



Java Essentials

Hoofdstuk 11

De opsomming

**DE HOGESCHOOL
MET HET NETWERK**

Hogeschool PXL – Elfde-Liniestraat 24 – B-3500 Hasselt
www.pxl.be - www.pxl.be/facebook



Inhoud

1. Inleiding
2. Definitie van het opsommingstype
3. Eigenschappen, methoden en constructors
4. Samenvatting



1. Inleiding

Het opsommingstype

- = klasse waarvan beperkt aantal instanties beschikbaar zijn
- = deze instanties worden tijdens de declaratie van de enumeratie opgesomd (vandaar de naam enumeration of opsomming)
 - => we kunnen zelf geen instanties aanmaken
- = deze instanties worden als constanten gebruikt
 - => met hoofdletters weergegeven
- = deze instanties kan men beschouwen als statistische eigenschappen van de klasse



Voorbeeld: klasse Colors

Oplossing zonder het opsommingstype

```
package examples.enumerations;

public class Colors {
    public final static int BLACK = 0;
    public final static int WHITE = 1;
    public final static int RED = 2;
    public final static int GREEN = 3;
    public final static int BLUE = 4;
    public final static int YELLOW = 5;

    public static void main(String[] args) {
        int color = GREEN;
        printColor(color);
    }

    public static void printColor(int color) {
        String text = null;
        switch (color) {
            case BLACK: text = "Black"; break;
            case WHITE: text = "White"; break;
            case RED: text = "Red"; break;
            case GREEN: text = "Green"; break;
            case BLUE: text = "Blue"; break;
            case YELLOW: text = "Yellow"; break;
        }
        System.out.println(text);
    }
}
```

Enkele nadelen:

- **onveilig:**
variabele color kan andere waarden aannemen
- **onduidelijk:**
afdruk van variabele color
=> primitieve waarde



2. Definitie van het opsommingstype

Definitie

```
public enum Color {  
    BLACK, WHITE, RED, GREEN, BLUE, YELLOW;  
}
```

toegelaten instanties

Gebruik

```
package examples.enumerations;
```

```
public class ColorApp {  
    public static void main(String[] args) {  
        Color color1 = Color.RED;  
        Color color2 = Color.GREEN;  
        printColor(color1);  
        printColor(color2);  
    }
```

gebruik vergelijkbaar
met static members
van een klasse

```
    public static void printColor(Color color) {  
        String text = null;  
        switch (color) {  
            case BLACK: text = "Black"; break;  
            case WHITE: text = "White"; break;  
            case RED: text = "Red"; break;  
            case GREEN: text = "Green"; break;  
            case BLUE: text = "Blue"; break;  
            case YELLOW: text = "Yellow"; break;  
        }  
        System.out.println(text);  
    }
```

alleen een instantie
die in de definitie
van de enum
voorkomt

Nuttige methodes

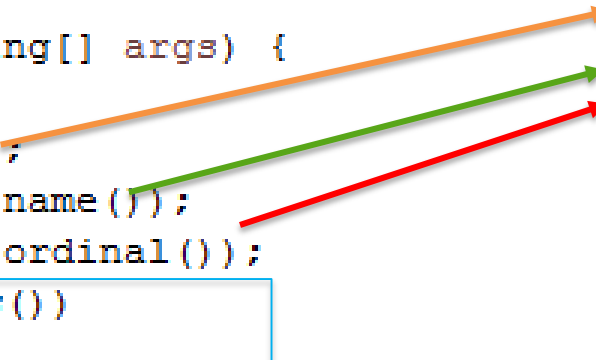
```
package examples.enumerations;
```

```
public enum Color {  
    BLACK, WHITE, RED, GREEN, BLUE, YELLOW;  
}
```

```
package examples.enumerations;
```

```
public class ColorAppBis {  
    public static void main(String[] args) {  
        Color color1 = Color.RED;  
        System.out.println(color1);  
        System.out.println(color1.name());  
        System.out.println(color1.ordinal());  
        for (Color c : Color.values())  
            System.out.println(c);  
    }  
}
```

output



```
RED  
RED  
2  
BLACK  
WHITE  
RED  
GREEN  
BLUE  
YELLOW
```

Opmerkingen

- elke enum is een subklasse van de abstracte klasse Enum => zie Javadoc voor methodes
- waarden van opsommingstype vaak nodig?
import static examples.enumerations.Color.*;
Color color1 = RED;

ipv Color color1 = Color.RED;



