Module 4: JavaScript 2

Introductie

In de module JavaScript 1 (module 2 van de leerijn BASIS1) hebben we geleerd HOE je met JavaScript code moet schrijven (syntax), wat variabelen zijn en hoe je ze moet gebruiken (declareren en initialiseren) en bewerken en hoe je hiermee berekeningen kan maken. Tevens heb je nu door, hoe je uit een HTML-invoerveld de gegevens van de gebruiker kan opvragen en verwerken door middel van een variabele. Als laatste heb je geleerd hoe je met beslissingstructuren (if-else) je code kan laten reageren op invoer van de gebruiker.

Mocht bovenstaande nog niet (helemaal) duidelijk zijn dan kan je het lesmateriaal van JavaScript nog eens doornemen en gebruiken om deze module tot een succes te maken.

In dit vervolg gaan we verder met JavaScript en behandelen we de volgende technieken:

- Herhalingslussen maken en toepassen (‘for‘ en ‘while’).
- Een array maken en de gegevens binnen een array manipuleren en uitlezen.

We gaan ons eerst, door middel van kleine oefeningen, deze technieken eigen maken. Daarna ga je aan de slag met een aantal opdrachten waarin je de geleerde technieken zelf gaat toepassen.

Beoordeling

In deze module worden de volgende leerdoelen beoordeeld:

Opdracht	Leerdoel	Taal	Gekoppeld werkproces
Opdracht 1	Ik benoem 2 situaties waarin ik een for-lus kan gebruiken.	JavaScript	B1-K1-W3
Opdracht 1	Ik benoem 2 situaties waarin ik een while-lus kan gebruiken.	JavaScript	B1-K1-W3
Opdracht 2	Ik optimaliseer mijn JavaScript code door herhalende code te verwerken in een for-lus.	JavaScript	B1-K1-W3
Opdracht 3	Ik optimaliseer mijn JavaScript code door herhalende code te verwerken in een while-lus.	JavaScript	B1-K1-W3
Opdracht 4	Ik gebruik JavaScript om een div-element te maken in een HTML-pagina.	JavaScript	B1-K1-W3
Opdracht 5	Ik gebruik arrays om meerdere gegevens in 1 variabele op te slaan.	JavaScript	B1-K1-W3
Opdracht 6	Ik gebruik een lus om gegevens uit een array in een HTML-pagina te tonen.	JavaScript	B1-K1-W3

Tijdsplanning

Week 1	
2 uur	LUSSEN: Theorie en oefeningen 1 en 2
6 uur	Werken aan opdrachten 1,2,3,4
Week 2	
2 uur	Arrays: Theorie en oefeningen 3 en 5
6 uur	Werken aan opdrachten 5, 6 en verdieping 6

Week 1 - Herhalingslussen

THEORIE

In de vorige module heb je geleerd HOE je JavaScript-code moet schrijven, maar soms wil je een stukje code meerdere keren, op ongeveer dezelfde manier, uitvoeren. Kijk maar eens naar het volgende voorbeeld.

Voorbeeld 1a:

```
"use strict";
var leeftijd = 16;
document.getElementById("info").innerHTML += "Je bent nu " + leeftijd + " jaar.<br/>";
leeftijd++;
document.getElementById("info").innerHTML += "Je bent nu " + leeftijd + " jaar.<br/>";
leeftijd++;
document.getElementById("info").innerHTML += "Je bent nu " + leeftijd + " jaar.<br/>";
leeftijd++;
document.getElementById("info").innerHTML += "Je bent nu " + leeftijd + " jaar.<br/>";
leeftijd++;
document.getElementById("info").innerHTML += "Je bent nu " + leeftijd + " jaar.<br/>";
```

Het is niet nodig om zo vaak dezelfde code te schrijven.
We gebruiken een **herhalingslus** (**“loop**”) om de code in te korten.
Er zijn twee methodes om een herhalingslus te schrijven: "FOR" en "WHILE"

FOR of WHILE?

FOR:

De ‘for-lus’ heeft een vaste syntax.

Voorbeeld 1b:

```
for (var leeftijd = 16; leeftijd < 21; leeftijd += 2)
{
    document.getElementById("info").innerHTML += "Je bent nu " + leeftijd + " jaar.<br/>";
}
```

In een “for-lus” gebruik je een variabele. Bijvoorbeeld ‘leeftijd’.

