Blok 3 - Web Front End

- HTML - Phoenix
- CSS - Phoenix
- Intro JS
- Portfolio Challenge
- Kennis Check Blok 3

HTML - Phoenix

cursus 28568

1 Introductie Phoenix Code

datasource

In deze les leer je werken met Phoenix Code: een makkelijke code-editor in de browser. Je maakt je eerste HTML-bestand aan en bekijkt het resultaat in je browser.

? Leerdoelen

- Je weet wat Phoenix Code is en waarvoor je het gebruikt
- Je kunt een HTML-bestand maken en bewerken
- Je kunt je werk bekijken in de browser

? Wat is Phoenix Code?

- Phoenix Code is een gratis code-editor die in je browser werkt
- Je kunt er HTML, CSS en JavaScript mee maken
- Het werkt zonder installatie en je kunt alles opslaan als ZIP

? Wat gaan we doen?

- We openen Phoenix Code
- We maken een HTML-bestand aan
- We bekijken het resultaat in de browser

?? Stappenplan

1. Open Phoenix Code

- Ga naar https://phoenixcode.dev
- Klik op**File - StartProject** en kies **HTML5 project**.

Je hebt nu een standaard HTML bestand (index.htm).

Tip: je kunt Phoenix Code ook downloaden en installeren. Dat is makkelijker omdat je dan je bestanden kunt bewaren.

2. Maak een nieuw HTML-bestand

- Klik links op de map
- Klik op `+` en kies **File**
- Geef het bestand de naam `myfirst.html`

3. Zet deze voorbeeldcode erin

```
<!DOCTYPE html>
<html>
<head>
  <title>Hallo wereld</title>
</head>
<body>
  <h1>Hallo allemaal!</h1>
  <p>Dit is mijn eerste HTML-pagina in Phoenix Code.</p>
</body>
</html>
```

4. Bekijk het resultaat

- Klik bovenaan op **Run** of **Live Server**
- De pagina verschijnt rechts of opent in een nieuw tabblad
- Klik aan de rechterkant op de verschillende onderdelen; je ziet nu welke HTML bij dezeonderdelen horen.
- Je kunt ook op het bestand dubbelklikken

Opdracht

Waarvoor dient HTML? Zet deze vraag en het antwoord op jouw HTML pagina.

AI-tip:
Vraag een AI-assistent (zoals ChatGPT) om in 3 zinnen uit te leggen wat HTML doet. Vergelijk dat met jouw eigen uitleg in Phoenix Code — komt het overeen?

? Vragen

- Wat denk je dat <h1> en </h1> doet?
- En .... , wat doen die denk je?

? Inleveren

1. Screenshot van je browser waarin de vraag "Waarom dient HTML?" en jouw antwoord zichtbaar zijn.

Voorbeeld screenshot

image.png

datasource

2 Wat is HTML?

In deze les leer je wat HTML is, waarom het belangrijk is voor websites, en maak je je eerste eenvoudige webpagina met koppen en tekst.

? Leerdoelen

- Je weet waar HTML voor wordt gebruikt
- Je kent de structuur van een HTML-pagina
- Je kunt zelf een eenvoudige HTML-pagina maken

? Wat is HTML?

- HTML betekent **HyperText Markup Language**
- Het is de **basis van elke webpagina**

- HTML vertelt de browser wat er op een pagina moet staan: tekst, afbeeldingen, knoppen, enzovoort

?Video - korte uitleg HTML

https://www.youtube.com/watch?v=it1rTvBcfRg

https://www.youtube.com/embed/it1rTvBcfRg

? Wat gaan we doen?

