4. AuD Tafelübung T-C3

Simon Ruderich

17. November 2010

Arrays

Unregelmäßige Arrays

```
int[][] x = new int[3][4];
x[2] = new int[2];

for (int i = 0; i < x.length; i++) {
    for (int j = 0; j < x[i].length; j++) {
        System.out.print("x ");
    }
    System.out.println();
}

// Ausgabe:
// x x x x
// x x x x
// x x
```

Arrays

Längenberechnung

```
String[][] x = new String[3][4];
x[2] = new String[2];

int length = 0;
for (int i = 0; i < x.length; i++) {
    if (x[i] == null) { continue; }
    for (int j = 0; j < x[i].length; j++) {
        // if je nach Aufgabe noetig oder nicht
        if (x[i][j] == null) { continue; }
        length++;
    }
}
// length = 10
```

Integerdivison

Integerdivison

```
double result;

result = 1/5;    // = 0, Integerdivison!

result = 1/5.;   // = 0.2
result = 1./5;   // = 0.2
result = 1./5.;  // = 0.2
```

Abstrakte Klassen

Abstrakte Klassen

- Klasse die Variablen und Methoden vorgibt, aber nicht instantiiert werden kann
- werden mit **abstract** erstellt
- zur Verwendung müssen Unterklassen gebildet werden
- eine nicht-abstrakte Klasse muss alle abstrakten Methoden überschreiben

Abstrakte Klassen

Beispiel

```
public abstract class Lok {
    public double geschwindigkeit;
    public double gewicht;
    public abstract void losfahren();
}
public class ELok extends Lok {
    public void losfahren() {
        // ...
    }
}

Lok  l = new Lok(); // Compiler-Fehler
ELok e = new ELok();
Lok  f = new ELok();
```

Interfaces

Schnittstellen

- Schnittstellen schreiben die Existenz von Methoden vor
- Klassen, die Interfaces implementieren, müssen die Methoden der Interfaces überschreiben
- Klassen können mehrere Interfaces implementieren
- Definition durch **implements**

Beispiel

```
public interface Logger {
    public void log(String string);
}
```

Interfaces

Beispiel

```
public interface Drawable {
    public void draw();
}

public interface Printable {
    public void print(Printer p);
}

public class Picture implements Printable,Drawable {
    public void print(Printer p) { ... }
    public void draw() { ... }
}
```

Überladen

Überladen

- Methoden mit gleichem Namen aber unterschiedlichen Parametern

Beispiel

```
public void print(String string) { ... }
public void print(int a) { ... }
public void print() { ... }
```

Überschreiben

Überschreiben

- Methode mit gleichem Namen und Parametern die in einer Unterklasse neu definiert wird

Beispiel

```
public class A {
    void print() { System.out.println("a"); }
}
public class B extends A {
    void print() { System.out.println("b"); }
}
A a = new A(); B b = new B();
a.print(); // a
b.print(): // b
```

UML

UML

- Unified Modeling Language
- Graphische Modellierungssprache
  - Spezifizieren
  - Konstruieren
  - Visualisieren
  - Dokumentieren

  von Softwaresystemen[a] in Form von Diagrammen

[a] Quelle: Konzeptionelle Modellierung, Vorlesungsfolie UML 1

Ziel

Exakte, eindeutige, unmissverständliche, umfassende Spezifikation des Problems.

Use-Case Diagramm

Use-Case Diagramm

- Funktionale Beschreibung des Systems
- Elemente: Akteure und Anwendungsfälle

- Akteure werden als Strichmännchen dargestellt
- Anwendungsfälle als Ovale
- Akteur und Anwendungsfälle werden durch Striche verbunden
- Anwendungsfälle finden innerhalb des Systems statt (als Rechteck dargestellt)

Use-Case Diagramm (Beispiel)

Das umzusetzende System soll es einem Passagier ermöglichen, einen (beliebigen aber noch nicht ausgebuchten) Flug zu buchen. Eine bereits erfolgte Buchung kann der Passagier bis 14 Tage vor dem Flug stornieren. Die Buchung kann der Passagier entweder über ein Webinterface selbst vornehmen bzw. stornieren oder telefonisch mit einem Call-Center-Agent unseres Unternehmens organisieren. Dazu muss er seine persönlichen Daten (Name, Geburtsdatum, Adresse, Kreditkartennummer oder Kontonummer) sowie die gewünschten Reisedaten angeben (Reisedatum und Flugnummer).
Unsere Verwaltungsangestellten drucken die Flugscheine eine Woche vor dem Flug aus, sofern der Zahlungseingang erfolgt ist, und versenden sie per Post an die Passagiere.
Die Reiseabwicklung erfolgt am Reisetag wie üblich zweistufig: Zunächst liefert der Passagier sein Gepäck am Check-In-Schalter ab, vereinbart dabei den Sitzplatz mit dem Check-In-Agent (diese Information muss natürlich auch im System gespeichert werden) und bekommt dafür sein Bordticket. Anschließend erfolgt das Boarding, bei dem der Kontrollabriss des Bordtickets vom Boarding-Agent eingescannt wird, so dass im System registriert werden kann, wer tatsächlich an Bord des Flugzeugs ging.

