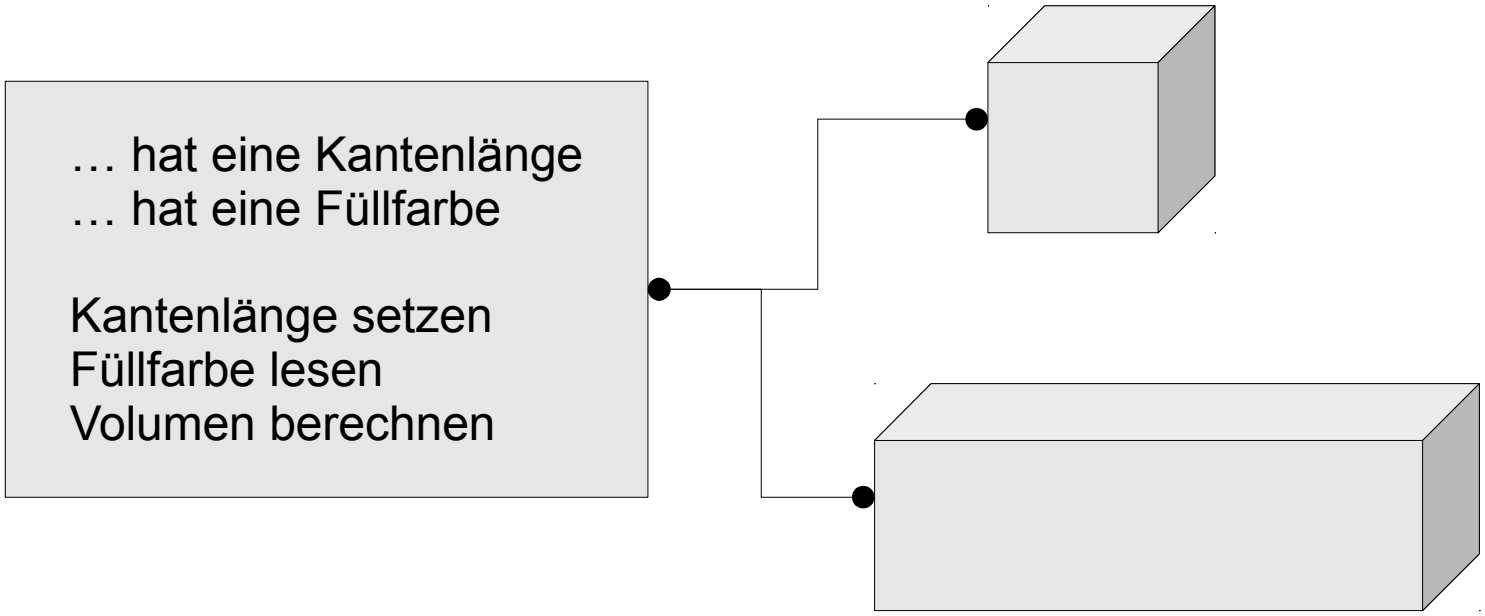
Java - Polymorphie



Überladen von Methoden und Konstruktoren

- Innerhalb einer Klasse werden Methoden mit dem gleichen Namen definiert.
- Die Methoden sind anhand ihrer Parameterliste unterscheidbar.
- Der Rückgabewert spielt bei der Überladung keine Rolle.

Konstruktoren

```
public clsVektor()
{
    this.xPunkt = 0;
    this.yPunkt = 0;
    this.zPunkt = 0;
}
```

- Konstruktoren haben den gleichen Name wie die Klasse, in der sie definiert werden.
- Diese Methoden werden automatisch bei der Erzeugung einer Instanz einer Klasse aufgerufen.

Überladung des Konstruktors

```
public clsVektor(double x, double y, double z)
{
    this.xPunkt = x;
    this.yPunkt = y;
    this.zPunkt = z;
}
```

- Die Namen der Konstruktoren sind gleich.
- Die Parameterliste unterscheidet sich aber in der Anzahl der Parameter.
- Anhand der Anzahl der Parameter wird dem Aufruf der Konstruktor zugeordnet.

Aufruf eines Konstruktors im Konstruktor

```
public clsVektor(double x, double y)
{
    this(x, y, 0);
}
```

- Dem Schlüsselwort `this` folgen die runden Klammern.
- Entsprechend der Parameter in den Klammern wird der passende Konstruktor für `this` aufgerufen.

