

Inhoudsopgave: Datum:

[1 Hoofdstuk 1: JavaScript, de basis 2](#_Toc158723974)

[1.1 Situering 2](#_Toc158723975)

[1.2 Aan de slag met JS - Code structuur 2](#_Toc158723976)

[1.3 Bouwstenen: variabelen 3](#_Toc158723977)

[1.4 Bouwstenen: basis datatypes 4](#_Toc158723978)

[1.5 Bouwstenen: wrapper objects 6](#_Toc158723979)

[1.6 Bouwstenen: Math & Date object 7](#_Toc158723980)

[1.7 Controlestructuren 8](#_Toc158723981)

[1.8 Operatoren 8](#_Toc158723982)

[1.9 Javascript debuggen 8](#_Toc158723983)

[2 Hoofdstuk 2: Functies – Arrays - Modules 9](#_Toc158723984)

[2.1 Functies 9](#_Toc158723985)

[2.2 Arrays 11](#_Toc158723986)

[2.3 Modules 14](#_Toc158723987)

[3 Hoofdstuk 3: Objecten en functies 17](#_Toc158723988)

# Hoofdstuk 1: JavaScript, de basis

## Situering

Afbeelding met schermopname, Multimediasoftware, software, Kleurrijkheid

Automatisch gegenereerde beschrijvingPopulariteit:

Samen met Python is **JavaScript** de meest populaire programmeertaal die er is.

Omschrijving JavaScript:

JavaScript® (afkorting: **JS**) is een licht **tweetalige, geïnterpreteerde, objectgeoriënteerde taal met eersteklas functies**, en is het best bekend als de scripttaal voor webpagina's, maar wordt ook in veel andere browsers gebruikt. JS is gestandaardiseerd in de **ECMA – 262** specificatie

Omschrijving ECMAscript:

ECMAScript is een objectgeoriënteerde programmeertaal voor het **uitvoeren van berekeningen en het manipuleren van rekenobjecten** binnen een hostomgeving.

Afbeelding met tekst, schermopname

Automatisch gegenereerde beschrijvingFunctie JavaScript:

Samen met HTML en CSS zorgt JavaScript ervoor dat de website **dynamisch** is met **functionaliteiten** zoals het drukken van een knop.

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijvingJava vs JavaScript:

* beide zijn **object georiënteerde** programmeertalen
* JavaScript is **sterk beïnvloed** door functionele programmeertalen
* **functies zijn first-class** citizens in JavaScript
  + spoiler: functies zijn ook objecten

## Aan de slag met JS - Code structuur

JavaScript toevoegen aan de website:

JavaScriptcode toevoegen doen we via het **<script>** element met een **src=""** waarbij we een link kunnen leggen naar een **apart bestand**.

Tekst toevoegen (twee manieren):

* **Console.log("de inhoud die je wil typen in de console")**
  + Schrijft tekst in de *console* van de website.
* **Document.getElementById("naamId").textContent = "de inhoud van dit element";**
  + Voegt tekst toe als inhoud van het aangeduide *element*.

alert("");

Voegt een venster toe bij het openen van de website.

Comments:

Commentaar zetten van **één** lijn: //

Commentaar zetten van **meerdere** lijnen: /\* \*/

"strict mode";

* deze moet **HELEMAAL BOVENAAN** staan in de JavaScriptcode
  + Je kan dit aanroepen met **“use strict”;**
* Het is veiliger en beter om JS te runnen in **‘strict mode’**
  + Sommige JS silent errors werpen nu **exceptions**
  + De JS engine kan **betere** **optimisatie** doen
  + **Verbied het gebruik van syntax** die nog niet officieel in de ecmascript werd opgenomen
* Vanaf **classes** en **modules** moeten we dit niet meer typen

## Bouwstenen: variabelen

Naamgeving variabelen:

* De naam van een variabele mag enkel **letters, cijfers en de symbolen $ en \_** bevatten en de naam mag niet starten met een cijfer
* Gebruik **camelcasing**
* ! Hoofdlettergevoelig !

twee soorten variabelen:

* **const (final in java):**
  + variabelen die je **moet** **initialiseren bij declaratie** en die nadien niet meer van waarde kunnen veranderen
* **let:** 
  + variabelen waarvan de waarde nog kan veranderen.

