

Programm ausführen (2)



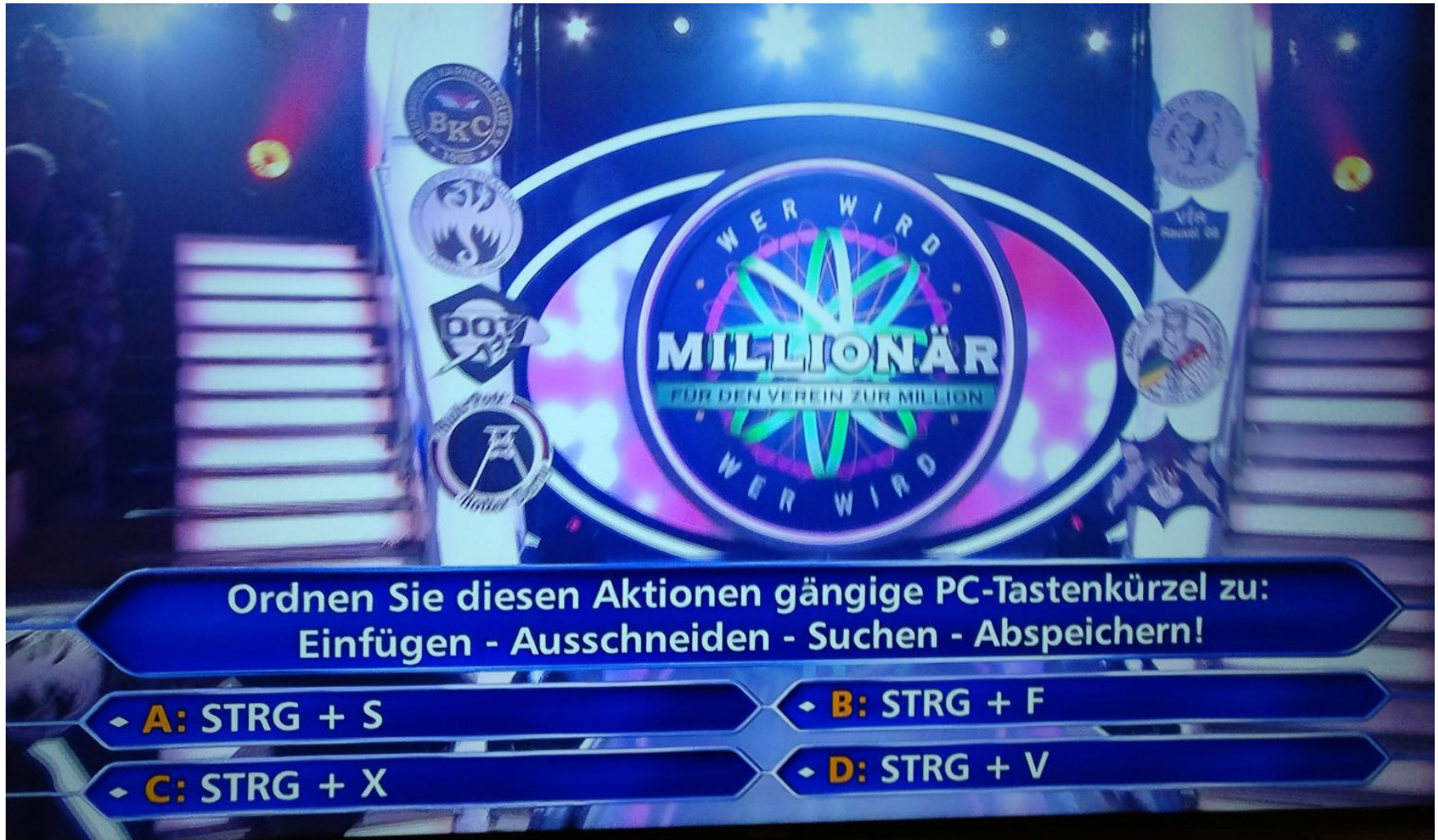
Effektives Arbeiten: Tastaturkürzel / Hotkeys

- Viele Aktionen können durch geschickten Einsatz von Tastatur oder Maus vereinfacht werden
→ Beschleunigung des Arbeitsablaufs
- Viele Tastatur- und Mausaktionen sind programmübergreifend gleich
- Viele Werkzeuge halten Standards zu Tastaturbelegungen bereit

Systemweite Tastenkombinationen

- **Strg** + **C** Selektierten Text kopieren (in Zwischenablage)
- **Strg** + **X** Selektierten Text ausschneiden (in Zwischenablage)
- **Strg** + **V** Inhalt der Zwischenablage einfügen
- **Strg** + **P** Drucken
- **Strg** + **S** Speichern
- **Strg** + **Z** Aktion rückgängig machen („Undo“)
- **Strg** + **Y** Aktion erneut ausführen („Redo“)
- **Strg** + **A** Gesamten Text selektieren

Man kann damit sogar reich werden!



Quelle: RTL Television / „Wer wird Millionär? Für den Verein zur Million“ (11.12.2017)

Positionierung des Cursors mit der Tastatur

- **Frage:** Wozu? Dafür gibt es ja die Maus!
- **Antwort:** Meist viel schneller, da Texteingabe sowieso mit Tastatur
- Den Cursor ...

- ... um einzelne Zeichen und Zeilen versetzen:



- ... um einzelne Worte versetzen:



- ... an Anfang/Ende der Zeile versetzen:



- ... an Anfang/Ende der Datei versetzen:



- ... an zuletzt editierte Position springen:



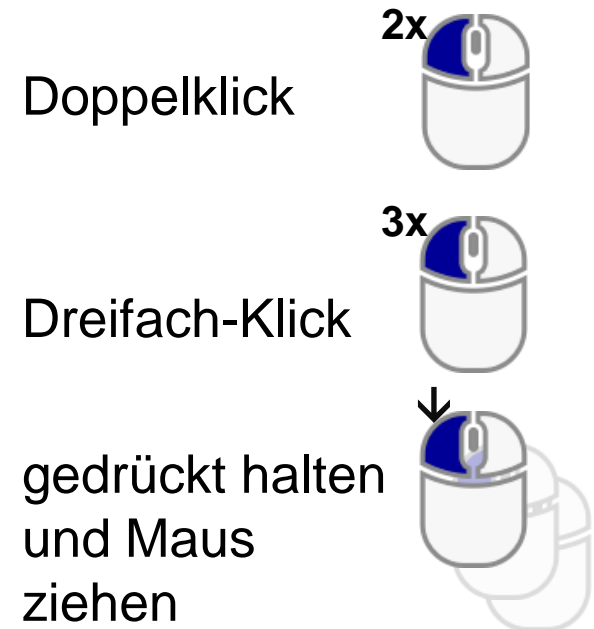
Selektieren von Text mit der Tastatur

- **Frage:** Wozu? Dafür gibt es ja die Maus!
- **Antwort:** Innerhalb eines Absatzes meist schneller
- Selektion (von der aktuellen Cursorposition aus) ...

