

# C# Essentials: Selectie

Lector: Tom Quareme

# if

- **Gebruik een if om keuzes te maken.**
  - **Als (voorwaarde), dan {code uitvoeren}.**
- `if (booleaanse expressie)`  
`{`  
    code  
`}`
- Voorwaarde is een booleaanse expressie. Kan true of false zijn.
  - Voorbeeld:
    - `if (getal > 0) { ... }`
  - Ander voorbeeld:
    - `bool expressie = (getal > 0); // true of false`  
    `if (expressie) { ... }`

# Waarheidstabel not operator (!)

- ! (not operator), keert booleaanse expressie om:
  - Bijvoorbeeld: `bool expressie = !(getal > 0);` // wordt `getal <= 0`

Stel een voorwaarde is:	! toepassen erop (tegengestelde nemen):
true	false
false	true

# Waarheidstabel and operator (&& of &)

- &&, & (and operator):
  - Aan alle voorwaarden moet voldaan zijn om in totaal true te zijn, anders false.
  - Gebruik meestal &&  $\Rightarrow$  stopt met voorwaardes checken als er eentje false is
  - & controleert alle voorwaardes, zelfs al is er eentje false.
  - Bijvoorbeeld: `bool expressie = (getal > 0) && (getal < 10) && (getal != 5);`

Stel voorwaarde 1 is:	Stel voorwaarde 2 is:	<b>&amp;&amp; toepassen ertussen</b>
<b>true</b>	<b>true</b>	<b>true</b>
false	true	false
true	false	false
false	false	false

# Waarheidstabel or operator (|| of |)

- ||, | (or operator):
  - Aan minstens 1 van de voorwaarden moet voldaan zijn om in totaal true te zijn.
  - Gebruik meestal ||  $\Rightarrow$  stopt met voorwaardes checken als er eentje true is
  - | controleert alle voorwaardes, zelfs al is er eentje true.
  - Bijvoorbeeld: `bool expressie = (getal == 5) || (getal == 10);`

Stel voorwaarde 1 is:	Stel voorwaarde 2 is:	<b>   toepassen ertussen</b>
true	true	true
false	true	true
true	false	true
false	false	false

# Waarheidstabel xor operator (^)

- ^ (xor operator):
  - Aan minstens 1 van de voorwaarden moet voldaan zijn om in totaal true te zijn.
  - Is hetzelfde als or, MAAR: als alle voorwaarden true zijn  $\Rightarrow$  toch false in totaal.
    - Daarom spreken we van exclusive or (xor).
  - Bijvoorbeeld: `bool expressie = (getal == 5) ^ (getal == 10);`

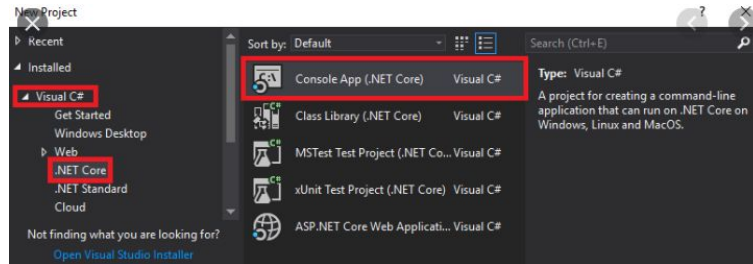
Stel voorwaarde 1 is:	Stel voorwaarde 2 is:	^ toepassen ertussen
true	true	<u>false</u>
false	true	true
true	false	true
false	false	false

# if (voorbeeld console programma)

```
static void Main(string[] args)
{
    string tekst = Console.ReadLine();
    int getal = Convert.ToInt32(tekst);
    if (getal == 10)
    {
        Console.WriteLine("Dit getal is gelijk aan 10.");
    }

    getal = 100;
    if (getal == 5)
    {
        // Deze blok wordt niet uitgevoerd! Waarom?
        Console.WriteLine("Dit getal is gelijk aan 5.");
    }
}
```

if  
Code snippet for if statement  
Note: Tab twice to insert the 'if' snippet



# if met maar 1 statement als code

- Als er maar 1 statement binnen { } van “if” wordt uitgevoerd, mag je { } weglaten.

```
static void Main(string[] args)
{
    int getal = 10;
    if (getal == 10)
        Console.WriteLine("Dit getal is gelijk aan 10.");

    getal = 100;
    if (getal == 5) Console.WriteLine("Dit getal is gelijk aan 5.");
}
```