Declaratie en assignment (initialisatie):

**Declaratie =** naamgeving variabele

**Assignment (initialisatie) =** waarde van de variabele

Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijving

Gebruik zoveel mogelijk const, **bepert** het gebruik van let.

## Bouwstenen: basis datatypes

JavaScript kent acht verschillende datatypes:

* Bij het maken van een variabele **(let of const)** moeten we geen datatype toekennen, maar hij heeft er wel één.
* Dit kunnen we checken door ***console.log(typeof naamVariabele);*** te typen.

|  |  |
| --- | --- |
| DataType: | *Omschrijving:* |
| Number | Getallen |
| Bigint | Hele grote getallen |
| String | Tekst |
| Boolean | Logische waarden **true/false** |
| Null | Voor **ongekende** waarden |
| Undefined | Voor **niet toegekende** waarden |
| Object | Voor meer **complexe datastructuren** |
| Symbol | Voor **unieke letters/cijfers** |

JavaScript is dynamisch:

* Afbeelding met tekst, schermopname, Lettertype, nummer

  Automatisch gegenereerde beschrijvingJavaScript is dynamisch wat betekent dat types worden bepaalt via de **runtime**
* één enkele variabele kan daardoor meerdere types hebben per run van de website.

JS is a loosely typed language:

* Afbeelding met tekst, Lettertype, lijn, schermopname

  Automatisch gegenereerde beschrijvingEr wordt redelijk los omgesprongen met het type van een variabele
* Er gebeuren veel **impliciete type conversies**

Datatype Number:

* Afbeelding met tekst, schermopname, Lettertype, nummer

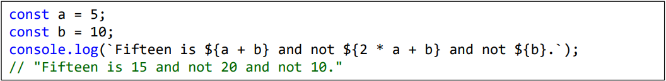
  Automatisch gegenereerde beschrijvingGehele én floating point getallen
* 64-bit double precision floating point representation
  + Gehele getallen: **[-(253 - 1), 253 – 1]**
* Integer literals
  + Decimaal: 3, 10, 10000000, 10\_000\_000, …
  + Prefix **0x** voor **hexadecimaal**
  + Prefix **0o** voor **octaal**
  + Prefix **0b** voor **binair**
* Floating point literals **[digits][.digits][(e|e)[(+|-)]digits]**
  + 3.14, 2345.789, .33333333333333, 6.02e23
* Basis operatoren : **+, - , \*, /, %**
* Kommagetallen
  + Je hebt exact **17** getallen na de komma
* Speciale waarden
  + **NaN (Not a Number)**
    - om iets dat niet kan voorgesteld worden als getal te beduiden
      * bv. resultaat wanneer je een string wil omzetten naar een getal maar de string bevat geen geldige representatie voor een getal
      * bv. resultaat wanneer je de vierkantswortel van -1 berekent
    - **isNan()** is een boolse functie die detecteert of een **waarde NaN is of niet**
  + **Infinity** 
    - een berekening die een getal retourneert buiten het waardenbereik, retourneert **Infinity of -Infinity** 
      * bv. resultaat van 3 / 0, of van 1010000 o
    - **isFinite()** is een boolse functie die detecteert of een waarde **finite is of niet**
  + **string – number conversie functies** 
    - parseInt(aString [, radix])
      * **converteert string naar geheel getal** 
        + leading spaces in de string worden genegeerd
        + eerste beduidende positie moet +,- of cijfer zijn
        + de radix is optioneel
      * parseFloat(aString)
        + **converteert string naar floating point**
      * beide doorlopen de string tot ze het eerste ongeldige karakter tegenkomen

Datatype String:

* **Algemeen**
* bevat een **sequentie** **van** 0 of meer Unicode **karakters** vervat in ‘ of “
* Afbeelding met tekst, schermopname, Lettertype, nummer

  Automatisch gegenereerde beschrijvingopening en closing quote stijl moet dezelfde zijn
* gebruik \ voor escape sequences
  + \”, \’, \n, \\
* **+ operator: concatenatie van 2 strings**
* merk op: er is geen type char voor één enkel karakter voor te stellen
* **ToString()**
  + Afbeelding met tekst, Lettertype, lijn, schermopname