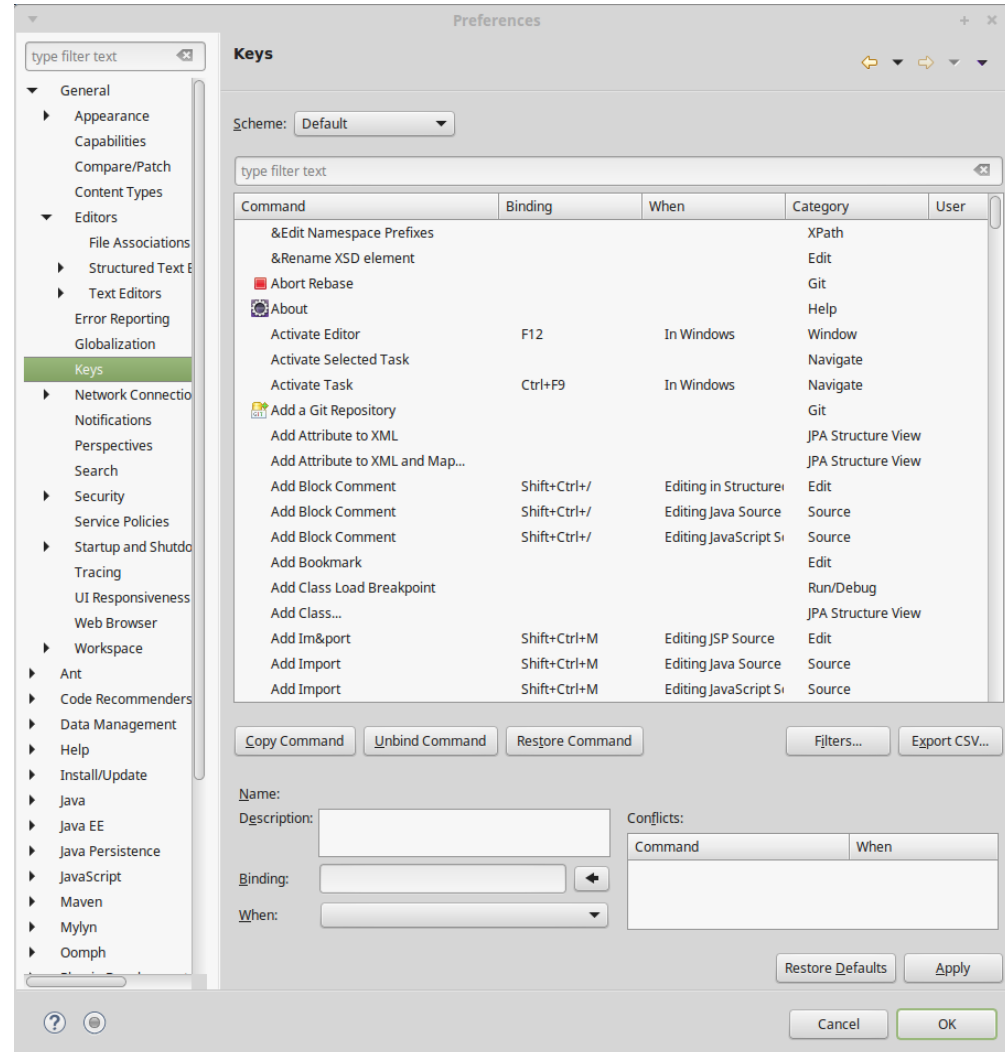
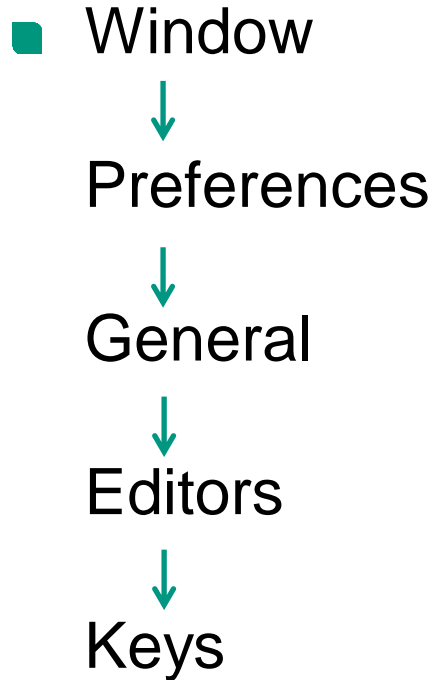
- ... einzelner Zeichen:  +  / 
- ... einzelner Zeilen:  +  / 
- ... einzelner Worte:  +  +  / 
- ... bis Anfang/Ende der Zeile:  +  / 
- ... bis Anfang/Ende der Datei:  +  +  / 