- We maken een nieuw HTML-bestand in Phoenix Code
- We schrijven een pagina over onszelf met koppen en tekst

?? Stappenplan

1. Maak een nieuw project in Phoenix Code

- Ga naar https://phoenixcode.dev
- Voeg een bestand toe met de naam `opdracht2.html`

2. Typ deze HTML-code

```
<!DOCTYPE html>
<html>
<head>
  <title>Over mij</title>
</head>
<body>

  <h1>Hallo! Ik ben [jouw naam]</h1>
  <p>Ik ben [leeftijd] jaar oud en ik zit op school bij [schoolnaam].</p>
  <p>Mijn favoriete hobby is [hobby].</p>

</body>
```

```
</html>
```

3. Pas de tekst aan

- Vervang `[jouw naam]`, `[leeftijd]`, `[schoolnaam]` en `[hobby]` door je eigen gegevens

4. Bekijk je werk

- Klik op **Run** of **Live Server** om je webpagina te bekijken

Uitleg

Tekst staat tussen tags. Deze tags geeft aan hoe een tekst moet worden weergegeven.

Vrijwel alle HTML tags bestaan uit een open-tag (bijvoorbeeld en een sluit tag bijvoorbeeld

** ... ** betekent dat de tekst een **pragraaf** is.

<h1>...</h1> betekent dat de tekst een **header** is.

Je hebt nog meer headers: <h2>, <h3>, ...

Opdracht

1. Vervang de header die tussen <h1>...</h1> staat door een kleinere header.
2. Zoek op hoe je tekst schuingedrukt (italic) kan afdrukken en laat jouw naam schuin afgedrukt worden.

? Inleveren

1. Maak een screenshot van je webpagina met aangepasgte header en jouw naam ver gedrukt.

3 Tekst en opmaak

In deze les leer je hoe je tekst opmaakt met HTML. Je leert koppen, paragrafen, vetgedrukte tekst, cursieve tekst en regels afbreken.

? Leerdoelen

- Je kunt kopjes en paragrafen maken met HTML
- Je weet hoe je tekst vet of schuin maakt
- Je kunt regels afbreken met een
-tag

? Wat kun je met HTML-opmaak?

- HTML laat je tekst er beter uitzien en duidelijker maken voor de lezer
- Je gebruikt kopjes om structuur aan te brengen, en `<p>` om losse alinea's te maken
- Je kunt woorden **vet** of *schuin* maken om ze op te laten vallen

? Wat gaan we doen?

- We maken een pagina met meerdere kopjes en paragrafen
- We oefenen met `<strong>`, `<em>` en `<br>`

? Eenvoudige HTML-tags voor tekstopmaak

Tag	Betekenis	Voorbeeld	Uitleg
`<h1>` t/m `<h6>`	Koppen	`<h1>Grote titel</h1>`	`<h1>` is de grootste kop, `<h6>` de kleinste
`<p>`	Paragraaf / alinea	`<p>Dit is een stukje tekst.</p>`	Zorgt voor witruimte vóór en na
`<strong>`	Vetgedrukt (belangrijk)	`<strong>Let op!</strong>`	Wordt ook door screenreaders benadrukt
`<em>`	Cursief (nadruk)	`<em>Heel belangrijk</em>`	Geeft nadruk, ook voor screenreaders
` `	Regellijn afbreken	`Eerste regel Tweede regel`	Breekt een regel zonder nieuwe paragraaf (deze tag bestaat alleen uit een openingstab)

?? Stappenplan

1. Maak een nieuw project of nieuw bestand in Phoenix Code

- Noem het bestand `tekst.html`

2. Typ of plak deze HTML-code

```
<!DOCTYPE html>
<html>
<head>
  <title>Mijn interesses</title>
</head>
<body>

  <h1>Over mij</h1>
  <p>Hallo! Ik ben <strong>[jouw naam]</strong>. Ik ben <em>enthousiast</em> over programmeren.</p>

  <h2>Mijn hobby's</h2>
  <p>Ik hou van muziek maken<br>voetballen<br>en gamen.</p>

  <h2>Toekomst</h2>
  <p>Later wil ik werken als programmeur bij een groot bedrijf.</p>

</body>
</html>
```

3. Pas de inhoud aan

- Vervang de tekst door jouw eigen naam en interesses
- Gebruik minstens één `<strong>`, één `<em>` en één `<br>`

4. Bekijk je werk

- Klik op **Run** of **Live Server** om het resultaat te zien

□□ *AI-uitbreiding:* Laat een AI de HTML-code verbeteren door te vragen:
“Hoe kan ik deze HTML beter structureren of leesbaarder maken?”
Bekijk of de suggesties zinvol zijn — pas alleen aan wat je begrijpt.

