



# LEREN PROGRAMMEREN MET JAVASCRIPT

*Beginnerscursus*

Tom Vanhoutte  
Docent Full Stack Developer

Versie: 2.0

Laatste update: 1 augustus 2022

# **DISCLAIMER**

© In2Sites.be  
**Copyright**

Het cursusmateriaal is auteursrechtelijk beschermd en mag niet worden gereproduceerd zonder uitdrukkelijke schriftelijke toestemming van de auteur.

Elk direct of indirect plagiaat van materiaal van deze cursus zal resulteren in snelle juridische stappen.

**IN2SITES**

Zaakvoerder  
Tom Vanhoutte

<b>HOOFDSTUK 1: INLEIDING TOT JAVASCRIPT .....</b>	<b>4</b>
Wat is JAVASCRIPT? .....	4
Historiek.....	5
Wanneer gebruik je geen javascript meer.....	5
Javascript onderwerpen.....	6
Console! .....	7
Variabelen en datatypes.....	9
Datatype conversion.....	16
PLACEHOLDERS.....	19
Operatoren.....	20
REKENKUNDIGE OPERATOREN.....	20
LOGISCHE OPERATOREN .....	21
<b>HOOFDSTUK 2: CONTROLESTRUCTUREN.....</b>	<b>23</b>
If / Else If / Else .....	23
Switch.....	24
While Loop.....	26
Do While.....	27
Foreach.....	28
<b>HOOFDSTUK 3: ARRAYS.....</b>	<b>29</b>
Arrays en loops.....	33
ARRAYS EN FOR LOOP .....	33
ARRAYS EN WHILE LOOP .....	33
ARRAYS EN FOREACH LOOP .....	34
ARRAYS EN FOR IN LOOP.....	34
ARRAYS, LOOP EN CONTINUE .....	35

## Javascript onderwerpen

Hieronder zie je reeds een beknopt overzicht van javascript onderwerpen die je in eerste instantie zal te verwerken krijgen. Al deze onderwerpen worden verder in de cursus in detail onder de loep genomen.

- Data types
  - Numeriek
  - Strings
  - Booleans
  - Arrays
  - Objects
    - built-in: window, document, ... zijn allemaal objecten met hun eigen properties (eigenschappen).
- Variabelen
- Operatoren
  - + teken wordt gebruikt voor het optellen van getallen of het samenvoegen van strings
  - = teken wordt gebruikt om waarden toe te kennen aan variabelen
  - == teken wordt gebruikt om 2 waarden met elkaar te vergelijken
  - != teken is idem als het voorgaande maar dan verschillend van elkaar.
- Functies
  - zijn collecties (verzamelingen) van statements die een waarde retourneren.
- Controle structuren
  - If, else, switch
- Loops
  - For, while
- De DOM (DOCUMENT OBJECT MODEL) voor javascript
  - Veel methodes
    - Bijv. document.getElementById,...
  - Veel properties
    - value, checked, className, id,...

ARRAYS, LOOP EN BREAK.....	35
ARRAYS MAP.....	36
ARRAYS FILTER.....	36
<b>HOOFDSTUK 3: FUNCTIES.....</b>	<b>37</b>
STRINGS.....	37
NUMBERS.....	39
MATH.....	41
ANONYMOUS FUNCTIONS.....	44
NAMED FUNCTIONS.....	45
BLOCKED SCOPING.....	45
FUNCTIONS MET PARAMETERS.....	48

# HOOFDSTUK 1: INLEIDING TOT JAVASCRIPT

**H**eb je geen ervaring met webdevelopment of programmeren, dan is dit de perfecte cursus die je doelen zal helpen te bereiken. Javascript is de beste taal op in te stappen en te leren programmeren voor het web.

## Wat is JAVASCRIPT?

Javascript IS GEEN Java. Java is een object georiënteerde programmeertaal en heeft niets te maken met Javascript. De keuze van de naamgeving was in het verleden een ongelukkig toeval.