Substantivanalyse

Substantivanalyse

- Klassen beschreiben „Objekte“ in der realen Welt
- Substantivanalyse macht es einfacher potentielle Klassen zu finden
- nicht alle Substantive sind sinnvolle Klassen, z. B.
  - außerhalb des Systems
  - Ereignisse/Aktionen
  - zu allgemeine Substantive
  - mögliche Klassenvariablen
  - Aktoren die nicht zum Kontext passen
  - etc.
- zuerst so wenig Klassen wie möglich modellieren
- erweitern ist einfacher als Klassen zu streichen

Substantivanalyse (Beispiel)

Das umzusetzende System soll es einem Passagier ermöglichen, einen (beliebigen aber noch nicht ausgebuchten) Flug zu buchen. Eine bereits erfolgte Buchung kann der Passagier bis 14 Tage vor dem Flug stornieren. Die Buchung kann der Passagier entweder über ein Webinterface selbst vornehmen bzw. stornieren oder telefonisch mit einem Call-Center-Agent unseres Unternehmens organisieren. Dazu muss er seine persönlichen Daten (Name, Geburtsdatum, Adresse, Kreditkartennummer oder Kontonummer) sowie die gewünschten Reisedaten angeben (Reisedatum und Flugnummer).
Unsere Verwaltungsangestellten drucken die Flugscheine eine Woche vor dem Flug aus, sofern der Zahlungseingang erfolgt ist, und versenden sie per Post an die Passagiere.
Die Reiseabwicklung erfolgt am Reisetag wie üblich zweistufig: Zunächst liefert der Passagier sein Gepäck am Check-In-Schalter ab, vereinbart dabei den Sitzplatz mit dem Check-In-Agent (diese Information muss natürlich auch im System gespeichert werden) und bekommt dafür sein Bordticket. Anschließend erfolgt das Boarding, bei dem der Kontrollabriss des Bordtickets vom Boarding-Agent eingescannt wird, so dass im System registriert werden kann, wer tatsächlich an Bord des Flugzeugs ging.

Substantivanalyse (Beispiel)

Das umzusetzende System soll es einem **Passagier** ermöglichen, einen (beliebigen aber noch nicht ausgebuchten) **Flug** zu buchen. Eine bereits erfolgte **Buchung** kann der Passagier bis 14 Tage vor dem Flug stornieren. Die Buchung kann der Passagier entweder über ein **Webinterface** selbst vornehmen bzw. stornieren oder telefonisch mit einem Call-Center-Agent unseres Unternehmens organisieren. Dazu muss er seine persönlichen Daten (Name, Geburtsdatum, Adresse, Kreditkartennummer oder Kontonummer) sowie die gewünschten Reisedaten angeben (Reisedatum und Flugnummer).
Unsere Verwaltungsangestellten drucken die Flugscheine eine Woche vor dem Flug aus, sofern der Zahlungseingang erfolgt ist, und versenden sie per Post an die Passagiere.
Die Reiseabwicklung erfolgt am Reisetag wie üblich zweistufig: Zunächst liefert der Passagier sein Gepäck am Check-In-Schalter ab, vereinbart dabei den Sitzplatz mit dem Check-In-Agent (diese Information muss natürlich auch im System gespeichert werden) und bekommt dafür sein Bordticket. Anschließend erfolgt das Boarding, bei dem der Kontrollabriss des Bordtickets vom Boarding-Agent eingescannt wird, so dass im System registriert werden kann, wer tatsächlich an Bord des Flugzeugs ging.

CRC-Karte

CRC-Karte

beschreibt

- wofür die Klasse zuständig ist
- was die Klasse auslöst
- mit wem die Klasse zusammen arbeitet

Klassendiagramm

Klassendiagramm

- stellt Attribute, Eigenschaften, Methoden und Beziehungen einer Klasse dar
- Beziehungen zwischen Klassen sind Kanten
- Vererbungsbeziehungen werden durch Pfeile dargestellt
- Sichtbarkeitsmodifikatoren:
  - + public
  - ∼ default
  - # protected
  - − private
- *abstract*, «interface»(gestrichelter Pfeil), static

Klassendiagramm

Assoziationen

- zeigen Beziehungen zwischen Klassen
- werden durch einen Strich zwischen den Klassen visualisiert

Multiplizitäten

- werden zu Assoziationen hinzugefügt
- geben an wie viele Objekte an der Beziehung teilnehmen
- stehen jeweils am anderen Ende der Kante

Komposition

- geben einen Bestandteil an, der nicht unabhängig existieren kann
- werden durch eine gefüllte Raute dargestellt

Mathematische Funktionen

Mathematische Funktionen

- Funktionen in der Java-API beschrieben
- Konstante `Double.NaN` definiert ungültige Zahl („Not a Number“)
- Beispiele: $\sqrt{-1}$ oder $0.0/0.0$.

.jar-Dateien

.jar-Dateien

- komprimierte Verzeichnisstruktur (eigentlich zip-Datei)
- stellt Pakete bzw. Klassen zur Verfügung
- Implementierung der Klassen ist unbekannt
- API der Klassen bekannt, sie können direkt genutzt werden

Verwendung

```
$ javac -cp .:Core.jar CLC.java

$ java -cp .:Core.jar CLC add 23 12
```

Fragen

Fragen?