

1.0.0

### **H3: Strings en hun methoden**

# Strings

## String

Een string is een reeks van 0, 1 of meerdere lettertekens, zoals je ook kan zien als je even met je muis boven een string keyword *hoovert* in je code:

```
string eenTekst = "Wat een mooi zin";
```



class System.String

Represents text as a sequence of UTF-16 code units.

## Strings declareren

Merk op dat we bij een string literal gebruik maken van aanhalingstekens ( " ) terwijl bij chars we een apostrof gebruiken ( ' ). Dit is de manier om een string van een char te onderscheiden.

Volgende code geeft twee keer het cijfer 1 onder elkaar op het scherm, maar de eerste keer gaat het om de weergave van een `string` (reeks van tekens) en de tweede keer van een `int` (effectief getal):

```
string eenString = "1";  
int eenGetal = 1;  
  
Console.WriteLine(eenString);  
Console.WriteLine(eenGetal);
```

De output van dit programma zal dan zijn:

```
1  
1
```

Fout gebruik van strings zal code geven die niet zal gecompileerd worden:

```
string eenString = 1; //fout  
int eenGetal = "1"; //fout
```

1. In de tweede toekenning proberen we een literal van het type **int** toe te kennen aan een variabele van het type **string**.
2. In de laatste toekenning proberen we een literal van het type **string** toe te kennen aan een variabele van het type **int**.

# Strings samenvoegen

✓ Kennisclip voor deze inhoud

## Strings samenvoegen

Je kan strings en variabelen samenvoegen tot een nieuwe string op verschillende manieren. We bekijken volgende twee mogelijkheden:

- `+` -operator
- `$` string interpolation

⚠ Gebruik zelf stringinterpolatie tenzij het anders gevraagd wordt. Dit is bijna altijd de handigste manier. Online kom je nog (vooral oudere) code tegen die het anders doet, maar we geven deze bewust **niet** omdat stringinterpolatie bijna altijd het beste werkt.

### In dit hoofdstuk

We gaan van volgende informatie uit:

- Stel dat je 2 variabelen hebt `int age=13` en `string name="Finkelstein"`.
- We willen de inhoud van deze variabelen samenvoegen in een nieuwe `string result` die zal bestaan uit de tekst:  
`Ik ben Finkelstein en ik ben 13 jaar oud.`

Volgende 2 manieren tonen hoe je steeds tot voorgaande string zal komen.


### Manier 1: String samenvoegen met de `+` -operator

Als je de `+` tussen strings plaatst, krijgt deze operator een andere betekenis dan tussen getallen. De strings worden dan achter elkaar geplakt. Als iets geen string is en op deze manier wordt gebruikt, wordt eerst een tekstvoorstelling bepaald, zoals hieronder bij `age` (want dat is een `int`).

```
string result = "Ik ben " + name + " en ik ben " + age + " jaar oud.";
```

Op het eerste zicht is dit een eenvoudige manier om strings op te bouwen, maar ze heeft een paar belangrijke nadelen:

- je moet vaak afwisselen tussen aanhalingstekens en plustekens
- het is lastig spaties en leestekens juist te noteren op deze manier (merk op dat de stukken tekst in het voorbeeld spaties op de zijkanten bevatten)
- er komt vrij veel extra werk bij kijken als je data in een specifiek formaat wil weergeven, bijvoorbeeld met een specifiek aantal cijfers na de komma
- als je grote, complexe strings op deze manier opbouwt, kost het erg veel rekentijd

 We geven deze manier van werken vooral mee omdat ze in héél veel programmeertalen bestaat en omdat ze simpel is. Ze is niet bijzonder *goed*.

## Manier 2: String interpolation met `$`

Via stringinterpolatie schrijf je je string min of meer zoals hij er uiteindelijk moet uitzien, maar vervang je de niet-letterlijke delen door geformatteerde waarden. Dit levert een goed leesbaar resultaat.