Javascript is de programmeertaal van het web die in staat is interactiviteit te genereren zoals: click, drag, swipe, tap, ... Javascript als programmeertaal wordt voornamelijk gebruikt bij front-end development.

Javascript kan je in elke website terugvinden. Zelfs de allergrootste sites zoals facebook en twitter gebruiken javascript. Javascript is ook CASE-SENSITIVE (=hoofdlettergevoelig). D.w.z. dat bijvoorbeeld de hoofdletter A inderdaad verschillend is van de kleine letter a. Javascript wordt voornamelijk gebruikt in de front-end, nl. client-side in de browser.

Dankzij frameworks zoals bijvoorbeeld node.js, react, angular,... kunnen we javascript ook server-side gaan gebruiken. In deze cursus zullen we het over client-side ontwikkeling hebben.

In deze cursus zullen we het eerst hebben over ES5! In de volgende cursus zien we de verschillen met ES6.

## Historiek

In het begin van het internet had iedere browser zijn eigen scripttaal:

- Netscape: livescript
- internet explorer: JScript

Uiteindelijk werd er overgegaan naar een universele scripttaal, nl. javascript.

Javascript is ook de enige front-end taal die samen met HTML wordt gebruikt.

Sedert 2009 spreken we van ECMA die de standaard is geworden, ook wel ES5 genoemd.

We spreken hier eigenlijk over ES6 (=ECMA SCRIPT 6 vanaf 2015),ES7 (=ECMA SCRIPT 7 vanaf 2016),....

De regel die nu wordt toegepast zorgt ervoor dat elk jaar er nieuwe features worden toegevoegd. D.w.z. dat we voor 2018 reeds spreken over ES9. Dit is de standaard die nu wordt gebruikt en is nog steeds aan de basis JAVASCRIPT! We zullen hier in deze cursus ECMA ook toelichten.

**OPMERKING:** De huidige standaarden die gebruikt worden momenteel zijn nog steeds ES5 en ES6 omdat de features van bijvoorbeeld ES7, .... in sommige oudere browsers gewoon niet werken. Het is aan te raden om dus nog steeds de features te gebruiken van ES5 en ES6 die ruim voldoende zijn voor ons als webdevelopers.

## Wanneer gebruik je geen javascript meer

Door de evolutie in CSS3 en HTML5 zijn er veel zaken die verplaatst werden van javascript en werden opgevangen door HTML5 en CSS3. Voorbeelden hiervan zijn:

- CSS zal zich ontfermen over oa.:
  - image swaps (rollover)
  - rollover menus
  - tooltips,...
- CSS3 transitions worden nu gebruikt i.p.v. de vroegere javascript animations.
- HTML5 form controls worden nu gebruikt i.p.v. javascript widgets

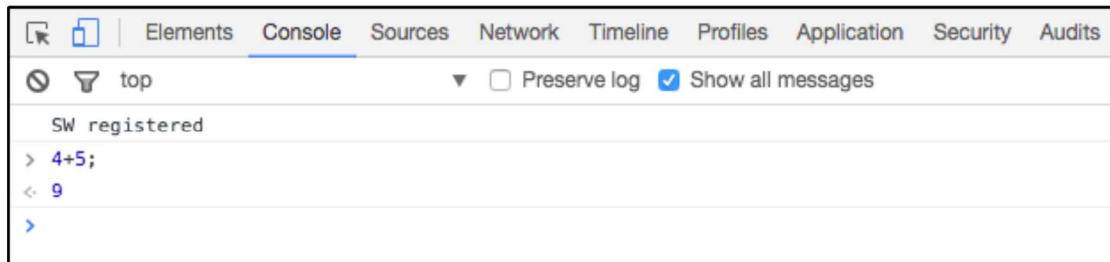
Niettegenstaande zijn er nog veel zaken die we wel dienen op te vangen met javascript en die we in deze cursus zullen overlopen.

## Console!

De Console is het belangrijkste tabblad die je terug kan vinden in iedere webbrowser. Wij gebruiken google chrome hiervoor.

De Console is in staat om onmiddellijk javascript code uit te voeren.

