



Oefeningen

JAVASCRIPT ESSENTIALS



Basisconcepten JavaScript	2
Oefening basisconcepten, variabelen en aanspreken van HTML-elementen.....	2
Logica en herhalingen	3
Oefening Tafel van x	3
Oefening Raad een getal van 1 t.e.m. 10	3
Functies en interacties	4
Oefening Tafel van x met functies	4
Events	5
Oefening Kleurenwisselaar	5
Miniproject	6
Oefening Kleurenspeel	6

Oefening **basisconcepten**, **variabelen** en **aanspreken** van HTML-elementen

Open het bestand “**oefening_favorieten_START.html**”.

Je gaat een eenvoudige webpagina maken waarin jouw favoriete dingen (kleur, eten en hobby) worden getoond. Een van de gegevens wordt opgevraagd via een `prompt()`-venster.

1. Maak drie **constante variabelen** aan en geef kleur en hobby een **waarde**:
 - één voor je favoriete kleur
 - één voor je favoriete hobby
 - één voor je favoriete eten (**opvragen** via een `prompt()`-venster)
2. Haal de drie HTML-elementen (paragrafen) op met `document.querySelector()`
3. Vul elk element met de juiste waarde via `.innerHTML`
4. Log eventueel alle waarden overzichtelijk in de **console**
 - Gebruik een duidelijke tekst bij elke log

Mijn favoriete dingen

Rood

Pizza

Programmeren

Controleer in de **console** dat je geen fouten hebt!

Bewaar de oplossing in de map **oefeningen** als **oefening_favorieten.html**

Oefening Tafel van x

Open het bestand “[oefening_tafel_x_START.html](#)”.

Je leert hoe je een for-lus gebruikt om herhalende berekeningen uit te voeren en het resultaat dynamisch toont in een HTML-element.

Plaats de tafel van tien op het scherm, maar nu met de for()-lus.



Bewaar de oplossing in de map **oefeningen** als [oefening_tafel_x.html](#)

Oefening Raad een getal van 1 t.e.m. 10

Open het bestand “[oefening_getal_raden_START.html](#)”.

Schrijf JavaScript-code waarmee de gebruiker een getal tussen 1 en 10 moet raden.

1. Het programma genereert een **willekeurig** getal van 1 t.e.m. 10
p Gebruik **Math.random** en **Math.floor**.
2. De gebruiker krijgt telkens een **prompt** om een gok in te geven.
3. Gebruik een **do-while-lus** zodat het programma **blijft vragen** tot het juiste getal is ingevoerd.
4. Wanneer het juiste getal geraden is, toon je op de webpagina **hoeveel pogingen** nodig waren en wat het **juiste getal** was.

Raad het getal tussen 1 en 10:

☐ Voorkomen dat deze pagina extra dialoogvensters maakt

Annuleren

OK

Raad een getal van 1 t.e.m. 10

Je had 2 pogingen nodig. Het juiste getal was: 6

Bewaar de oplossing in de map **oefeningen** als [oefening_getal_raden.html](#)

Oefening **Tafel van x met functies**

Omschrijving: Je hebt eerder een programma gemaakt dat met een `prompt()` het getal vroeg waarvan de tafel werd getoond. In deze versie gaan we hetzelfde doen, maar **met een invoerveld en een knop op de webpagina**.

Open het bestand "**oefening_tafel_x_functies_START.html**". Pas deze zodanig aan dat de gebruiker de waarde voor de tafel in een **invoerveld** (number) kan ingeven. Bij het **drukken** op de knop "Berekenen" **verschijnt** de tafel van dit getal.

Wat voeg je toe of pas je aan?

1. Gebruik het meegegeven HTML-bestand met een invoerveld, knop en resultaatvak.
2. Haal in JavaScript de elementen op met `getElementById`.
3. Maak een **functie** `toonTafel()` die:
 - Het **resultaatvak** leegt (`innerHTML = ''`)
 - Het **getal** uitleest uit het **invoerveld** (`value`)
 - Met een **for-lus** toon je de tafel van het ingegeven getal in het resultaatvak (= code vorige opdracht).
4. Voeg een **addEventListener** toe aan de **knop** die de functie `toonTafel()` uitvoert bij een klik.

Voer een getal in:

1 x 7 = 7
2 x 7 = 14
3 x 7 = 21
4 x 7 = 28
5 x 7 = 35
6 x 7 = 42
7 x 7 = 49
8 x 7 = 56
9 x 7 = 63
10 x 7 = 70

Uitdaging: Voeg een extra **eventListener** toe die de `toonTafel` functie uitvoert telkens we een **karakter ingeven (input)**.

Uitdaging 2: Controleer of het een **geldig getal** is (getal geen nummer (`isNaN`) is OF leeg is) **anders** toon je een (rode) foutmelding en je stopt de functie (`return`).

Voer een getal in:

Gelieve een geldig getal in te geven

Bewaar de oplossing in de map **oefeningen** als **oefening_tafel_x_functies.html**

Oefening Kleurenwisselaar

Omschrijving: Je krijgt een HTML-bestand met een gekleurd vak (div). In deze oefening reageert het vak op verschillende **muishandelingen** en **toetsenbordinvoer**.