    Automatisch gegenereerde beschrijvingconversie naar een string
  + retourneert elke andere waarde naar een string
* **Template literals strings** 
  + **Kort: wiskunde doen in Strings**
  + Die variabelen & expressies bevatten.
  + Worden omsloten door backticks `expressie`
  + Multiline strings (zonder \n) zijn mogelijk
  + Expressies in de string evalueren: **${expressie}**

****

Datatype Null:

* Bevat enkel de **waarde null**
  + Staat voor een **lege object pointer** **(geen waarde ingevoerd)**
  + Typeof null retourneert “object”
    - Dit is een foutje dat in javascript sloop...
  + Evalueert false in een boolean expressie

Datatype bigint:

om gehele getallen voor te stellen die **buiten het bereik** van number vallen **< -253 –1 of > 253 -1** gebruik de suffix n voor bigint literals.



Datatype Undefined:

* Bevat enkel de speciale waarde **undefined**
  + Staat voor een onbekende waarde
    - Een variabele waaraan **nog geen waarde** is toegekend bevat de waarde **undefined**
    - Een functie in js retourneert steeds een waarde, indien er **niet** expliciet een **waarde wordt geretourneerd** dan wordt er impliciet **undefined** geretourneerd

Datatype Object:

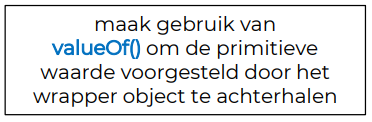
Datatype object is een **voorstelling** van meer **complexe datastructuren**

Datatype Symbol:

* Datatype symbol is om **unieke identifiers voor te stellen**:
  + kan gebruikt worden om bv. **unieke namen** te hebben voor **properties** in objecten
  + zullen we niet verder behandelen in deze cursus

## Bouwstenen: wrapper objects

Wrapper object:

* is een object dat **een primitief datatype ‘omsluit’ bestaat voor elk primitief datatype** behalve **null en undefined**
  + String
  + Number
  + BigInt
  + Boolean
* het wrapper object geeft ons de mogelijkheid **properties en methodes te gebruiken**
  + properties - eigenschappen
  + methodes – functies

Wrapper object Number:

* JavaScript past impliciete conversie toe, het primitieve getal **wordt automatisch omgezet** naar een **Number object** als je een methode van Number oproept,
  + **voorbeelden van methodes:** 
    - toFixed(x) : afronden tot x cijfers na de komma
    - toString(x) : string representatie, x is de radix, default 10
  + **voorbeelden van properties** 
    - Number.MIN\_VALUE : constante voor kleinste getal (5e-324)
    - Number.MAX\_VALUE : grootste getal (-1.7976931348623157e+308)
    - Number.NaN : idem als NaN, maar Property van Number

Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijvingWrapper object Boolean:

de functie **Boolean(x)** cast x naar een **Boolean**

Wrapper object String:

* **String()** 
  + Type-casting met String() roept in feite tostring() aan maar **één verschil** is dat String() ook een **null of undefined kan omzetten zonder fout te genereren**:
  + Afbeelding met tekst, Lettertype, schermopname, lijn

    Automatisch gegenereerde beschrijvingHet primitief wrapper object String bevat heel wat handige **methodes:**

Methodes wrapper object String:

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

Wrapper object BigInt:

Je kan de functie BigInt(x) gebruiken om een **bigint te maken** een bigint is gelijkaardig aan een **number**, maar er zijn ook **significante verschillen.**

Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijving

## Bouwstenen: Math & Date object

Math-object:

* Is een built-in object dat properties en methodes bevat voor wiskundige constanten en functies bevat enkel static properties en methods.
  + Gebruik steeds math. Om te refereren naar een property/methode.
* Afbeelding met tekst, Lettertype, schermopname, lijn

  Automatisch gegenereerde beschrijving**Math.round() - Math.trunc()** 
  + afronden of afkappen
* **Math.max() - Math.min()** 
  + grootste/kleinste getal
* **Math.random()** 
  + pseudo random getal tussen 0 (incl) en 1(excl)

Date-object:

* **Houdt datums bij in milliseconden sinds 1/1/1970 UTC**
  + Creatie van een date object:

**Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving**

* + Enkele methodes:

**Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving**

## Controlestructuren

Alle controlestructuren:

* Selectie: if - else / switch -case
* Iteratie: while / do..while /
* Iteraties onderbreken: break/continue

Afbeelding met tekst, Lettertype, schermopname, nummer

Automatisch gegenereerde beschrijvingAccolades:

Code wordt afgebakend met accolades.

selectie:

* if (..) {} en if(..) {} else if (..) …
  + Afbeelding met tekst, schermopname, Lettertype, nummer

    Automatisch gegenereerde beschrijving
* switch (...) case ...
  + Afbeelding met tekst, schermopname

    Automatisch gegenereerde beschrijving

## Operatoren

Berekeningsoperatoren:

+ - \* / % ++ -- \*\*(macht) unary - unary +

Toewijzingsoperatoren:

= \*= /= %= += -=

Vergelijkingsoperatoren:

== != === !== > >= <=

Logische operatoren:

&& || ! ?? String operatoren : + en +=

Conditionele operatoren:

condition ? ifTrue : ifFalse

## Javascript debuggen

# Hoofdstuk 2: Functies – Arrays - Modules

## Functies

Defintie functie:

* Een **functie in JavaScript** bevat een verzameling **beweringen die een taak uitvoeren**.
* Functies zijn een van de fundamentele bouwstenen in JavaScript. Het **neemt** wat **invoer** en **geeft** een **uitvoer** terug waarbij er een duidelijke relatie is tussen de invoer en de uitvoer.
* Om een functie te gebruiken, moet je deze ergens definiëren in het bereik van waaruit je de functie wilt oproepen.

Nut van functies:

* functies leiden tot **herbruikbare code**
* laten toe om **dubbele code** te **elimineren**
* laten toe om oplossingen voor **grotere problemen** op te delen **in** **kleine stukjes**

Skelet functie + oproepen van een functie:

Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving

**Een functie heeft altijd een return waarde!**

Doorgeven van parameters in een functie:

Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijving

* **Wordt een bepaalde parameter niet ingevuld? Waarde = undefined**
* Waarden van de parameters gaan van **links naar rechts**

Default parameters:

Door gebruik van default parameters kan je vermijden om instellingen voor **parameters in de body van de functie te doen.**

Afbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijving

Hoisting van functies:

* Hoisting is een JavaScript-mechanisme waarbij declaraties worden verplaatst **naar de top van hun scope voordat code wordt uitgevoerd**.
* Als gevolg daarvan kun je een hoisted declaratie gebruiken in zijn scope voor de regel waarop hij is gedeclareerd.

Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving

Het is dus mogelijk om functies zowel **boven als onder aan te roepen**. En het zal nog steeds geldige uitvoer geven.

Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijvingHoisting van let/const variables

* Er is een andere vorm van hoisting **voor de let/const declaratie** van variabelen
* Deze vorm van hoisting laat niet toe **dat je een variabele gebruikt vóór zijn declaratie!**

Predefined functions:

Javascript kent verschillend top-level built-in functions:

* Isnan
* Parseint
* Pars

globalThis:

* JavaScript’s globalThis
  + deze property bevat de globale **this** waarde, het verwijst naar het globale object
  + dit is een object die **steeds is gedefinieerd** en bestaat in de **global** scope
  + in browser omgevingen is deze globalThis gelijk aan **window**
  + dit is dus een referentie naar het **browser venster** die het DOM bevat
* indien je methodes of properties van dit globale object wenst te gebruiken hoef je deze **niet te laten voorafgaan** door **globalThis. of window**.
* Afbeelding met tekst, schermopname, Lettertype, nummer

  Automatisch gegenereerde beschrijvinghet **gebruik van methodes** **van window** lijkt dan ook sterk op het gebruik van **built-in functies**

Dit betekent dat we functies kunnen casten naar een **window** op de website.

