



Programmeren 3

Lesmateriaal

Moodle-course:

- "1920 Inf1.3S Databases"

Boek:

- *geen fysiek boek (op Moodle staan verwijzingen)*

Opdrachten:

- Wekelijks, individueel
- Verplicht, steeds volgende les afvinken
(vragen/feedback in de les)

Lessen en toetsing

Lessen:

- Programmeren 3 theorie (theorie les, alle klassen samen)
- Programmeren 3 praktijk (praktijk les, elke klas apart)

Toetsing:

- praktijk tentamen (opdrachten op computer)
- wekelijkse opdrachten

Programmeer paradigma's

- Er zijn verschillende concepten van programmeren

- Imperatief programmeren (o.a. C, Pascal)

uitvoeren van commando's, toestand

```
printf("Hello World!\n");
```

- **Object-georiënteerd programmeren** (o.a. C#, Java)

werken met objecten

```
class Fiets : Voertuig { ... }
```

- Functioneel programmeren (o.a. Haskell, ML)

definieren / uitvoeren van functies

```
> list1 = [1, 2, 3, 4, 5]  
> filter odd list1
```

- Logisch programmeren (o.a. Prolog)

```
ouder_van(julia, augustus)  
vrouw(julia)  
moeder_van(X,Y) :- ouder_van(X, Y), vrouw(X)
```

Object Oriëntatie

- Programma bestaat uit objecten die gezamenlijk de verantwoordelijkheid dragen voor de uitvoering van het programma
- Object is een instantie van een class
- Het gaat dus om verantwoordelijkheden

Classes

- Een class bevat data en functionaliteit
- Classes worden gedeclareerd door de keyword class, gevolgd door de class name en een verzameling van "class members" omsloten door { accolades }

```
class Employee  
{  
    // ...  
}
```

- Afspraak: De class name begint met een Hoofdletter

Classes - class members

■ Fields (data) en methods (code)

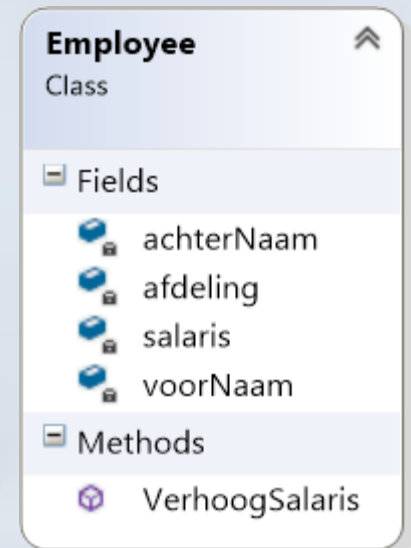
```
class Employee
{
    // fields
    string voorNaam, achterNaam;
    string afdeling;
    float salaris;

    // method
    public void VerhoogSalaris(float verhoging)
    {
        salaris = salaris + verhoging;
    }
}
```

Een class bevat
'fields' en 'methods'
die bij elkaar horen.

Elke 'Employee' heeft een
voornaam, achternaam, afdeling
en salaris, en het salaris kunnen
we verhogen.

(class diagram)



Classes - instanties

- Een class is een blauwdruk / sjabloon waarvan objecten gemaakt kunnen worden
- Via keyword new kan een instantie / object aangemaakt worden

```
// maak instanties (objecten) aan van de class Employee
Employee employee1 = new Employee();
Employee employee2 = new Employee();

// verhoog salaris van employee1
employee1.VerhoogSalaris(100);
```


- Constructor is een methode die wordt aangeroepen bij het aanmaken van een object; naam v/d methode = class naam
- Geen constructor: default constructor
(verdwijnt als er een parameter-constructor is...)
- Een class kan meerdere constructors bevatten (met verschillende parameters)

Classes - constructors

(2/4)