Hieronder een voorbeeld van een som binnen de console. We zullen dus eerst heel wat tijd besteden aan de console zelf, om die beter te leren kennen.



The screenshot shows the Google Chrome DevTools interface with the 'Console' tab selected. At the top, there are tabs for Elements, Console, Sources, Network, Timeline, Profiles, Application, Security, and Audits. Below the tabs, there's a header with a refresh icon, a square icon, and the text 'top'. To the right of the header are two checkboxes: 'Preserve log' (unchecked) and 'Show all messages' (checked). The main area contains the following text:  
SW registered  
> 4+5;  
< 9  
>

Om dit te openen in google chrome: druk F12 en klik op het tabblad CONSOLE of

CTRL + SHIFT + J

Om dit te openen in firefox: druk F12 en klik op het tabblad WEB CONSOLE of

CTRL + SHIFT + K

## Editor

We zullen de console nog veel gebruiken, maar vanaf dit punt coderen we in een editor. Er zijn verschillende editors die zich lenen tot het coderen van javascript. Er zijn gratis editors zoals: Sublime Text, Notepad++, ...

Maar als professioneel developer kijk je best naar professionele editors zoals: vscode of phpsstorm/webstorm van Jetbrains.

Wij zullen het beste pakket hiervoor gebruiken nl. phpsstorm/webstorm die een professioneel pakket is en door zeer veel softwarehuizen wordt gebruikt. Voor developers die voornamelijk met frontend bezig zijn kan webstorm gebruikt worden, maar voor mensen die Full Stack coderen gebruiken we beter PHPStorm die speciaal voor PHP developers werd ontwikkeld. Wij zullen PHPSTORM installeren. Dit pakket heeft een gratis geldigheid van 1 jaar met een studielicentie.

## Variabelen en datatypes

Het declareren van variabelen in de console is eenvoudig. Onderstaand voorbeeld declareert de variabele `x` en geeft de waarde 5 mee. Telkens wanneer we nu `x` intikken wordt de waarde teruggegeven in de sessie van de browser.

Wanneer je een browser refresh uitvoert is die waarde terug verdwenen.

```
> var x=5;
< undefined
> x
< 5
>
```

`var` zorgt dus voor het **declareren** van de **variabele**. Javascript kent ook nog `let` en `const` om variabelen te declareren. `Let` werd toegevoegd vanaf Javascript ES6. We weten reeds dat Javascript case-sensitive (hoofdlettergevoelig) werkt. Daarnaast dienen we ook rekening te houden met de regels voor opmaak voor variabelen. Zorg ervoor dat je variabelen een verstaanbare inhoud hebben.

Bijvoorbeeld: `a+b` vervang je beter door `getal1 + getal2`

Variabelen worden voorafgegaan door het gereserveerde woord **var** of **let**. Tussen beide zit een wezenlijk verschil die we verder in de cursus zullen aankaarten.

**Een var of let is undefined als datatype tot er een waarde aan wordt gegeven.**

Hoe mogen variabelen NIET geschreven worden:

- GEEN gereserveerde worden gebruiken! (`var`, `break`,...)
- NIET BEGINNEN met een cijfer
- GEEN gebruik van het koppelteken
- GEEN gebruik van ongeldige tekens (`&`, `!`,...)

Een datatype zorgt ervoor dat je variabele gedefinieerd wordt. Javascript dient te weten of een variabele een nummer is, tekst kan zijn,...

De datatypes die we kunnen gebruiken zijn:

- Primitive values:
  - Number: Floating point numbers (decimalen) of integers
  - String: tekenreeks, tekst
    - Schrijfwijze: **een string wordt steeds tussen enkele of dubbele quotes geschreven!**
  - Boolean: true or false
  - Undefined: Data type werd niet toegekend
  - Null: 'bestaat niet'
  - BigInt: nummers gelimiteerd tot het geheugen van de pc.
  - Symbol: zorgt voor een UNIEKE waarde (value)
- Objects



Maak een folder op je pc aan met de naam **javascript**. Daarin maak je terug een folder aan met de naam **1\_variabeldatatype**. In deze folder maak je een bestand aan met de naam **index.html**.