* Afbeelding met tekst, Lettertype, schermopname, lijn

  Automatisch gegenereerde beschrijvingFuncties kunnen ook worden gedeclareerd als functie **expressies**.
* De declaratie van een functie wordt toegekend aan een **variabele.**

## Arrays

Array:

* Afbeelding met schermopname, Lettertype, tekst, lijn

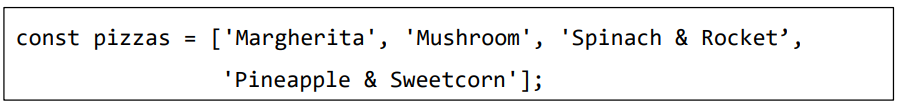
  Automatisch gegenereerde beschrijvingEen array is een **geordende verzameling van elementen**, deze mogen van een verschillend type zijn.
* Elk element heeft een **genummerde positie in de array**, index genaamd: het eerste element heeft **index 0**.
* Arrays zijn **dynamisch**: er wordt geen grootte gegeven bij creatie, **ze kunnen groeien/krimpen**
* **Array notatie: [ ]**

Declaratie van een array (2 manieren):

Arrays kunnen ook via een **literal notatie** worden **gedeclareerd** en **geinitializeerd.**

Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving



Toekennen van een waarde aan een array: Wijzigen van een arraywaarde:

Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving

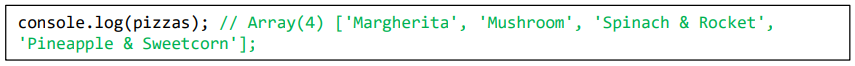
Raadplegen van informatie aan de hand van een index:

* Index start **vanaf 0**, gebruik [ ] **-** notatie
* **Length**: aantal elementen in array

Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijving

Weergeven van een volledig array:



Itereren over een array (2 manieren):

Afbeelding met tekst, Lettertype, lijn, wit

Automatisch gegenereerde beschrijving

Afbeelding met tekst, lijn, Lettertype, schermopname

Automatisch gegenereerde beschrijving

Afbeelding met tekst, Lettertype, schermopname, Post-it-briefje

Automatisch gegenereerde beschrijvingElementen verwijderen (pop):

*‘Pop’* verwijdert altijd het **laatste element** uit de array.



Elementen verwijderen (shift):

Afbeelding met tekst, Lettertype, Post-it-briefje, Rechthoek

Automatisch gegenereerde beschrijvingVerwijdert de **eerste waarde** in array en returnt deze waarde



Elementen toevoegen aan een array (push):

Afbeelding met tekst, Lettertype, schermopname, Post-it-briefje

Automatisch gegenereerde beschrijvingVoegt één of meerdere waarden toe **aan het** **einde van de array** en retourneert de nieuwe lengte van de array.



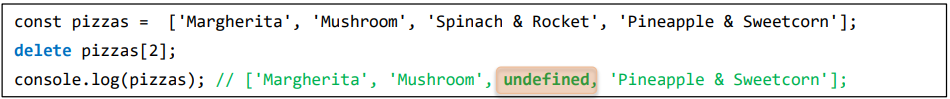
Elementen toevoegen aan een array (unshift):

Afbeelding met tekst, Lettertype, schermopname, Rechthoek

Automatisch gegenereerde beschrijvingVoegt één of meerdere waarden toe **aan het** **begin van de array** en retourneert de nieuwe lengte van de array.



Gaten in een array:

* Er wordt een element toegevoegd op index 30, maar daarvoor staan enkel 0, 1, 2 en 3.
* Afbeelding met tekst, Lettertype, lijn, schermopname

  Automatisch gegenereerde beschrijvingHiertussen staan geen andere indexen.
* Gebruik van de delete operator

Zoeken van indexen aan de hand van de waarde van een bepaald element:

* Het **zoeken** van een **index** van een bepaalde waarde gebeurt met **indexOf**.
  + Retourneert index van eerste voorkomen of -1 als waarde niet voorkomt.

Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving

Extra functies arrays:

* **concat()** 
  + voegt 2 arrays samen
* **reverse()** 
  + Volgorde elementen omdraaien
* **slice(start\_index, upto\_index)** 
  + returnt een nieuwe array als een stuk van de oorspronkelijke array met als argumenten de begin- en een eindpositie.
* **splice(start\_index, numberofitemsToRemove, waarde1,…, waardex)** 
  + verwijdert numberofItemsToRemove waarden uit de array startend op positie start\_index en voegt dan de nieuwe waarden toe waarde1,… waardex
* **sort()** 
  + sorteert de elementen in de array
* **indexOf(searchElement[, fromIndex])** 
  + de index van het eerste voorkomen van het element vanaf fromIndex
* **lastIndexOf(searchElement[, fromIndex])** 
  + idem indexOf maar begint achteraan
* **join()** 
  + converteert alle elementen van een array tot 1 lange string

Afbeelding met tekst, Lettertype, nummer, lijn

Automatisch gegenereerde beschrijving

Destructuring assignment:

Een manier om meerdere waarden te extraheren uit een array en toe te kennen aan variabelen.

