Java - Kurzreferenz

Basis-Syntax

Variablen

```
public class Variablen {
 public static void main(String[] args) {
   // Variablen
   int zahl1 = Integer.parseInt(args[0]);
   int zahl2 = Integer.parseInt(args[1]);
   // Kontrollanweisungen
   if (zahl1 < zahl2){
     // Vorherige Überprüfung der Bedingung
     while (zahl1 <= zahl2){
       if (zahl1 < 0) break;
       System.out.println(zahl1);
       zahl1++;
     }
   }
   else if (zahl1 > zahl2){
     // Spätere Überprüfung der Bedingung
     do {
       System.out.println("1: " + zahl1
                  +  " / 2: " + zahl2);
     }
   while (zahl1 < zahl2);
     // Vorherige Überprüfung inkl. Zähler
     for(int i = zahl2; i < zahl1; i++){
        System.out.println(zahl1);
     }
   }
   else {
     System.out.println("Keine Verarbeitung möglich.");
   }
  }
}
```

Arrays

```
// Eindimensional
int[] ganzzahlen1 = {1, 2, 3};
int[] ganzzahlen2 = new int[]{4, 5, 6};
// Mehrdimensional
double[][] matrix1 = new double[2][3];
matrix1[1][1] = 1.1;  // ...
double[][] matrix2 = {{1.1, 1.2, 1.3}, {2.1, 2.2, 2.3}};
```

Aufzählungsklassen

```
// Erstellung
public enum Jahreszeit { FRUEHLING, SOMMER,
                         HERBST, WINTER };
// Verwendung
Jahreszeit jahreszeit1 = Jahreszeit.FRUEHLING;
// Wert einzeln abrufen
Jahreszeit jahreszeit2 =
  (Jahreszeit) Enum.valueOf(Jahreszeit.class, "WINTER");
// Werte komplett abrufen
Jahreszeit[] werte = Jahreszeit.values();
```

Klassen und Objekte

Einfache Bausteine einer Klasse

```
package adressbuch;

class Adresse {
  // Eigenschaften
  private String strasse;
  private String plz;
  private String stadt;

  // Konstruktor
  public Adresse(String strasse,
                 String plz,
                 String stadt) {
    this.setStrasse(strasse);
    this.setPlz(plz);
    this.setStadt(stadt);
  }

  // Getter und Setter
  public String getStrasse() {
    return strasse;
  }

  public void setStrasse(String strasse) {
    this.strasse = strasse;
  }

  // ...
}
```

Objekte

```
public class PersonTester {
  public static void main(String[] args) {
    // Adresse und Person erstellen

    Adresse hochhaus = new Adresse("OOP-Weg 6",
                                   "12345", "Java-Stadt");
    Person anton = new Person("Anton", "Ebenhof", hochhaus);
    Person elvira = new Person("Elvira", "Hülzemann",
                               hochhaus);
    // Test-Ausgabe
    System.out.println(anton.export(Person.EXPORT_XML));
    // Änderung der Adresse außerhalb
    // und durch Referenz innerhalb
    hochhaus.setPlz("54321");
    if (anton.getAdresse().getPlz().equals("54321")){
      System.out.println(elvira.export(Person.EXPORT_CSV));
    }
  }
}
```

Ausgabe

```
<Person Nr="1">
  <Vorname>Anton</Vorname>
  <Nachname>Ebenhof</Nachname>
  <Alter>31</Alter>
</Person>
2;Elvira;Hülzemann
```

Vererbung

Einfache Elternklasse

```
package produktkatalog;

// Eltern-Klasse
public class Produkt {
  // Eigenschaften
  private int nr;
  private String name;
  protected double preis;
  // Konstruktor
  public Produkt(int nr, String name,
                 double preis){
    this.setName(name);
    this.setNr(nr);
    this.setPreis(preis);
  }
  // In Kind-Klasse überschriebene Methode
  public String getBeschreibung(){
    return this.getName() + " ("
           + this.getNr() + ") ";
  }
  // Getter und Setter
  public int getNr() {
    return nr;
  }
  public void setNr(int nr) {
    this.nr = nr;
  }
  // ...
}
```

```
//Kind-Klasse
public class Buch extends Produkt {
  // Zusätzliche Eigenschaft
  private int seiten;
  // Konstruktor
  public Buch(int nr, String name,
              double preis, int seiten) {
    // Aufruf des Eltern-Konstruktors
    super(nr, name, preis);
    // Weitere Anweisungen
    this.setSeiten(seiten);
  }
  // Überschriebene Methode
  public String getBeschreibung(){
    return super.getBeschreibung()
           + this.getSeiten() + " S.";
  }
  // Weitere Methoden
  public int getSeiten() {
    return seiten;
  }
  // ...
}
```