Javascript zal steeds in combinatie werken met een html bestand. HTML (HyperText Markup Language), zorgt enkel voor de weergave van TEKST binnen een pagina. Het onderscheid tussen HTML en javascript code scheiden we door een `<script>` tag te gebruiken zoals je hieronder kan zien in ons voorbeeld. De script tag heeft een begin en een eindtag. De script eindtag wordt voorafgegaan door een slash. Tussen deze tags schrijven we onze code, net zoals we die in de console hebben uitgevoerd. In onderstaande oefening zullen we variabelen volgens de verschillende datatypes declareren.

```
<script>
```

```
var voorNaam = "Tom";
console.log(voorNaam);
console.log(typeof(voorNaam));
var familieNaam = "Vanhoutte";
console.log(familieNaam);
console.log(typeof(familieNaam));
var geboorteJaar = 1973;
console.log(geboorteJaar);
console.log(typeof(geboorteJaar));
var functie = "Docent";
console.log(functie);
console.log(typeof(functie));
var gehuwd = true;
console.log(gehuwd);
console.log(typeof(gehuwd));
var niets; //datatype of waarde
console.log(niets);
console.log(typeof(niets));
niets = "niets is niet langer undefined";
console.log(niets);
console.log(typeof(niets));
var x = Number("fullstack");
console.log(x); //resulteert in NaN (Nota applicable number);
console.log(voorNaam + " " + familieNaam + ' is een ' + functie);
</script>
```

**OPMERKING:** **typeOf** is een ingebouwde functie van javascript. Functies komen later aan bod in deze cursus!

### Commentaar plaatsen in een javascript bestand:

// de dubbele slash zorgt voor het plaatsen van **commentaar** bij je code. Commentaar heeft geen enkele invloed voor je code en wordt enkel gebruikt om uitleg te verschaffen.

**Slash asterisk, asterisk slash = /\* meerdere**

Lijnen code in commentaar zetten \*/

```

Tom
string
Vanhoutte
string
1973
number
Docent
string
true
boolean
undefined
undefined
niets is niet langer undefined
string
Tom Vanhoutte is een Docent

```

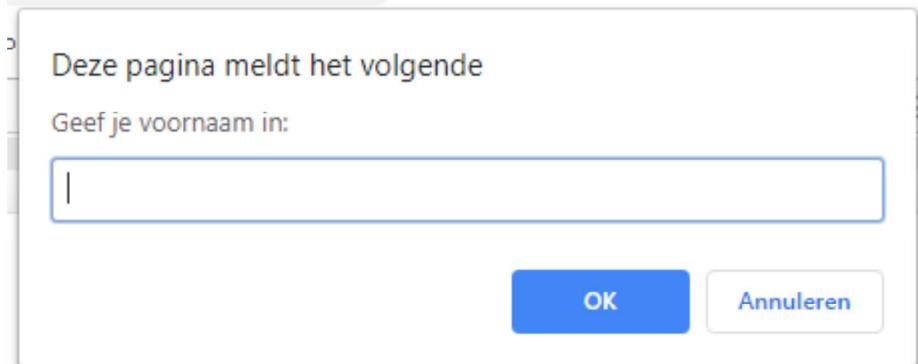
Merk op dat de laatste lijn volledig als string wordt ingelezen.



We kunnen variabelen ook laten inlezen door de gebruiker d.m.v. de **prompt** functie. Open de folder op je pc aan met de naam **javascript**. Daarin maak je terug een folder aan met de naam **prompt**. In deze folder maak je een bestand aan met de naam **index.html**.

```
<script>
var voorNaam = prompt('Geef je voornaam in:');
console.log(voorNaam);

</script>
```



Een andere mogelijkheid om een vraag te stellen aan de gebruiken is `window.confirm` ipv `window.prompt`. Aangezien confirm en prompt globale variabelen zijn dienen we `window.` niet te laten voorafgaan aan de methode.