? Reflectie

- Wat doet `<br>` precies?
- Wanneer zou je `<strong>` gebruiken en wanneer `<em>`?
- Hoe maak je een tekst extra duidelijk voor de lezer?

? Inleveren

1. Maak een screenshot van jouw pagina waarin opmaak en breekregels zichtbaar zijn

4 Afbeeldingen en links

In deze les leer je hoe je een afbeelding op een webpagina toont en hoe je een klikbare link toevoegt naar een andere website.

? Leerdoelen

- Je kunt een **afbeelding** invoegen met HTML
- Je kunt een **link** maken naar een andere website
- Je begrijpt wat attributen zoals `src`, `alt` en `href` doen

? Wat zijn afbeeldingen en links in HTML?

- **Afbeelding:** je gebruikt de tag `<img>` om een foto of plaatje te tonen
- **Link:** je gebruikt de tag `<a>` om een klikbare link te maken
- Je gebruikt attributen zoals:
  - `src` = locatie van de afbeelding
  - `alt` = tekst als de afbeelding niet geladen wordt
  - `href` = doel van de link

?Voorbeeld

Hoe maak je een link?

https://www.youtube.com/watch?v=gOioxltfh48

https://www.youtube.com/embed/gOioxltfh48

Hoe voeg je een plaatje toe op je web pagina?

https://www.youtube.com/watch?v=Hh_se2Zqsdk

https://www.youtube.com/embed/Hh_se2Zqsdk

? Wat gaan we doen?

- We voegen een afbeelding toe aan onze pagina
- We maken een link naar een favoriete website

?? Stappenplan

1. Maak een nieuw HTML-bestand in Phoenix Code

- Noem het bestand `afbeelding.html`

2. Typ of plak deze HTML-code

```
<!DOCTYPE html>
<html>
<head>
  <title>Afbeelding en link</title>
</head>
<body>

  <h1>Mijn favoriete dier</h1>

  <img src="https://upload.wikimedia.org/wikipedia/commons/3/3a/Cat03.jpg" alt="Foto van een kat" width="300">
```

```
  <p>Meer informatie over katten vind je op
    <a href="https://nl.wikipedia.org/wiki/Kat_(dier)" target="_blank">Wikipedia</a>.
  </p>

</body>
</html>
```

3. Pas de inhoud aan

- Vervang de afbeelding met een plaatje dat jij leuk vindt (gebruik Google of Wikipedia en kopieer de afbeeldingslink)
- Verander de link naar jouw favoriete website
- Verander de tekst zodat het bij jouw afbeelding past

4. Bekijk je werk

- Klik op **Run** of **Live Server** om je pagina te testen

? Reflectie

- Wat gebeurt er als je een verkeerde link of afbeelding gebruikt?
- Waarom is het belangrijk om de `alt`-tekst te gebruiken?

? Inleveren

1. Maak een screenshot van je webpagina met een afbeelding én een werkende link

5 Lijsten in HTML

In deze les leer je hoe je lijsten maakt met HTML. Je leert zowel opsommingen als genummerde lijsten maken.

? Leerdoelen

- Je weet wat het verschil is tussen een genummerde en een opsomming

- Je kunt een lijst maken met HTML
- Je kunt zelf kiezen welk type lijst het best past

? Wat zijn lijsten in HTML?

- Je gebruikt een lijst om dingen op een rijtje te zetten
- **Opsomming (bullets):** gebruik je met `<ul>` = **unordered list**
- **Genummerde lijst:** gebruik je met `<ol>` = **ordered list**
- Elk item in een lijst staat in een `<li>` = list item

? Wat gaan we doen?

