



# Workshop Dev3

Door Raoul, Hayder en Dion  
Peercoaches



# Onderwerpen

- Datatypes
- Methodes en functies
- Classes
- Arrays
- Patroonherkenning

# Datatypes

- Dynamische en Statische talen.
  - Python
  - C# en Java

# Datatypes

- Standaard datatypes:

- Void (void)
- String (string)
- Boolean (bool)
- Integer (int)
- Float (float)

- Special data types:

- Arrays
- Classes

# Datatypes

## Implicit casting

- $\text{int} + \text{int} = \text{int}$
- $\text{double} + \text{double} = \text{double}$
- $\text{float} + \text{float} = \text{float}$
- $\text{int} + \text{double} = \text{double}$
- $\text{int} + \text{float} = \text{float}$
- $\text{float} + \text{double} = \text{double}$
- Omdat je dezelfde type nummers altijd kan optellen.
- Het datatype met nummers dat het meest precies is, wordt gekozen.

# Datatypes

## Explicit casting

- $\text{int} + (\text{double}) \text{int} = \text{double}$
- $\text{double} + (\text{int}) \text{int} = \text{int}$
- Je kan een deel van een waarde verliezen.
- Bijvoorbeeld:
  - $13 + (\text{int}) 16,23 = 29$

# Datatypes

## Oefeningen

- $\{ a := \text{Func}\langle C, \text{int}[] \rangle [ ] [], i := \text{int}, j := \text{int}, l := \text{int}, x := C, C := \dots \}$
- $a[i][j](x)[l]$
- $\text{int}$

# Datatypes

## Oefeningen

- $\{ x := \text{string}, z := C, y := \text{Func}\langle C, \text{string} \rangle, C := \dots \}$
- $x \Rightarrow y \Rightarrow z \Rightarrow x + y(z)$
- $\text{Func}\langle \text{string}, \text{Func}\langle \text{Func}\langle C, \text{string} \rangle, \text{Func}\langle C, \text{string} \rangle \rangle \rangle$





# Methodes en functies

- Methodes kunnen hetzelfde als functies.
- Methodes zijn onderdeel van een class.

# Methodes en functies

- Closure
  - Het gebruiken van variable buiten de functie variable om.

# Methodes en functies

- Voorbeeld closure:

```
int z = 5
```

```
Func<int, int> f = x => x + z;
```

```
var k = f(10);
```

- Antwoord: 15

```
int z = 5
```

```
int f(int x)
```

```
{
```

```
    return x + z;
```

```
}
```

# Methodes en functies

- Currying
  - Dat is een techniek waarbij je functies terug geeft uit andere functies. Dit veroorzaakt een soort ketting aan functies.

# Methodes en functies

- Voorbeeld currying:

```
Func<int, Func<int, int>> f = x => z => x + z;
```

```
var k = f(10);
```

```
var y = k(90);
```

Antwoord: 100

```
Func<int, int> f(int x)
{
    int u(int z)
    {
        return x + z;
    }
    return u;
}
```