Open het bestand “[oefening_kleurwisselaar_START.html](#)”.

Schrijf de JavaScript-code om volgende **events** te verwerken:

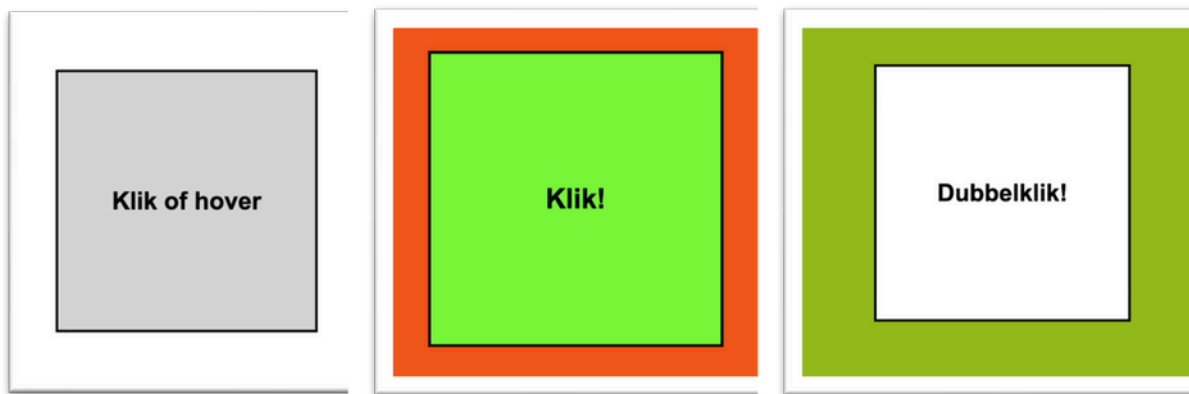
- **Klik op het vak** → verander de **achtergrondkleur van het vak** willekeurig en toon de tekst "Klik!".
- **Dubbelklik op het vak** → zet de achtergrondkleur terug naar **wit** en toon de tekst "Dubbelklik!".
- **Muis over het vak** → toon de tekst "Hovering!" in het vak.
- **Muis uit het vak** → herstel de tekst naar "Klik of hover" in het vak.
- **Toets indrukken (op het toetsenbord)** → verander de **achtergrondkleur van de volledige pagina (body)** willekeurig.

Gebruik de gegeven functie `willekeurigeKleur()` om een willekeurige kleuren te genereren:

Gebruik `addEventListener` om de **juiste events** te koppelen aan het **juiste element**.

Tip

- Je kunt `console.log()` gebruiken om te **testen** of de **eventlisteners** correct werken.



Bewaar de oplossing in de map **oefeningen** als [oefening_kleurenwisselaar.html](#)



Oefening Kleurenspel

Omschrijving: In deze oefening maak je een klein spel waarbij de gebruiker moet raden welke kleur hoort bij een willekeurige kleurcode (zoals #3366FF). Bij elke poging krijgt de speler drie kleurvlakken om uit te kiezen.

Open het bestand “[oefening_kleurenspel_START.html](#)”.

Je krijgt een **HTML-bestand** waarin de basis structuur klaarstaat, de CSS voor de opmaak en de functie [randomMakkelijkeKleur\(\)](#) die een (makkelijke) willekeurige kleur genereert.

Vul het JavaScript-gedeelte verder aan zodat:

1. Bij het laden van de pagina start het spel automatisch => [startSpel\(\)](#)
2. Een **willekeurige** kleurcode (#RRGGBB) wordt gekozen en als de **juiste kleur** en in de span weergegeven.
3. De **drie kleurvakken** krijgen elk een **willekeurige kleur** (kleurA, kleurB of kleurC).
4. Slechts **één** van de drie vakken heeft de juiste kleur => los dit willekeurig (**random**) op en gebruik best een **keuzestructuur**.
5. Maak een functie [controleerKleur\(gekozenKleur, juisteKleur\)](#) met 2 argumenten:
 - o gekozenKleur → de kleur van het vak waarop de gebruiker klikt
 - o juisteKleur → de kleurcode die geraden moet worden
 - o In deze functie controleer je of gekozenKleur === juisteKleur.
 - § Juist → toon “ Correct!”
 - § Fout → toon “ Fout! Probeer opnieuw.”
6. **Koppel elk kleurvak** aan deze **functie** met twee argumenten wanneer erop geklikt wordt.
7. Wanneer op de **knop** “Nieuw spel” wordt geklikt, **start** het spel opnieuw => [startSpel\(\)](#)

Gebruik de gegeven functie [randomMakkelijkeKleur\(\)](#) om een willekeurige kleuren te genereren:

Gebruik [addEventListener](#) om om functies te koppelen aan de drie kleurvakken en de knop.



Bewaar de oplossing in de map **oefeningen** als [oefening_kleurenspel.html](#)