Prof. Dr. Sven Strickroth

# Einführung in die Programmierung

Wintersemester 2025/26

## Debugging

# Fehlersuche aka. „Debugging“

- Debugging bezeichnet die Tätigkeit, Fehler zu diagnostizieren und aufzufinden, sei es unter Verwendung eines Debuggers oder anderer Methoden.
- Ein Debugger ist ein Werkzeug zum Diagnostizieren und Auffinden von Fehlern in Programmen.
  - ist in der Regel in IDEs, wie Eclipse, integriert

# Debugging mit Eclipse (1)

- Beispiel: Suchen nach der Ursache eines Fehlers, z.B. `ArrayIndexOutOfBoundsException`

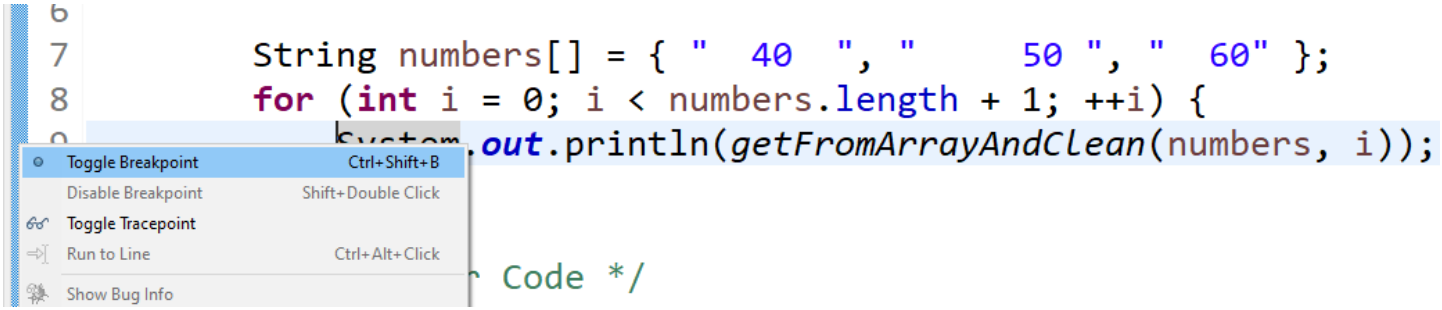
```
String numbers[] = { "  40  ", "     50 ", "  60" };
for (int i = 0; i < numbers.length + 1; ++i) {
    System.out.println(getFromArrayAndClean(numbers, i));
}
```

```
40
50
60
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 3 out of bounds for length 3
        at DebuggingExample.getFromArrayAndClean(DebuggingExample.java:17)
        at DebuggingExample.main(DebuggingExample.java:9)
```

- Der Fehler ist nicht offensichtlich, weil er z. B. in einer anderen Methode auftritt, wie können wir diesen Fehler mit dem Debugger identifizieren?

# Debugging mit Eclipse (2)

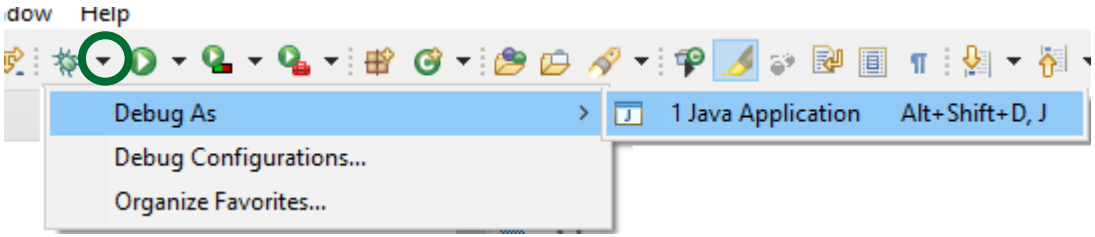
1. Setzen eines Breakpoints
   - Rechtsklick auf den linken Rand des Editors in der Zeile, wo der Breakpoint gesetzt werden soll
   - Auswahl „Toggle Breakpoint“




→ blauer Punkt erscheint links von der Zeile

# Debugging mit Eclipse (3)

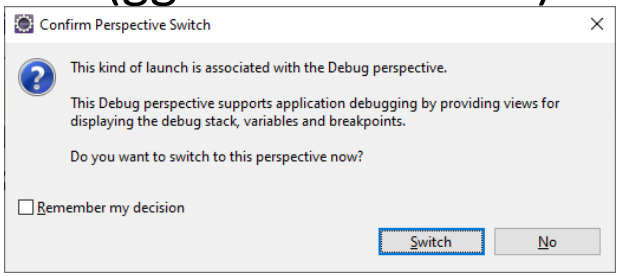
2. Starten des Programms mit dem Debugger



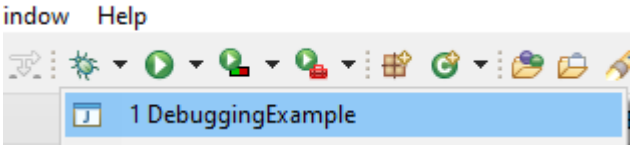

- evtl. erscheint ein Dialog, der fragt, ob zur Debugging-Perspective gewechselt werden soll: „Switch“ anklicken (ggf. Haken setzen)

