Programmiervorkurs

Tag 3

Carina Kühnert

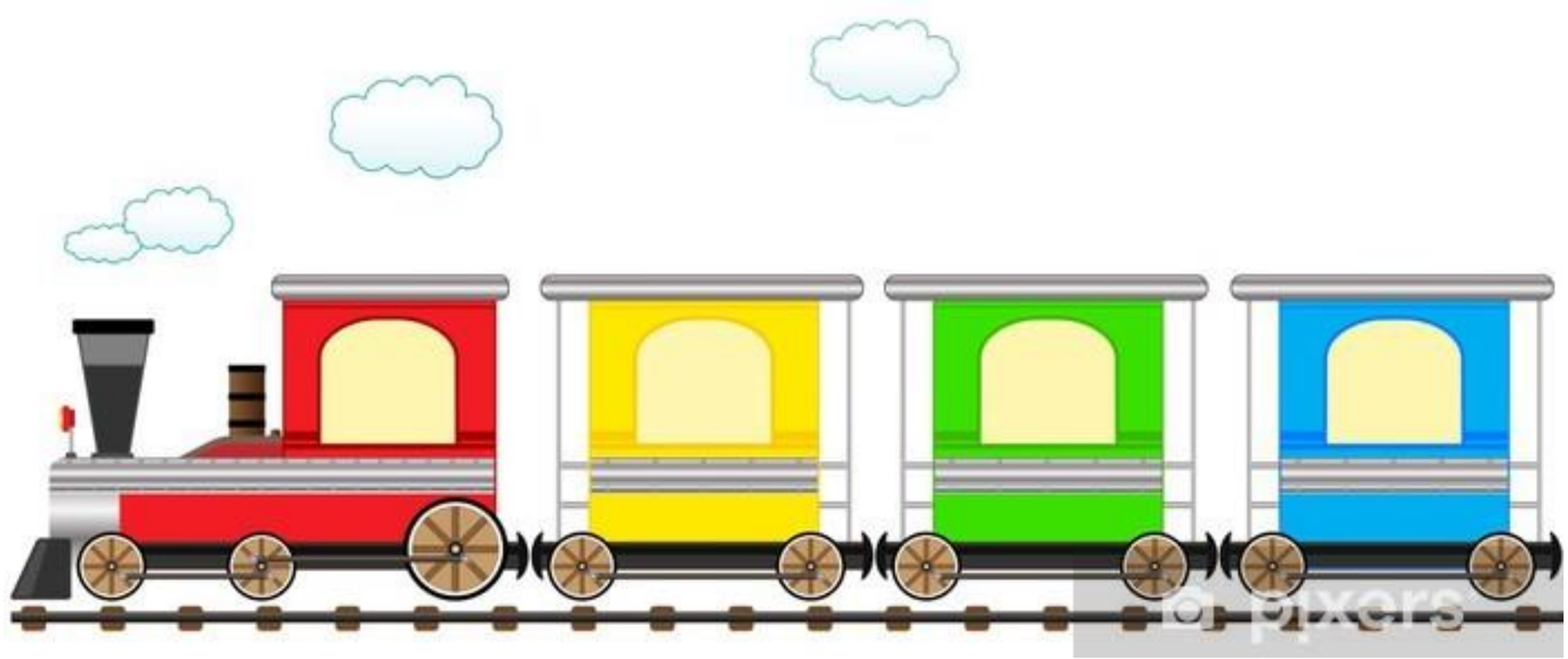
Ablauf

09:30	Tag 2 Übungsbesprechung
10:00	Vorlesung
12:00	60 Minuten Mittagspause
13:00	Übungen im LI 137

Inhaltsübersicht Vorkurs

Tag 1	Zustände, Variablen, Datentypen, Konvertierungen, Arithmetik, Eclipse Live-Demo
Tag 2	Kommentare, Boolesche Ausdrücke, If-Abfragen, Switch-Case
Tag 3	**Arrays, (Do-)While-Schleife, For-Schleifen, Debugging**
Tag 4	Methoden, Klassen, JavaDoc

Arrays



Arrays

- **Ein Array fasst mehrere Variablen des gleichen Typs zusammen.**

  Ein Array von Integern enthält Ganzzahlen:

  { 4, 8, 15, 16, 23, 42 }

  Ein Array von Strings enthält Wörter:
  { "Hallo", "Array", "Strings", "Peter" }

- **Alle Werte müssen vom gleichen Typ sein.**

  Falsch: **{ 3, 18, 3.14, 'r ', "Hallo" }**

Arrays erstellen

- **Ein Array Typ *type* zu erstellen:**

  type[ ] arrayName;

- **Ein Array Typ *type* zu erstellen, mit der Größe *n*:**

  type[ ] arrayName = type[n];

  Das Array wird dann mit Standardwerten gefüllt (bei Zahlen mit 0)

Arrays erstellen

- **Ein Array mit Werten Initialisieren:**

  type[ ] arrayName = new type[ ] {w1, w2};

  kürzer:

  type[ ] arrayName = {w1, w2};

Die Größe eines Arrays kann nachträglich nicht geändert werden. Zum Vergrößern oder Verkleinern muss ein neues Array angelegt werden.

Alternativen kommen in der Vorlesung

Arrays erstellen

Beispiele in der IDE

Arrayzugriff

```
int[ ] arrayInt = new int[ ]{1, 2, 3};
```

i 0 1 2

- Arrayzugriff an der Stelle i:

  ```
  arrayInt[ i ];
  ```

- Größe eines Arrays

  ```
  arrayInt.length
  ```

Arrayzugriff

Beispiele in der IDE

Übung

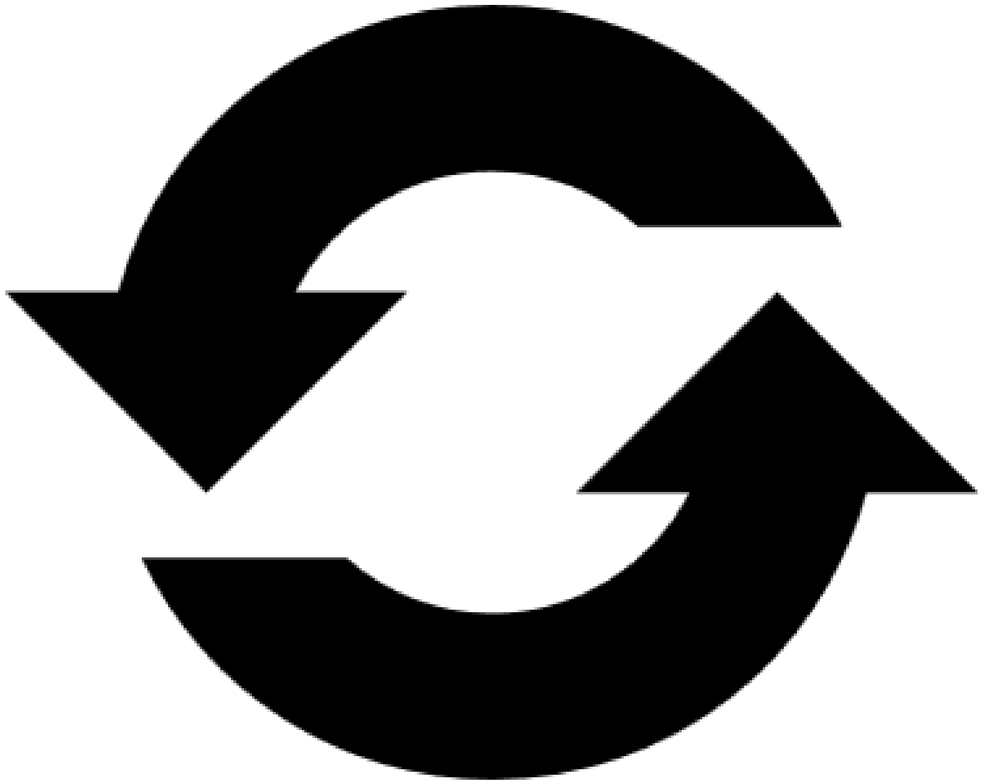
Aufgabe:
Was passiert, wenn man versucht, auf ein Feld außerhalb des Arrays zuzugreifen?

```
int[] arrayWithInt = {1, 2, 3, 4};
System.out.println(arrayWithInt[4]);
```

Arrays

Fragen?

Schleifen



While-Schleife

```
while (Bedingung) {
  Anweisung1;
  Anweisung2;
  // ...
}
```

While-Schleife

```
int counter = 0;
while (counter < 10) {
    System.out.println("Hello World");
    counter++;
}
```

Was macht die Schleife genau?