Van die variabele moet je drie dingen vastleggen:

- De beginwaarde
  - *var leeftijd = 16*
- De voorwaarde - Hoe lang blijf je in de lus
  - *leeftijd < 21* (dus: zolang de leeftijd kleiner is dan 21)
- De stapgrootte - Met welke waarde wordt de leeftijd verhoogd
  - *leeftijd ++* (dus: elke keer één hoger)

WHILE:

Bij de ‘while-lus’ leg je dezelfde dingen vast, alleen op een andere manier.

Voorbeeld 1c:

```
var leeftijd = 16
while (leeftijd < 21)
{
    leeftijd++;
    document.getElementById("info").innerHTML += "Je bent nu " + leeftijd + " jaar.<br/>";
}
```

Ook hier zie je de drie stappen terug:

Beginwaarde – voorwaarde – stapgrootte
Maar: in een andere volgorde! Dit is van belang.

Een groot verschil tussen de twee herhalingslussen is dat:

- Bij een ‘for-lus’ het aantal keren van de herhaling van tevoren vaststaat.
- Bij een while-lus kan deze bijvoorbeeld afhankelijk zijn van de invoer van de gebruiker.

Op internet is meer informatie over herhalingslussen te vinden, bijvoorbeeld:

- https://JavaScript.info/while-for
- https://www.w3schools.com/js/js_loop_while.asp
- https://youtu.be/E6xX51Zcrl8

Oefening 1 For loop

Maak een map aan genaamd “JavaScript2”. In de map maak je een map genaamd “Week1” en hierin maak je twee bestanden, oefening1.html en oefening1.js. Koppel de JavaScript aan het HTML-bestand.

Console log je naam het aantal keer als je oud bent. Dus als je 17 jaar bent, console log je je naam 17 keer. Je mag in je script maar 1x console.log() gebruiken en je maakt gebruik van de for loop.

Oefening 2 While loop

In de “Week1” map maak je twee nieuwe bestanden, oefening2.html en oefening2.js. Koppel de JavaScript aan het HTML-bestand.

Console log je naam het aantal keer als je oud bent. Dus als je 17 jaar bent, console log je je naam 17 keer. Je mag in je script maar 1x console.log() gebruiken en je maakt gebruik van de while loop.

Opdracht 1 Onderzoek

Voorbeeld 1: Om een webshop te vullen gebruikt een developer de while loop. De while loop wordt gebruikt om alle artikelen uit de database op het scherm te zetten zodat de developer dit niet handmatig 1 voor 1 hoeft te doen. “Zo lang er artikelen in de database staan. Zet ze op het scherm neer”.

Ga op zoek naar hoe en wanneer je een for en/of while loop gebruikt. Maak een HTML pagina aan, opdracht1.html en beschrijf in deze pagina wanneer je als developer een for of while loop gebruikt. Beschrijf 2 situaties voor de for loop en 2 situaties voor de while loop. Het voorbeeld telt niet mee.

Opdracht 2 For loop

In de map maak je twee bestanden, opdracht2.html en opdracht2.js. Koppel de JavaScript aan het HTML-bestand.

In het JavaScript bestand maak je een variabele (let) aan en vul het met een getal. Wanneer de gebruiker de website opent komt de tekst “Ik feliciteer je x keer” op het scherm. Het aantal keer dat de tekst op het scherm komt is op basis van de variabele. En x is het getal hoe vaak je iemand feliciteert. Dit getal loopt op. Voor deze opdrache maak je gebruik van een for loop.

Voorbeeld: Gebruiker vult het getal 5 in. Dan komt de tekst 5 keer op het scherm op de volgende manier.
Ik feliciteer je 1 keer.
Ik feliciteer je 2 keer.
Ik feliciteer je 3 keer.
Ik feliciteer je 4 keer.
Ik feliciteer je 5 keer.

Opdracht 3 While loop

In de map maak je twee bestanden, opdracht3.html en opdracht3.js. Koppel de JavaScript aan het HTML-bestand.

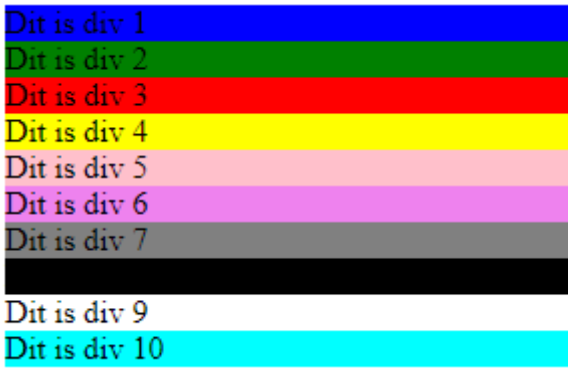
Los opdracht 2 op door middel van een while loop in plaats van een for loop.