→ Eclipse wechselt in die Debug-Ansicht

(zurückwechseln über oder Window, Perspective -> Open Perspective -> Java

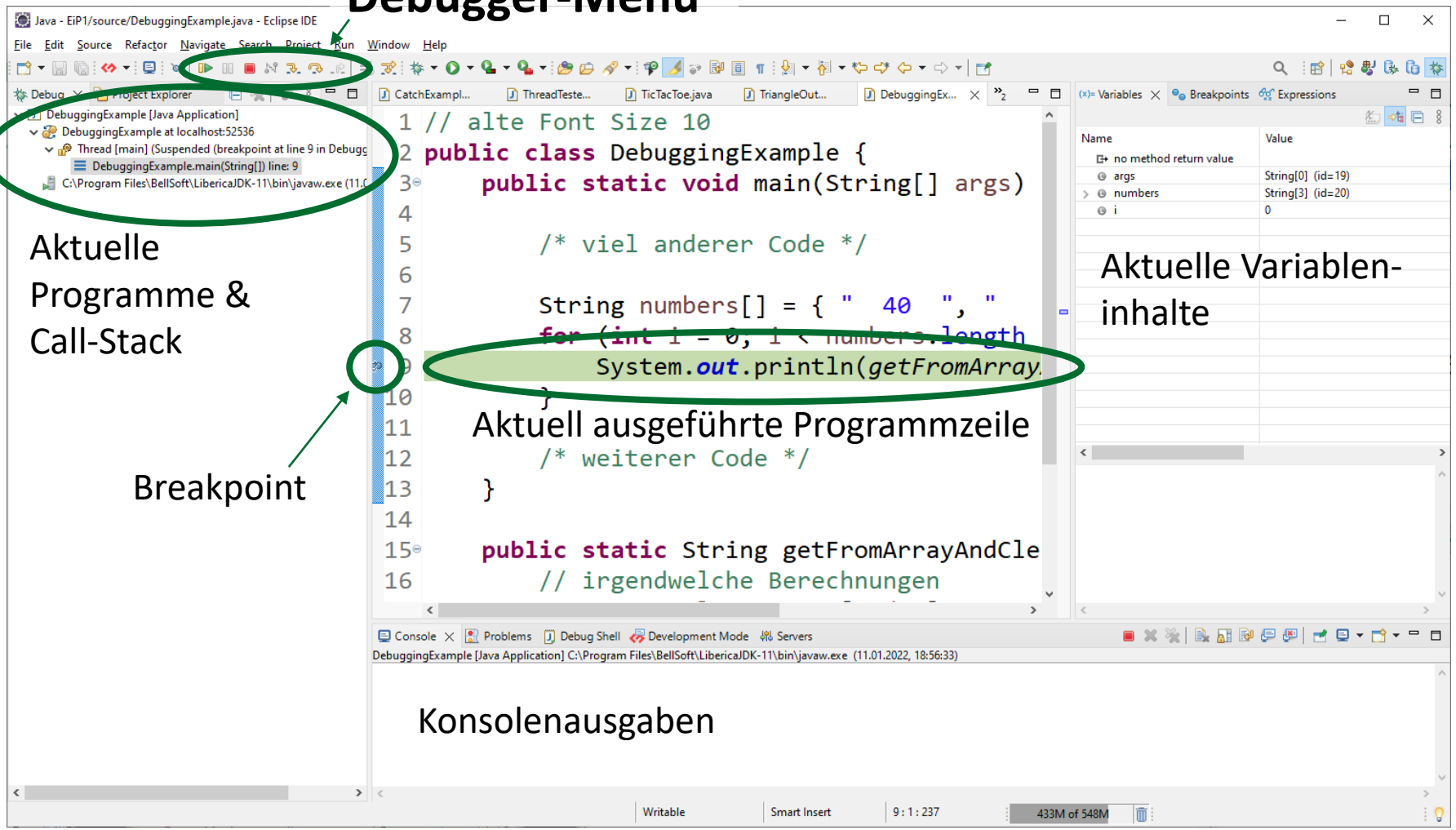



- Ab dem zweiten Mal kann direkt das Programm über den Klassennamen gewählt werden:



# Debugging mit Eclipse (4)

- Programm wird bis zum Breakpoint ausgeführt und hält an



# Debugging mit Eclipse (5)

- Das Debugger-Menü:


  - 1: Programm fortfahren (bis zum nächsten Breakpoint oder Ende)
  - 2: Programmausführung (gewaltvoll) beenden
  - 3: In die Abarbeitung einer Anweisung hineinspringen (z. B. in eine aufzurufende Methode hineingehen)
  - 4: Die aktuelle Anweisung komplett ausführen (z. B. Methodenaufruf wird als ein Schritt betrachtet)
  - 5: Wenn wir in einer Methode sind, diese bis zum Beenden ausführen und wieder anhalten

  - Durch klicken auf 3 und 4 kann man das Programm schrittweise ausführen.
    - Dabei erkennt man in jeder Iteration die Variablenbelegungen für *i* und kann so dem Fehler auf die Spur kommen

# Prof. Dr. Sven Strickroth

Ludwig-Maximilians-Universität München
Institut für Informatik
Lehr- und Forschungseinheit PLAI
Oettingenstraße 67
80538 München

Telefon: +49-89-2180-9300
sven.strickroth@ifi.lmu.de