- We maken een lijst van hobby’s
- We maken een genummerde top 3 van favoriete dingen

?Voorbeeld

https://www.youtube.com/watch?v=2O8pkybH6po

https://www.youtube.com/embed/-kXZvKxs9oA

?? Stappenplan

1. Maak een nieuw HTML-bestand in Phoenix Code

- Noem het bestand `lijsten.html`

2. Typ of plak deze HTML-code

```
<!DOCTYPE html>
<html>
<head>
```

```
  <title>Mijn lijsten</title>
</head>
<body>

  <h1>Mijn hobby's</h1>
  <ul>
    <li>Gamen</li>
    <li>Voetballen</li>
    <li>Tekenen</li>
  </ul>

  <h2>Top 3 favoriete snacks</h2>
  <ol>
    <li>Pizza</li>
    <li>Frikandel</li>
    <li>Chips</li>
  </ol>

</body>
</html>
```

Opdracht

Verander de pagina en maak:

1. een **numbered list** van dingen die jij leuk vind om te doen. Op nummer 1 staat dan wat je het leukst vind, op 2 wat je daarna het leukts vind, etc. etc. Noem tenminst 5 dingen.
2. een **unnumbered lists** van dingen die je tot nu hebt geleerd van software development. Wees specifiek, dus antwoord (Python is onjuist omdat dit niet specifiek genoem is).

Klik op **Run** of **Live Server** om je pagina te testen

? Inleveren

1. Maak een screenshot van jouw webpagina met beide lijsten zichtbaar

6 Tabellen in HTML

In deze les leer je hoe je een eenvoudige tabel maakt met HTML. Je gebruikt dit om informatie netjes in kolommen en rijen te tonen.

? Leerdoelen

- Je weet wat een HTML-tabel is en waarvoor je die gebruikt
- Je kunt een eenvoudige tabel maken met rijen en kolommen
- Je gebruikt de tags `<table>`, `<tr>`, `<td>` en `<th>`

? Wat is een tabel in HTML?

- Een tabel is een manier om informatie te tonen in rijen (horizontaal) en kolommen (verticaal)
- `<table>` = de hele tabel
- `<tr>` = een rij (table row)
- `<td>` = een cel met gegevens (table data)
- `<th>` = een kopcel (table header)

In een HTML tabel moet elke regel evenveel kolommen bevatten. Om dat voor elkaar te krijgen zou je lege kolommen kunne toevoegen als een regel te weinig kolommen heeft.

> In een tabell moet elke regel precies evenveel kolommen bevatten!

?Voorbeeld

https://www.youtube.com/watch?v=2O8pkybH6po

https://www.youtube.com/embed/iDA0kF5lrVk

? Wat gaan we doen?

- We maken een tabel met onze schoolrooster of favoriete eten

- We geven de bovenste rij een titel (kopjes)

?? Stappenplan

1. Maak een nieuw HTML-bestand in Phoenix Code

- Noem het bestand `tabel.html`

2. Typ of plak deze HTML-code

```
<!DOCTYPE html>
<html>
<head>
  <title>Mijn schoolrooster</title>
</head>
<body>

  <h1>Mijn schoolrooster</h1>

  <table border="1">
    <tr>
      <th>Dag</th>
      <th>Vakken</th>
    </tr>
    <tr>
      <td>Maandag</td>
      <td>Nederlands, Wiskunde</td>
    </tr>
    <tr>
      <td>Dinsdag</td>
      <td>Engels, Gym</td>
    </tr>
  </table>

</body>
</html>
```

3. Pas de inhoud aan

- Vervang de dagen en vakken door jouw eigen rooster

  AI-opdracht: Vraag ChatGPT:
  “Hoe kan ik mijn tabel beter leesbaar maken zonder CSS?” Probeer één van de suggesties uit.

4. Bekijk je werk

- Klik op **Run** of **Live Server** om je tabel te bekijken

? Inleveren

1. Maak een screenshot van jouw tabel met minstens 6 rijen (kopregel plus maamndag t/m vrijdag)

7 Formulieren in HTML

In deze les leer je hoe je een formulier maakt in HTML. Je leert invoervelden gebruiken en een knop toevoegen om informatie te versturen.