Klassen und Objekte

Weitere Bausteine einer Klasse

```
public class Person {

  // Objekt-Eigenschaften
  private String vorname;
  private String nachnachme;
  // Standardwert
  // (automatisch, Ini-Block, dynamisch oder statisch)
  private int alter = Person.berechneAlter(); // = 31
  private Adresse adresse;
  // Verpflichtende nicht änderbare Eigenschaft
  private final int nr;
  // Klassen-Eigenschaften
  // Standardwert
  // (automatisch, Ini-Block, dynamisch oder statisch)
  private static int zaehler;

  // Konstanten
  // Wertvorgabe bei Deklaration
  public static final int EXPORT_XML = 1;
  public static final int EXPORT_CSV = 2;

  // Initialisierungsblock
  // Statisch
  static
  {
    Person.zaehler = 0;
  }
  // Objektbezogen
  {
    Person.zaehler++;
  }
// Konstruktor (mit Überladung)
  public Person (String vorname, String nachname,
                 Adresse adresse){
    this.setVorname(vorname);
    this.setNachnachme(nachname);
    this.setAdresse(adresse);
    // Methodenaufruf oder durch Initialisierungsblock
    // Person.erhoeheZaehler();
   this.nr = Person.getZaehler();
  }

  public Person (String vorname, String nachname,
                 String strasse, String plz,
                 String stadt){
    // Expliziter Aufruf der Konstruktor-Variante
    this(vorname, nachname, new Adresse(strasse, plz,
                                        stadt));
  }
```

```
// Objekt-Methoden
  public String export(int format){
    String text;
      switch (format){
        case Person.EXPORT_CSV:
          text = + this.getNr() + ";"
                 + this.getVorname() + ";"
                 + this.getNachnachme() + "\n";
        break;
        case Person.EXPORT_XML:
          text = "<Person Nr=\""+ this.getNr()
                 + "\"><Vorname>" + this.getVorname()
                 + "</Vorname><Nachname>" + this.getNachnachme()
                 + "</Nachname><Alter>" + this.getAlter()
                 + "</Alter></Person>";
      break;
      default:
        text = "";
    }
    return text;
  }

  private static int berechneAlter(){
      return 31;
  }
// Klassen-Methoden (statisch)
  private static void erhoeheZaehler(){
    Person.zaehler++;
  }
  public static int getZaehler(){
    return Person.zaehler;
  }

  // Getter und Setter
  public String getVorname() {
    return vorname;
  }
  public void setVorname(String vorname) {
    this.vorname = vorname;
  }
  // ...
  // Nur lesbar
  public int getAlter(){
    return this.alter;
  }
  public int getNr() {
    return this.nr;
  }
}
```

Implementierung

```
// Schnittstelle
// Vererbung auch möglich
// Verwendung auch Rückgabe-/Übergabe-Typ
public interface Druckbar {
   public String exportXML();
   public String exportCSV();
}

// Implementierung
class Person2 implements Druckbar,
                         Comparable<Person2> {
  // Implementierte Methoden
  // ... selbstdefinierte Schnittstelle
  public String exportXML(){
    return "export-XML-Format";
  }
  public String exportCSV(){
    return "export-CSV-Format";
  }
  // ... Framework-Schnittstelle
  public int compareTo(Person2 person) {
  // Vergleiche mit Rückgabe 1, -1, 0
    return 0;
  }
}
```

Vererbung

Abstraktion

```
// Abstrakte Klasse = abstraktes Konzept
public abstract class Zahlungsweise {
  // Konkrete Eigenschaften
  protected double summe;
  protected String empfaenger;
  // Konstruktor
  public Zahlungsweise(double summe,
                       String empfaenger){
    this.setEmpfaenger(empfaenger);
    this.setSumme(summe);
  }
  // Abstrakte Methode
  public abstract void durchfuehren();
  // Konkrete Getter und Setter
  public double getSumme() {
    return this.summe;
  }
  public void setSumme(double summe) {
    this.summe = summe;
  }
  // ...
}
```

```
public class Kreditkarte extends Zahlungsweise {
  // Konstruktor
  public Kreditkarte(double summe, String empfaenger) {
    super(summe, empfaenger);
  }
  // Überschriebene Methode
  public void durchfuehren() {
    System.out.println("Zahlung mit Kreditkarte an "
                       + this.getEmpfaenger()
                       +  " über " + this.getSumme() + ".");
  }
}

public class WebshopTester {
  public static void main(String[] args) {
    Zahlungsweise kreditKarte =
                  new Kreditkarte(99.99, "Webshop");
    kreditKarte.durchfuehren();
  }
}
```

Ausgabe:

```
Zahlung mit Kreditkarte an Webshop über 99.99.
```

Zusammengestellt von Marco Skulschus
Satz und Layout Beatrice Walendy

Comelio GmbH
Goethestraße 34, 13086 Berlin
Web: www.comelio.com

Terrashop GmbH
Lise-Meitner-Str. 8, 53332 Bornheim
Web: www.terrashop.de

ISBN 978-3-939701-38-5