University of Zurich[UZH]

s.e.a.l. software evolution & architecture lab

Informatik I – Eprog HS11

Übung 5

1 Aufgabe: Konstruktoren und statische Variablen

1.1 Lernziele

1. Sie können ein Java Projekt in Eclipse anlegen.
2. Sie können mehrere Werkzeuge, die Ihnen Eclipse bietet, effizient nutzen.
3. Sie können mehrere Konstruktoren in einer Klasse implementieren und diese gezielt aufrufen.
4. Sie kennen den Unterschied zwischen Instanz- und Klassenvariablen

1.2 Aufgabenstellung

1. Installieren Sie die IDE[1] Eclipse (wählen Sie Eclipse Classic).
2. Arbeiten Sie das zur Verfügung gestellte Eclipse-Tutorial durch.
3. Legen Sie ein neues Eclipse Projekt an und geben Sie ihm einen sinnvollen Namen.
4. Schreiben Sie eine Klasse `Employee` mit folgenden Attributen: `firstName`, `lastName`, `street`, `zip` und `residence`. Im Kontextmenü von Eclipse finden Sie den Eintrag *Source/Generate Getters and Setters*. Nutzen Sie diese Funktionalität, um die Accessor-Methoden zu implementieren.
5. Nutzen Sie ebenfalls *Source/Generate Constructor using fields*, um die Klasse um einen Konstruktor zu erweitern, welcher es ermöglicht, folgende Attribute bereits bei der Objekterzeugung mit zu initialisieren: `firstName`, `lastName`, `street`, `zip` und `residence`.
6. Erstellen Sie eine zusätzliche Instanzvariable `id` vom Typ `int`, inklusive Getter. Fügen Sie der Klasse eine Klassenvariable `numberOfCreatedEmployees` vom Typ `int` hinzu und initialisieren Sie diese mit 0. Erweitern Sie den Konstruktor aus 5. nun dahingehend, dass dieser den aktuellen Wert von `numberOfCreatedEmployees` der Instanzvariablen `id` zuweist und erstere Variable anschliessend um eins erhöht. So soll gewährleistet werden, dass jedes neue `Employee`-Objekt eine eindeutige Personalnummer erhält.
7. Schreiben Sie einen weiteren Konstruktor, welcher keine Argumente erwartet. Er soll den in 6. erweiterten Konstruktor mit „Dummy"-Werten aufrufen. So könnte für `lastName` der Wert „TestNachname", für `firstName` „TestVorname", etc. übergeben werden. Danach soll der Konstruktor zudem auf dem Bildschirm ausgeben, dass ein Testobjekt generiert wurde und welche Personalnummer dieses besitzt:

[1] *IDE* steht für *Integrated Development Environment*

Ein Testobjekt mit der Personalnummer 12 wurde erzeugt!

8. Implementieren Sie eine Methode `equals()`, welche als Argument ein `Employee`-Objekt erwartet. Die Methode soll die Personalnummern der beiden Angestellten vergleichen. Stimmen diese überein, so soll die Methode wahr zurückliefern, andernfalls falsch.
9. Implementieren/Überschreiben Sie die Methode `toString()`, sodass ein `Employee`-Objekt Information in folgender Form zurückgibt:

   Hans Muster
   Rämistrasse 74
   8001 Zürich
   Personalnummer: 15

10. Lassen Sie nun noch Eclipse Ihren Code formatieren, indem Sie *Source/Format* verwenden.
11. Schreiben Sie einen TestDriver. Verwenden Sie ein Template von Eclipse, um darin die `main()`-Methode zu erzeugen. Dazu schreiben Sie main, drücken anschliessen *CTRL+space* und wählen den obersten Eintrag in der eingeblendeten Liste. Eclipse erzeugt dadurch automatisch den Rumpf für die `main()`-Methode. Instanzieren Sie darin verschiedene `Employee`-Objekte und testen Sie insbesondere die `equals()`-Methode. Nutzen Sie „code completion“, indem Sie wiederum während des Schreibens *CTRL+space* drücken.
12. Wie Sie sicher gemerkt haben, erleichtern Ihnen die „code completion“ und die Templates das Programmieren erheblich. In den Einstellungen von Eclipse unter *Java/Editor/Templates* finden Sie die vordefinierten Templates. Stöbern Sie darin herum, um in Zukunft von diesen Gebrauch machen zu können.
13. Warum erzeugt folgendes Codefragment, wenn die Klasse `Employee` damit ergänzt wird, einen Fehler zur Übersetzungszeit? Begründen Sie Ihre Antwort und diskutieren Sie den Sinn der Methode.

```
1 public static int getID() {
2    return this.id;
3 }
```

Nehmen Sie Ihre `Employee`-Klasse als Java-File in die Übungsbesprechung mit!