Door het `$`-teken **VOOR** de string te plaatsen geef je aan dat alle delen in de string die tussen accolades staan als code mogen beschouwd worden. Een voorbeeld maakt dit duidelijk:

```
string result = $"Ik ben {name} en ik ben {age} jaar oud.";
```

In dit geval zal dus de inhoud van de variabele `name` tussen de string op de plek waar nu `{name}` staat geplaatst worden. Idem voor `age`. Dit mag, zelfs al is `age` geen string: hetgeen tussen de accolades staat, wordt altijd intern omgezet naar een string voor het in het resultaat wordt geplaatst.

# Omzetten van en naar strings

## Input van de gebruiker verwerken

✓ Kennisclip voor deze inhoud

### Conversie

Zoals eerder aangegeven, kan je geen strings toekennen aan variabelen van type `int` of omgekeerd. Toch wil je soms een getal beschouwen als tekst, of tekst omzetten naar een getal. .NET heeft hiervoor methoden die je kunnen helpen om data van het ene type naar het andere te brengen. Deze methoden zitten binnen de **Convert**-klasse.

Het gebruik hiervan is zeer eenvoudig. Enkele voorbeelden:

```
int userAge = Convert.ToInt32("19"); // string to int
string ageAsText = Convert.ToString(19); // int to string
```

Je plaatst bij gebruik van `Convert` tussen de ronde haakjes de variabele of literal die je wenst te converteren naar een ander type. Merk op dat naar een `int` converteren met `.ToInt32()` moet gebeuren. Voor andere types zijn er overeenkomstige methoden. Je kan [alle conversie-mogelijkheden hier bekijken](#). Volgende conversies zullen je al ver vooruit helpen:

gewenst type	methode
int	<code>ToInt32</code>
double	<code>.ToDouble</code>
string	<code>ToString</code>
boolean	<code>ToBoolean</code>
byte	<code>ToByte</code>

### Foutloze input

Voorgaande code veronderstelt dat de gebruiker géén fouten invoert. De conversie zal namelijk mislukken indien de gebruiker bijvoorbeeld `IKWEEG10KG` invoert in plaats van `10,3`.

In het begin van de leercurve **moet** je er altijd van uitgaan dat de gebruiker foutloze input geeft. Later leer je wel hoe je dit kan afhandelen.

⚠ **Opgelet:** de invoer van kommagetallen door de gebruiker is afhankelijk van de landinstellingen van je besturingssysteem. Staat deze in Belgisch/Nederlands dan moet je kommagetallen met een **KOMMA**( , ) invoeren (dus 9,81 ), staat deze in het Engels dan moet je een **PUNT**( . ) gebruiken ( 9.81 ).

⚠ **Opgelet 2:** In je C# code moet je doubles **ALTIJD** met een punt schrijven. Dit is onafhankelijk van je taalinstellingen.



# Functionaliteit van strings

Strings bevatten veel ingebouwde functionaliteit. Als je deze leert gebruiken, kan je al snel nuttige programmaatjes voor tekstverwerking schrijven.

## Length

De lengte van een string is het aantal symbolen in de weergegeven versie. Je plaatst `.Length` achter de string om de lengte te weten te komen. Enkele voorbeelden:

```
Console.WriteLine("hallo".Length); // levert 5 want: 5 symbolen in de uiteindelijke weergave
Console.WriteLine("hallo ".Length); // levert 6 want: 6 symbolen in de uiteindelijke weergave
```

## Substring

Een string is een reeks van 0 of meer symbolen in een bepaalde volgorde. Dat wil zeggen dat we elk symbool een nummer kunnen toekennen. Misschien wat vreemd, maar het eerste symbool krijgt nummer 0, het tweede krijgt nummer 1, het derde nummer 2,... en het laatste krijgt een nummer gelijk aan de `Length` van de string min één. Dit nummer heet de **index** van het symbool.