# if else

- Voer code uit in { } van “else” indien niet voldaan aan voorwaarde van de “if”.

```
static void Main(string[] args)
{
    int getal = 5;
    if (getal == 10)
    {
        Console.WriteLine("Dit getal is gelijk aan 10.");
    }
    else
    {
        Console.WriteLine("Dit getal is NIET gelijk aan 10.");
    }
}
```

# if else met maar 1 statement als code

- Als er maar 1 statement binnen { } van “if” of “else” wordt uitgevoerd, mag je { } weglaten.

```
static void Main(string[] args)
{
    int getal = 5;
    if (getal == 10)
        Console.WriteLine("Dit getal is gelijk aan 10.");
    else
        Console.WriteLine("Dit getal is NIET gelijk aan 10.");
}
```

# Ternary operator

- (voorwaarde) ? resultaat als voorwaarde true is : resultaat als voorwaarde false is;

```
static void Main(string[] args)
{
    int getal = 15;
    string waarde = (getal > 10) ? "juist" : "fout";

    // Print "juist" af, want getal is groter dan 10
    Console.WriteLine(waarde);
}
```

# if, else if, else

- Je kan onbeperkt aantal “else if” blokken zetten.

```
static void Main(string[] args)
{
    double getal = 15.2;
    if (getal < 10.0)
    {
        Console.WriteLine("Dit getal is kleiner dan 10.");
    }
    else if (getal >= 10 && getal < 20)
    {
        Console.WriteLine("Dit getal is groter of gelijk aan 10 en kleiner dan 20.");
    }
    else
    {
        Console.WriteLine("Dit getal groter of gelijk aan 20");
    }
}
```

# Geneste if's

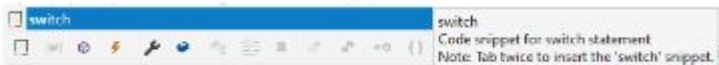
- Een “if”-blok kan je ook zetten in een “if”-blok, “if-else”-blok of “else”-blok.
- Nesting is onbeperkt.

```
double getal = 2.718;
if (getal < 3)
{
    if (getal > 2)
        Console.WriteLine("tussen 2 en 3");
    else if (getal > 1)
        Console.WriteLine("tussen 1 en 2");
    else
        Console.WriteLine("<= 1");
}
```

# switch

- Test op expliciete gevallen (cases). Enkel echte gelijkheden, geen < of > toestanden.
- Is performanter dan if, else-if's indien mogelijk.
- Met break spring je terug uit de switch.

```
string fruit = Console.ReadLine();  
bool isLekker = false;  
switch (fruit)  
{  
    case "peer":  
        isLekker = true;  
        break;  
    case "kers":  
        isLekker = true;  
        break;  
    case "citroen":  
    case "pompelmoes":  
        isLekker = false;  
        break;  
    default:  
        break;  
}
```



# TryParse

- Controleer of conversie zal lukken.
  - Geeft bool terug: true indien gelukt, false indien niet gelukt.
  - Geconverteerde waarde opslaan in variabele getal via **out getal**.

```
static void Main(string[] args)
{
    int getal;
    string tekst = "10";
    bool isGelukt = int.TryParse(tekst, out getal);
    Console.WriteLine(getal + " is gelukt of niet: " + isGelukt);
}
```

- Declaratie mag ook binnen de TryParse en geldt voor de blok erbuiten ook:

```
static void Main(string[] args)
{
    string tekst = "10";
    bool isGelukt = int.TryParse(tekst, out int getal);
    Console.WriteLine(getal + " is gelukt of niet: " + isGelukt);
}
```

# TryParse

- Controleer of conversie zal lukken.
  - Geeft bool terug: true indien gelukt, false indien niet gelukt.
  - Geconverteerde waarde opslaan in variabele getal via **out getal**.

```
static void Main(string[] args)
{
    int getal;
    string tekst = "10";
    bool isGelukt = int.TryParse(tekst, out getal);
    Console.WriteLine(getal + " is gelukt of niet: " + isGelukt);
}
```



