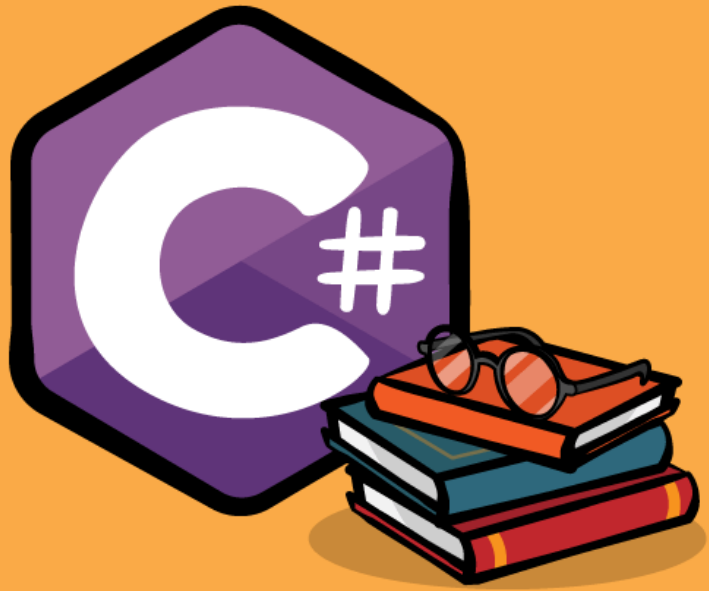
Windesheim



Classes, methods en properties

Weekplanning

Windesheim

Week	Onderwerp
1	**A-mazing**
2	.NET, Visual Studio, Solutions, Classes, objects, methods, properties
3	Arrays, types, Interfaces, Generics, polymorphism
4	Exception handling en Unittesting
5	Delegates en events
6	Linq introductie, Functioneel programmeren en lambda expressies
7	**Geen college**
8	WPF, GDI en bespreken oefenpracticum
9	**TOETSPRACTICUM**

Week 2b

Windesheim

- Instance variables en properties
- Methods, overriding en overloading
- Constructors
  - default constructors,
  - constructor chaining
- Static
- Casting

~~Getters and setters~~

Windesheim

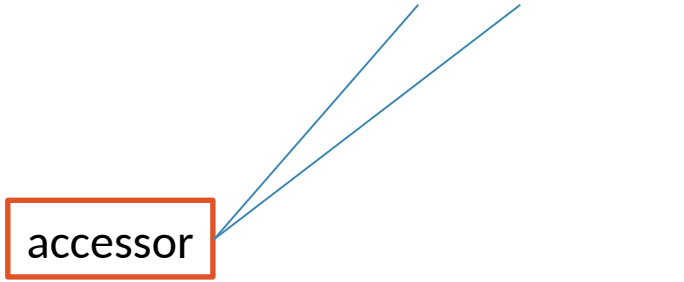
```
class Boot
{
    private int snelheid;

    public void SetSnelheid(int snelheid)
    {
        this.snelheid = snelheid;
    }

    public int GetSnelheid()
    {
        return snelheid;
    }
}
```

Veel code voor één instance variabele.

C# oplossing: Properties

Windesheim

```
class Boot
{
    public int Snelheid { get; set; }
}
```



Oefening properties

- Voeg aan de Solution een console-project toe met de naam Watersport.
- Schrijf een klasse Boot
- Type in de klasse Boot ‘prop’, gevolgd door twee tabs. Maak zo de property snelheid (int) en naam (string). Spring steeds verder met de tab-toets.
- Maak in de klasse Program een instantie van Boot (snelheid: 40, naam: ‘Druppie’) en print de eigenschappen van de boot uit.

Data validatie en properties

Windesheim

```
class Boot
{
    private int snelheid;

    public int Snelheid
    {

        set
        {
            if (value < 0)
            {
                throw new ArgumentOutOfRangeException($"{nameof(value)} moet hoger zijn dan 0");
            }
            snelheid = value;
        }

    }
}
```

Oefening accessoren

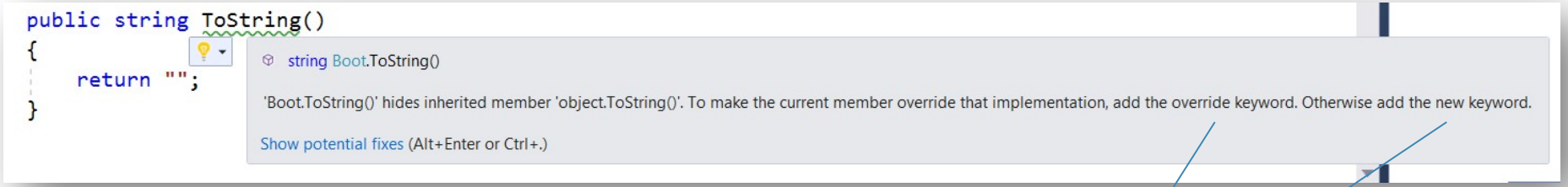
- Geef de property Naam een get accessor die de naam als volgt retourneert: ‘De naam is: <naam>’.