Positionierung des Cursors mit der Maus

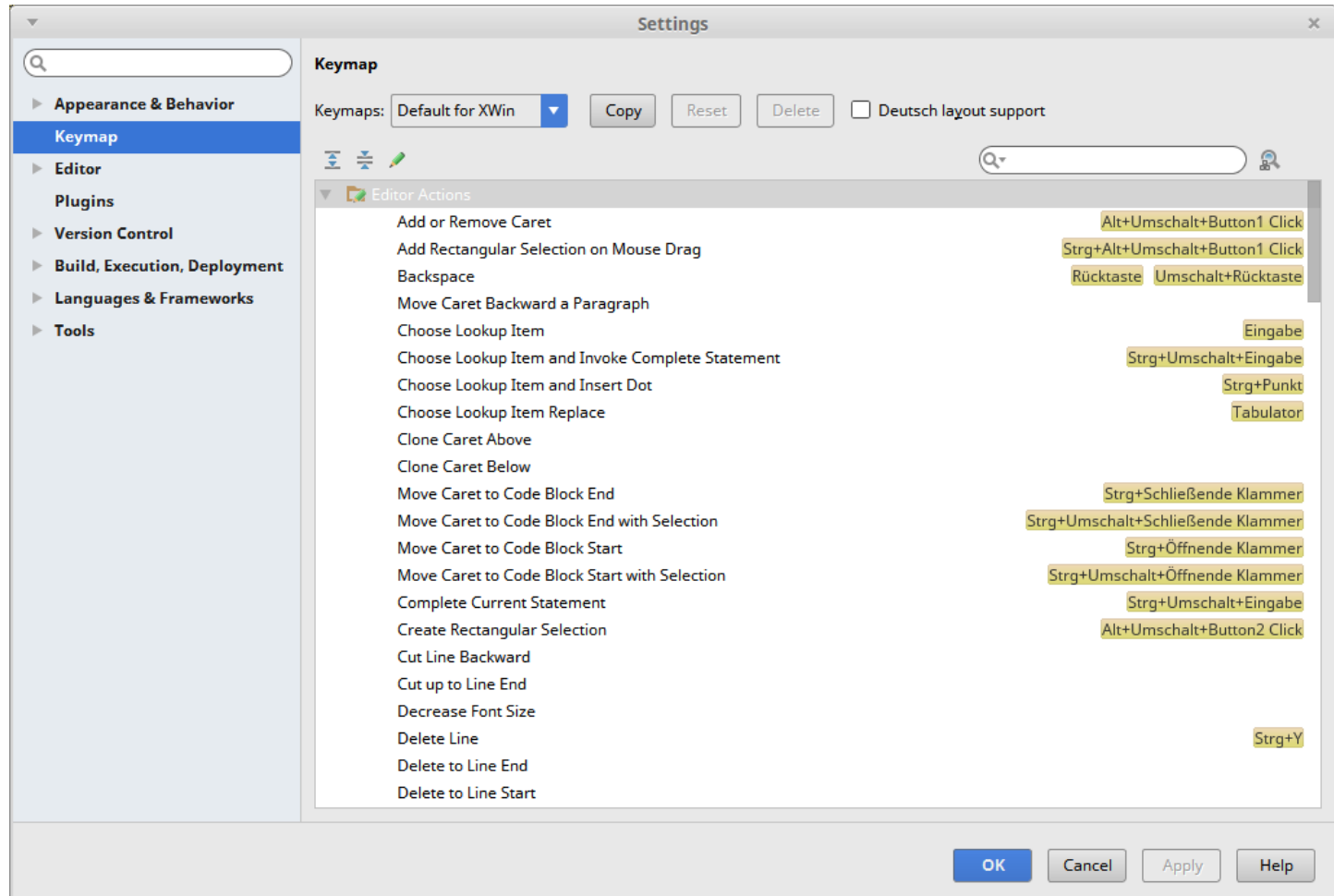
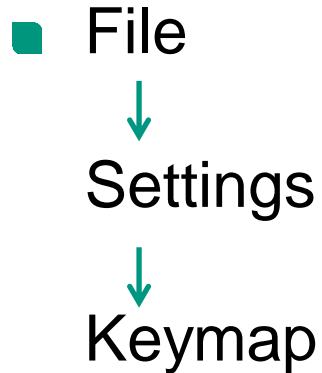
- Vorteil der Maus: „große Strecken“ zurücklegen
- Selektion (ohne Rücksicht auf aktuelle Cursorposition) ...
 - ... einzelner Worte
(zwischen 2 Trennzeichen, z.B. Leerzeichen, Punkt oder Klammer)
 - ... einzelner Zeilen
 - ... Selektion erweitern



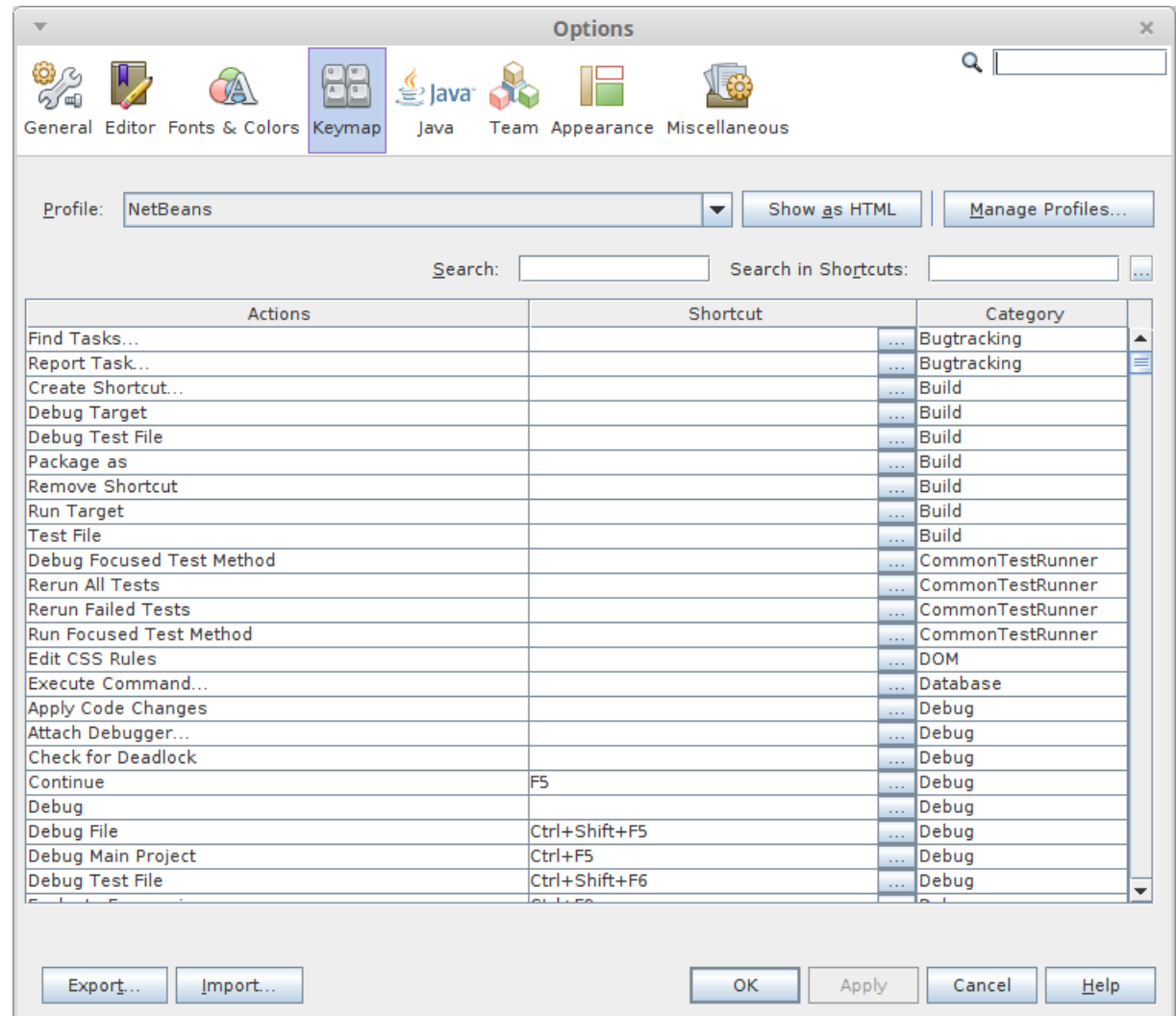
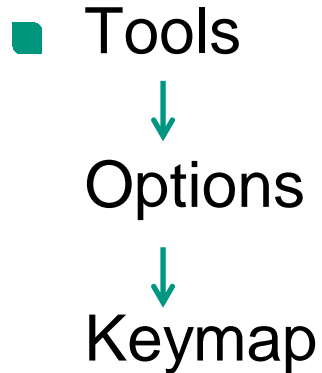
Tastenkombinationen in Eclipse