2 Aufgabe: Vorlesung

2.1 Lernziele

1. Sie können Konstruktoren schreiben.
2. Sie wissen, wie man Methoden überladet.
3. Sie kennen den Unterschied zwischen Instanz- und Klassenvariablen.
4. Vertiefung der programmatischen Abbildung einer Problemstellung.

2.2 Ausgangslage

Für eine Vorlesung kann man sich einschreiben, jedoch nur solange es noch freie Plätze gibt. Vorlesungen sind durch einen Namen gekennzeichnet und sind durch eine Vorlesungsnummer eindeutig identifizierbar. Normalerweise bieten sie Platz für 100 Anmeldungen, jedoch kann der Dozent beim Eröffnen der Vorlesung eine andere maximale Teilnehmerzahl festlegen. Jedermann hat Einsicht in die Anzahl freier Plätze und ob man sich für die Vorlesung noch einschreiben kann. Auch Abmeldungen können eingehen.

a) Aufgabenstellung

Implementieren Sie obigen Sachverhalt in einer Klasse und testen Sie diese in einem Test-Driver. Versuchen Sie, die Aufgabe mit Konstruktoren, Overloading und statischen Variablen zu lösen.

3 Aufgabe: Chain of Words

3.1 Lernziele

1. Das Wissen über Referenzen vertiefen.
2. Die Verwendung von Loops wiederholen, insbesondere den Unterschied zwischen `while`- und `for`-Schleifen vertiefen.
3. `String`-Operationen wiederholen.

a) Aufgabenstellung: Die Elemente

1. Erstellen Sie eine Klasse namens `WordElement`.
2. Definieren Sie nun einen Konstruktor, welcher einen `String` als Parameter erwartet.
3. Deklarieren Sie nun eine private Instanzvariable des Typs `WordElement` und benennen sie diese mit `nextElement`.
4. Machen Sie dasselbe für eine zweite Variable namens `previousElement`.
5. Implementieren Sie Getter- und Setter-Methoden.
6. Fügen Sie desweiteren zwei Methoden hinzu, welche überprüfen, ob die jeweiligen Instanzvariablen mit einem Wert initialisiert wurden.

b) Aufgabenstellung: Die Kette

Im vorherigen Teil haben wir den Grundstein für unsere Kette gelegt: Die einzelnen Elemente verfügen über die Methoden, um sich aneinander zu ketten.

1. Erstellen Sie eine Klasse namens `Chain`.
2. Definieren Sie für diese Klasse eine private Instanzvariable namens `firstElement`.
3. Desweiteren muss eine Methode `addWord(String word)` definiert werden, welche keinen Wert zurückliefert. Diese Methode erstellt bei Aufruf jeweils eine neue Instanz der Klasse `WordElement` und übergibt dieser gleich das erhaltene Wort. Ist `firstElement == null`, so wird diese Variable mit dem neuen Objekt initialisiert. Ist dies nicht der Fall, so muss nun über eine `while`-Schleife das letzte Element der Wortkette ermittelt werden und danach das Objekt angefügt werden.
4. Erweitern Sie die Klasse dahingehend, dass eine Methode namens `getWordCount()` die Anzahl Wörter der Kette zurückgibt.
5. Ebenfalls muss eine Methode namens `getSentence()` erstellt werden, welche die gesamte Kette als `String` (mit Leerzeichen zwischen den Wörtern!) zurückgibt.
6. Erstellen Sie nun eine Methode `getReverse()`, welche bei folgendem Input:

   ```
   1 Chain myChain = new Chain();
   2 myChain.addWord("Anna");
   3 myChain.addWord("liebt");
   4 myChain.addWord("Thomas");
   ```

   diesen Ouptut in Form eines Strings zurückliefert: `Thomas liebt Anna`.
7. Erstellen Sie eine Methode namens `getEverySecond()` welche zuerst eine lokale Variable mit der aktuellen Anzahl Wörter initialisiert und danach mit einer `for`-Schleife jedes zweite Wort zu einem String zusammenhängt. z.B. Die Wörter `Ich lebe und arbeite in der Schweiz` sollten wie folgt ausgegeben werden: `lebe arbeite der`.
8. Testen Sie die Funktionalität in einem TestDriver.