Extra:

- Geef de property Naam een set accessor die een controle uitvoert op de lengte van de naam. De lengte mag niet korter zijn dan drie tekens. Bovendien mogen er geen twee spaties achter elkaar zitten in de naam.

Method overriding

Windesheim



Bewust overriden:
override, als het is toegestaan
new, om het te forceren

Oefening overriden

Windesheim

- Voeg aan de klasse Boot een methode Sturen toe, met de eigenschappen:
  - Returntype: void
  - Parameter: richting (string)
  - Maak de methode Overridable
  - Output/printen: ‘De boot stuurt naar <richting>’. Maak gebruik van string interpolation

- Maak een klasse Speedboot en override de methode Sturen. De output van de methode Sturen is: ‘De boot stuurt *heel snel* naar <richting>’

Default constructor

Windesheim

```
class Boot
{

}
```

Geen constructor geschreven, dan een default constructor zonder parameters gegenereerd bij compilatie.

Constructor verplicht

Windesheim

```
class Boot
{
    public string Naam { get; set; }

    public Boot(string naam)
    {
        Naam = naam;
    }
}
```

```
class Speedboot : Boot
{

}
```

```
class Speedboot : Boot
{
    public Speedboot(string naam) : base(naam)
    {
    }
}
```

Constructor chaining

Windesheim

```
class Speedboot : Boot
{

    public Speedboot(string naam, int snelheid): base(naam)
    {
        Snelheid = snelheid;
    }

    public Speedboot() : this("geen naam", 0){}

    public Speedboot(string naam) : base(naam){}
}
```

```
class Boot
{
    public string Naam { get

    public Boot(string naam)
    {
```

Oefening constructors

Windesheim

- Schrijf een klasse Zeilboot die erft van Boot
- Geef de klasse drie constructoren:

  Let op:

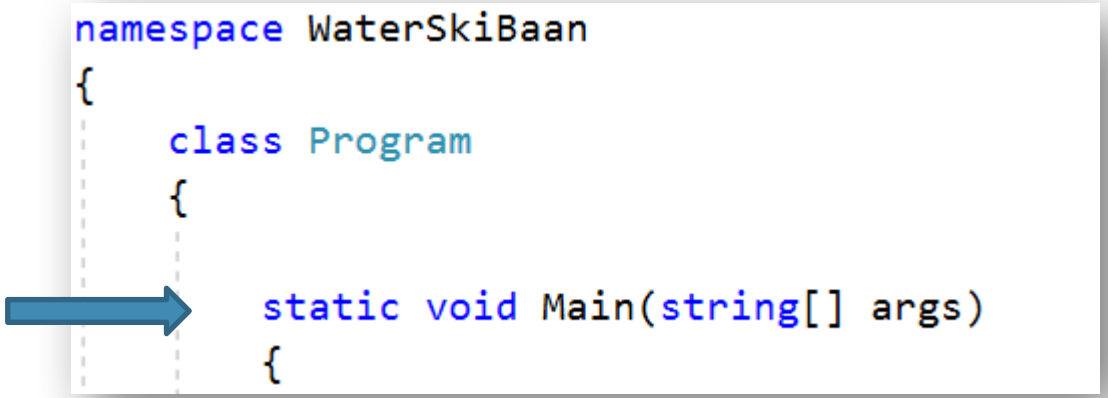
  - Er is maar één constructor die base mag gebruiken!

```
public Zeilboot(string naam)
public Zeilboot(int snelheid)
public Zeilboot(int snelheid, int zeiloppervlak) //maak voor de opslag van de waarde zeiloppervlak een property
```

Static

Windesheim

- static eigenschappen horen bij een class, niet bij een instantie
- static toepasbaar op classes, fields, methods, properties, operators, events, and constructors



Casting

Windesheim

```
class Animal { }

class Olifant : Animal { }

class Leeuw : Animal { }

class Program
{
    static void Main(string[] args)
    {
        Olifant o = new Olifant();
        Leeuw l = new Leeuw();

        Animal a = o;

        Olifant o2 = a;

        Olifant o3 = (Olifant)a;

        Leeuw oliLeeuw = o;

        Leeuw oliLeeuw2 = (Leeuw)o;

    }
}
```

Toegestaan, Olifant o is een Animal

Niet toegestaan, het is niet zeker dat Animal a een Olifant is

Expliciet aangeven dat a een Olifant is, **Explicit Casting**

Niet verbonden in rechtstreekse lijn van inheritance

Explicit Casting mag dus niet

Implicit cast:
int i = 10;
double d = i;

Object initializers

Windesheim

```
class Olifant {
    public int LengteSlurf { get; set; }
}

class Program
{
    static void Main(string[] args)
    {
        Olifant o = new Olifant();
        o.LengteSlurf = 10;

        Olifant m = new Olifant() { LengteSlurf = 20 };

        List<Olifant> reservaat = new List<Olifant>()
        {
            new Olifant(){LengteSlurf = 10 },
            new Olifant(){LengteSlurf = 20 },
            new Olifant(){LengteSlurf = 30 }
        };
```

Vooral erg handig bij het vullen van een lijst!

Huiswerk

Windesheim

- Race simulator