3. Eigenschappen, methoden en constructors

- Net als bij alle andere klassen beschikken opsommingen over eigenschappen, methoden en constructors.
- Het opsommingstype erft (inherit) eigenschappen en methoden over van zijn superklasse Enum.
- Eigenschappen en methoden kunnen toegevoegd en vervangen (override) worden.



```
public enum Color {  
    BLACK(0x000000),  
    WHITE(0xFFFFFFFF),  
    RED(0xFF0000),  
    GREEN(0x00FF00),  
    BLUE(0x0000FF),  
    YELLOW(0xFFFF00);  
  
    private int rgb;
```

```
    private Color(int rgb) {  
        this.rgb = rgb;  
    }
```

```
    public int getRgb() {  
        return rgb;  
    }
```

```
    public String toString() {  
        return name() + "(" + Integer.toHexString(rgb) + ")";  
    }  
}
```



Constructor kan nooit afzonderlijk opgeroepen worden.
Enkel in opsomming van de elementen

Geen constructor gedefinieerd
=> compiler neemt default constructor

In opsomming mogen ronde haken weggelaten worden.

```
public enum Color {  
    BLACK, WHITE, RED, GREEN, BLUE, YELLOW;  
}
```

is equivalent met

```
public enum Color {  
    BLACK(), WHITE(), RED(), GREEN(), BLUE(), YELLOW();  
  
    private Color() {  
    }  
}
```



```

public class ColorApp {

    public static void main(String[] args) {
        Color color1 = Color.RED;
        Color color2 = Color.GREEN;
        printColor(color1);
        printColor(color2);
    }

    public static void printColor(Color color) {
        System.out.println(color.name());
        System.out.println(color.ordinal());
        System.out.println(color.getRgb());
        System.out.println(color);
        String text = null;
        switch (color) {
            case BLACK: text = "Black"; break;
            case WHITE: text = "White"; break;
            case RED:    text = "Red";   break;
            case GREEN: text = "Green"; break;
            case BLUE:   text = "Blue";  break;
            case YELLOW: text = "Yellow"; break;
        }
        System.out.println(text);
    }
}

```

Methoden van het
opsommingstype
Color

Methoden
overgeërfd van
klasse Enum



Voor een element uit de opsomming kunnen bepaalde methoden vervangen worden

```
public enum Color {  
    BLACK(0x000000) {  
        public String toString() {  
            return "Black";  
        }  
    },  
    WHITE(0xFFFFFFFF),  
    RED(0xFF0000),  
    GREEN(0x00FF00),  
    BLUE(0x0000FF),  
    YELLOW(0xFFFF00);  
  
    private int rgb;  
  
    private Color(int rgb) {  
        this.rgb = rgb;  
    }  
  
    public int getRgb() {  
        return rgb;  
    }  
  
    public String toString() {  
        return name() + "(" + Integer.toHexString(rgb) + ")";  
    }  
}
```

Methode toString voor de instantie Black vervangen



Opdracht

- Maak een opsommingstype Dag met de dagen van de week.
- Voeg een private variabele toe waarin aangegeven wordt of dit een dag in de week is of dag in het weekend.
- Maak een hoofdprogramma dat alle dagen van de week overloopt en de naam, de ordinale waarde en weekenddag of weekday afdrukt.



4. Samenvatting

Het opsommingstype of enumeratie

- = speciale klasse waarvan er een beperkt aantal vooraf gedefinieerde instanties bestaan
- = deze instanties worden meestal als constanten gebruikt