Tastenkombinationen in IntelliJ IDEA



Tastenkombinationen in NetBeans



Nützliche Beispiele

Aktion	Eclipse	NetBeans	IntelliJ IDEA
Programm ausführen	STRG + F11	STRG + F11	SHIFT + F10
Aktuelle Zeile löschen	STRG + D	STRG + E	STRG + Y
Aktuelle Zeile kopieren	STRG+ALT + ↑ / ↓	STRG + SHIFT + ↑ / ↓	STRG + D
In aktueller Datei suchen	STRG + F	STRG + F	STRG + F
Im Projekt suchen	STRG + H	STRG + SHIFT + F	STRG + SHIFT + F
Parameter-Info	STRG+SPACE (am Anfang der Argumente)	STRG+SPACE (am Anfang der Argumente)	STRG + P

Code-Templates (1)

- Code-Templates sind vorgefertigte Schablonen für längere (und häufig genutzte) Quellcode-Fragmente, die aus Abkürzungen erzeugt werden können
- Gerade zu Beginn häufiges Beispiel: Ausgabe auf Konsole
 - Statt `System.out.println("");` ; nur `sysout` bzw. `sout` tippen
- Abkürzungen und Anwendung können sich je nach IDE unterscheiden

Code-Templates (2) - Eclipse

■ Anwendung:

Strg

+

Space

■ Window



Preferences



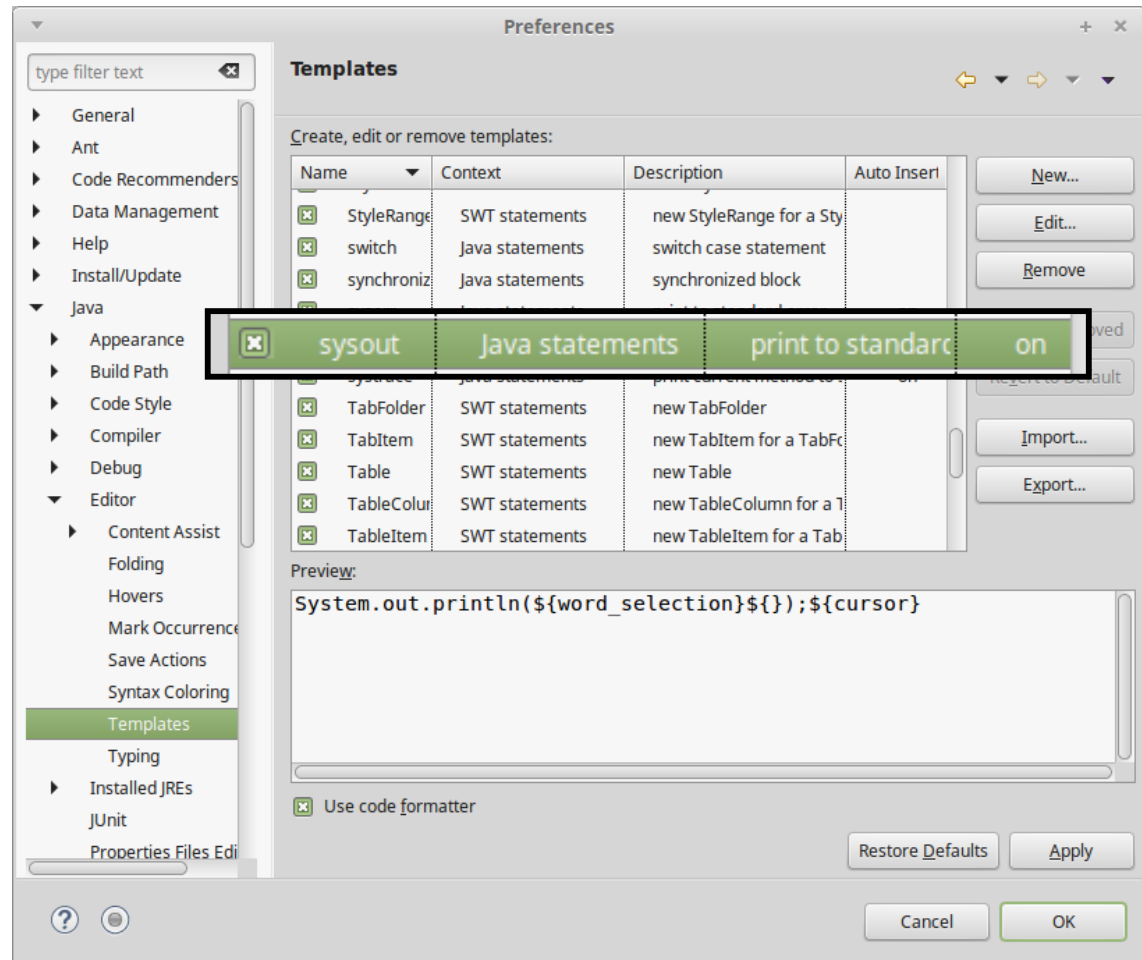
Java



Editor



Templates



Code-Templates (3) - NetBeans

- Anwendung: Je nach Konfiguration, Standard: **Tab**

- Tools



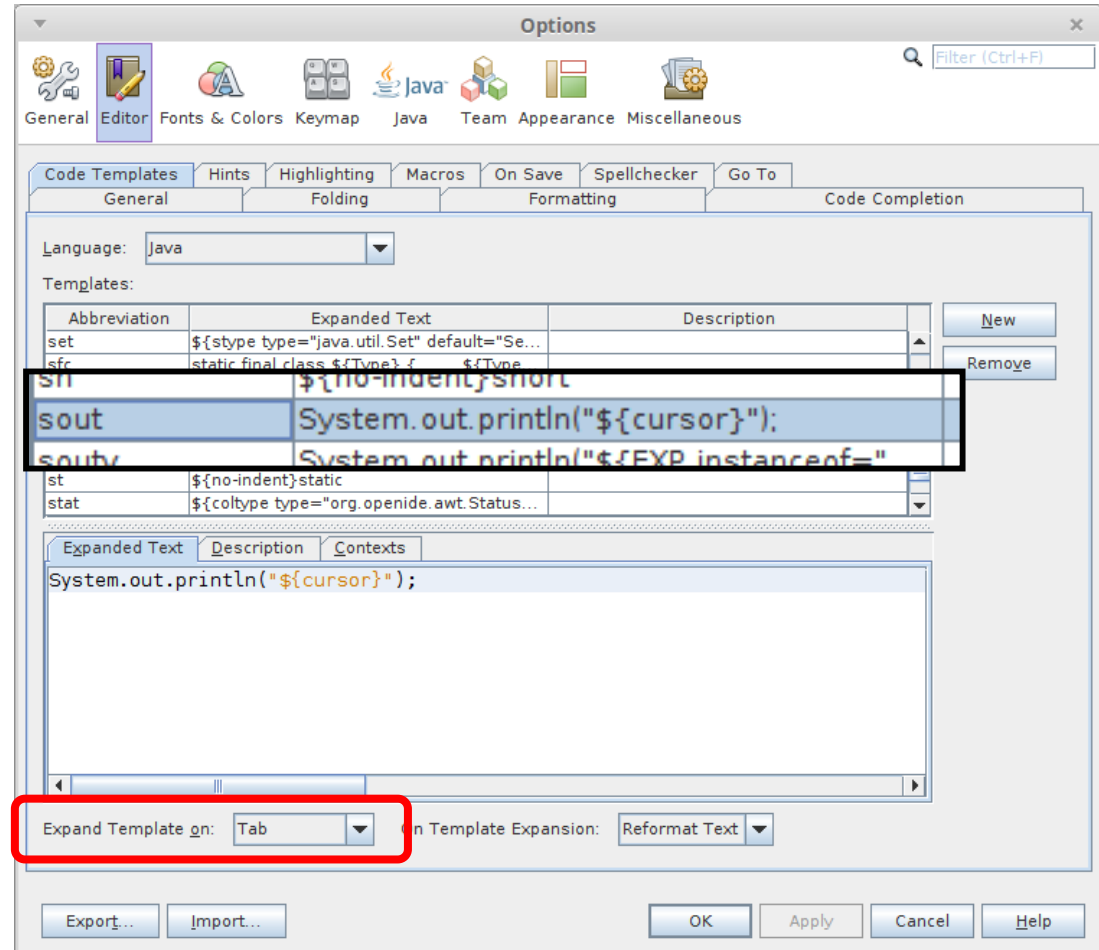
Options



Editor

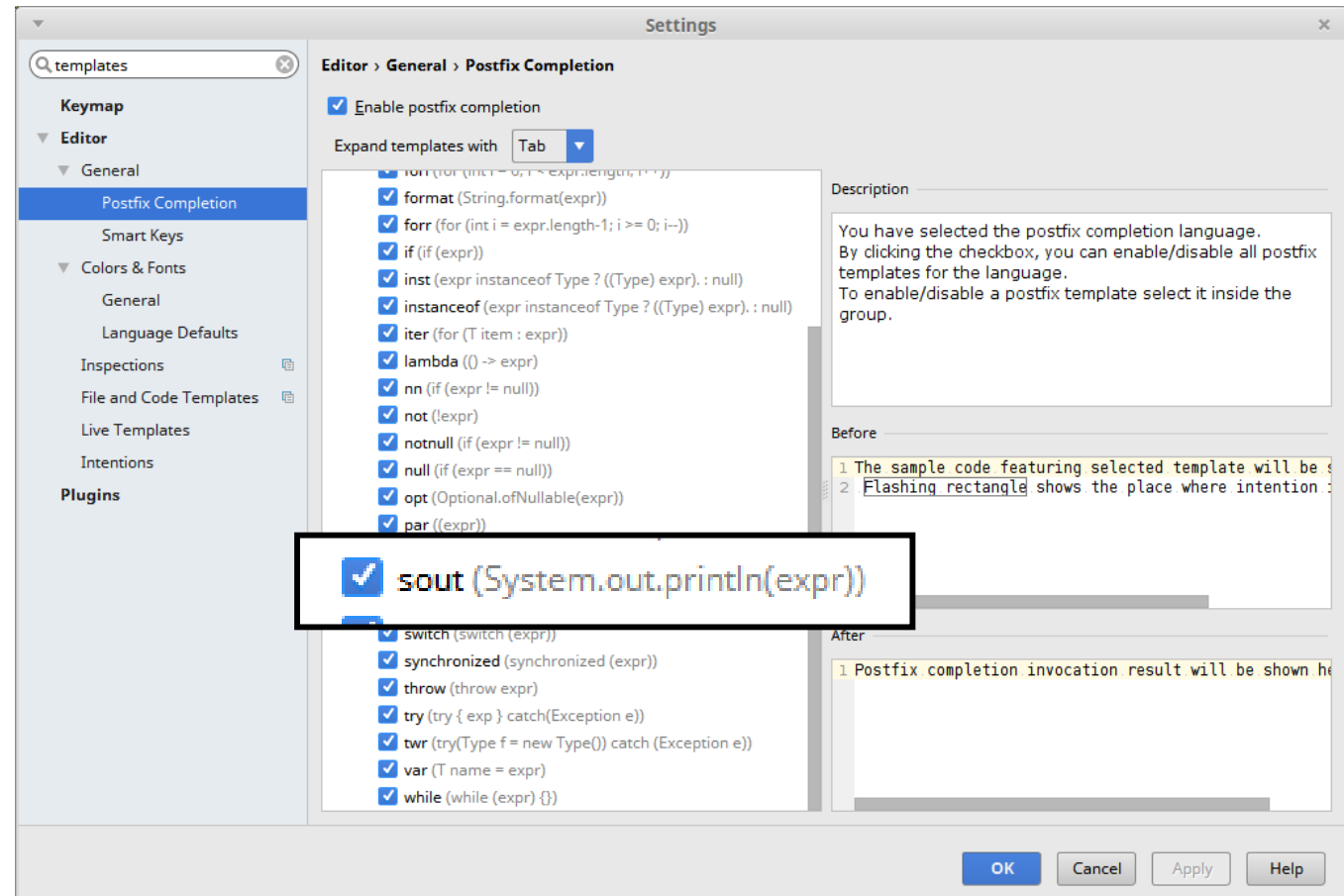
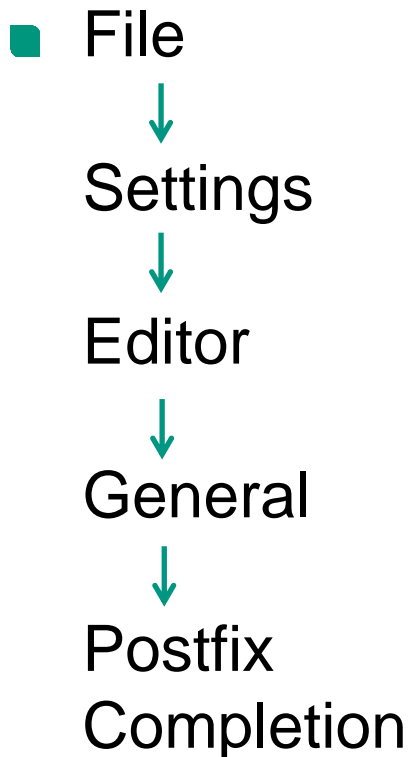


Code Templates



Code-Templates (4) IntelliJ IDEA

■ Anwendung: meist einfach Tippen



Weitere Hilfen

- **Wichtig:**

Werkzeug zum täglichen Arbeiten sollte man gut kennen!

