universität
innsbruck

Aufzählungstypen

Programmiermethodik

Lukas Kaltenbrunner, Simon Priller
Universität Innsbruck

Motivation

- Variable sollte nur einen Wert aus einer Menge vordefinierter Werte annehmen können.
  - Beispiel: Farbvariable, welche die Werte Rot, Grün und Blau besitzen kann.
- Simpler Ansatz

```
…
private static final int RED = 0, GREEN = 1, BLUE = 2;
…
```

  - Compiler kann dabei aber nicht garantieren, dass einer Farbvariable nur Farbwerte zugewiesen werden.

```
…
int color = RED;
…
color = 100;
…
```

Falsch!
Keiner der vordefinierten Werte!

- Typsicherheit ist nicht gegeben!

Aufzählungstypen

- Aufzählungstypen werden auch Enumerationstypen oder kurz Enums genannt.
- Ein Aufzählungstyp definiert die Menge seiner Werte (Aufzählungskonstanten) durch namentliche Aufzählung.
- Jede Aufzählungskonstante definiert ein Exemplar des Enumerationstyps.
  - Es existieren neben den Aufzählungskonstanten keine weiteren Exemplare eines Enumertationstyps.
  - Der Versuch weitere Exemplare zu erzeugen führt zu einem Kompilierfehler.
- Beispiel

```
enum Color {RED, GREEN, BLUE}
enum Direction {NORTH, SOUTH, WEST, EAST}
Color c = Color.RED;
Direction d = Direction.NORTH;
// c = 100; // compilation error (incompatible types)
// c = d;   // compilation error (incompatible types)
…
```

Aufzählungstypen als Klassen

- Aufzählungstypen sind spezielle Klassen.
- Aufzählungstypen können beispielsweise auch Objektvariablen, Methoden und Konstruktoren enthalten.
- Es können keine neuen Exemplare durch `new` erzeugt werden.
- Konstruktoren sind bei Aufzählungstypen immer `private`.
  - Der Versuch einen Konstruktor als `public` oder `protected` zu deklarieren führt zu einem Kompilierfehler.
- Das Laufzeitsystem stellt sicher, dass Aufzählungskonstanten nicht kopiert werden (es existiert nur ein Exemplar von jedem Wert).
  - Kopieren der Referenz auf eine Aufzählungskonstante ist möglich.

Aufzählungskonstanten

- Können mit == und != verglichen werden.
- Sind in der Reihenfolge ihrer Deklaration geordnet.
  - Compiler ordnet jeder Aufzählungskonstante eine Ordinalzahl zu.
  - Aufzählungskonstanten sind aber keine Zahlen.
  - Im Farbbeispiel
    - `Color.RED` hat den Wert 0
    - `Color.GREEN` den Wert 1
    - `Color.BLUE` den Wert 2
- Können als `case`-Labels in `switch`-Anweisungen und `switch`-Ausdrücken verwendet werden.

```
…
switch(c) {
    case RED: …
    case GREEN: …
    case BLUE: …
}
```

Auszug bereitgestellter Methoden

- Aufzählungskonstanten bieten
  - `name`: liefert den exakten Name der Aufzählungskonstante
  - `ordinal`: liefert die Ordinalzahl der Aufzählungskonstante
  - `toString`: liefert den Name der Aufzählungskonstante
    - Kann eine lesbarere Repräsentation des Namen liefern

```
…
Color c = Color.GREEN;
System.out.println(c.name());      // GREEN
System.out.println(c.ordinal());   // 1
System.out.println(c.toString()); // GREEN
…
```

- Aufzählungstypen bieten
  - *values*: liefert ein Array zurück, das alle Aufzählungskonstanten des Enums enthält.

Beispiel (1)

```
public enum Roman {

    I(1), V(5), X(10), L(50), C(100), D(500), M(1000);

    private final int value;

    Roman(int value) {
        this.value = value;
    }

    public int getValue() {
        return value;
    }
}
```

Konstruktoraufruf

Konstruktoren sind bei Aufzählungstypen implizit `private`

Beispiel (2)

```
public class RomanApplication {

    public static void main(String[] args) {
        Roman r = Roman.V;
        System.out.println(r.getValue());

        for (Roman x : Roman.values()){
            System.out.print(x + " ");
        }
    }
}
```

```
Ausgabe:
5
I V X L C D M
```

Quellen

- Christian Ullenboom: **Java ist auch eine Insel: Einführung, Ausbildung, Praxis**, Rheinwerk Verlag, 16. Auflage, 2022 (Java 17)
- Joachim Goll, Cornelia Heinisch: **Java als erste Programmiersprache**, Springer Vieweg, 8. Auflage, 2016
- James Gosling, Bill Joy, Guy Steele, Gilad Bracha, Alex Buckley, Daniel Smith, Gavin Bierman: **The Java® Language Specification (***Java SE 17 Edition***)**, Oracle, 2021