Einführung in die Programmierung für NF

Übung 12

22.01.2014

Inhalt

- Korrektur Blatt 11
  - Teilnehmerliste
  - Interfaces
  - Exceptions

- Verwendung von Interfaces
- Die Spezialklasse „Enum“

Korrektur Blatt 11

- Teilnehmerliste
- ArrayList mit Strings im Model
- Vollständige Umsetzung von MVC- und Observer-Pattern
- Kommunikation zwischen View und Model nur über den Controller

Interfaces

- Nachdem wir nun Interfaces geschrieben haben und Klassen diese importieren haben lassen, stellt sich die Frage: wozu?
- Ein Grund für Interfaces ist die Austauschbarkeit der dahinter stehenden Klasse in anderen Klassen
- Zum Beispiel: Das Model wird im Controller durch ein anderes Model ersetzt

Interfaces

- Im Code des Controllers ersetzen wir dazu die folgende Zeile

```
public class Controller {

    public Controller(){

        Model model = new Model();
        View view = new View();
        model.addObserver(view);

    }
}
```

durch

```
public class Controller {

    public Controller(){

        IModel model = new Model();
        View view = new View();



    }
}
```

- Nun sind nur noch die Methoden aus dem Interface bekannt

Enum

- Ein Enum-Typ (von „Enumeration“) ist ein spezieller Datentyp für eine Variable, die eine vordefinierte Menge an Werten haben kann
- Diese Werte müssen fix definiert werden
- Solche Datentypen werden sehr häufig verwendet, weil in vielen Fällen Variablen nur eine bekannte Menge von Werten einnehmen können

Enum

- Beispiel: Ein Datentyp für Wochentage
- Mit Grunddatentypen ist nur eine Kodierung möglich, z.B. 0 für Montag und 7 für Sonntag
- Dies ist aber sehr unübersichtlich, kaum lesbar und sehr fehleranfällig
- Besser also: ein Datentyp mit den Werten
  MONTAG, DIENSTAG, MITTWOCH,
  DONNERSTAG, FREITAG, SAMSTAG, SONNTAG

Enum in Java

- Beispiel:

```
public enum Wochentag {
    MONTAG, DIENSTAG, MITTWOCH, DONNERSTAG, FREITAG, SAMSTAG, SONNTAG
}
```

… ja, das ist alles ;)

- Nun gibt es einen Datentyp „Wochentag“, der die definierten Werte einnehmen kann
- Diese werden üblicherweise groß geschrieben

Enum in Java

- Verwendung:

```
public class EnumTest {

    Wochentag tag1;
    Wochentag tag2;

    public EnumTest(){
        tag1 = Wochentag.MONTAG;
        tag2 = Wochentag.SONNTAG;
    }

}
```

- Der Datentyp wird also wie üblich verwendet
- Die Werte werden mit Punkt-Notation dargestellt: EnumName.WERT

Enum in Java

- Ausnahme: In switch-case-Blöcken werden die Werte ohne Punktnotation dargestellt

- Das erhöht zusätzlich die Lesbarkeit

```
public EnumTest() {
    Wochentag tag = Wochentag.MONTAG;

    switch (tag) {
    case MONTAG:
        System.out.println("Montage sind doof");
        break;

    case FREITAG:
        System.out.println("Freitage sind besser");
        break;

    case SAMSTAG:
    case SONNTAG:
        System.out.println("Wochenende ist toll");
        break;

    default:
        break;
    }

}
```

Enum in Java

- Aufgabe: Schreiben Sie einen Enum-Typ für die Speicherung von Belegungen von Spielfeldern in einem Schachspiel.
- Folgende Belegungen sind möglich:
  - Leeres Feld
  - Alle Schachfiguren in weiß
  - Alle Schachfiguren in schwarz

Enum in Java

- Enum-Typen können auch mehr enthalten, z.B. weitere Daten pro Wert oder spezielle Funktionen, die mit diesen Werten rechnen
- Im Rahmen dieser Vorlesung genügt jedoch die Kenntnis über den Enum-Typ und seinen praktischen Nutzen

Fragen zum Übungsblatt?