Einführung in die Programmierung

by André Karge
Übung - Packages & Recursion

letzte Woche

- Enum Typen & Random
- Referenzen
- Copy

diese Woche

- Besprechung Übungsblatt 5
- Packages
- Recursion
- Besprechung Übungsblatt 6

Übungsblatt 5

Übungsblatt 5

Aufgaben 1-3

Code Beispiel

Anmerkungen

Umlaute im Programmcode

- Bitte verwendet nirgendwo in eurem Programm Umlaute (vorallem nicht als Bezeichner von Variablen, Objekten oder Klassen)
- Wenn ihr trotzdem eure Variablen in Deutsch benennen wollt, nutzt die International verfügbaren Buchstaben des lateinischen Alphabetes: $\ddot{a} \rightarrow ae$, $\ddot{u} \rightarrow ue$, $\ddot{o} \rightarrow oe$, $ß \rightarrow ss$
- Wenn ihr eine andere Dateiformatierung verwendet, als derjenige, der euer Programm compiled, werden Umlaute falsch dargestellt und der Compiler bricht ab.

Anmerkungen

Konstruktoren

- Manche Gruppen verstehen noch nicht genau, wozu Konstruktoren verwendet werden
- Konstruktoren sind Methoden zum Initialisieren von Objekten

Leerer Konstruktor

```
public class Sheep {
  public Fur fur;
  public String name;
  public Sheep() {
    // leeren Konstruktor vermeiden
  }
  // ...
}
// ...
Sheep newSheep = new Sheep();
newSheep.fur = fur; // gehört in den Konstruktor
newSheep.name = name; // gehört in den Konstruktor
```

Anmerkungen

Redundante Zuweisung

```
public class Sheep {
  public Fur fur;
  public String name;
  public Sheep(Fur fur, String name) {
    this.fur = fur;
    this.name = name;
  }
  //...
}
//...
Sheep newSheep = new Sheep(fur, name);
newSheep.fur = fur; // redundant
newSheep.name = name; // redundant
```

Packages

Packages - Insel 3.6

- ermöglicht Kapselung
- muss in definiertem Schema geschrieben werden (siehe Vorlesung)
- Pakete können überall eingebunden werden (damit alle Klassen, die das Paket kapselt)
- Pakete und imports stehen immer am Anfang von Java-Dateien
- Stichwort *import*

Beispiel - Import

```
import java.util.Random; // lädt das Paket Random aus util
//import java.util.*; //lädt alle Pakete in util
public class MyClass {
  // attribute
  // methoden
}
```

Packages - Insel 3.6

Beispiel - Paketdefinition

```
package de.uni_weimar.medien.karge.proggen; //definiert, dass die folgende Klasse teil dieses Pakets ist
public class MyClass {
  public static void sayHello() {
    System.out.println("Hello");
  }
}
```

Beispiel - Einbinden des Pakets

```
import de.uni_weimar.medien.karge.proggen.MyClass; // Einbindung der Klasse
//import de.uni_weimar.medien.karge.proggen.*; // Einbindung des gesamten Packages

public class Test {
  public static void main(String[] args) {
    MyClass.sayHello(); // Aufruf der statischen Methode sayHello aus der Klasse im Paket
  }
}
```

Packages - Insel 3.6

Übungsaufgabe - (10min)

1. Erstellen Sie ein neues Projekt mit 2 Packages:
   de.uni_weimar.medien.proggen.application und *de.uni_weimar.medien.proggen.aufg1*
2. Erstellen Sie zwei Klassen im Paket *aufg1*: *Person* und *Guess*, mit jeweils einer statischen Methode
3. Erstellen Sie eine Klasse *Application* im Paket *application* mit einer main-Methode
4. Importieren Sie das Paket *aufg1* in der Datei der Klasse *Application* und testen Sie die statischen Methoden der importierten Klassen

Packages - Insel 3.6

Standard Packages

- Java bietet eine Hand voll Pakete, die Standardmäßig ausgeliefert werden
- Beispiele:
  - java.lang (fundamentale Java Klassen wie String)
  - java.util (Hilfsklassen wie Random)
  - java.io (System Klassen wie input und output Streams)
  - java.text (Klassen zur Arbeit mit Text)
- Eine vollständige Liste für alle Standardpaketen findet ihr hier: Link

Recursion

Recursion - Insel 2.7.14

Beispiel

Auf dem Weg durch den Wald treffen wir eine Fee, die uns 3 Wünsche gibt. Man könnte jetzt sich 3 Explizite Wünsche ausdenken. Damit wären die 3 Wünsche aber verbraucht.
Was wäre, wenn wir uns mit dem letzten Wunsch nochmal 3 Wünsche wünschen würden? Das könnten wir so lange machen, bis alle Wünsche der Welt erfüllt sind.

```
static void feenWunsch() {
  wunsch(); // Frieden auf der Welt
  wunsch(); // Beim Programmieren nicht mehr Semikolon vergessen
  feenWunsch(); // 3 Wünsche
}
```

Recursion - Insel 2.7.14

- Rekursion ist der Aufruf einer Methode in sich selbst
- kann die Laufzeit des Programmes verändern
- Kann bei zu tiefer Rekursion einen Stack Overflow hervorrufen

Recursion - Insel 2.7.14

Übungsaufgabe (15min)

Erweitern Sie Ihre Klasse Guess um Methoden, die eine von Ihnen gedachte Zahl zwischen 1 und 1000 erraten.
Dazu sollen die Methoden eine Zahl vorschlagen und Sie antworten mit $<$, $>$ oder $=$

1. Implementieren Sie eine Methode mit einer while-Schleife
2. Implementieren Sie eine Methode mit Recursion

Das Programm benötigt hier keinen Zufallwert. Überlegen Sie sich eine Taktik, mit der Sie möglichst schnell auf die Lösung kommen.

Übungsblatt 6

Fibonacci

Formel

$$f_n = f_{n-1} + f_{n-2} \text{ für } n > 2$$
$$f_1 = f_2 = 1$$

Beispiel

$$f_1 = 1 \tag{1}$$
$$f_2 = 1 \tag{2}$$
$$f_3 = f_2 + f_1 = 1 + 1 = 2 \tag{3}$$
$$f_4 = f_3 + f_2 = (f_2 + f_1) + f_2 = (1 + 1) + 1 = 3 \tag{4}$$
$$f_5 = f_4 + f_3 = 3 + 2 = 5 \tag{5}$$

Fragen?