JS - (DOM1)

1 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 `dom1.html`.
2. Maak hierin een kopje, een paragraaf met een id, en een knop.

3. Als je op de knop klikt, moet de tekst in de paragraaf veranderen naar iets anders (bijv. “Hallo wereld!”).

Gebruik bijvoorbeeld:

```
document.getElementById("knop").addEventListener("click", function() {
  document.getElementById("mijnParagraaf").textContent = "Hallo wereld!";
});
```

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 `dom1.html` in via Teams of Canvas.
- Beantwoord de reflectievragen in een .txt of .pdf bestand en lever deze mee in.

2 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>

<script>
  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 `dom2.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 `actief` krijgen.

Gebruik bijvoorbeeld:

```
document.getElementById("knop").addEventListener("click", function() {
  document.querySelectorAll(".item").forEach(function(el) {
    el.classList.add("actief");
  });
});
```

Stijl de class `actief` in je <style> met bijvoorbeeld een achtergrondkleur of rand.

□□ Reflectie

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

□□ Inleveren

- Lever het bestand `dom2.html` in via Teams of Canvas.
- Beantwoord de reflectievragen in een .txt of .pdf bestand en lever die ook in.

3 Interactie met knoppen en events

□□eerdoelen

- 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.”

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

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";
  });
});
```

□□ Opdracht – Events in actie

1. Maak een nieuw HTML-bestand `dom3.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.

Voorbeeld CSS:

```
.kleurvak {
  width: 100px;
  height: 100px;
  display: inline-block;
  margin: 10px;
  text-align: center;
  line-height: 100px;
  background-color: lightblue;
}
```

□□ 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 `dom3.html` in via Teams of Canvas.
- Beantwoord de reflectievragen in een .txt of .pdf bestand en lever die ook in.

4 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>
```

□□ Opdracht – Invoer toevoegen

1. Maak een bestand `dom4.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.

Bonus: Als je het leuk vindt, laat de invoervelden na het klikken automatisch leeglopen.

□□ Reflectie

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

□□ Inleveren

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

5 Elementen verwijderen of aanpassen via `event.target`

□□eerdoelen

- 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 klik), kun je met `event.target` achterhalen welk element 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>
```

Of met `this` als shorthand:

```
item.addEventListener("click", function() {
  this.remove();
});
```

□□ Opdracht – Klik en verwijder

1. Maak een bestand `dom5.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()` of `this.remove()` binnen je event handler.

□□ Reflectie

- Wat is `event.target` en waar gebruik je het voor?
- Wat is het verschil tussen `event.target` en `this` in een event?
- Waarom zou je een lijst dynamisch willen kunnen aanpassen?

□□ Inleveren

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

6 Klassennamen wisselen met `classList.toggle`

□□eerdoelen

- Je kunt met JavaScript classes toevoegen of verwijderen.
- Je weet wat `classList.toggle` doet.
- Je kunt styling aanpassen afhankelijk van de class van een element.

□□ Uitleg

Met `classList` kun je een class toevoegen, verwijderen of omwisselen (“toggelen”). Dit is handig om styling of gedrag van elementen aan te passen wanneer de gebruiker iets doet.

Bijvoorbeeld: klik op een element om het te markeren:

```
<ul id="taken">
  <li>Boodschappen doen</li>
  <li>Afwassen</li>
  <li>Hond uitlaten</li>
</ul>

<style>
  .afgevinkt {
    text-decoration: line-through;
    color: grey;
  }
</style>

<script>
  document.querySelectorAll("#taken li").forEach(function(taak) {
    taak.addEventListener("click", function() {
      this.classList.toggle("afgevinkt");
    });
  });
</script>
```

□□ Opdracht – Actieve selectie

1. Maak een bestand `dom6.html`.
2. Maak een lijst (bijv. favoriete games, liedjes, sporters).
3. Als je op een item klikt, moet de class `geselecteerd` worden toegevoegd of verwijderd.
4. Stijl de class `geselecteerd` in CSS met bijvoorbeeld een andere kleur en achtergrond.

Voorbeeld CSS:

```
.geselecteerd {
  background-color: lightgreen;
  font-weight: bold;
}
```

Gebruik `element.classList.toggle("geselecteerd")` bij het klikken.

□□ Reflectie

- Wat is het voordeel van `toggle` ten opzichte van `add` en `remove`?
- Wat gebeurt er als je meerdere keren op hetzelfde item klikt?
- Waar zou je dit in een echte webapp kunnen gebruiken?

□□ Inleveren

- Lever het bestand `dom6.html` in via Teams of Canvas.
- Beantwoord de reflectievragen in een .txt of .pdf bestand en lever die ook in.

Revision #7
Created 6 June 2025 14:06:12 by Max
Updated 13 June 2025 19:35:27 by Max