Opdracht 4

Maak 10 divjes aan met behulp van JavaScript. Geef alle divjes ook een unieke ID met JavaScript. Als laatste geef je elke div een unieke tekst en kleur. Het geven van de kleur aan de divs mag wel handmatig via css. De rest moet via een loop in JavaScript.

***Hulp**: Om een div aan te maken met JavaScript gebruik je de create element functie let div = document.createElement('div');*
Om een ID mee te geven gebruik je de attribute functie: div.setAttribute("id", "div1");

Mogelijke uitwerking



Week 2 - ARRAYS

THEORIE

Computers bewerken vaak hele lijsten met gegevens tegelijk. Om dat mogelijk te maken zijn er **arrays**.
Een array is niks anders dan een hele lijst van variabelen achter elkaar. Stel dat je de leden van een sportvereniging wil bijhouden, dan kan je wel een heleboel variabelen maken (lid1, lid2, lid3, etc.) maar dat is erg onhandig. Je kan eigenlijk beter een variabele maken waar weer een heleboel andere variabelen in bewaard kunnen worden. Zo'n variabele heet een array, een lijst met gegevens dus. In JavaScript kun je op de volgende manier een array aanmaken:

```
var mijnArray = new Array();
```

Nu is het natuurlijk nog een lege lijst; er moeten nog waardes ingevuld worden. Dat kan doormiddel van de functie push. Hiermee voeg je een waarde toe aan het einde van de array.

```
var mijnArray = new Array();
mijnArray.push(11);     // Voeg de waarde 11 toe aan mijnArray
mijnArray.push(27);     // Voeg de waarde 27 toe aan mijnArray
mijnArray.push(8);      // Voeg de waarde 8 toe aan mijnArray
```

Er is ook en snellere manier om dit te schrijven. Je kan namelijk de array gelijk al wat waardes meegeven. Dit gaat als volgt:

```
var mijnArray = [11, 27, 8]; // Zet in mijnArray de waardes 11, 27, 8
```

De array ‘mijnArray’ bevat nu dus 3 waardes, namelijk 11, 27 en 8 (in die volgorde). Elke waarde in de array heeft een eigen nummertje, een eigen id. Dat nummertje of id wordt een key genoemd. Met die key heb je toegang tot de waarde die daarbij hoort. Deze keys werken heel simpel: de eerste waarde heeft key 0, de tweede waarde heeft key 1, de derde key 2, etc. **Let dus wel op dat het tellen bij 0 begint, en niet bij 1!** Je zou de array mijnArray dus als volgt kunnen voorstellen:

key	value
0	11
1	27
2	8

Om een waarde, of zoals één zo'n waarde in een array wordt genoemd: element, via zo'n key op te vragen type je eerst de naam van de array: mijnArray in dit geval. Daarna zet je de key tussen blokhaken (dus tussen [ ]).

```
alert(mijnArray[1]);    // Laat het 2e(!) element uit mijnArray zien
```

Een waarde die je opvraagt met een key gedraagt zich als een gewone variabele. Je kan er dus bewerkingen op uitvoeren (optellen, vermenigvuldigen, etc.) en je kan hem wijzigen.

```
// Vraag een nieuwe waarde en sla deze op in het 2e element uit mijnArray
mijnArray[1] = parseInt(prompt("Geef een nieuwe waarde in", mijnArray[1]));
// Op elementen in een array kunnen gewoon bewerkingen uitgevoerd worden
mijnArray[2] = mijnArray[0] + mijnArray[1];
alert(mijnArray[0] + " + " + mijnArray[1] + " = " + mijnArray[2]);
```

EEN ARRAY DOORLOPEN

Bij arrays gebruik je vaak loops, omdat je voor elk element dezelfde code wil uitvoeren. Je doorloopt de array dan met een loop. Er zijn veel redenen waarom dat nodig kan zijn, je kan bijvoorbeeld naar een bepaald element zoeken of je wil bij elk element een bepaalde code uitvoeren. Het doorlopen van een array kan het best met een for-loop. Daarbij maak je gebruik van de eigenschap length, daarmee vraag je namelijk het aantal elementen in de array op.

```
var lengte = mijnArray.length;              // Bepaal de lengte (aantal elementen)van mijnArray
for(var i=0; i<lengte; ++i)                 // Doorloop de array van key = 0 tot de laatste key
{                                           // Daarbij stelt i de huidige key voor
    document.write(mijnArray[i]+"<br/>");   // Zet het element dat bij de key i hoort op het scherm
}
```

Merk op dat het laatste element in de array de key mijnArray.length-1 heeft, want length geeft het aantal elementen aan. Omdat er bij 0 begonnen wordt met het tellen van de keys, is de laatste key dus altijd mijnArray.length-1 (tenzij er geen elementen in de array zitten). Vandaar dat in de for-lus het ‘kleiner dan’ teken (‘<’) wordt gebruikt in plaats van ‘kleiner dan of gelijk aan’ (‘<=’).

