JS 2 - (DOM1)

1 Introductievideo

datasource

? Leerdoelen

- Je weet wat de DOM is.
- Je weet dat je met JavaScript het DOM kan lezen en kan aanpassen.

? Uitleg

Bekijk de volgende video

https://www.roc.ovh/link/902#bkmrk-6-formulieren-in-html

https://www.youtube.com/embed/NO5kUNxGIu0

Code

Gebruik dit als start-code.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Simple Page</title>
  <style>body{font-family:Arial,sans-serif;background-color:#f0f4f8;padding:60px;text-
align:center}.container{background-color:#fff;padding:40px;border-radius:10px;box-shadow:0 4px
12px rgba(0,0,0,0.1);max-width:600px;margin:auto}h1{color:#2c3e50}p{color:#555;font-
size:1.1em}</style>
```

```
</head>
<body>

<div class="container">
  <h1 id="header">Embrace the Simplicity</h1>
  <p>Sometimes, all you need is a quiet page, a clear message, and a bit of whitespace. Let your thoughts breathe and focus on what truly matters.</p>
</div>

</body>
</html>
<script>

</script>
```

?? Opdracht

Bekijk de video daarin worden dingen voorgendaan, die jij ook moet toepassen.

1. Neem de start-code over in het bestand `dom1.html` en
2. Op regel 18 plaats code waarmee je de `backgroundColor` van de body grijs (grey) maakt.
3. Plaats vervolgens JavaScript code om tekst van de header met id "header" aan te passen naar:
   "Hallo slimme student".
4. In de video wordt dit beiden voorgedaan!

Je ziet aan het eind dus:



? Inleveren

- De aangepaste code (`dom1.html` bestand).

2 Elementen ophalen en aanpassen

? Leerdoelen

- Je weet wat de DOM is.
- Je kunt HTML-elementen selecteren met JavaScript.
- Je kunt de inhoud en stijl van elementen aanpassen via JavaScript.

? Uitleg

De DOM (Document Object Model) is de structuur van je HTML-document zoals de browser die begrijpt. Met JavaScript kun je deze structuur lezen en aanpassen.

Voorbeeld – een paragraaf veranderen:

```
<p id="mijnParagraaf">Oude tekst</p>
<script>
  const p = document.getElementById("mijnParagraaf");
  p.textContent = "Nieuwe tekst!";
  p.style.color = "blue";
</script>
```

?? Opdracht – Tekst aanpassen

1. Maak een nieuw bestand aan met de naam `dom2.html`.
2. Begin met de start-code (hieronder)
3. Als je op de knop klikt, moet de tekst in de paragraaf veranderen naar iets anders (bijv. “Hallo wereld!”).

Start code

```
<!DOCTYPE html>
<html lang="nl">
<head>
  <meta charset="UTF-8">
  <title>Voorbeeld</title>
```

```
</head>
<body>

<p id="mijnParagraaf">Dit is de originele tekst.</p>
<button onclick="verander()">Verander</button>

<script>
  function verander() {
    // plaats hier de code die de inhoud van de pragraaf veranderd.
  }
</script>

</body>
</html>
```

? Reflectie

- Wat is de DOM in eigen woorden?
- Wat doet `getElementById` precies?
- Waarom zou je de stijl van een element met JavaScript aanpassen en niet met CSS?

? Inleveren

- Lever je bestand `dom2.html` in.
- Beantwoord de reflectievragen in een .txt of .pdf bestand en lever deze mee in.

3 Meerdere elementen aanpakken

? Leerdoelen

- Je kunt meerdere elementen selecteren met `querySelectorAll`.
- Je kunt met `forEach` een actie uitvoeren op elk element.
- Je kunt een class toevoegen of verwijderen met `classList`.

? Uitleg

Als je meerdere elementen tegelijk wilt aanpakken (zoals alle <p>-elementen of alle knoppen), gebruik je `querySelectorAll`. Dit geeft je een lijst (een zogenaamde “NodeList”) van alle elementen die matchen.

Met `forEach` kun je vervolgens over deze lijst heen lopen en elk element iets laten doen:

```
<p>Item 1</p>
<p>Item 2</p>
<p>Item 3</p>

<button onclick="verander()">Verander</button>

<script>
  function verander() {
      const alleP = document.querySelectorAll("p");
      alleP.forEach(function(p) {
        p.style.color = "green";
      });
    }
</script>
```

Je kunt ook classes toevoegen of weghalen met `classList`:

```
p.classList.add("actief");
p.classList.remove("verborgen");
p.classList.toggle("geselecteerd");
```

?? Opdracht – items markeren

1. Maak een bestand `dom3.html`.
2. Maak een lijst van minimaal 5 <p>-elementen met een class `item`.
3. Maak een knop met de tekst “Markeer alles”.
4. Wanneer je op de knop klikt, moeten alle <p class="item"> elementen de class `geselecteerd` krijgen.
5. Als je weer op de knop drukt, dan moet de <p> weer terug veranderen (tip: gebuik **toggle**!).

6. Gebruik de voorbeeld code en pas die zelf aan.

Voorbeeld (gebruik zelf andere items)

Item 1, vliegtuig Item 2, trein Item 3, auto Item 4, motor Item 5, fiets Verander	Item 1, vliegtuig Item 2, trein Item 3, auto Item 4, motor Item 5, fiets Verander

? Reflectie

- Wat doet `querySelectorAll(".item")` precies?
- Wat is het verschil tussen `getElementById` en `querySelectorAll`?
- Waarom gebruik je `forEach` bij een NodeList?
- Wat doet `p.classList.toggle("geselecteerd");` ?

? Inleveren

- Lever het bestand `dom3.html` in.
- Beantwoord de reflectievragen in een .txt of .pdf bestand en lever die ook in.

4 Interactie met knoppen en events

? Leerdoelen

- Je begrijpt wat een event is in JavaScript.
- Je kunt reageren op een klik of muisactie met `addEventListener`.
- Je kunt een actie koppelen aan meerdere elementen.

? Uitleg

Een event is iets wat gebeurt in de browser: een klik, het bewegen van de muis, een toets indrukken…

Met `addEventListener` kun je zeggen: “Als dit gebeurt, doe dan dat.”

```
<button id="klikMij">Klik mij (addEventListener)</button>

<script>
  const knop = document.getElementById("klikMij");
  knop.addEventListener("click", function() {
    alert("Je klikte op de knop!");
  });
</script>
```

Dit doet hetzelde als:

```
<button onclick="toonMelding()">Klik mij (via functie)</button>

<script>
  function toonMelding() {
    alert("Je klikte op de knop!");
  }
</script>
```

Waarom zou je eventlisteners gebruiken?

Denk eerst zelf na en klik dan open!

Voordelen:

Scheidt structuur (HTML) van **gedrag (JS)** → netter.

Je kunt **meerdere functies** aan één event koppelen.

Flexibeler: makkelijk dynamisch events toevoegen of verwijderen.

Werkt ook bij elementen die pas later op de pagina verschijnen (bijv. via JavaScript geladen).

Nadelen:

Iets meer code nodig.

Niet altijd direct zichtbaar in de HTML wat er gebeurt (iets minder beginner-vriendelijk).

Andere events

Ook andere events zijn mogelijk, zoals `mouseover`, `mouseout`, `keydown` enzovoort.

Je kunt ook meerdere elementen selecteren en daar een event aan koppelen:

```
document.querySelectorAll(".kleurvak").forEach(function(el) {
  el.addEventListener("mouseover", function() {
    el.style.backgroundColor = "yellow";
  });
});
```

Wat doet dit? Denk eerst na en controleer dan door hier te klikken.

Elk vak met de class "*kleurvak*" krijgt een gele achtergrond kleur als je er met de muis overheen gaat.

?? Opdracht – Events in actie

1. Maak een nieuw HTML-bestand `dom4.html`.
2. Voeg 5 divjes toe met de class `kleurvak`, elk met een vaste afmeting en een andere begin-kleur.
3. Als je met de muis over een vakje gaat, verandert de achtergrondkleur in geel.
4. Als je erop klikt, moet de tekst in het vakje veranderen naar “Geklikt!”.

Gebruik zowel `mouseover` als `click` events.

Gebruik dit als basis voor de CSS:

```
  <style>
    .kleurvak {
      width: 100px;
```

```
      height: 100px;
      display: inline-block;
      margin: 10px;
      text-align: center;
      line-height: 100px;
      background-color: lightblue;
      font-weight: bold;
      cursor: pointer;
    }
  </style>
```

Resultaat

https://youtu.be/deDyTevLTgk

https://www.youtube.com/embed/deDyTevLTgk

? Reflectie

- Wat is een event in je eigen woorden?
- Wat doet `addEventListener` precies?
- Wat is het verschil tussen `mouseover` en `click`?

? Inleveren

- Lever je bestand `dom4.html` in.
- Beantwoord de reflectievragen in een .txt of .pdf bestand en lever die ook in.

5 Elementen toevoegen met JavaScript

? Leerdoelen

- Je kunt een nieuw HTML-element aanmaken met `createElement`.

- Je kunt dat element toevoegen aan de DOM met `appendChild`.
- Je kunt invoer van de gebruiker gebruiken om dynamisch iets te maken.

? Uitleg

Je kunt nieuwe HTML-elementen maken en ze toevoegen aan je pagina met JavaScript. Dit is handig als je bijvoorbeeld automatisch lijstjes wilt uitbreiden of reacties wilt tonen.