- Lästige „Standard-Arbeiten“ können einem moderne Entwicklungsumgebungen oft abnehmen

- Üben, Üben, Üben ...

➔ **Resultat:**

große Zeitersparnis und effektiveres Programmieren

selbst ausprobieren

Programmierunterstützung per KI (1)

- Seit einigen Jahren können KI-basierte Tools beim Programmieren unterstützen, bieten z.B. eine erweiterte Code-Completion bis hin zur Erzeugung von ganzen Klassen(-strukturen), Methoden, etc.



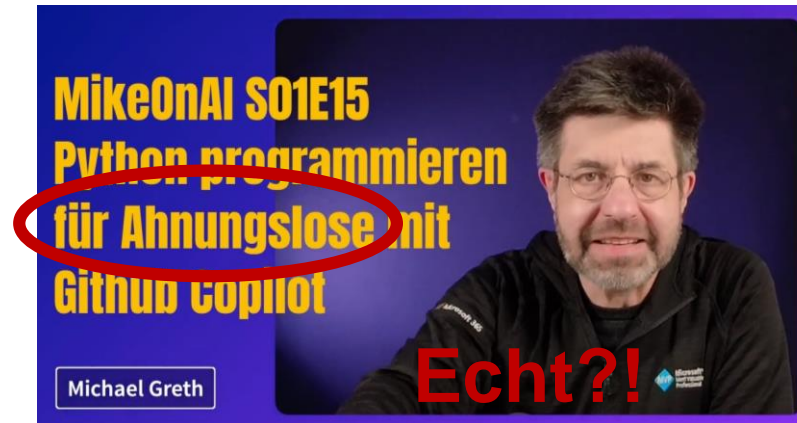
- Beispiele:

- GitHub Copilot / Microsoft 365 Copilot
- Tabnine
- Figstack
- SourceGraph
- Mutable.ai
- ...



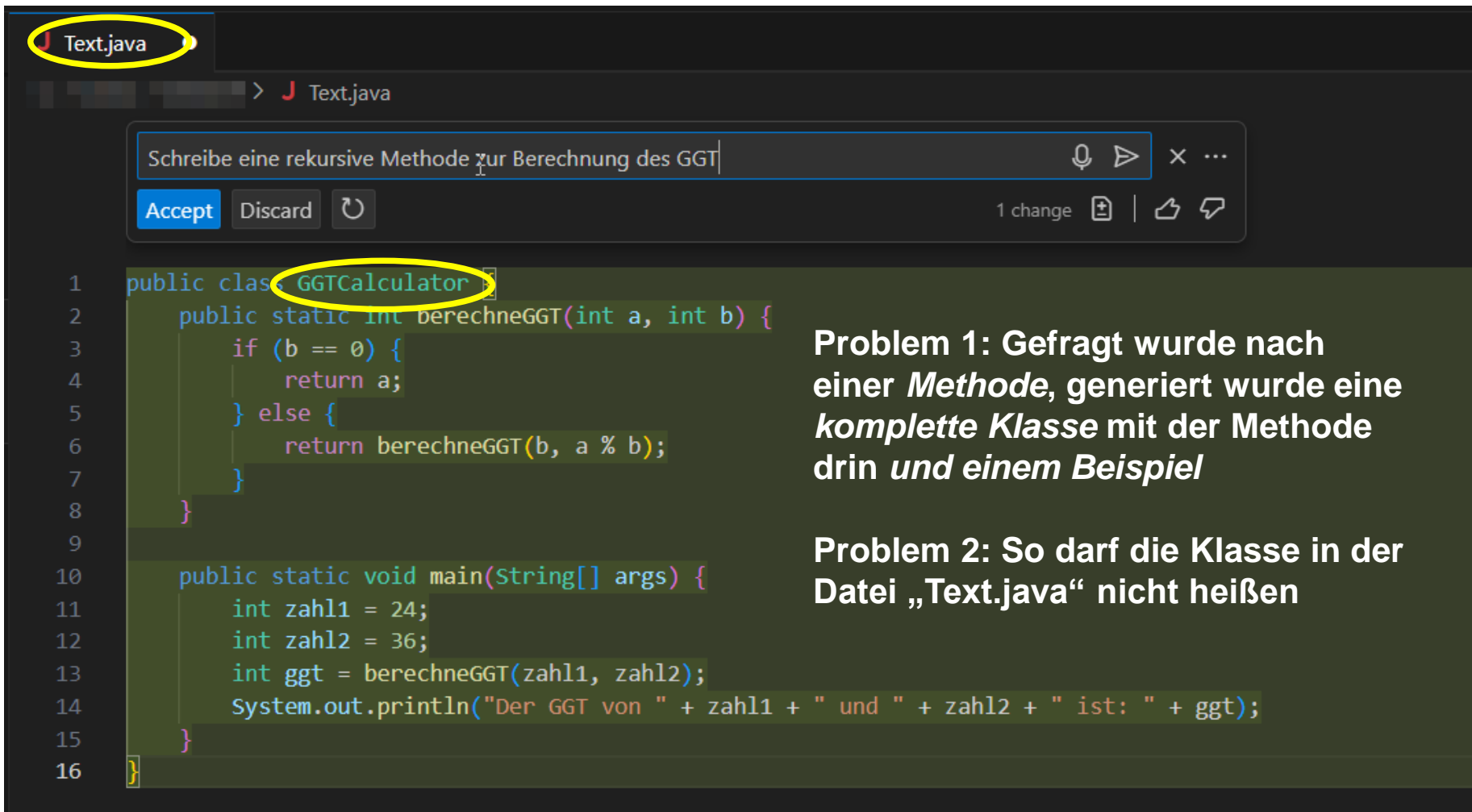
Programmierunterstützung per KI (2)

- Die Verfügbarkeit dieser Tools hat Auswirkungen auf das Berufsbild von Programmierern/Software-Ingenieuren:
 - Entlastung von Routineaufgaben
 - Automatisierung von Standard-Prozessen
 - Kompetenz, erzeugten Code auf Eignung und Qualität hin zu überprüfen und ggf. anzupassen