# Methodes en functies

Grande omega

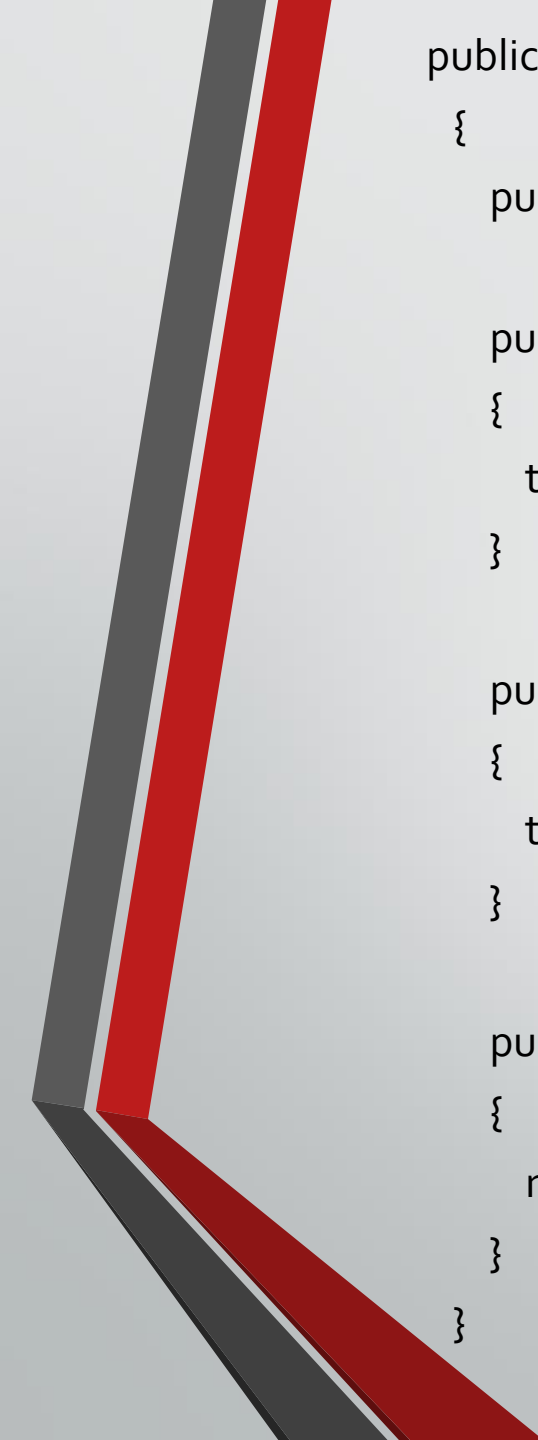
# Classes

- Een class is een datatype.
- Wat zit er in een class?
  - Constructor
  - Velden (fields) met informatie
    - Variabelen die weer een bepaald datatype hebben.
  - Het gedrag van de class.
    - Methodes.

# Classes

- In de volgende voorbeelden gaan we uit van een class dat hier na wordt gedefinieerd.





```
public class intValue
{
    public int value;

    public intValue()
    {
        this.value = 0;
    }

    public void add(int valueToAdd)
    {
        this.value = this.value+ valueToAdd;
    }

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




# Classes

```
var variable = new intValue();  
variable.add(10);  
var integerValue = variable.getValue();
```



# Classes

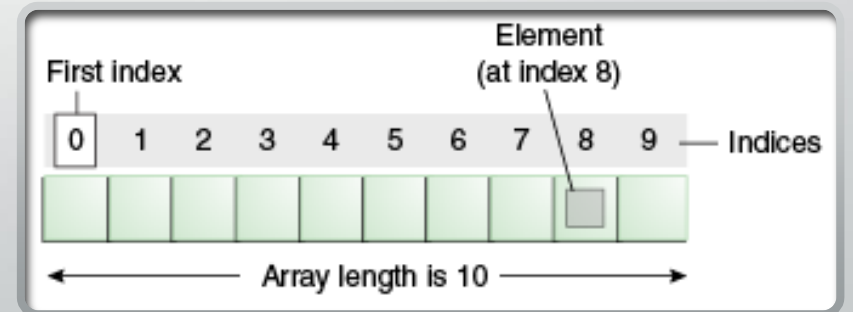
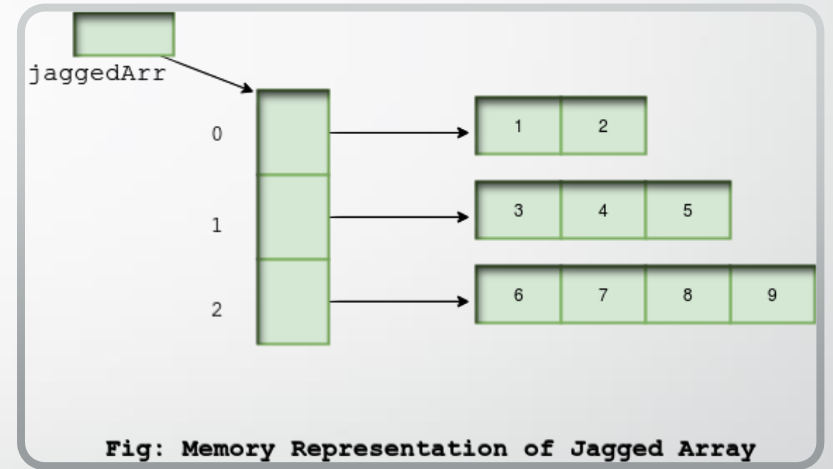
```
var variable = new intValue();  
variable.add(13);  
variable.add(3);  
var doubleValue = (double) variable.getValue();
```



Arrays en loops

# Arrays

- Wat is een array?
- Index
- Hoe maak je een nieuwe array aan?



# Loops

- De standaardmethode: for-loops
- Opbouw van een for-loop

# Grenzen

- Begin en eindwaarde van bijvoorbeeld  $i$  (index)
- Booleaanse operatoren

# Aan de slag

- Unit 3: 3-BA, 21-BA





# Patroonherkenning

Opdelen van stukken code in functionele onderdelen

Matchen van typen

Aan de slag

# Opdelen in functionele onderdelen

- Vaak in combinatie met bewerkingen van arrays en/of condities
  - Toevoegen of verwijderen van elementen
  - Overslaan van bepaalde elementen
- Welke functionele onderdelen kunnen we onderscheiden? Waarom?



# Hoe bepaal ik een patroon?

- Basiswaarde
- Verandering per stap

# Voorbeelden van patronen

- De variabele  $i$  is in het begin gelijk aan nul (0) en neemt in elke stap met één (1) toe
- 1, 5, 9, 13, 17, 21, ...
- 2, 8, 32, 128, 512, 2048, ...

# Matchen van types

- Variabelen en returntypes van methoden
- Waarom?

# Aan de slag

- Unit 3: 5-BA, 19-BA
- Unit 4: BA 3