Programmierkurs Java

UE 9 – Klassen und Objekte I

Dr.-Ing. Dietrich Boles

Gliederung

- Motivation
- Klassendefinition (erste Version)
- Objektvariablen
- Objekterzeugung
- Attribute
- Arrays mit Objekten
- Arrays und Objekte als Attribute
- Zusammenfassung

Motivation (1)

Definition „Klasse":

Datenstruktur/-typ zur Zusammenfassung von mehreren **Elementen u.U. unterschiedlichen Typs** zu einer Einheit. Der Wertebereich einer Klasse ist das kartesische Produkt der Wertebereiche seiner einzelnen Elemente (auch **Attribute** genannt)

Definition „Objekt":

Konkrete Ausprägung einer Klasse

Anmerkungen:

- Klassen sind Referenzdatentypen!
- Eine Klasse ist quasi ein Bauplan für gleichartige Objekte

- **Wichtig**: die Definitionen werden später noch erweitert!!!

Motivation (2)

Beispiel: Personaldatei

Mitarbeiter:

Name:	String name;
Alter:	short alter;
Geschlecht:	boolean maennlich;
Gehalt:	float gehalt;

Definition der Klasse:

```
class Mitarbeiter {
  String name;
  short alter;
  boolean maennlich;
  float gehalt = 2000.0F;
}
```

Definition einer Objektvariablen:

```
Mitarbeiter karl;
```

Erzeugung eines Objektes:

```
karl = new Mitarbeiter();
```

Zugriff auf ein Objekt:

```
karl.alter = 42;
```

Klassendefinition (erste Version)

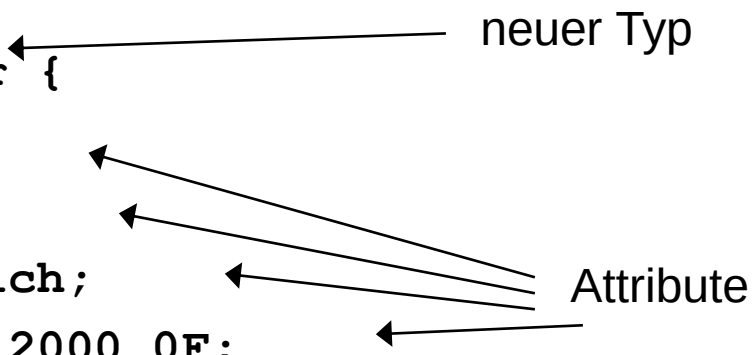
```
<Klassen-def> ::= [ "public" ] "class" <Bezeichner>
                  "{" { <Variablendefinition> } "}"
```

Anmerkungen:

- Klassen sind Typen
- der Bezeichner legt den Namen für den neuen Typ fest
- der neue Typ kann überall dort stehen, wo auch Standarddatentypen stehen dürfen
- die Variablen nennt man auch Elemente, Attribute oder Instanz-Variablen
- Code-Konvention: Klassenname beginnt mit Großbuchstaben

Beispiel:

```
class Mitarbeiter {
  String name;
  short alter;
  boolean maennlich;
  float gehalt = 2000.0F;
}
```



Klassen und Dateien

- In einer Datei dürfen mehrere Klassen definiert werden.
- In einer Datei darf nur eine Klasse als `public` deklariert werden. Der Name dieser public-Klasse muss dem Namen der Datei entsprechen.
- In der Regel besitzt die `public`-Klasse die `main`-Prozedur.
- Klassen dürfen auch in anderen Dateien desselben Verzeichnis definiert werden.
- In einem Verzeichnis dürfen keine zwei Klassen mit demselben Namen definiert werden.
- Klassen dürfen nicht innerhalb einer anderen Klasse definiert werden (gilt nur für uns!)
- **Wichtig**: Diese Regeln werden später noch korrigiert bzw. ergänzt!