We kunnen een **substring** (= deel van een string) opvragen door de index van het eerste symbool en de lengte mee te geven als volgt:

```
Console.WriteLine("Hallo, wereld".Substring(4,5)); // toont o, we
```

Je mag de lengte achterwege laten om vanaf de gegeven index tot het einde van de string te gaan:

```
Console.WriteLine("Hallo, wereld".Substring(4)); // toont o, wereld
```

Deze methode **verandert een string niet**. Er is geen enkele methode die dat doet, want **je kan een string niet veranderen in C#**. Je kan er alleen een nieuwe mee bouwen. De methode berekent wel een nieuwe string met de gewenste eigenschappen. Dit is een belangrijk onderscheid. Volgende drie voorbeelden tonen het verschil. Voer uit en verklaar.

```
string hallo1 = "hallo";
hallo1.substring(0,2);
Console.WriteLine(hallo1);
```

```
string hallo1 = "hallo";
Console.WriteLine(hallo1.substring(0,2));
```

```
string hallo1 = "hallo";

string hallo2 = hallo1.Substring(0,2);
Console.WriteLine(hallo2);
```

⚠ Onthoud het goed: je kan een string niet aanpassen in C#. We kunnen alle gevolgen hiervan nog niet uitleggen, maar het is wel zo.

## IndexOf

Een index is, zoals hierboven aangegeven, de positie van een teken in de string. Als we willen weten waar een bepaalde substring in een string voorkomt, gebruiken we `IndexOf`:

```
Console.WriteLine("C# is cool".IndexOf("cool")); // geeft 6
Console.WriteLine("C# is cool".IndexOf("z")); // geeft -1 want komt niet voor
Console.WriteLine("C# is cool".IndexOf("Cool")); // geeft -1 want "Cool" MET HOOFDLETTER komt niet voor
```

## ToUpper / ToLower

Met deze twee methodes bereken je een versie van de string waarop je ze toepast, maar dan in hoofdletters of in kleine letters. Let op: je past de oorspronkelijke string niet aan!

```
Console.WriteLine("C# is cool".ToUpper()); // C# IS COOL
Console.WriteLine("C# is cool".ToLower()); // c# is cool
string tekst = "DiT iS EnGe TeKsT";
tekst.ToLower();
Console.WriteLine(tekst); // DiT iS EnGe TeKsT -> tekst wordt niet aangepast door ToLower
tekst.ToUpper();
Console.WriteLine(tekst); // DiT iS EnGe TeKsT -> tekst wordt niet aangepast door ToLower
string berekendeTekst = tekst.ToUpper();
Console.WriteLine(berekendeTekst); // DIT IS ENGE TEKST -> uit tekst is iets anders berekend
```

## Replace

Met deze methode kan je een substring vervangen door een andere substring. Ook deze methode past de oorspronkelijke tekst niet aan.

```
Console.WriteLine("C# is cool".Replace("C#", "Racket")); // Racket is cool
Console.WriteLine("C# is cool".Replace("Java", "Racket")); // C# is cool -> Java kwam niet voor
string tekst = "C# is cool";
tekst.Replace("C#", "Racket");
Console.WriteLine(tekst); // C# is cool -> uit tekst is iets anders berekend, tekst is niet veranderd
```

## TrimStart / TrimEnd / Trim

Met deze methodes verwijder je witruimte (spaties, tabs, newlines,...) aan het begin of aan het einde van een string:

```
Console.WriteLine("    C# is cool".TrimStart()); // C# is cool, eerste teken is C en geen s
Console.WriteLine("C# is cool    ".TrimEnd()); // zelfde resultaat
Console.WriteLine("    C# is cool    ".Trim()); // zelfde resultaat
```

# Oefeningen

Al deze oefeningen maak je in een klasse `StringsEnHunMethoden`

## Oefening: VariabelenEnHoofdletters

Leerdoelen

- gebruik van variabelen om input en output op te slaan en te tonen
- functionaliteit van strings

Functionele analyse

Een applicatie vraagt je tekst in te voeren die dan daarna zal worden getoond met allemaal hoofdletters.

Technische analyse

Noem de methode voor deze oefening `VariabelenEnHoofdletters`.

voorbeeldinteractie(s)

```
Welke tekst moet ik omzetten?  
> Hello World  
HELLO WORLD
```

Technische hulp

## Programmaverloop

Lees de gebruikersinvoer van de console en sla deze op in een variabele.

Zet de inhoud van deze variabele om in hoofdletters. Je kan dit doen door `ToUpper()` te gebruiken.

Uiteindelijk geef je dan het resultaat weer in de console.