4 Aufgabe: GUI – First Window

4.1 Lernziele

1. GUI-Elemente kennenlernen.
2. Bestehenden Code verwenden und anpassen können.

4.2 Aufgabenstellung

Verwenden Sie für diese Aufgabe die zur Verfügung gestellten Klassen-Files `Gui.java` und `GuiTest.java`.[2]

1. Importieren Sie die beiden Files in Eclipse und führen Sie den TestDriver aus.
2. Schauen Sie sich die Klasse `Gui` und deren `show()`-Methode an. Darin werden u.a. drei Hilfsmethoden aufgerufen, um Muster zu erzeugen: `drawRandomColorSquarePattern`, `drawGreekPattern` und `drawSmiley`. Das Ziel ist, diese so zu vervollständigen, dass in etwa dasselbe Bild (siehe unten) herauskommt.
3. In der `drawRandomColorSquarePattern`-Methode geht es darum, im durch die Übergabewerte vorgegebenen Quadrat für jedes Pixel eine zufällige Farbe zu setzen. Ein einzelnes Pixel (x, y) in einem `BufferedImage image` verändern Sie farblich wie folgt – r, g und b stehen dabei für die Rot-, Grün- und Blauanteile und können Werte von 0 bis 255 annehmen.

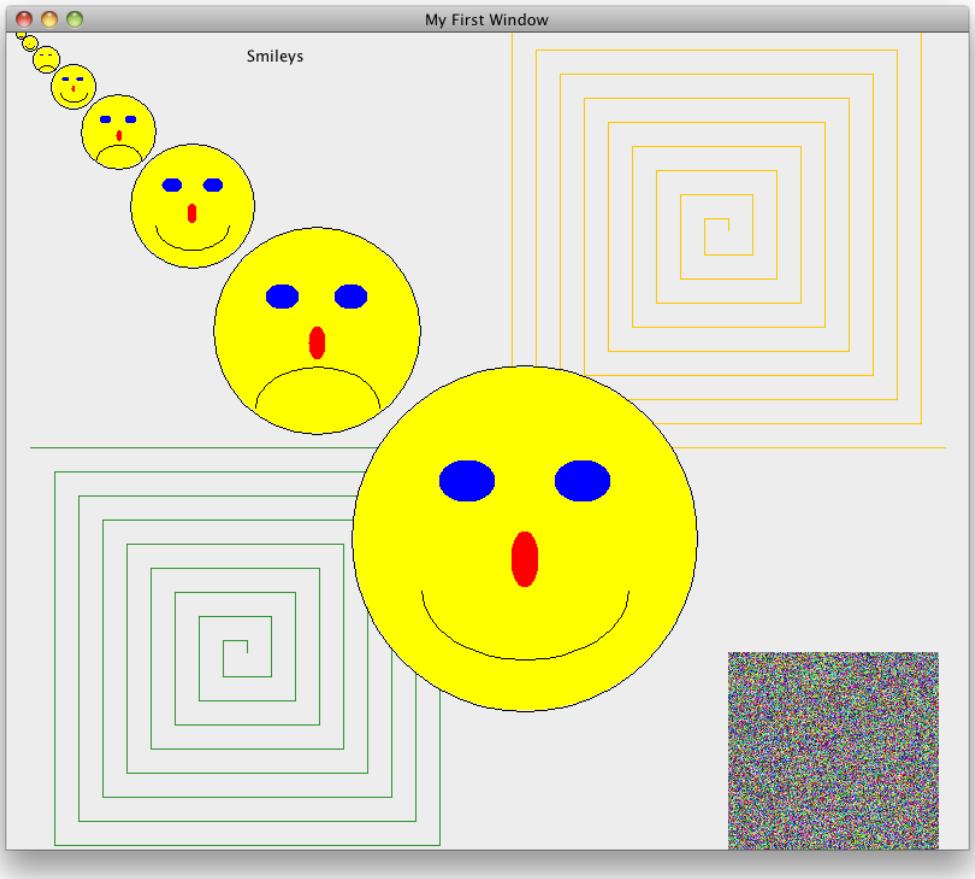
```
1  Color color = new Color(r, g, b);
2  image.setRGB(x, y, color.getRGB() );
```

4. Schauen Sie sich für die beiden anderen Methoden die `Graphics`-Klasse – insbesondere die im Methodenrumpf angegeben Methoden – in der Java-API an und vervollständigen Sie sie ebenfalls, indem Sie sich an existierendem Code orientieren. Statt Farben wie oben beschrieben selber zu definieren, können Sie auch bereits vorhandene Konstanten verwenden.

```
1  Color color = Color.BLACK;
2  graphics.setColor(color);
3  //oder direkt
4  graphics.setColor(Color.BLACK);
```

5. Probieren Sie weitere Dinge aus; Verändern Sie die `show`-Methode, um andere Muster zu erzeugen.

[2] Diese finden Sie entweder direkt auf der Übungsseite oder im OLAT.


My First Window
Smileys