- ***counter*** **ist am Anfang auf 0**
- **Prüfung: ist *counter kleiner als 10*?**
  - **Ja: es wird „*Hello World*“ *ausgegeben* und *counter* wird um eins erhöht und wieder geprüft**
  - **Nein: die Schleife wird Verlassen**

While-Schleife

Beispiele in der IDE

Do-While-Schleife

```
do {
  Anweisung1;
  Anweisung2;
  // ...
} while (Bedingung);
```

Do-While-Schleife

Beispiele in der IDE

Übung

Aufgabe:
Wie oft werden diese Schleifen ausgeführt?

```
int counter = 10;
do {
    System.out.println(“Hello World”);
    counter ++;
} while (counter < 10);
```

```
int counter = 10;
while (counter < 10 ){
    System.out.println(“Hello World”);
    counter ++;
};
```

While vs Do-While

Endlosschleife

Wie oft wird diese Schleife ausgeführt?

```
Int counter = 0;

while (counter < 10) {
     System.print.out(“Hilfe”);
}
```

For-Schleife

```
for (Initialisierung; Bedingung; Schritt) {
  Anweisung1;
  Anweisung2;
  // ...
}
```

For-Schleife

```
for (Initialisierung; Bedingung; Schritt) {
    Anweisung1;
    Anweisung2;
    // ...
}
```

Was macht die Schleife genau?

- **zuerst wird die Initialisierung ausgeführt**
- **Prüfung der Bedingung**
- **Die Schritt-Anweisung wird ausgeführt**

For-Schleife

Beispiele in der IDE

Wann welche Schleife?

For-Schleifen:
wenn die Anzahl der Durchläufe im voraus bekannt ist
(Wenn man einen Zähler braucht)

While/Do-While Schleifen:
wenn die Anzahl der Durchläufe unbekannt ist
(Wenn man keinen Zähler braucht)

Schleifen

Fragen?

Debugging

Debugging

Durch Doppelklick in einer Zeile setzt man einen Breakpoint:

```
System.out.println(i);
```

Dann durch Klick auf den Käfer den Debuggingmodus starten

Über die Taskleiste kann man den Debuggingmodus steuern

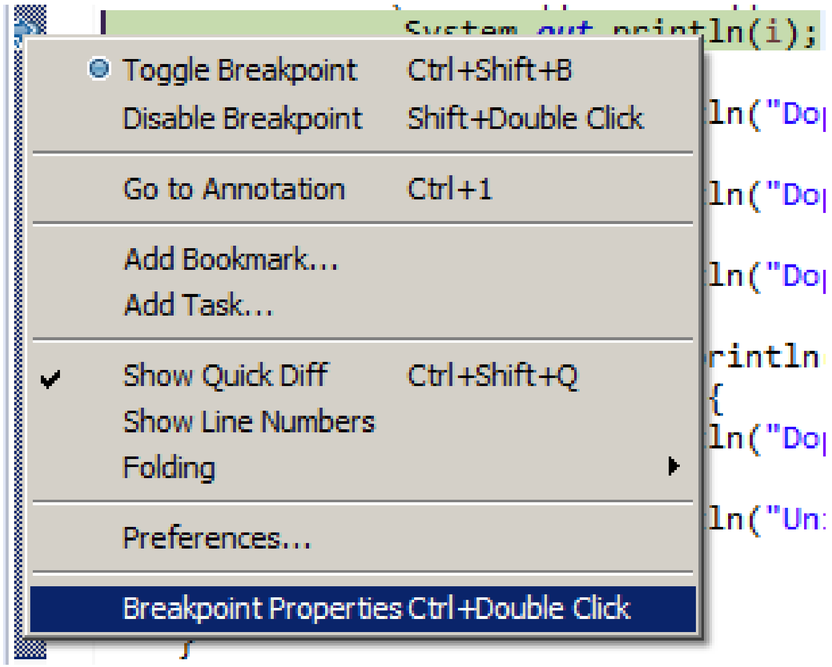
Debugging

Beim Debugging von Schleifen sind Conditional Breakpoints nützlich.