Eén constructor

```
class Employee
{
    // fields
    string voorNaam, achterNaam;
    string afdeling;
    float salaris;

    public Employee()
    {
        voorNaam = "";
        achterNaam = "";
    }

    // method
    public void VerhoogSalaris(float verhoging)
    {
        salaris = salaris + verhoging;
    }
}
```

Classes - constructors

(3/4)

Meerdere constructors

```
class Employee
{
    // fields
    string voorNaam, achterNaam;

    public Employee()
    {
        voorNaam = "";
        achterNaam = "";
    }

    public Employee(string voorNaam, string achterNaam)
    {
        this.voorNaam = voorNaam;
        this.achterNaam = achterNaam;
    }
}
```

Classes - constructors

(4/4)

■ Instanties aanmaken

```
// maak instanties (objecten) aan van Employee
Employee emp11 = new Employee();
Employee emp12 = new Employee();
```

▲ 2 of 2 ▼ Employee.Employee(string vn, string an)

```
class Employee
{
    // fields
    string voorNaam, achterNaam;

    public Employee()
    {
        voorNaam = "";
        achterNaam = "";
    }

    public Employee(string vn, string an)
    {
        voorNaam = vn;
        achterNaam = an;
    }
}
```

Welke constructor bij welke instantie?


```
// maak instanties (objecten) aan van Employee
Employee emp11 = new Employee();
Employee emp12 = new Employee("Piet", "Paulusma");
```

What's this?

- In methoden van een object kun je keyword 'this' gebruiken
- Met 'this' verwijst je naar het object zelf
- Het wordt o.a. gebruikt om onderscheid te maken tussen object-velden en parameters van een methode

```
class Player
{
    public string name;

    public Player(string name)
    {
        name = name;
    }
}
```

 (parameter) string name

Assignment made to same variable; did you mean to assign something else?

```
class Player
{
    public string name;

    public Player(string name)
    {
        this.name = name;
    }
}
```

Classes - destructors

(1/2)

- Destructor is een methode die wordt aangeroepen bij het opruimen van een object; naam v/d methode = `~class naam`
- één destructor mogelijk, zonder parameters
- Garbage collector -> opruimen kan ff duren...

Classes - destructors

(2/2)

■ Voorbeeld

Wanneer wordt de destructor aangeroepen?

```
// maak instanties (objecten) aan van Employee
Employee empl1 = new Employee();
Employee empl2 = new Employee();

// ...

// ruim employees op
empl1 = null;
empl2 = null;
```

```
class Employee
{
    // fields
    string voorNaam, achterNaam;

    // constructor
    public Employee()
    {
        voorNaam = "";
        achterNaam = "";
    }

    // destructor
    ~Employee()
    {
        voorNaam = "";
        achterNaam = "";
    }
}
```

Of zodra object 'out of scope' raakt (geen referenties meer heeft)...

Classes - instance vs static

- Instance class members behoren bij een specifieke instantie / object. Elk object heeft zijn eigen gegevens (status van een object)
(*voorbeeld: voornaam, achternaam van Employee*)
- Static class members behoren bij de class
- System.Math

```
int maxWaarde = Math.Max(getal1, getal2);
int geheugenGrootte = (int)Math.Pow(2, 16);
```
- OO: GEEN static gebruiken, tenzij ...

Classes - static members

■ Voorbeeld

```
class TemperatureConvertor
{
    public static double CelsiusToFahrenheit(double celsius)
    {
        // convert Celsius to Fahrenheit
        double fahrenheit = (celsius * 9 / 5) + 32;

        return fahrenheit;
    }
}
```

```
double celsius = 24.6;
double fahrenheit = TemperatureConvertor.CelsiusToFahrenheit(celsius);
double celsius2 = TemperatureConvertor.FahrenheitToCelsius(fahrenheit);

if (celsius != celsius2)
{
    // ...
}
```

Huiswerk voor volgende week

- Bestudeer de aangegeven paragrafen uit het 'Yellow Book' (zie Moodle)
- Week 1 opdrachten (zie Moodle)