Hoofdstuk 1: Variabelen

Doel van variabelen

Variabelen komen voor in elke programmeertaal en worden gebruikt om **data bij te houden** en te manipuleren tijdens de uitvoering van het programma. Variabelen geven elk stuk data **in het geheugen** een duidelijke naam zodat de code van een programma gemakkelijker te lezen is door de developer.

Variabelen gebruiken

Vooraleer je een variabele kan gebruiken dien je eerst de variabele te **declareren**. Dit betekent dat je de variabele een unieke zelfgekozen naam geeft. Na - of tijdens - de declaratie kan je een **waarde toekennen** aan de variabele. Je kan deze waarde later **aanpassen** of een meerdere keren **ophalen**.

Een veelgebruikte vergelijking is dat variabelen in een programmeertaal werken als containers of dozen. Elke variabele is een container of doos. Je kan er een waarde instoppen, deze waarde er later terug uithalen en je kan ze aanpassen.

Variabelen in JavaScript

Declareren

Om een variabele te declareren in JavaScript maak je gebruik van het keyword const of let gevolgd door de naam van de variabele die je zelf kiest. Er is een groot verschil tussen const en let:

- **const**: gebruik je voor een variabele waarvan je de waarde later **niet meer** wenst te **wijzigen**. Je **initialiseert** dus direct ook de variabele door een waarde toe te kennen.
- **let**: gebruik je voor een variabele waarvan je later de waarde wel **nog wenst** te **wijzigen**. Je kan deze variabele direct **initialiseren**, maar dit is **niet verplicht**.

OPMERKING 1: het lijkt vreemd dat je een variabele aanmaakt om nooit meer te wijzigen, maar in de praktijk zal je merken dat de meeste variabelen enkel een initiële waarde nodig hebben.

OPMERKING 2: in JavaScript kan je ook nog het keyword var gebruiken om een variabele te declareren. Niettegenstaande je dit nog in veel online demo's en tutorials zal zien gebruiken we var niet meer in deze cursus. Dit heeft te maken met scope (zie later bij hoofdstuk 4: functies).

Waarde toekennen

Je kent een waarde toe aan een variabele door het **= teken** te gebruiken. Bij const moet je dit direct doen bij de declaratie, bij let kan je dit later nog doen en kan je de waarde ook nog wijzigen. Je kan de demo bekijken en aanpassen via https://codepen.io/fdc/pen/BaBWxZd

```
// variabele met directe verplichte initialisatie
const naam = `Frederik`;
// declaratie van een variabele zonder initialisatie
let leeftijd;
// waarde toekennen aan de variabele
leeftijd = 27;
// andere waarde toekennen
leeftijd = 37;
// declaratie met directe -niet verplichte - initialisatie
let stad = `Kortrijk`
```

Nog enkele opmerkingen bij de bovenstaande code:

- Je merkt dat in JavaScript elke regel code afgesloten wordt door een ";".
- Om een lijn commentaar te schrijven in JavaScript gebruik je //. Als je een volledige blok code in commentaar wil plaatsen gebruik je /* dit is commentaar */.
- Je merkt dat tekst tussen backticks (Bijvoorbeeld: `Kortrijk`) staat. Hier gaan we dieper op in bij het onderdeel datatypes en template literals.

Pas de pen aan zodat de variabele *naam* een andere waarde krijgt. Op het eerste zicht lukt dit, maar als je de inspector van de browser opent en in het tabblad "console" kijkt merk je dat dit een **foutmelding** geeft. Dit komt omdat de variabele *naam* aangemaakt is via het keyword const en dus niet meer gewijzigd kan worden.

```
Uncaught TypeError: Assignment to constant variable.
    at pen.js:4
```

In hoofdstuk 3 rond debugging gaan we dieper in op foutmeldingen en hoe ze op te lossen.