Dazu erst wie gewohnt einen Breakpoint setzen:

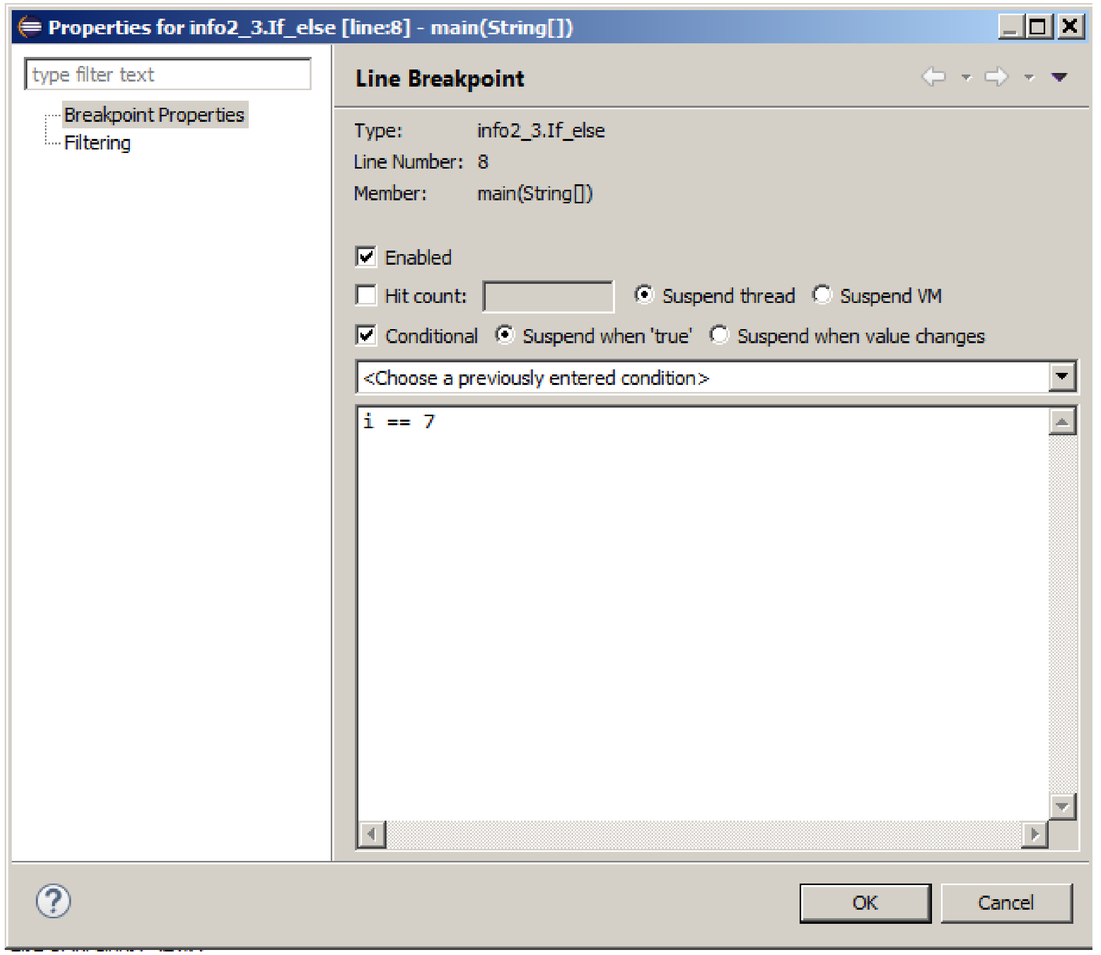
```
System.out.println(i);
```

Dann per Rechtsklick die Eigenschaften des Breakpoints öffnen:



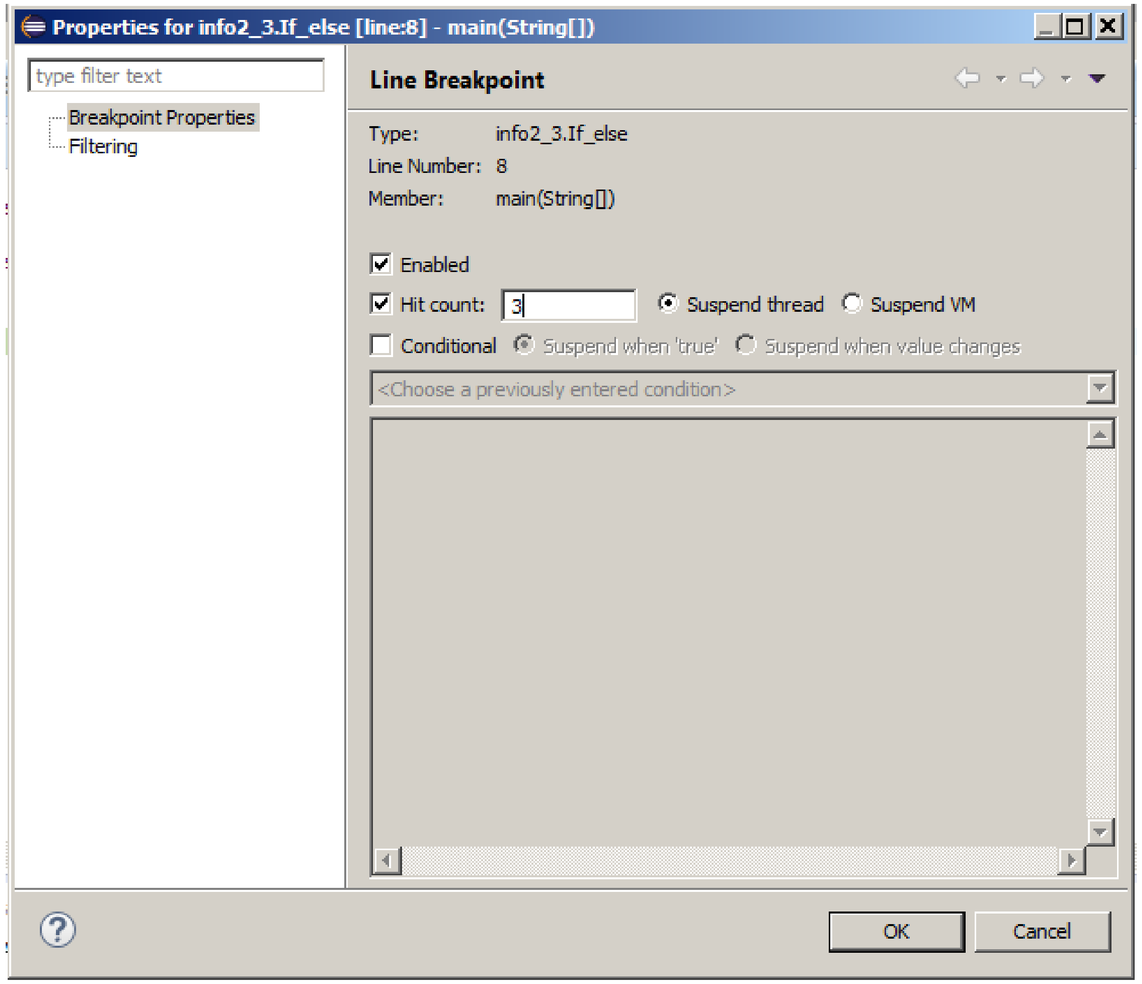
Debugging

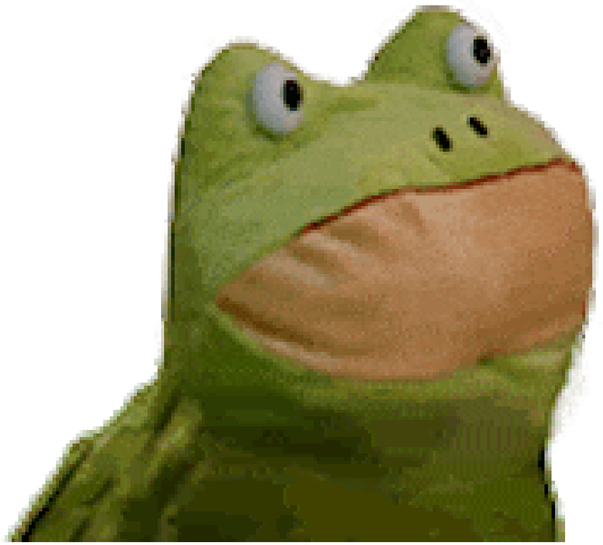
Über Condition kann z.B. die Laufvariable auf einen bestimmten Wert überprüft werden.
Nur wenn die Bedingung wahr ist, wird am Breakpoint angehalten.



Debugging

Mit dem *HitCount* kann man z.B. einstellen, dass erst ab dem 3. Durchlauf am Breakpoint gestoppt werden soll. Der HitCount ist hilfreich, wenn die Schleife keine Laufvariable hat.





Ende! Fragen?