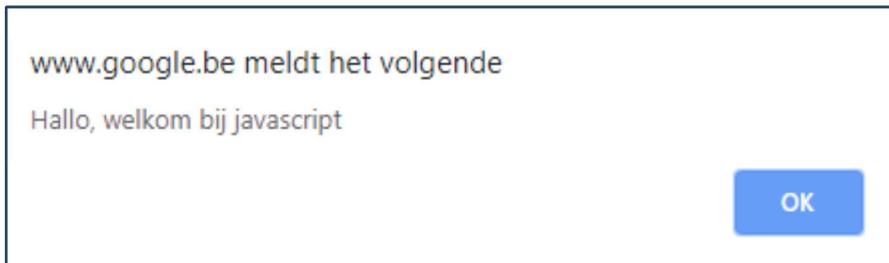
Een confirm heeft in tegenstelling tot de prompt als return een boolean, nl. true or false.

Voorbeeld:

```
if(window.confirm("Ben je zeker dat je dit wil verwijderen?")){
    deleteIets(id);
}
```



Een alert is de derde mogelijkheid. Hiermee kan je enkel een bericht (message) weergeven aan de gebruiker.



Een laatste en meest gebruikte mogelijkheid is het combineren van een html pagina met een invoerveld en javascript. Wanneer je nog geen HTML gezien hebt is dit buiten de scope van deze cursus javascript. Uiteindelijk zal javascript altijd met HTML samenwerken! Hier gaan we voor de eerste keer gebruik maken van een **EXTERN** javascript bestand. Dit is de meest gebruikte manier en heeft als voordeel dat we HTML en javascript code proper van elkaar scheiden. Om de verbinding te maken met het javascript bestand dienen we deze als script tag te linken binnen ons HTML bestand.

**index.html:**

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-
scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
</head>
<body>
    <label>Enter your name:</label>
    <input type="text" id="myText">
    <button type="button" id="activateTextbox">Activate</button>

    <script src="index.js"></script>
</body>
</html>
```

**index.js:**

```
document.getElementById("activateTextbox").onclick = function(){
    name = document.getElementById("myText").value;
    console.log(name);
}
```

**OPMERKING:** wanneer we via alle bovenstaande inputmogelijkheden een waarde (value) ontvangen, dan is deze steeds van het datatype STRING.

Wanneer je dus een cijfer zou intikken dan zou deze geïnterpreteerd worden als “2”

## DATATYPE CONVERSION

### NUMBER, PARSEINT, PARSEFLOAT

In bepaalde gevallen afhankelijk van je programma, zal het nodig zijn om type conversion uit te voeren. Type conversion is het wijzigen van het datatype. Je kan bijvoorbeeld een number omzetten naar string of omgekeerd,...



Open de folder op je pc aan met de naam **javascript**. Daarin maak je terug een folder aan met de naam **3\_typeconversion**.

In je javascript bestand index.js tik je het volgende in:

```
let yearOfBirth = window.prompt("What is your birth year?");
// yearOfBirth = yearOfBirth + 1 //lange schrijfwijze
yearOfBirth += 1 //korte schrijfwijze
//onderstaande geeft je nog steeds een STRING terug.
console.log("When you are one year of age, you would live in the year:", yearOfBirth);

//typeconversion naar een number
let yearOfBirth2 = window.prompt("What is your birth year?");
yearOfBirth2 = Number(yearOfBirth2);
yearOfBirth2 += 1;

//onderstaande geeft je nog steeds een STRING terug.
console.log("When you are one year of age, you would live in the year:", yearOfBirth2);
```