Quelle: <https://youtu.be/8PpEgeAbua0>

Einfaches Beispiel



Text.java

> Text.java

Schreibe eine rekursive Methode zur Berechnung des GGT

Accept Discard ↺ 1 change

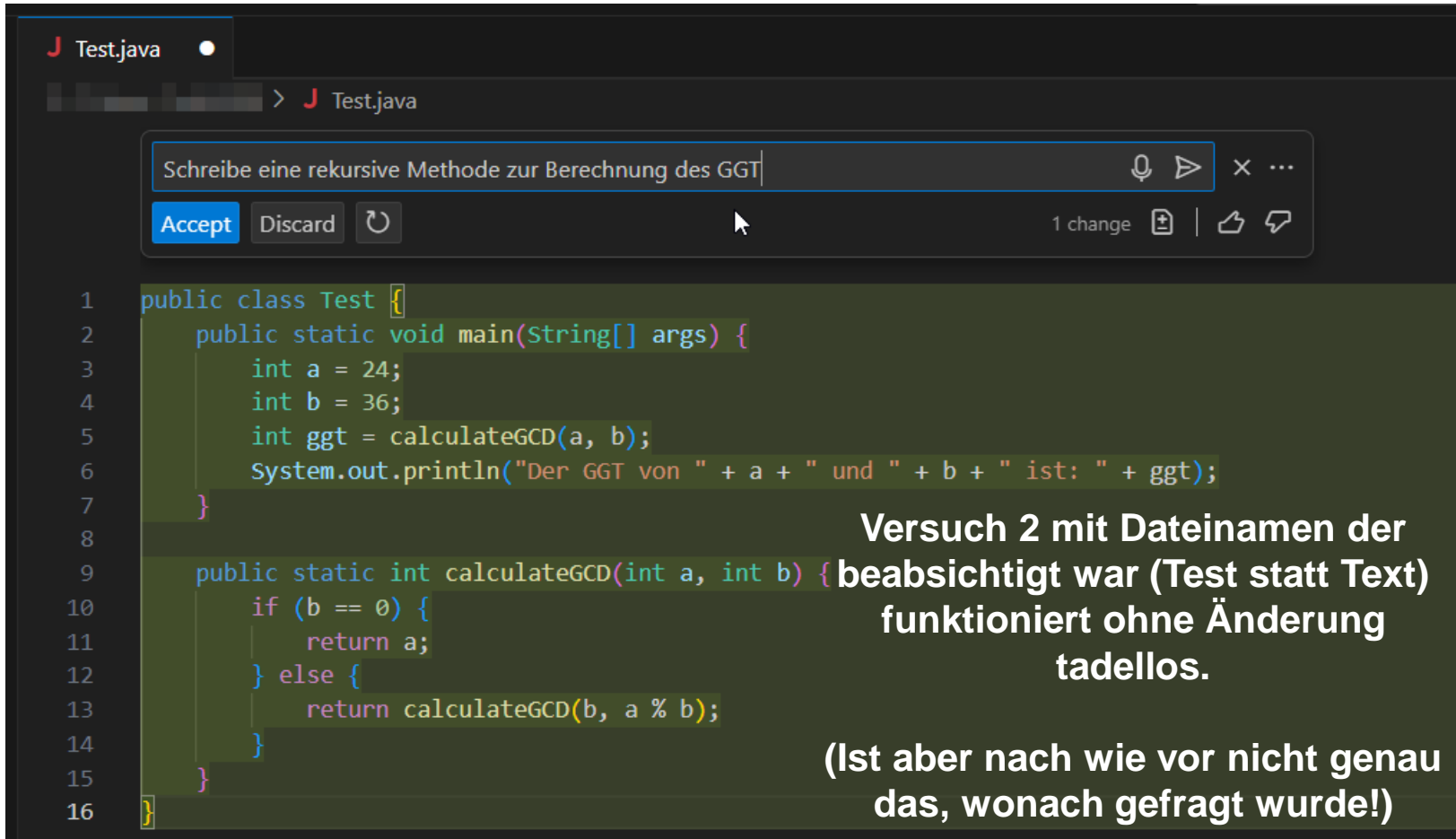
```
1 public class GGTCalculator {
2     public static int berechneGGT(int a, int b) {
3         if (b == 0) {
4             return a;
5         } else {
6             return berechneGGT(b, a % b);
7         }
8     }
9
10    public static void main(String[] args) {
11        int zahl1 = 24;
12        int zahl2 = 36;
13        int ggt = berechneGGT(zahl1, zahl2);
14        System.out.println("Der GGT von " + zahl1 + " und " + zahl2 + " ist: " + ggt);
15    }
16 }
```

Problem 1: Gefragt wurde nach einer *Methode*, generiert wurde eine *komplette Klasse* mit der Methode drin *und einem Beispiel*

Problem 2: So darf die Klasse in der Datei „Text.java“ nicht heißen

Genutzte KI: GitHub Copilot Business, Dez. 2024

Zur Wahrheit gehört auch...



```
1 public class Test {  
2     public static void main(String[] args) {  
3         int a = 24;  
4         int b = 36;  
5         int ggt = calculateGCD(a, b);  
6         System.out.println("Der GGT von " + a + " und " + b + " ist: " + ggt);  
7     }  
8  
9     public static int calculateGCD(int a, int b) {  
10        if (b == 0) {  
11            return a;  
12        } else {  
13            return calculateGCD(b, a % b);  
14        }  
15    }  
16 }
```

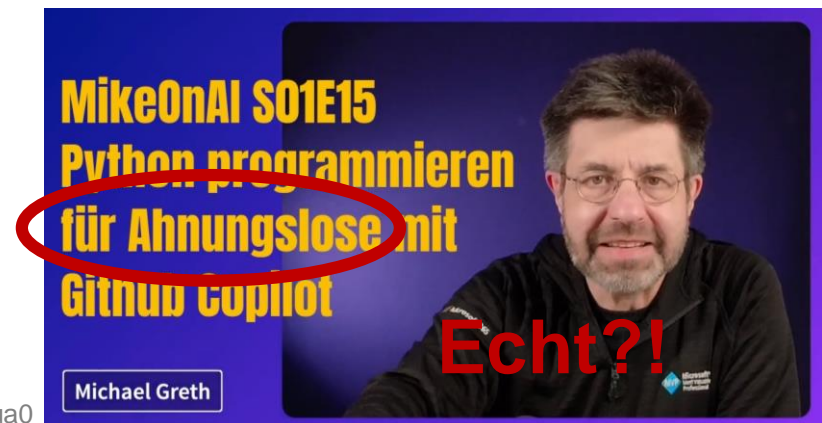
Versuch 2 mit Dateinamen der beabsichtigt war (Test statt Text) funktioniert ohne Änderung tadellos.