Afbeelding met tekst, Lettertype, lijn, schermopname

Automatisch gegenereerde beschrijving

Meerdimensionale arrays (matrixen en boven):

Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving

Uitgebreid voorbeeld meerdimensionale matrixen:

Afbeelding met tekst, nummer, Lettertype, software

Automatisch gegenereerde beschrijving

## Modules

Definitie modules:

* Zolang **scripts** **beperkt** zijn in size is één js-bestand voldoende voor je applicatie.
* Indien de **code** **uitgebreid en complex** (meerdere functies – klassen(zie later)) wordt, kan het gebruik van **aparte bestanden** interessant worden om je code overzichtelijk en mooi gestructureerd te houden.
* Ook het hergebruik van klassen en functies kan het gebruik van meerdere aparte bestanden interessant zijn.
* **Zo een apart bestand is een module.**
* Een **module** is een js-bestand.
* Modules kunnen elkaar laden, daarvoor wordt gebruik gemaakt van de directives:
  + **Export**: variabelen, functies, klassen **worden toegankelijk buiten de huidige module**
  + **Import**: staat toe om **functionaliteit** (variabelen, functies, klassen) van andere modules te **importeren en gebruiken**

Voorbeelden modules:

Afbeelding met tekst, Lettertype, lijn, schermopname

Automatisch gegenereerde beschrijving

* In het script element wordt enkel naar main.js verwezen
* Je moet **expliciet aanduiden** dat je met modules werkt**: type=“module”**

Afbeelding met tekst, Lettertype, lijn, schermopname

Automatisch gegenereerde beschrijving

Vervolg modules:

* Modules werken steeds in **strict mode**
* Elke module heeft zijn **eigen scope** (top-level) o variabelen, functies en klassen zijn niet automatisch zichtbaar voor ander modules
* **Export** directive
  + Om aan te geven wat je toegankelijk wil maken voor andere modules
* **Import** directive
  + Om aan te geven wat je wil gebruiken uit een andere module
* Een module wordt **één keer geëvalueerd**, namelijk bij het importeren

Afbeelding met tekst, schermopname, tekenfilm, Lettertype

Automatisch gegenereerde beschrijvingVervolg voorbeelden modules:

Afbeelding met tekst, Lettertype, lijn, schermopname

Automatisch gegenereerde beschrijvingModules: export:

* Export wordt vaak voor de declaraties geplaatst.
* Export mag ook na de declaraties geplaatst worden.

Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijvingAfbeelding met tekst, Lettertype, nummer, lijn

Automatisch gegenereerde beschrijving

Modules: import:

* Import wordt gebruikt om elementen (variabelen, functies, klassen, …) uit een andere module te gebruiken.
* Specifieer deze in een **lijst** (tussen accolades)
* De relatieve verwijzing (begint met ./) naar de module
* Import kan ook met een \* gebruikt worden.
* Dit gebeurt als er **veel moet geïmporteerd worden**.
* Je moet dan een **alias** (=object) voor \* ingeven (na keyword **as**).
* Je kan dan de **geïmporteerde functionaliteiten** benaderen als een **object**.

Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving Afbeelding met tekst, Lettertype, schermopname, lijn

Automatisch gegenereerde beschrijving

Modules: As:

As: je kan zowel voor de **import als de export een alias gebruiken** voor de geëxporteerde/geïmporteerde functionaliteiten.

Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijving

Modules: export default

* Vaak wordt een klasse in een apart bestand geplaatst en beschikbaar gesteld via een export.
  + Het gebruik van de accolades bij de import is dan een beetje overbodig en verwarrend
* Hiervoor wordt de **default export** geïntroduceerd.
  + Let op: er kan maar één default export gebruikt worden per js-bestand.

# Hoofdstuk 3: Objecten en functies