```
When you are one year of age, you would live in the year: 19731
When you are one year of age, you would live in the year: 1974
```

OPMERKING: **Number()** is hier een functie die zorgt voor de omzetting van een string naar een number (getal). Omgekeerd evenredig bestaat er ook de functie **String()** die een number naar een string (tekst) omzet.

Je kan ook gebruiken maken van **parseInt()** of **parseFloat()** om strings om te zetten naar een integer of een kommagetal.

Het verschil tussen het gebruik van `parseInt()` en `Number()` is de volgende: een `number` converteert het type en `parseInt` perst de waarde dus de inhoud van de input samen tot een getal.

Voorbeelden:

`parseInt('56test')` = 56

`parseInt('6e2')` = 6

`Number('56test')` = Nan (Not applicable Number)

`Number('6e2');` = 600

OPVANGEN VAN NaN:

Om te vermijden dat een programma een foutmelding geeft van `NaN` (Nota applicable Number), bestaat de functie `isNaN()`. Met deze functie kan je controleren of er true of false wordt gereturneerd. De functie `isNaN()` zorgt dus voor een boolean return. Zoals jullie reeds weten kent het datatype boolean slechts 2 waarden namelijk true or false.

Voorbeeld:

`Console.log(isNaN(Number('56test')));` = true

## CONST

Een constante is een variabele waarvan de waarde binnen een programma die niet gewijzigd kan worden.



Open de folder op je pc aan met de naam **javascript**. Daarin maak je terug een folder aan met de naam **4\_const**.

In je index.js bestand plaats je de volgende code:

```
const pi = 3.14;  
console.log(pi);  
pi = 9.14;  
console.log(pi);
```

Enkel de eerste 2 lijnen van dit programma zullen uitgevoerd worden. We proberen hier namelijk een constante te overschrijven. Dit zal resulteren in de volgende error in je console.

```
3.14  
✖ Uncaught TypeError: Assignment to constant variable.  
  at index.js:3:4
```

## PLACEHOLDERS

Placeholders (%)s) bepalen de locatie waar de waarde van de variabele in een string wordt weergegeven.

Placeholder	Omschrijving
%s	format voor een string
%i of %d	format voor een integer (number)
%f	format voor een floating point of decimal
%c	css toevoegen



Open de folder op je pc aan met de naam **javascript**. Daarin maak je terug een folder aan met de naam **5\_placeholders**.

```
> var begroeting="Goeiedag", persoon = "Tom";
< undefined
> console.log("%s, %s!", begroeting, persoon);
Goeiedag, Tom!
```

Iedere tag van een html pagina kan in de console.log worden weergeven. Wanneer je de body tag wil weergeven dan doe je dit als volgt:

console.log(document.body)

Voorbeeld:

```
console.log("%cFull stack developers", "color:blue");
Full stack developers
```

# Operatoren

## REKENKUNDIGE OPERATOREN

De basis operatoren zijn:

- deling
- vermenigvuldiging
- optelling
- aftrekking
- Modulus (rest)



Open de folder op je pc aan met de naam **javascript**. Daarin maak je terug een folder aan met de naam **6\_rekenkundige\_operatoren**.

```
let getal1, getal2, quotient, verschil, som, product;
getal1 = 8;
getal2 = 4;

som = getal1+getal2;
verschil = getal1 - getal2;
product = getal1 * getal2;
quotient = getal1 / getal2;
rest = getal1 % getal2;

console.log('Som: ' + som);
console.log('Verschil: ' + verschil);
console.log('Product:' + product);
console.log('Quotient:' + quotient);
console.log('Rest:' + rest);
//Voorrang haakjes, rekenregels
console.log(getal1 + getal2 * getal1);
console.log((getal1 + getal2) * getal1);
```

Som: 12
Verschil: 4
Product: 32
Quotient: 2

## LOGISCHE OPERATOREN

Bij een logische operator kunnen we 2 of meerdere variabelen met elkaar vergelijken. Het resultaat ervan zal een boolean datatype zijn, m.a.w. true or false.

Bijvoorbeeld: wanneer getal 1 > getal 2 dan krijg je **true** als resultaat.

Hieronder heb je een overzicht van alle mogelijk logische operatoren die we kunnen gebruiken

OPERATOR	BESCHRIJVING
&&	(=AND) retourneert true als beide waarden voldoen aan de conditie Voorbeeld: waarde1=0 && waarde2 =0
	(= OR) retourneert true als één van beide waarden voldoen aan de conditie Voorbeeld: waarde1=0 && waarde2 =0
!	als de operand true is wordt er false teruggegeven. Als de operand false is, wordt er true weergegeven
==	gelijk aan
===	gelijk aan en hetzelfde datatype
!=	verschillend van
!==	verschillend van en verschillend van data type
>	groter dan
<	kleiner dan
>=	groter of gelijk aan
<=	kleiner of gelijk aan

OPMERKING: maak er een algemene regel van om steeds 3 gelijkheidstekens te gebruiken ipv 2. Controleer steeds datatype en value! Dit is best practice.



Open de folder op je pc aan met de naam **javascript**. Daarin maak je terug een folder aan met de naam **7\_logische\_operatoren**. In dit voorbeeld zie je duidelijk het verschil tussen het gebruik van 3 gelijkheidstekens of 2 gelijkheidstekens.

