Einführung in die Informatik 1

– Klassen und Objekte in Java –

Sven Kosub

AG Algorithmik/Theorie komplexer Systeme
Universität Konstanz

E 202 | Sven.Kosub@uni-konstanz.de | Sprechstunde: Freitag, 12:30-14:00 Uhr, o.n.V.

Wintersemester 2008/2009

Klassen

Erinnerung:

(Objekt-)Klasse

- Zusammenfassung von Objekten mit gleichen Zustandsvariablen und Methoden
- Objekte einer Klasse durch Werte der Zustandsvariablen beschrieben
- modelliert einen bestimmten Typ von Objekten

Beschreibung von Typen als Klassen bedeutet:

- dynamisch: Menge aller (erzeugten) Objekte der Klasse
- statisch: Festlegung des Aufbaus (Struktur) aller Objekte der Klasse

Klassen: Bestandteile

- **Instanzen**
  Objekte der Klasse
- **Instanzenvariablen**
  speichern individuellen Datensatz bzw. Zustand eines Objekt
- **Instanzenmethoden**
  werden auf einem Objekt aufgerufen, laufen auf diesem, haben Zugriff auf Instanzenvariablen des Objektes
- **Klassenvariablen**
  nur ein Satz pro Klasse, speichern globalen Zustand der Klasse
- **Klassenmethoden**
  werden wie normale Funktionen aufgerufen, unabhängig von Objekten (müssen nicht einmal existieren)

Klassen: Deklaration

Klassen:

ClassDeclaration :
Modifier$_{\text{opt}}$ `class` *Identifier* { *DeclarationStatements* }

DeclarationStatements :
DeclarationStatement$_{\text{opt}}$
DeclarationStatements *DeclarationStatement*

DeclarationStatement :
VariableDeclaration ;
MethodDeclaration
ConstructorDeclaration
ClassDeclaration

```
class Messwert2D {
  int x;    // x-Koordinate
  int y;    // y-Koordinate
  double wert; // Messwert
}
```

- Klasse `Messwert2D` modelliert reinen Verbundtyp
- nur Felder (Variablen), keine Methoden
- Container-Klasse für Datenbeziehungen

Klassen: Deklaration

```
class Time {
  int sec=0;    // Sekunden, 0 ≤ sec ≤ 60
  int min=0;     // Minuten, 0 ≤ min ≤ 60
  int hrs=0;           // Stunden, 0 ≤ hrs
  void tick() {
    sec++;
    if (sec>=60) {
      sec -=60;
      min++;
      if (min>=60) {
        min -=60;
        hrs++;
      }
    }
  }
}
```

- Klasse `Time` zur elektronischen Zeitnahme
- Methoden `tick()` implementiert die Operation des Tickens des Sekundenzählers

Klassen: Deklaration

```
class M2DT {
  // Instanzenvariablen
  Messwert2D m;                          // Messdaten in 2D
  Time t;                              // Zeitpunkt des Messens
  // Konstruktor
  M2DT() {
    m=new Messwert2D();
    t=new Time();
  }
}
```

- Klasse `M2DT` verbindet ein Objekt (Messwert) vom Typ `Messwert2D` mit einem Objekt vom Typ `Time` (Zeitpunkt der Messung)
- Variablen `m` und `t` sind Referenzvariablen
- Konstruktor `M2DT` initialisiert Objekt vom Typ `M2DT`

Modifikatoren

Modifikatoren (für Klassen, Variablen, Methoden):

Modifier : ein Terminal aus

```
public  private  protected
```

Bedeutung:

- regulieren Zugriffsrechte unter den Objekten
- **public** erlaubt globalen Zugriff auf Klasse, Variable oder Methode
- **private** erlaubt Zugriff innerhalb der Klasse
- **protected** erlaubt Zugriff von abgeleiteten Klassen aus
- ohne Angabe (*Modifier* ist optional) gilt Zugriff innerhalb eines Pakets (`package`)

Modifikatoren

```
public class Time {
  // Sekunden, 0 ≤ sec ≤ 60
  private sec=0;
  // Minuten, 0 ≤ min ≤ 60
  private int min=0;
  // Stunden, 0 ≤ hrs
  private int hrs=0;
  public void tick() {
    sec++;
    if (sec>=60) {
      sec -=60;
      min++;
      if (min>=60) {
        min -=60;
        hrs++;
      }
    }
  }
}
```

Zugriffsrechte:

- jedes Objekt darf auf Klasse `Time` zugreifen
- auf die Instanzenvariablen `sec`, `min` und `hrs` darf nur das besitzende Objekte zugreifen
- Methoden `tick` darf von allen Objekten aufgerufen werden

Wie können die Instanzenvariablen gelesen und verändert werden?

Selektoren

Selektoren:

- spezielle Methoden für Zugriff auf Variablen
- stellen Schnittstelle für andere Objekte zur Verfügung
- regulieren wie Variablenwerte von außen gesehen werden
- Zugriff über Selektoren meist langsamer als direkter Variablenzugriff

```
public class Time {
  private sec=0;                                  // Sekunden, 0 ≤ sec ≤ 60
  private int min=0;                               // Minuten, 0 ≤ min ≤ 60
  private int hrs=0;                                    // Stunden, 0 ≤ hrs
  public int getSeconds() { return sec; }
  public int getMinutes() { return min; }
  public int getHours() { return hrs; }
  public int getSecondsInTotal() { return sec+60*min+3600*hrs; }
  ...
  public void tick() { ... }
}
```

Selektoren

Selektoren:

- spezielle Methoden für Zugriff auf Variablen
- stellen Schnittstelle für andere Objekte zur Verfügung
- regulieren wie Variablenwerte von außen gesehen werden
- Zugriff über Selektoren meist langsamer als direkter Variablenzugriff

```
public class Time {
  private sec=0;                                  // Sekunden, 0 ≤ sec ≤ 60
  private int min=0;                               // Minuten, 0 ≤ min ≤ 60
  private int hrs=0;                                    // Stunden, 0 ≤ hrs
  ...
  public void setSeconds(int s) { sec=s; }
  public void setMinutes(int m) { min=m; }
  public void setHours(int h) { hrs=h; }
  public void setSecondsInTotal(int s) {
                              hrs=s/3600; min=((s%3600)/60); sec=s%60; }
  public void tick() { ... }
}
```