```
const nieuwElement = document.createElement("p");
nieuwElement.textContent = "Hallo, ik ben nieuw!";
document.body.appendChild(nieuwElement);
```

Je kunt ook iets maken op basis van wat de gebruiker invoert:

```
<input type="text" id="tekstvak">
<button id="voegToe">Voeg toe</button>
<div id="resultaat"></div>

<script>
  document.getElementById("voegToe").addEventListener("click", function() {
    const invoer = document.getElementById("tekstvak").value;
    const nieuwP = document.createElement("p");
    nieuwP.textContent = invoer;
    document.getElementById("resultaat").appendChild(nieuwP);
  });
</script>
```

- Op regel **7** wordt de waarde van het HTML elemenet met het id `tekstvlak` gelezen en in de variabel `invoer` gezet.
- Op regel **8** wordt er een nieuwe paragraaf gemaakt.
- Op regel **9** wordt het nieuwe element gevuld met tekst.
- Op regel **10** wordt het nieuwe element toegevoegd aan de de `div` met id *resultaat*

?? Opdracht – Invoer toevoegen

1. Maak een bestand `dom5.html`.
2. Voeg een invoerveld toe waarin de gebruiker een hobby, film of favoriet eten kan typen.

3. Voeg een knop toe met de tekst “Toevoegen”.
4. Telkens wanneer je klikt, moet er een nieuw <p>-element met de ingevoerde tekst verschijnen onder een lijstdiv.
5. Bonus
   Als je het leuk vindt, maak je het invoerveld na het klikken automatisch weer leeg. Gebruik daaarvoor `input.value = ""` om het veld te legen.

? Reflectie

- Wat doet `createElement` precies?
- Wat is het verschil tussen `textContent` en `innerHTML`?
- Waarom moet je `appendChild` gebruiken?

? Inleveren

- Lever je bestand `dom5.html` in.
- Beantwoord de reflectievragen in een .txt of .pdf bestand en lever die ook in.

6 Elementen aanpassen via `event.target`

? Leerdoelen

- Je begrijpt wat `event.target` doet.
- Je kunt een klik koppelen aan een specifiek element dat je wilt aanpassen of verwijderen.
- Je kunt met JavaScript elementen verwijderen uit de DOM.

? Uitleg

Als een event plaatsvindt (zoals een `click`), kun je met `event.target` achterhalen welk element er precies geklikt is.

Voorbeeld

"klikbare lijst waarin een item verdwijnt"