Waarde ophalen

De waarde van een variabele ophalen kan je doen door de naam van de variabele in te geven. Aangezien je tot nu toe nog niet veel JavaScript commando's en functies gezien hebt log je de waarde van de variabele voorlopig in de console. Dit doe je als volgt:

```
// variabele met directe verplichte initialisatie
const naam = `Frederik`;
console.log(naam);
// …
```

Open https://codepen.io/fdc/pen/pozeKPM en open de inspector van de browser. Ga naar het tabblad console en bekijk de output.

```
Frederik
undefined
27
37
Kortrijk
```

Merk op dat de variabele leeftijd eerst de waarde ***undefined*** krijgt. Dit betekent dat de variabele aangemaakt is, maar nog geen waarde heeft gekregen.

Naamgeving

Er zijn in JavaScript een paar regels omtrent de naamgeving van variabelen:

- Alle **letters en cijfers zijn toegelaten** net als een underscore (_) en een dollar-teken ($)
- Een variabelenaam mag **niet starten met een cijfer**
- Elke naam moet **uniek** zijn
- Als de naam uit meerdere woorden bestaat gebruik je **camelCase**. Bijvoorbeeld: dayOfBirth of mainColor
- Een variabelenaam kan **geen keyword** van JavaScript zijn. Een keyword is een woord dat in JavaScript een speciale betekenis heeft. Je hebt er al enkele gezien: const, let en var. De volledige lijst kan je vinden op MDN.

Daarnaast denk je best goed na over een **duidelijke naam** voor je variabele. Je wil namelijk dat voor iedereen direct duidelijk is waar een variabele precies voor dient. Variabelenamen als *data, getal, tekst, content,...* vertellen niets over je variabele. Je kiest beter voor een naam als *leeftijd, achtergrondKleur, lijstVanFilms,...*

Je bent vrij om te kiezen of je de voorkeur geeft aan **engelstalige** of **nederlandstalige** variabelenamen. Wees wel consequent en begin niet plots af te wisselen.

Datatypes

Elke variabele bevat een bepaald type data. Je hebt al gemerkt dat er soms een stuk tekst in opgeslagen wordt of een getal. JavaScript kent zeven verschillende types data. Binnen deze cursus beperken we ons tot de vier voornaamste:

- **Boolean**: is een variabele die true of false / waar of onwaar / 1 of 0 bevat
- **Number**: een variabele die een getal bevat. Dit kan een geheel getal of een kommagetal zijn
- **String**: een variabele die een stuk tekst bevat. De tekst kan tussen single quotes, double quotes of backticks komen. In deze cursus kiezen we meestal voor backticks.
- **Undefined**: is een variabele die geen waarde bevat

Type ophalen

Je kan via de operator typeof opvragen wat het **type** van je variabele is. De demo op https://codepen.io/fdc/pen/XWrMBWg toont een voorbeeld van de vier voornaamste datatypes.

JavaScript is dynamically typed

Dynamically typed betekent dat **JavaScript zelf bepaalt** wat het datatype van een variabele is. Dit type kan **wijzigen** tijdens de uitvoering van het programma zoals je in de vorige demo gemerkt hebt met de variabele *leeftijd*. Eerst had de variabele *undefined* als type, na het toekennen van een waarde werd dit *number*.
Pas de code aan zodat *leeftijd* een stuk tekst bevat. Vraag dan nogmaals het type op en je merkt dat dit aangepast is naar string.

Wiskundige operatoren

De vier wiskundige **basisbewerkingen** zijn **mogelijk** in JavaScript. Je kan waarden optellen (+), aftrekken (-), vermenigvuldigen (*) en delen (/). Er zijn ook enkele **shortcuts** mogelijk die je kan bekijken op https://codepen.io/fdc/pen/NWKpBrq. De demo toont enkel het optellen van waarden. De andere operaties verlopen gelijkaardig. Test andere bewerkingen ook eens uit.

Weakly typed

JavaScript is naast dynamically typed ook weakly typed. Dit betekent dat JavaScript **afhankelijk** van de **context** het datatype kan **aanpassen**. Test onderstaand voorbeeld uit:

```
const result = 1 + `2`;
console.log(result);
```

Het resultaat zal 12 opleveren. Je probeert namelijk een number op te tellen met een string wat een fout zou opleveren. Daarom zet JavaScript het cijfer 1 automatisch om naar een string. Het + teken voegt de twee strings samen wat 12 oplevert.