```
getal1= 2;  
getal2= 3;  
let grootsteKleinste = getal2 > getal1;  
console.log(grootsteKleinste); //true  
  
let getal3 = '5';  
let getal4 = 5;  
let gelijk1 = getal3 == getal4; //true  
console.log(gelijk1);  
let gelijk2 = getal3 === getal4;  
console.log(gelijk2); //false
```

# HOOFDSTUK 2: CONTROLESTRUCTUREN

**D**e controlestructuren zijn de basis bouwstenen van ieder programma die in iedere programmeertaal voorkomt. Deze controlestructuren kent iedere developer wereldwijd. Na verloop van tijd zullen deze controlestructuren een automatisme worden in je dagelijkse opdrachten.

## If / Else If / Else

Met dit statement wordt een beslissing gevraagd binnen je code. Voorbeeld: als **getal1 > getal1** dan voer je bepaalde **code** uit. De tegenhanger is de **else**. De **IF** staat voor het true gedeelte, de **ELSE** staat voor het false gedeelte;

Notatie in de code:

```
if(conditie)
```

```
{ waar}
```

```
else
```

```
{ onwaar}
```



Open de folder op je pc aan met de naam **javascript**. Daarin maak je terug een folder aan met de naam **8\_if\_elseif\_else**.

```
let getal5 = prompt('Geef een eerste getal in:');
let getal6 = prompt('Geef een tweede getal in:');
if(getal5>getal6){
    console.log(getal5 + ' is groter dan ' + getal6);
}else{
    console.log(getal5 + ' is kleiner dan ' + getal6);
}

let naam= prompt('Geef uw naam in:');
let beroep = prompt('Geef uw beroep in, maak een keuze: bediende, arbeider, werkzoekend');
if(beroep == 'bediende'){
    console.log('Het beroep van ' + naam + ' is ' + beroep);
}else if(beroep == 'arbeider'){
    console.log('Het beroep van ' + naam + ' is ' + beroep);
```

```
}else{
    console.log('Het beroep van ' + naam + ' is ' + beroep);
}
```

5 is kleiner dan 6
Het beroep van Tom is bediende

OPMERKING:

#### SHORTHAND NOTATIE (CONDITIONAL OPERATOR)

Een if statement kan je ook verkort schrijven. Na het vraagteken komt het **true** gedeelte, na het dubbelpunt komt het **false** gedeelte.

Voorbeeld:

```
(conditie) ? waar : onwaar;

getal5>getal6 ?
    console.log(getal5 + ' is groter dan ' + getal6) :
    console.log(getal5 + ' is kleiner dan ' + getal6);
```

## Switch

Wanneer je meer dan 3 keuze mogelijkheden hebt, dan is een switch statement gemakkelijker in gebruik in vergelijking met het if - else if - else statement.

Het **switch** statement onderzoekt de waarde **die** een variabele bevat en voert code uit **die** voldoet aan **die** waarde. Iedere **case** stelt een mogelijk waarde voor, **die** dan ook zijn eigen code bevat. Wanneer de code wordt uitgevoerd, wordt door een **break** het programma beeindigd. Wanneer geen enkele **case** voldoet aan de variabele dan wordt **default** uitgevoerd.