```
<ul id="lijst">
  <li>Appel</li>
  <li>Banaan</li>
  <li>Peer</li>
</ul>

<script>
  document.querySelectorAll("#lijst li").forEach(function(item) {
    item.addEventListener("click", function(event) {
      event.target.remove();
    });
  });
</script>
```

Er wordt een HTML lijst gemaakt en er wordt op elke element van de lijst een click event geplaatst. Dit event verwijderd zichzelf.

?? Opdracht – Klik en verwijder

1. Maak een bestand `dom6.html`.
2. Voeg een lijst toe (bijv.) met minstens 5 items (bijv. films, dieren of snacks).
3. Schrijf JavaScript die ervoor zorgt dat je een item uit de lijst verwijdert zodra je erop klikt.
4. Bonus: Toon boven de lijst hoeveel items er nog over zijn.

Gebruik `event.target.remove()` binnen je event handler.

? Reflectie

- Wat is `event.target` en waar gebruik je het voor?
- Bedenk en beschrijf een nuttige toepassing waarbij je meet `event.target` iets wil aanpassen of verwijderen.

? Inleveren

- Lever je bestand `dom6.html` in.
- Beantwoord de reflectievragen in een .txt of .pdf bestand en lever die ook in.

7 Herhaling – Interactieve favorietenlijst

We gaan een aantal onderdelen die we in de opdrachten hebben behandeld nu combineren en we gaan onze eigen interacteive lijst maken.

Wat gaan we maken?

https://www.youtube.com/embed/CxEQ1B4m-I0

? Leerdoelen

- Je kunt elementen aanmaken, toevoegen en verwijderen met JavaScript.
- Je kunt reageren op events zoals klikken en rechtermuisklikken.
- Je kunt meerdere elementen selecteren en aanpassen.

? Uitleg

In deze opdracht herhaal je alles wat je geleerd hebt in de vorige opdrachten. Je maakt een interactieve lijst waarin je dingen kunt toevoegen, markeren en verwijderen.

?? Opdracht – Favorietenlijst

1. Maak een bestand `dom7.html`.
2. Voeg een **invoerveld** toe waarin de gebruiker een favoriete hobby, snack of film kan typen.
3. Voeg een **knop** toe met de tekst `Voeg toe`.
4. Onder de knop komt een lijst (<ul>) waarin elke keer een **nieuw item** verschijnt als de gebruiker op de knop klikt.

5. Wanneer je op een item klikt, **verandert** de **tekstkleur** of opmaak (gebruik `classList.toggle()`).
6. Wanneer je met de **rechtermuisknop** op een item klikt, moet dat item **verdwijnen** (gebruik `event.preventDefault()`).
7. Boven de lijst staat een **teller** met hoeveel items er in de lijst staan. Deze teller werkt automatisch.

? Reflectie

- Wat heb je allemaal moeten combineren uit de vorige lessen?
- Waarom is `event.preventDefault()` nodig bij het gebruik van de rechtermuisknop?
- Wat is het voordeel van `classList.toggle()` ten opzichte van `add()` en `remove()`?
- Waarom moet je elementen ophalen met `getElementById` of `querySelector` nadat de HTML geladen is?

? Inleveren

- Lever je bestand `dom7.html` in.
- Beantwoord de reflectievragen in een .txt of .pdf bestand en lever die ook in.

8 Debuggen met de console

? Leerdoelen

- Je weet wat de `console` is in de browser.
- Je kunt fouten (errors) in JavaScript herkennen in de console.
- Je kunt `console.log()` gebruiken om waarden te controleren.

? Uitleg

De console is een hulpmiddel in je browser waarmee je kunt zien wat er “onder de motorkap” van je code gebeurt. Je opent de console met **F12** of via **Rechtermuisknop → Inspecteren → Console**.

Als er iets misgaat, verschijnt er een foutmelding (bijvoorbeeld in rood). Vaak vertelt die precies waar de fout zit — bijvoorbeeld in welke regel of welk bestand.

Voorbeeld

```
<script>
  const naam = "Max";
  console.log("De naam is:", naam);

  // Fout voorbeeld
  console.log(voornaam); // ReferenceError: voornaam is not defined
</script>
```

De foutmelding vertelt je dat de variabele `voornaam` niet bestaat. Zo kun je fouten snel vinden en oplossen.

?? Opdracht – Debuggen met de console

1. Maak een nieuw bestand `dom8.html`.
2. Kopieer onderstaande code:

```
<!DOCTYPE html>
<html lang="nl">
<head>
  <meta charset="UTF-8">
  <title>Debug oefening</title>
</head>
<body>
  <h1>Welkom op mijn pagina</h1>
  <p id="tekst">De tekst wordt hieronder aangepast...</p>

  <script>
    const tekstElement = document.getElementById("tekts");
    tekstElement.textContent = "Nieuwe tekst";
    console.log("Code is uitgevoerd!");
  </script>
</body>
</html>
```

3. Open je pagina in de browser.
4. Gebruik de **console (F12)** en zoek de foutmelding.
5. Noteer wat er misgaat en op welke regel de fout verschijnt.
6. Corrigeer de fout en controleer of de tekst nu wel verandert.
7. Gebruik daarna `console.log()` om te controleren of je variabele correct gevuld is.

? Reflectie

- Wat betekent de foutmelding die je kreeg?
- Hoe hielp de console jou om de fout te vinden?
- Wat is het verschil tussen een foutmelding (error) en een melding via `console.log()`?

? Inleveren

- Lever het bestand `dom8.html` in (met de fout opgelost).
- Lever een kort reflectieverslag (.txt of .pdf) waarin je de drie reflectievragen beantwoordt.

Revision #23
Created 2025-06-06 14:06:12 UTC by Max
Updated 2026-07-10 08:49:25 UTC by Max