Objektvariablen

```
<Objektvar-Def> ::= <Klassenbezeichner> <Bezeichner>
                    {"," <Bezeichner> } ";"
```

Anmerkungen:

- <Klassenbezeichner> muss gültige Klasse sein
- <Klassenbezeichner> ist der Typ der Objektvariablen
- Reservierung von Speicherplatz für Adressen (Referenzen auf Objekte)
- <Bezeichner> ist der Name der Objektvariablen
- kann Referenzen (Adressen) auf Objekte speichern

Beispiele:

```
Mitarbeiter karl;
Mitarbeiter hans, hubert;
Mitarbeiter heike = null; // Initialisierung
```

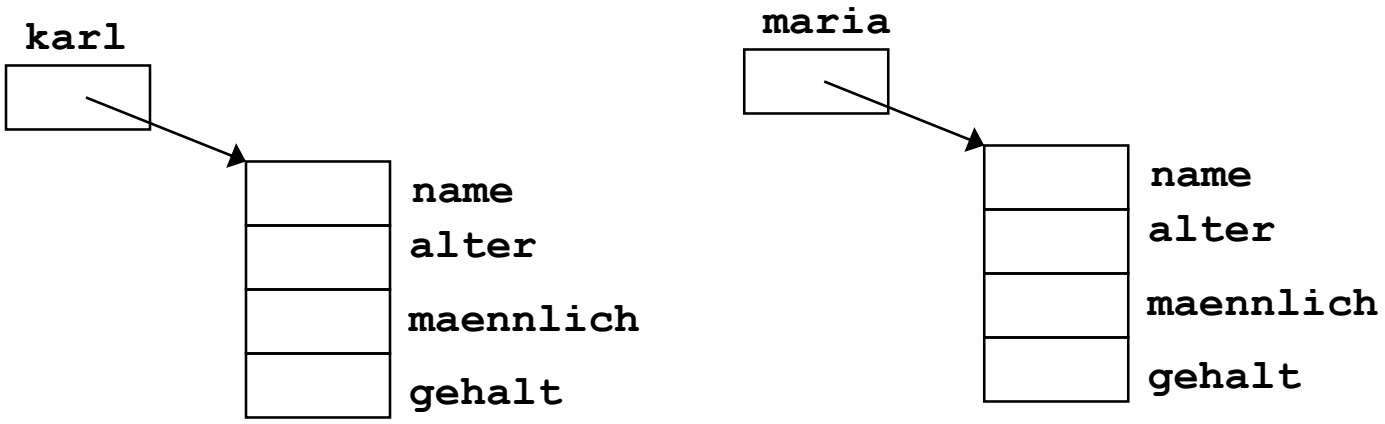
```
<Objekt-Erz> ::= "new" <Klassenbezeichner> "(" ")"
```

Anmerkungen:

- <Klassenbezeichner> muss gültige Klasse sein
- <Klassenbezeichner> ist der Typ des Objektes
- Reservierung von Speicherplatz für Attribute (auf dem Heap)
- liefert Referenz (Adresse) des Speicherbereiches

Beispiele:

```
Mitarbeiter karl = new Mitarbeiter();
Mitarbeiter maria = new Mitarbeiter();
```



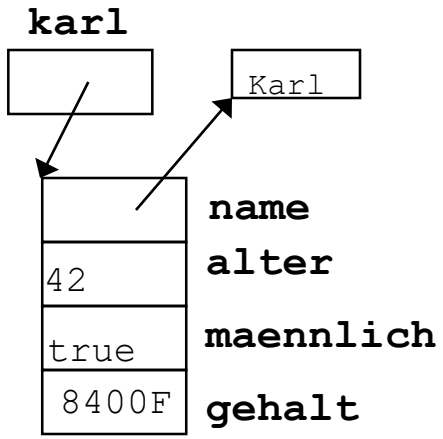
```
<Attr-Zugr> ::= <objektvar-bezeichner>
                "."
                <attr>
```

Anmerkungen:

- <objektvar-bezeichner> muss gültiger Name einer Objektvariablen sein
- <attr> muss Attribut einer Klasse vom Typ der Objektvariablen sein
- Objekt muss vorher erzeugt worden sein (-> Laufzeitfehler!)
- können überall dort stehen, wo auch Variablennamen stehen dürfen

Beispiele:

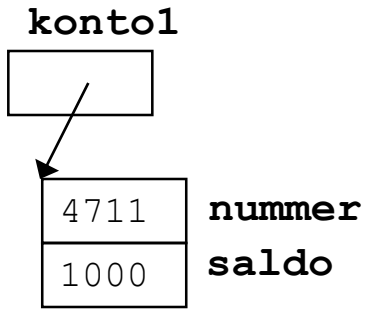
```
Mitarbeiter karl = new Mitarbeiter();
karl.name = "Karl";
karl.alter = 42;
karl.maennlich = true;
karl.gehalt = 2000.0F * (karl.alter / 10.0F);
```