# TryParse

- Declaratie mag ook binnen de TryParse en geldt voor de blok erbuiten ook:

```
static void Main(string[] args)
{
    string tekst = "10";
    bool isGelukt = int.TryParse(tekst, out int getal);
    Console.WriteLine(getal + " is gelukt of niet: " + isGelukt);
}
```

# Waarde van variabele omzetten naar een string

- Je kan een waarde van variabele omzetten naar een string via:
  - `Convert.ToString(variabeleNaam);`
  - `variabeleNaam.ToString();`

```
int getal = 10;  
string s1 = Convert.ToString(getal);  
string s2 = getal.ToString();  
// ToString() heeft betere foutmeldingen en is performanter!
```

```
TxtResultaat.Text = getal.ToString();
```

# string formatting

- Variabelen x en y invullen in een string via \$"geformatteerde waardes: {x} en {y}"

```
int x = 2;
int y = 4;
string resultaat = $" {x} + {y} = 6"; // De string resultaat is nu: "2 + 4 = 6"
Console.WriteLine(resultaat);
```

```
x = 1;
y = 2;
string boodschap = $"Dit is regel {x}.\r\n" +
    $"Dit is regel {y}\r\n";
```

// met \r\n nieuwe regel starten en met + ga je 2 strings aan elkaar plakken (concatteneren)

- Oudere manier is via String.Format():

```
string resultaat = String.Format("{0} + {1} = 6", x, y);
// x waarde komt op {0}, y waarde op {1}, dus:
// De string resultaat is nu: "2 + 4 = 6"
```

# string formatting

- Voorbeelden van verschillende formatting

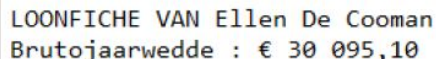
```
Label1.Text = $"Geheel getal: {123456} en floating point: {123.456}"; // Geheel getal: 123456 en floating point: 123,456
Label1.Text = $"{123:D5}"; // 00123
Label1.Text = $"{123456:N}"; // 123 456,00
```

// Voorbeeld getallen uitgelijnd wegschrijven

```
string a = $"|{1,5}|{20,5}|{300,5}|";           "|      1|    20|   300|"
string b = $"|{1,-5}|{20,-5}|{300,-5}|";         "|1      |20    |300   |"
string b = $"|{1,-5:d3}|{20,-5:d3}|{300,-5:d3}|"; "|001    |020   |300   |"
```

//Afdruk naar tekstvak

```
TxtResultaat.Text = $"LOONFICHE VAN {naam}\r\nBrutojaarwedde : {bruto:C}";
```



LOONFICHE VAN Ellen De Cooman  
Brutojaarwedde : € 30 095,10

# string formatting

- **Overzicht van alle soorten formatting**

Karakter	Omschrijving	Voorbeeld	Resultaat
C or c	Currency	<code>\$"{2.5:C}";</code> <code>\$"{-2.5:C}";</code>	€ 2,50 € -2,50
D or d	Decimal	<code>\$"{25:D5}";</code>	00025
E or e	Scientific	<code>\$"{250000:E}";</code>	2,500000E+005
F or f	Fixed-point	<code>\$"{25:F2}";</code> <code>\$"{25:F0}";</code>	25,00 25
G or g	General	<code>\$"{2458.512:G}";</code>	2458,512
N or n	Number	<code>\$"{2500000:N}";</code>	2 500 000,00
P or p	Percent	<code>\$"{2.5:P}";</code>	250,00%
X or x	Hexadecimal	<code>\$"{250:X}";</code>	FA

# string formatting

- Eigen string formatting maken

Vb. Label1.Text = \$"{3100.5: #,##0.00}"; // 3 100,50

Format	Getal	Weergave
0000.00	100.5	0100.50
0	100.5	101
#,##0	3100.5	3 101
#,##0.00	3100.5	3 100,50
# ##0.00 \\E\\U\\R	3100.5	3 100,50 EUR
0.00%	0.253	25,30%