## Testscenario's

- Voer tekst in met spaties
- Voer tekst in van meer dan 100 karakters
- Voer tekst in van 1 karakter
- Voer geen tekst in

## Oefening: H3-string-interpolation

## Leerdoelen

- gebruik van string interpolation

## Functionele analyse

Zelfde als oefeningen maaltafels en ruimte vorig hoofdstuk.

## Technische analyse

Je moet twee methoden schrijven. Noem de eerste `MaaltafelsStringInterpolatie` en de tweede `RuimteStringInterpolatie`. Deze doen net hetzelfde als hun tegenhangers uit het vorige hoofdstuk, maar je bouwt de getoonde tekst op met stringinterpolatie in plaats van via `+` en/of `Console.WriteLine`.

## voorbeeldinteractie(s)

Zie oefening H2-maaltafels en H2-ruimte.

## Technische hulp

Programmaverloop

Pas string interpolatie m.b.v. `$` toe om de veranderlijke onderdelen van de output in te vullen.

Testscenario's

- Zie oefening H2-maaltafels en H2-ruimte.

## Oefening: H3-bereken-btw

### Leerdoelen

- gebruik van string interpolation

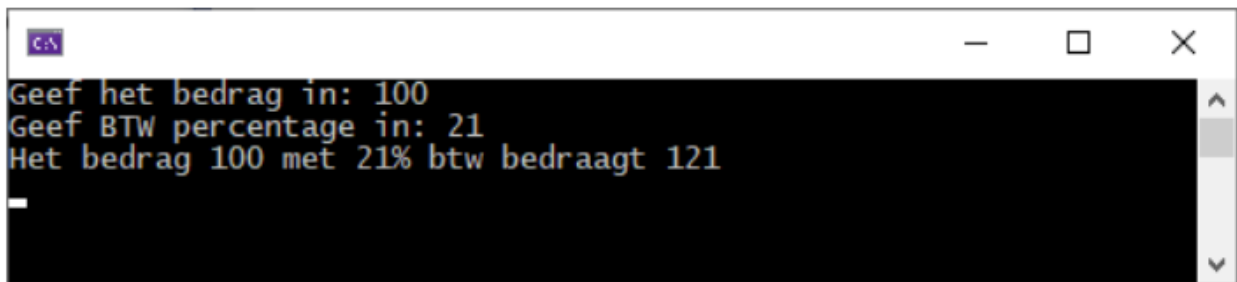
### Functionele analyse

Een programma vraagt een bedrag en vervolgens btw percentage in te geven waarna het bedrag incl. btw-percentage wordt weergegeven.

### Technische analyse

Noem de methode voor deze oefening `BerekenBtw`.

## voorbeeldinteractie(s)



```
Geef het bedrag in: 100
Geef BTW percentage in: 21
Het bedrag 100 met 21% btw bedraagt 121
```

## Technische hulp

### Programmaverloop

Het bedrag dat wordt ingevoerd moet geconverteerd worden naar een `int` met `Convert.ToInt32`.

Pas string interpolatie toe om de output te tonen.

### Testscenario's

- Typ tekst in
- Geef een veel te groot bedrag in

## Oefening: H3-leetspeak

### Leerdoelen

- functionaliteit van strings leren kennen

### Functionele analyse

We willen tekst omvormen naar een ander formaat. Laat de gebruiker een lijn tekst ingeven en haal er alle tussenliggende spaties uit en vervang de a's door @

### Technische analyse

Gebruik `Console.ReadLine` om tekst in te lezen en hou bij in een variabele. Pas de nodige string methodes toe om het resultaat te verkrijgen. Noem je methode voor dit programma `LeetSpeak`.

### Programmaverloop

```
Geef je tekst in
> Oefening baart kunst!
Oefeningb@@rtkunst!
```

## Testscenario's

- test met een zin zonder a's
- test met een zin met vijf a's of meer
- test met een lege string

## Oefening: H3-instructies

### Leerdoelen

- leren werken met stringinterpolatie
- leren werken met methodes van strings

### Functionele analyse

We willen met behulp van een programma instructies genereren voor de gebruiker. Meerbepaald wordt automatisch aangegeven in welke map de gebruiker bepaalde bestanden op een UNIX-achtig systeem moet bijhouden.

Voor deze oefening is het verplicht gebruik te maken van een (geïnterpoleerde) string.