Meer informatie over JavaScript arrays vind je hier:

- https://www.w3schools.com/js/js_arrays.asp
- https://javascript.info/array
- https://youtu.be/3tmpNF0BUdI

ASSOCIATIEVE ARRAYS

Je kan ook je eigen keys bepalen en in plaats van getallen namen gebruiken als key. Zoiets heet een associatieve array.

```
var mijnArray = new Array();     // Maak een lege array aan
mijnArray[15] = "vijftien";      // Key: 15, waarde: "vijftien"
mijnArray["leeftijd"] = 42;      // Key: "leeftijd", waarde: 42
mijnArray["naam"] = "Truus";     // Key: "naam", waarde: "Truus"
mijnArray[12] = 144;             // Key 12, waarde: 144
```

Nu staan er in mijnArray dus 4 waardes. Ieder met zijn eigen key:

key	value
15	"vijftien"
leeftijd	42
naam	"Truus"
12	144

Het opvragen van de waardes gaat hetzelfde als bij een normale array, maar wanneer je een string als key gebruikt, moeten er tussen de blokhaken ook aanhalingstekens gebruikt worden:

```
alert(mijnArray["leeftijd"]);    // geeft een alert-box met: 42
```

DE FOR-IN LOOP

Een nadeel van zo'n associatieve array is dat hij niet op een normale manier doorgelopen kan worden. De keys zijn namelijk niet netjes genummerd. Daarvoor is de for-in-loop bedacht. Die ziet er als volgt uit:

```
for(variabele in associatieve_array)
{
    // Code die uitgevoerd moet worden
}
```

Hierbij krijgt een variabele telkens de waarde van een key uit de array totdat alle keys geweest zijn. Zo kan de hele array alsnog doorgelopen worden. Merk op dat dit ook werkt voor gewone arrays, dus niet alleen voor associatieve arrays.

```
<script type="text/javascript">
  var mijnArray = new Array();     // Maak een lege array aan
  mijnArray[15] = "vijftien";      // Key: 15, waarde: "vijftien"
  mijnArray["leeftijd"] = 42;      // Key: "leeftijd", waarde: 42
  mijnArray["naam"] = "Truus";     // Key: "naam", waarde: "Truus"
  mijnArray[12] = 144;             // Key 12, waarde: 144

  for(key in mijnArray)            //Laat alle keys en waardes zien
  {
     document.write(key + ": " + mijnArray[key] + "<br />");
  }
</script>
```

Oefening 3

Maak een HTML-pagina aan en sla deze op als ‘oefening3.html’.
Maak in JavaScript een array aan met jouw favoriete games. Toon al de games op het scherm met gebruik van een loop.

Oefening 4

Maak een HTML-pagina aan en sla deze op als ‘oefening4.html’. Zet een div op het scherm met een breedte en hoogte van 400px en een knop.

Maak in JavaScript een array aan met 10 namen van kleuren. Als je op de knop klikt veranderd de div van kleur. Het is de bedoeling dat de volgende kleur uit de array gebruikt wordt. Als het einde van de array is bereikt is de volgende kleur weer de eerste kleur uit de array.

Opdracht 5

Maak een HTML-pagina aan en sla deze op als ‘opdracht5.html’.
Maak in JavaScript een array aan en zet daar namen van tenminste 4 landen per werelddeel. Zet in de HTML de werelddelen als div op het scherm. Via JavaScript ga je de juiste landen bij de juiste werelddeel-div plaatsen. **Gebruik hiervoor een loop!**

Europa: Nederland / België /
Zuid Amerika: Panama / Mexio /
Azie: Japan / Korea /
Afrika: Zuid Afrika / Nigeria /

Opdracht 6

Maak een HTML-pagina aan en sla deze op als ‘opdracht6.html’. Ga opzoek naar 10 foto’s die bij software development horen en zet deze in een map.
Maak in JavaScript een array aan en zet daar de namen in van de foto’s. Zet alle 10 de foto’s op het scherm via een loop. Daarbij zet je de naam uit de array ook bij elke foto erbij.

Tip: zoek naar “Create img with js”

Opdracht 6 Verdieping

Zorg dat de foto’s van opdracht 6 in een GRID layout worden getoond.