Parsen

Het feit dat JavaScript weakly typed is kan soms problemen opleveren. Daarom heb je de mogelijkheid om via de functie parseInt() een **stuk tekst om te zetten naar een getal**. Dit wordt getoond in volgende demo https://codepen.io/fdc/pen/ExYWpmQ. Indien je een kommagetal dient te parsen maak je gebruik van de functie parseFloat(). Deze werkt gelijkaardig aan parseInt().

Indien de string die je wenst om te zetten geen getal bevat zal het resultaat NaN (not a number) bevatten. Ironisch genoeg is deze variabele dan wel van het type number. Probeer dit via codepen.

Template literals

Naast het optellen van twee getallen kan je de + operator ook gebruiken om **meerdere strings met elkaar te verbinden**. Het is echter aan te raden om hiervoor template literals te gebruiken. Dit maakt de code een stuk leesbaarder.
Een template literal is een string tussen **backticks** die gebruik maakt van **placeholders** om de variabele waarden in te vullen. Een placeholder geef je aan met ${}. Binnen die placeholders kan je niet enkel variabelen oproepen, maar ook bewerkingen uitvoeren. Bekijk de demo op https://codepen.io/fdc/pen/vYBxavB.

Oefeningen 1: Variabelen

Maar eerst nog dit

De oefeningen binnen deze cursus worden altijd gemaakt met VS Code. Een tool als codepen is schitterend om een kleine demo te maken en te delen, maar niet om een volledig programma in op te bouwen.
Om JavaScript in de browser uit te voeren is echter wel een HTML pagina nodig waar het script aan gelinkt kan worden. Dit komt nog ruim aan bod in *hoofdstuk 7: Document Object Model*. Weet voorlopig dat elke oefening start via een HTML pagina waar via deze lijn code een JavaScript bestand aan gelinkt is.

```
<script src="js/script.js"></script>
```

Plaats deze lijn code steeds als laatste stukje code voor de afsluitende </body> tag. Op die manier wordt eerst alle HTML gerenderd en dan pas de JavaScript code uitgevoerd. Deze volgorde is ook belangrijk wanneer je begint met de HTML structuur te manipuleren via JavaScript. Het openen van de HTML pagina zal de uitvoering van het script starten.

Oefening 1: Hello Weather

Maak een nieuw HTML en JavaScript document aan en ga aan de slag in het JS document.
Open je console in de browser en check of er geen fouten optreden.

1. Initialisatie

Maak variabelen aan waarin we volgende info kunnen opslaan. Denk na over een gepaste naamgeving.

a. De stad/gemeente waar je je op dit moment in bevindt (je huidige locatie)
b. De postcode
c. De huidige temperatuur in Celcius
d. Het type weer (bewolkt, zonnig, onweer, ….)
e. Of het momenteel regent.

Beantwoord volgende vragen:

a. Koos je voor let, const of var? Waarom?
b. Maakte je gebruik van hoofdletters, speciale tekens, cijfers? Zoja, waar en waarom?
c. Koos je een duidelijke, bondige naamgeving?

2. Toekenning

Ken gepaste, willekeurige waarden toe aan deze variabelen. Laat de eenheden (°Celcius, …) achterwege zodanig dat bepaalde variabelen enkel een cijfer bevatten.

a. Toon de inhoud van deze variabelen in de console.
b. Toon het datatype van elke variabele. Welke types onderscheid je?

3. Aanpassen en uitbreiden

a. Stel, het begint te regenen. Pas de variabele aan.
b. Overschrijf via een nieuwe lijn code de waarde van de stad naar “Londen”, en postcode “EC1A”.
c. Maak een tweede stad “Parijs” aan, die altijd ongewijzigd zal blijven. Hou ook de temperatuur bij voor Parijs en toon beide variabelen in je console.