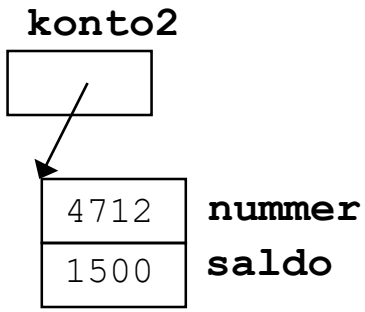


- bei der Definition der Klasse (allgemeingültig)
- durch Default-Werte des entsprechenden Typs
- explizit für jedes Attribut

Beispiele:

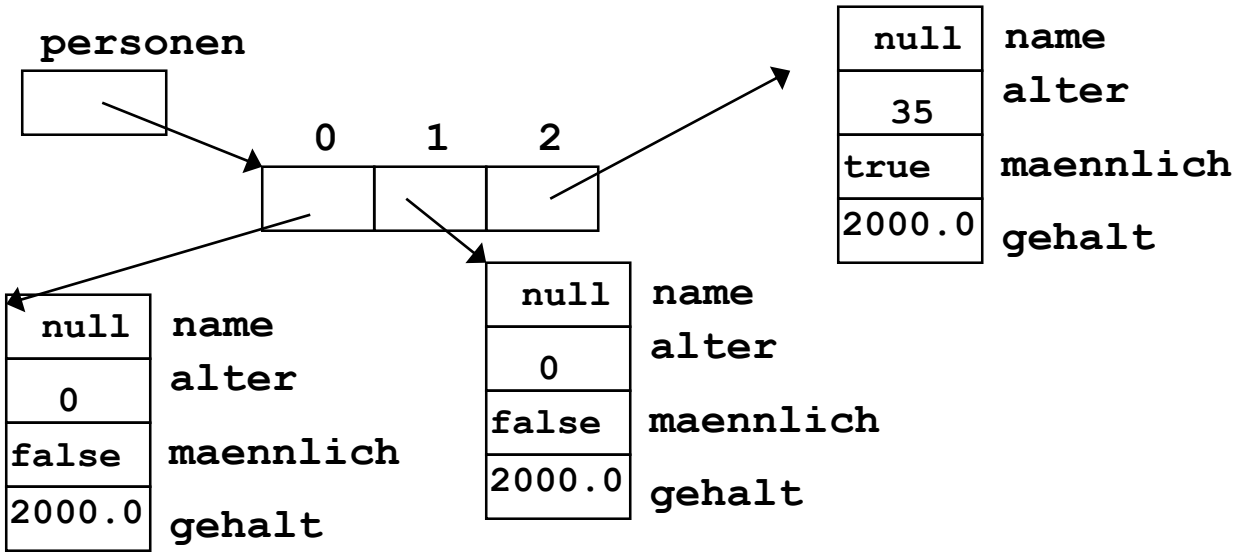
```
class Konto {
   int nummer;
   int saldo = 1000;
} ...
public static void main(String[] a){
  Konto konto1 = new Konto();
  konto1.nummer = 4711;
  Konto konto2 = new Konto();
  konto2.nummer = 4712;
  konto2.saldo += 500;
}
```





Arrays mit Objekten

```
Mitarbeiter[] personen = new Mitarbeiter[3];
for (int i=0; i<3; i++) {
  personen[i] = new Mitarbeiter();
}
personen[2].alter = 35;
personen[2].maennlich = true;
```



Arrays und Objekte als Attribute

```
class Produkt {
  String bez;
  float  preis;
}

class Kunde {
  int    nummer;
  String name;
}

class Warenkorb {
  Kunde     kaeufer;
  Produkt[] produkte;
}
```

```
public class Shop {

public static void main(String[] a) {

  Warenkorb korb = new Warenkorb();

  korb.kaeufer = new Kunde();
  korb.kaeufer.nummer = 4711;
  korb.kaeufer.name = "Karl";

  korb.produkte = new Produkt[3];

  korb.produkte[0] = new Produkt();
  korb.produkte[0].bez = "Hamster-Buch";
  korb.produkte[0].preis = 24.90F;
  ...
}}
```

Zusammenfassung

- Klasse: Zusammenfassung mehrerer Variablen u. U. unterschiedlichen Typs zu einer Einheit (wird erweitert!)
- Klasse: benutzerdefinierter Datentyp
- Klassen sind Referenzdatentypen
- Objekt: Ausprägung einer Klasse
- Objektvariable: Variable zur Referenzierung eines Objektes
- Zugriff auf die einzelnen Variablen (Attribute) eines Objektes via Objektvariable und Punktnotation