Schleifen

Inhaltsverzeichnis

TECHNISCHE UNIVERSITÄT DARMSTADT

Wofür Schleifen?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Manchmal muss man Teile seines Codes mehrfach ausführen
- Goto war der Vorgänger der Schleifen
- In Assembler ist es immer noch goto
- Sollte aber nicht verwendet werden
- Code wird schwer durchschaubar

```
int counter = 0; // Zähler initialisieren
loop: counter++; // Anfang der Schleife
System.out.print("Zaehler:" + counter + "\n");
if(counter < 5) goto loop; // Sprung an den Anfang der Schleife
System.out.print("Fertig. Zaehler:" + counter + "\n");
```

Abbildung: Schleife [1]

While-Schleife

TECHNISCHE UNIVERSITÄT DARMSTADT

- Syntax:

```
while(Bedingung){
  //code
}
```

- Solange die Bedingung zu true ausgewertet wird, werden Anweisungen wiederholt
- Alle logischen Operatoren können verwendet werden

```
int counter = 0;
while(counter < 5) {
    counter++;
    System.out.println("Zaehler:" + counter);
}
```

While-Schleife
Continue und break

TECHNISCHE UNIVERSITÄT DARMSTADT

- Manchmal soll eine Schleife von vorne beginnen, obwohl diese noch nicht komplett durchgelaufen ist
- **continue** setzt die Schleife beim Anfang fort
- Manchmal soll eine Schleife vorzeitig verlassen werden
- **break** bricht die Schleife sofort ab

Continue und break Beispiel

TECHNISCHE UNIVERSITÄT DARMSTADT

```
int counter = 0;
while(counter < 10) {

    if(counter > 5) {
        System.out.print("Ha␣nun␣per␣continue" + "\n");
        counter++;
        continue;
     }

    if(counter == 8) break;
    counter++;
}
```

While-Schleife
Do-While Schleife

TECHNISCHE UNIVERSITÄT DARMSTADT

- Körper von while Schleifen evtl. nie abgearbeitet
- Do-while wird mindestens einmal ausgeführt
- Syntax:

```
do {
    //code
} while(Bedingung);
```

- Fuß gesteuerte Schleife

```
boolean ready = true;
do {
ready = false;
System.out.print("Ich werde ausgeführt");
} while(ready);
```

For-Schleife

TECHNISCHE UNIVERSITÄT DARMSTADT

- Häufig legt man für eine Variable eine Startbelegung fest
- Testet auf eine Bedingung mit dieser Variablen
- Und verändert die Variable beim Durchlauf
- Speziell für diesen Fall, die for-Schleife
- Syntax:

```
for(Initialisierung; Bedingung; Aktion) { Code; }
```

```
for(int counter = 0; counter < 5; counter++) {
        System.out.print(counter + ". Durchlauf");
}
```

Hinweis

Auch hier nach dem Schleifenkopf kein **;**

Wissenswertes

TECHNISCHE UNIVERSITÄT DARMSTADT

- For-Schleifen sehr flexibel
- Erst ausführen der Initialisierung
- Dann testen, ob Bedingung zu true ausgewertet werden kann
- Ausführen des Codes im Schleifenrumpf
- Aktionsanweisung ausführen
- ...Bedingung erneut testen usw.
- Man kann auch Teile des Bereiches leer lassen

```
int counter = 0;
for( ; counter < 5; ) {
        System.out.print("Schleife");
        counter++;
}
```

For-Schleife
Verschachtelte Schleifen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Natürlich kann man Schleifen auch verschachteln

```
for(int y = 0; y < 10; y++) {
        for(int x = 0; x < 10; x++) {
                System.out.print("0");
        }
        System.out.print("\n");
}
```

Hinweis

Achtet auf die Variablen die zum Zählen verwendet werden! (Nicht gleiche Namen wählen!)

Teiler
Live-Coding

TECHNISCHE UNIVERSITÄT DARMSTADT

- Bestimme alle Teiler einer Zahl

```
public class Teiler {
        public static void main(String[] args) {
                int number = 666;
                for(int i = 2; i < number; i++) {
                        if(number % i == 0)
                                System.out.println("Teiler:␣" + i);
                }
        }
}
```

Einschub: Gültigkeitsbereich

TECHNISCHE UNIVERSITÄT DARMSTADT

- Variablen unterliegen einem Gültigkeitsbereich
- Variablen sind nur innerhalb ihres Blocks gültig und sichtbar
- Wird der Block verlassen, wird die Variable gelöscht

```
int i = 2;

if(true) {
 char c = '!';

 System.out.println(i);
 System.out.println(c);
}

System.out.println(i);
//System.out.println(c);  // nicht mehr sichtbar
```

Switch-Case Anweisung

TECHNISCHE UNIVERSITÄT DARMSTADT

```
int zahl = ...

if(zahl == 1)
   System.out.println("Die␣Zahl␣ist␣Eins.");
else if(zahl == 2)
   System.out.println("Die␣Zahl␣ist␣Zwei.");
else if(zahl == 3 || zahl == 4 || zahl == 5)
   System.out.println("Die␣Zahl␣ist␣Drei,␣Vier␣oder␣Fünf.");
else
   System.out.println("Die␣Zahl␣ist␣weder␣Eins␣noch␣...␣Fünf.");
```

Switch-Case Anweisung

TECHNISCHE UNIVERSITÄT DARMSTADT

```
int zahl = ...
switch(zahl)
{
     case 1:
        System.out.println("Die␣Zahl␣ist␣Eins."); break;
     case 2:
        System.out.println("Die␣Zahl␣ist␣Zwei."); break;
     case 3: case 4: case 5:
        System.out.println("Die␣Zahl␣ist␣Drei,␣Vier␣oder␣Fünf.");
            break;
     default:
        System.out.println("Die␣Zahl␣ist␣weder␣Eins␣noch..Fünf.");
}
```

Switch-Case Anweisung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Alternativ zu vielen else ... if
- Es sind nur Vergleiche mit primitiven Datentypen möglich z.b

```
char c = 'a';
switch(c) {
        case 'a': break;
        //...
        case 'g': break;
}
```

- Es können keine größeren primitiven Typen (long, float, double) oder Objekte benutzt werden
- Es wird nur auf Gleichheit geprüft
  - Verschachtelte if sind auf jede Art von Prüfung und jeden Datentyp anwendbar

Hinweis:

Seit Java 7 können auch Strings dem switch statement übergeben werden

Fragerunde

TECHNISCHE UNIVERSITÄT DARMSTADT

- Zähle die Schleifenarten auf und nenne deren Besonderheiten
- Wann sollte man **switch** verwenden?
- Wann sollte man **for** benutzen?
- Ist es möglich **while** Schleifen in **for** Schleifen zu verschachteln?

Finde 2 Fehler

```
int counter = 0;
while(counter > 10) {
{
        System.out.print("Ha..Schleife");
}
```

Quellen

TECHNISCHE UNIVERSITÄT DARMSTADT

- **[1]** http://www.bilderkiste.de/de/cliparts/schleifen-1.html