Beantwoord volgende vragen:

a. Moet er bij het hernoemen naar “Londen” opnieuw let of const geschreven worden? waarom?
b. Wat is de Type nu van de postcode van Londen?
c. Hoeveel verschillende types heeft deze postcode-variabele al gehad doorheen het script?
d. Koos je let of const om “Parijs” in op te slaan? Waarom?
e. Wat gebeurt er als je “Parijs” toch tracht aan te passen?
f. Koos je een gepaste, unieke naamgeving voor je nieuwe variabelen stad en temperatuur)?

4. Wiskundige operatoren

a. Maak een nieuwe variabele waarin je het temperatuurverschil tussen Londen en Parijs berekent en opslaat. Toon dit verschil in je console.

5. Template Literals

a. Maak gebruik van template literals om volgende boodschappen in de console weer te geven:
   - “In Londen is het bewolkt en 22°”
   - “De postcode van Londen is EC1A”
   - “In Londen is het 2° warmer dan in Parijs”.

Oefening 2: Mathness

Maak een nieuw HTML en JavaScript document aan en ga aan de slag in het JS document.
Open je console in de browser en check of er geen fouten optreden.

1. Initialiseer twee nieuwe variabelen: num1 en num2 en ken er respectievelijk de waarden 2 en 8 aan toe.
   a. Toon in je console achtereenvolgens volgende cijfers, door (enkel) deze twee variabelen in combinatie met wiskundige operatoren te gebruiken: 10, 4, 16, -6.
   b. Hoe verhoog je de waarde van num1, met 1?

i. Er zijn 3 mogelijke notaties. Test deze uit.
ii. Hoeveel notaties blijven er over als je deze met 2 wil verhogen?

c. Bonus: hoe krijg je, enkel door deze twee variabelen te gebruiken, "28" in je console?

Oefening 3: Template Literals

In een nieuw HTML + js document:

1. Instantieer een variabele myName en stel die in op jouw naam.
2. Instantieer een variabele myCity en stel die in op jouw stad.
3. Gebruik een template literal om de volgende zin weer te geven in de console.
   My name is <NAME>. My hometown is <CITY>.

Oefening 4: You're not my type

In een nieuw HTML + js document:

1. Booleans
   a. Instantieer een variabele atHome en stel deze in op true. Toon dit in de console.
   b. verander vervolgens de waarde naar false en controleer opnieuw.
   c. Van welk type is deze variabele?
2. Not a Number
   1. Declareer een variabele: let nan = `2`;
   2. Toon in je console het type en de inhoud van de variabele.
   3. Zet deze variabele om naar een number.
   4. Toon opnieuw het type en de inhoud in je console.

Oefening 5: CYA

Context: Voor deze oefening kan je uitgaan van een warenhuis-webshop waar je JS functionaliteit aan wil toevoegen.
Maak een nieuw HTML en JavaScript document aan en ga aan de slag in het JS document:

a. Declareer een variabele die het aantal items in je *shopping cart* bijhoudt. Denk na over een gepaste variabelenaam.
   i. *Welke naam koos je en waarom is dit een goede keuze?*
   ii. *Koos je let of const om de variabele te declareren? Waarom?*
b. Toon nu de inhoud van deze variabele in de console.
   i. Wat is het resultaat in de console?
c. Ken de waarde 7 toe aan deze variabele en toon dit opnieuw in de console.
   i. *Wat is het datatype van deze variabele en hoe kom ik dit te weten?*
d. Declareer een tweede variabele aan "bestSellingProduct" en ken er de waarde "Dr Oetker Mozarella" aan toe. Toon dit in de console.
e. Verander op een nieuwe lijn de inhoud van deze variabele in "Dr Oetker Pepperoni" en toon opnieuw in de console.
   i. *Kies je let of const voor deze variabele?*

   ii. *Wat is hier het datatype?*

f. Maak een volgende variabele aan waarin je de stukprijs (2,03 euro) opslaat.
g. Zorg ook dat je de totaalprijs kan bijhouden.
h. Voeg nu 5 items toe aan je winkelmandje en bereken de totaalprijs.
i. toon in je console de boodschap aan de hand van een template literal:
   Het winkelmandje bevat 5x Dr Oetker Pepperoni (2.03euro/stuk) voor een totaalbedrag van 10,15euro).