Aufruf der Konstruktoren

```
clsVektor vektorA = new clsVektor();
clsVektor vektorB = new clsVektor(4, 5);
clsVektor vektorC = new clsVektor(1, 2, 3);
```

- Ein Konstruktor wird durch das Schlüsselwort `new` automatisch aufgerufen.
- In Abhängigkeit der Anzahl der Parameter in der Parameterliste wird dem Aufruf ein Konstruktor zugeordnet.

Methoden

```
public void addiereVektor(double x, double y)
{
    this.xPunkt = this.xPunkt + x;
    this.yPunkt = this.yPunkt + y;
}
```

- Methoden werden von Objekten aufgerufen, um Attribute zu verändern.
- Methoden können einen Wert zurück geben oder nicht.

Parameterliste einer Methode

```
public void addiereVektor(double x, double y)
{
    this.xPunkt = this.xPunkt + x;
    this.yPunkt = this.yPunkt + y;
}
```

- Methoden haben eine Parameterliste, die leer sein kann.
- Jeder Parameter hat einen bestimmten Datentyp.
- Die Parameter werden in der Liste durch Kommata getrennt.

Überladung mit Hilfe der Anzahl der Parameter

```
public void addiereVektor(double x, double y,
                          double z)
{
    this.xPunkt = this.xPunkt + x;
    this.yPunkt = this.yPunkt + y;
    this.zPunkt = this.zPunkt + z;
}
```

- Die Namen der Methoden sind gleich.
- Die Parameterliste unterscheidet sich aber in der Anzahl der Parameter.
- Anhand der Anzahl der Parameter wird dem Aufruf die Methode zugeordnet.

Überladung durch den Datentyp des Parameters

```
public void addiereVektor(int x, int y)
{
    this.xPunkt = this.xPunkt + x;
    this.yPunkt = this.yPunkt + y;
}
```

- Wenn die Anzahl der Parameter in der Liste gleich sind, muss mindestens ein Parameter einen anderen Datentyp besitzen.

Aufruf der Methoden

```
vektorA.addiereVektor(2.3, 2.4);
vektorA.addiereVektor(2, 3);
vektorA.addiereVektor(1, 0, 4);
```

- In Abhängigkeit der Anzahl und des Datentyps wird dem Aufruf eine Methode zugeordnet.
- Falls dem Aufruf keine Methode zugeordnet werden kann, wird der Fehler `no suitable method found` ausgegeben.

Statische Methoden

```
public static clsVektor addiereVektor
          (clsVektor vektorL, clsVektor vektorR)
{
   clsVektor tmp =
      new clsVektor(vektorL.xPunkt, vektorL.yPunkt);

   tmp.addiereVektor(vektorR.xPunkt, vektorR.yPunkt);
   return tmp;
}
```

- Der Rückgabetyp und das Schlüsselwort `static` spielen keine Rolle bei der Überladung.

Nutzung von Arrays

```
public void setzeVektor(double koordinaten[]){
    switch(koordinaten.length){
        case 3:
            this.zPunkt = koordinaten[2];
        case 2:
            this.yPunkt = koordinaten[1];
        case 1:
            this.xPunkt = koordinaten[0];
            break;
    }
}
```

Erläuterung

- Mit Hilfe eines Arrays kann eine beliebige Anzahl von Parametern einer Methode übergeben werden.
- In der Liste können vor und nach der Angabe des Array-Parameters viele beliebige Parameter folgen.

… ab Java 5

```
public void setzeVektor(double... ){
    switch(koordinaten.length){
        case 3:
            this.zPunkt = koordinaten[2];
        case 2:
            this.yPunkt = koordinaten[1];
        case 1:
            this.xPunkt = koordinaten[0];
            break;
    }
}
```

Erläuterung

- Die drei Punkte im Anschluss eines Datentyps symbolisieren ein beliebig langes Array.
- Der Parameter darf nur als letzter Parameter in der Liste im Methodenkopf stehen. Dem Parameter dürfen keine weiteren Parameter folgen.
- Die Anzahl der Elemente kann mit `.length` abgefragt werden.
- Das Array kann mit einer foreach-Schleife durchlaufen werden.

Aufruf der Methode

```
double[] koordinaten = new double[]{6,7};

vektorA.setzeVektor(koordinaten);
```

- Der Methode wird ein Array von Elementen übergeben.