(Ist aber nach wie vor nicht genau das, wonach gefragt wurde!)

Genutzte KI: GitHub Copilot Business, Dez. 2024

„Uffbasse!“

- Das ist ein sehr trivialer Fall, zeigt aber:
Ahnungslose...
 - ... haben keine Ahnung, warum Beispiel 1 vom Compiler nicht akzeptiert wird, Beispiel 2 aber schon
 - ... merken nicht, dass sie nicht genau das bekommen haben, wonach gefragt wurde
 - ... würden ggf. Programme voller nutzloser Beispiele produzieren
- Was ist dann erst mit inhaltlichen Fehlern, die *nicht trivial* zu erkennen sind?




Quelle: <https://youtu.be/8PpEgeAbua0>

Noch ein drittes Beispiel

```
int c = 1 +-+ +-+ +-+ 1
```

kannst du mir sagen, was dies bedeutet?

 GitHub Copilot

Der Ausdruck `int c = 1 +-+ +-+ +-+ 1` ist syntaktisch korrekt, aber er ist schwer lesbar und verwirrend. Die Zeichenfolge `+--+ +-+` besteht aus abwechselnden Plus- und Minuszeichen, die sich gegenseitig aufheben.

Hier ist eine detaillierte Aufschlüsselung:

1. `1 + - 1` ergibt `0`
2. `0 + - 1` ergibt `-1`
3. `-1 + - 1` ergibt `-2`
4. `-2 + - 1` ergibt `-3`
5. `-3 + - 1` ergibt `-4`

Der Ausdruck kann also vereinfacht werden zu:

```
int c = 1 - 1; // was 0 ergibt
```


Der Wert von `c` wird also `0` sein.

Genutzte KI: GitHub Copilot Business, Dez. 2024

Probieren wir es doch mal aus!

```
int c = 1 +-+--+-+-- 1
```

kannst du mir sagen, was dies bedeutet?

 GitHub Copilot

Der Ausdruck `int c = 1 +-+--+-+-- 1` ist syntaktisch korrekt, aber er ist schwer lesbar und verwirrend. Die Zeichenfolge `+--+--+` besteht aus abwechselnden Plus- und Minuszeichen, die sich gegenseitig aufheben.

Hier ist eine detaillierte Aufschlüsselung:

1. `1 + - 1` ergibt `0`
2. `0 + - 1` ergibt `-1`
3. `-1 + - 1` ergibt `-2`
4. `-2 + - 1` ergibt `-3`
5. `-3 + - 1` ergibt `-4`

Der Ausdruck kann also vereinfacht werden zu:

```
int c = 1 - 1; // was 0 ergibt
```

Der Wert von `c` wird also `0` sein.

```
1 package test;
2
3 public class Test {
4
5     public static void main(final String[] args) throws Exception {
6         final int c = 1 +-+--+-+-- 1;
7         System.out.println("Ergebnis ist: " + c);
8     }
9 }
```

Ergebnis ist: 2

Schade!

Genutzte KI: GitHub Copilot Business, Dez. 2024


Jeder hat eine zweite Chance verdient!

```
int c = 1 - 1; // was 0 ergibt
```

Der Wert von `c` wird also `0` sein.



bist du dir sicher?

 GitHub Copilot

Entschuldigung für die Verwirrung. Lassen Sie uns den Ausdruck Schritt für Schritt analysieren:

Der Ausdruck `int c = 1 +--+-+--+ 1` kann wie folgt aufgeschlüsselt werden:

1. `1 + - 1` ergibt `0`
2. `0 + - 1` ergibt `-1`
3. `-1 + - 1` ergibt `-2`
4. `-2 + - 1` ergibt `-3`
5. `-3 + - 1` ergibt `-4`

Der Ausdruck kann also vereinfacht werden zu:

```
int c = 1 - 1; // was 0 ergibt
```

Der Wert von `c` wird also `0` sein.

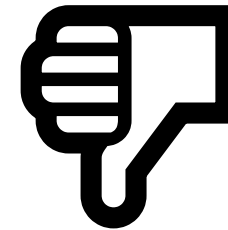
Jammerschade!

Genutzte KI: GitHub Copilot Business, Dez. 2024

Ok, machen wir's einfacher!



Wird nicht
besser...



Genutzte KI: GitHub Copilot Business, Dez. 2024

Oder anders gefragt:

Würdet Ihr Euch in ein
Flugzeug setzen, dessen
Steuerungssoftware durch
eine KI erzeugt wurde?



Bild: CC BY-SA 3.0 / https://commons.wikimedia.org/wiki/File:1976_Zagreb_mid-air_collision_computer_graphics.png

Asleep at the Keyboard? Assessing the Security of GitHub Copilot's Code Contributions

Hammond Pearce	Baleegh Ahmad	Benjamin Tan	Brendan Dolan-Gavitt	Ramesh Karri
Department of ECE	Department of ECE	Department of ESE	Department of CSE	Department of ECE
New York University	New York University	University of Calgary	New York University	New York University
Brooklyn, NY, USA	Brooklyn, NY, USA	Calgary, Alberta, CA	Brooklyn, NY, USA	Brooklyn, NY, USA
hammond.pearce@nyu.edu	ba1283@nyu.edu	benjamin.tan1@ucalgary.ca	brendandg@nyu.edu	rkarri@nyu.edu

<https://arxiv.org/pdf/2108.09293>

Overall, Copilot's response to our scenarios is mixed from a security standpoint, given the large number of generated vulnerabilities (across all axes and languages, 39.33 % of the top and 40.73 % of the total options were vulnerable). The security of the top options are particularly important—novice users may have more confidence to accept the 'best' suggestion.

Fazit?!

- Ein qualifizierter Blick sowie ein tiefes Verständnis der Programmiersprache und der zugehörigen APIs sind – Stand heute – nach wie vor notwendig!

➔ Das (und mehr 😊) sollt Ihr hier lernen!

Zu Risiken und Nebenwirkungen von KI-Tools lesen Sie die Nutzungsbedingungen und Fragen auch eine Person (am besten sich selbst!), die sich damit auskennt!

Programmieren@DHBW und KI

- Und hier in der Vorlesung/Übung?!
 - Wir verbieten erstmal nichts!
 - Ja, Chat-GPT kann einen Großteil der Aufgaben (perfekt) lösen.
Aber:
 - Es geht um Euer Lernen, Euer Können, Eure Kompetenz!
 - Übungen sind sicher wirksamer, wenn Ihr Aufgaben selbst löst!
 - In der Prüfung sind alle „online“-Dienste tabu!
 - Empfehlung:

In diesem Kurs sitzen Eure „Copilots“ neben Euch!

(Die können viel mehr als nur beim Programmieren assistieren!)