### Technische analyse

Op basis van de voornaam van de student en de naam van de cursus wordt de map gegeven die de student moet aanmaken ( /home/ , eerste 3 letters voornaam, in hoofdletters met submap de naam van de cursus. Noem je methode Instructies.

### Programmaverloop

```
Wat is je naam?
> Vincent
Wat is de naam van de cursus?
> Programmeren
Maak een map als volgt: /home/VIN/Programmeren
```

## Oefening: H3-lotto

### Leerdoelen

- functionaliteit van strings
- stringinterpolatie

### Functionele analyse

De gebruiker voert zijn lottocijfers in. We willen deze op een overzichtelijke manier weergeven.

### Technische analyse

Laat de lottocijfers allemaal achter elkaar ingeven, gescheiden door komma's, zonder spaties. De gebruiker wordt verondersteld cijfers onder de 10 in te geven voorafgegaan door een nul. Gebruik de juiste methode om de cijfers uit te string te "knippen" en gebruik het karakter `|` om de uitvoer te scheiden. Noem je methode `Lotto`.

### Voorbeeldinteractie

```
Wat zijn je cijfers (tussen 01 en 45)?
> 05,08,13,18,27,44
Je cijfers zijn:
05|08|13
18|27|44
```

## Oefening: H3-som-van-cijfers

### Leerdoelen

- functionaliteit van strings
- stringinterpolatie

### Functionele analyse

De gebruiker voert een getal in. Het programma berekent de som van de cijfers in de decimale voorstelling van dit getal.

### Technische analyse

We veronderstellen dat de gebruiker een getal van exact vijf cijfers ingeeft, desnoods vooraan opgevuld met nullen. Noem je methode `SomVanCijfers`.

### Voorbeeldinteractie

Onderstaand voorbeeld komt uit op 27, want  $6 + 3 + 9 + 2 + 7$  is 27.



```
Gelieve een getal in te voeren dat bestaat uit exact 5 decimale cijfers.  
> 63927
```

```
De som is 27.
```

## Oefening: H3-naam-uit-mail

### Leerdoelen

- functionaliteit van strings
- stringinterpolatie

### Functionele analyse

De gebruiker voert een e-mailadres in. Jouw programma toont hieruit het gedeelte dat de naam voorstelt, in hoofdletters.

### Technische analyse

We veronderstellen dat de gebruiker een juist mailadres invult. Noem je methode `NaamUitEmail`.

### Voorbeeldinteractie

```
Geef je e-mailadres:  
> ann.debrabandere@ap.be  
Je naam uit je e-mail is: ANN.DEBRABANDERE
```

## Oefening: H3-eerste-letter-en-achternaam

### Leerdoelen

- functionaliteit van strings
- stringinterpolatie

### Functionele analyse

De gebruiker voert zijn naam in. Je programma toont dan de eerste letter van de voornaam en de familienaam.

### Technische analyse

We veronderstellen dat de gebruiker een voornaam zonder spaties invult. Noem je methode `EersteLetterEnAchternaam`.

## Voorbeeldinteractie

```
Geef je naam:  
> Ann De Brabandere  
De eerste letter van je naam is: A.  
Je achternaam is: De Brabandere
```

## Oefening: H3-toegangscade

### Leerdoelen

- functionaliteit van strings
- stringinterpolatie
- omzetting tussen tekst en getal

### Functionele analyse

De gebruiker voert enkele persoonlijke gegevens in en op basis hiervan wordt een persoonlijke toegangscade gegenereerd.

### Technische analyse

Noem je methode `Toegangscade`. De code van vier tot vijf cijfers wordt als volgt bepaald:

- het eerste symbool is de voorlaatste letter van de naam, in kleine letters
- het tweede symbool is de laatste letter van de naam, in hoofdletters
- het derde symbool is het laatste cijfer van het geboortecade
- het vierde (en eventueel vijfde) symbool is het eerste cijfer van de postcode, in het kwadraat (dus vermenigvuldigd met zichzelf)

## Voorbeeldinteractie

```
Geef je naam: > Janssens  
Geef je geboortecade: > 2001  
Geef je postcode: > 2000  
Je toegangscade is nS14
```