Teil 6: Konstruktoren

-- deckt Lektion 7 des Kursbuches ab --

Modul „Grundlagen der objektorientierten Programmierung mit Java“

Prof. Dr. Cornelia Heinisch

Konstruktoren

Agenda

- Konstruktor zur Initialisierung
- Überladen von Konstruktoren
- Aufruf Konstruktor der Vaterklasse

Konstruktor zur Initialisierung

Eigenschaft eines Konstruktors

- **dient zum Initialisieren der Instanzvariablen**.
- **wird automatisch** – nach dem Erzeugen eines Objektes durch den `new`-Operator – **aufgerufen** (→ die Initialisierung kann damit nicht vergessen werden!)
- **spezielle Instanzmethode** einer Klasse, die den gleichen Namen wie die Klasse selbst besitzt.
- besitzt **keinen Rückgabetyp**.
- Compiler fügt automatisch einen Standard-Konstruktor ein, falls der Programmierer keinen eigenen Konstruktor erstellt.

Konstruktoren

Ein Konstruktor für die Klasse Person

```
public class Person
{
   private String vorname;  // privates Datenfeld vorname
   private String name;  // privates Datenfeld name

   public Person(String name, String vorname)
   {
         this.name = name;
         this.vorname = vorname;
   }
   . . . . .
}
```

Konstruktoren

Agenda

- Konstruktor zur Initialisierung
- Überladen von Konstruktoren
- Aufruf Konstruktor der Vaterklasse

Überladen von Konstruktoren

Mehrere Konstruktoren für eine Klasse

- Ein Konstruktor **kann überladen werden**.
- Es gibt dann mehrere Konstruktoren für eine Klasse, die sich in der Parameterliste unterscheiden.
- Innerhalb eines Konstruktors kann ein anderer Konstruktor der gleichen Klasse mit `this (Parameterliste)` aufgerufen werden.
- Ermöglicht Flexibilität beim Erstellen und Initialisieren von Objekten.
- Vermeidet Programmcode-Duplikate.

Agenda

- Konstruktor zur Initialisierung
- Überladen von Konstruktoren
- Aufruf Konstruktor der Vaterklasse

Aufruf eines Konstruktors der Vaterklasse durch den Programmierer

Durch `super(Parameterliste)` kann der Programmierer im Konstruktor der Sohnklasse selbst einen Konstruktor der Vaterklasse aufrufen.

Durch den expliziten Aufruf eines Konstruktors der Vaterklasse im Konstruktor der Sohnklasse durch den Programmierer kann Folgendes vermieden werden:

- Ein Zugriff auf die Instanzvariablen der Vaterklasse im Konstruktor der Sohnklasse ist nicht notwendig --> die Instanzvariablen der Vaterklasse `Person` können `private` bleiben.
- Die Klasse `Person` benötigt keinen selbst geschriebenen Default-Konstruktor. Ein Default-Konstruktor wird nur dann im Konstruktor der Sohnklasse aufgerufen, wenn nicht durch den Programmierer ein Aufruf eines Konstruktors der Vaterklasse erfolgt.
- Der Programmcode zur Initialisierung der Instanzvariablen der Klasse `Person` ist nicht sowohl im Konstruktor der Klasse `Person` als auch im Konstruktor der Klasse `Ausleiher` doppelt vorhanden, sondern wird nur in der Klasse `Person` implementiert.