? Leerdoelen

- Je weet wat een formulier is en waarvoor je het gebruikt
- Je kunt invoervelden maken voor tekst, getallen en knoppen
- Je kunt een formulier laten verzenden naar een ander HTML- of PHP-bestand

? Wat is een formulier?

- Een formulier gebruik je om gegevens in te voeren op een website
- Denk aan: contactformulieren, inlogvelden, zoekvelden, bestellingen
- Je gebruikt tags zoals:
  - `<form>` – de container
  - `<input>` – invoerveld
  - `<label>` – beschrijving van een veld
  - `<button>` of `<input type="submit">` – knop

Wil je een voorbeeld zien, dan kan je hier kijken.

? Wat gaan we doen?

- We maken een formulier met naam, leeftijd en hobby
- We laten het formulier "doen alsof" het iets verstuurt

?Voorbeeld

https://www.youtube.com/watch?v=2O8pkybH6po

https://www.youtube.com/embed/2O8pkybH6po

?? Stappenplan

1. Maak een nieuw HTML-bestand in Phoenix Code

```
<!DOCTYPE html>
<html>
<head>
  <title>Contactformulier</title>
</head>
<body>

  <h1>Vertel iets over jezelf</h1>

  <form action="#" method="post">
    <label for="naam">Naam:</label><br>
    <input type="text" id="naam" name="naam"><br><br>

    <label for="leeftijd">Leeftijd:</label><br>
    <input type="number" id="leeftijd" name="leeftijd"><br><br>

    <label for="hobby">Hobby:</label><br>
    <input type="text" id="hobby" name="hobby"><br><br>
```

```
    <input type="submit" value="Verstuur">
  </form>

</body>
</html>
```

- Noem het bestand `formulier.html`

2. Typ of plak deze HTML-code

Opdracht

Gebruik het formulier als voorbeeld en maak zelf een formulier.

Vraag de gebruiker om de volgende gegvens:

- voornaam
- achternaam
- adres
- postcode
- woonplaats
- teelfoonnummer

Bekijk je werk

- Klik op **Run** of **Live Server** en vul het formulier in
- Klik op "Verstuur" – er gebeurt nog niets, dat komt later met PHP!

? Reflectie

- Wat zou je met dit formulier willen doen als het "echt" was?
- □□ *AI-verkenning:* Vraag een AI:
  “Welke gegevens mag ik volgens de AVG (privacywet) niet zomaar vragen in een formulier?”

Noteer in je verslag 2 regels die je daaruit leert.

? Inleveren

1. Maak een screenshot van jouw formulier met ingevulde gegevens

8 Mini-project: Jouw persoonlijke homepage

In deze les ga je zelf een echte webpagina bouwen over jezelf. Je gebruikt alles wat je hebt geleerd over HTML: tekst, afbeeldingen, lijsten, tabellen en een formulier.

? Leerdoelen

- Je kunt zelfstandig een complete HTML-pagina maken
- Je gebruikt verschillende HTML-elementen op de juiste manier
- Je denkt na over opbouw, inhoud en structuur van een webpagina

? Wat gaan we doen?

- Je maakt een persoonlijke homepage met meerdere onderdelen
- Je laat zien wat je geleerd hebt in lessen 1 t/m 6

?? Opdracht

Maak een bestand `mijnhomepage.html` aan en zorg dat het deze onderdelen bevat:

Verplichte onderdelen:

- Een **titel** en `<h1>`-kop met je naam
- Minstens twee `<p>` paragrafen over jezelf
- Een `<img>` met een afbeelding die bij jou past (bijv. hobby, dier of sport)
- Een `<ul>` of `<ol>` met je favoriete dingen (bijv. films, games, hobby's)
- Een `<table>` met 2 of meer rijen, bijv. je rooster of een eetplanner
- Een `<a>`-link naar een website die jij vaak bezoekt

- Een klein `<form>` met 2-3 invoervelden (bijv. naam + boodschap + knop)

Basis HTML code

```
<!DOCTYPE html>
<html>
<head>
  <title>Contactformulier</title>
</head>
<body>



</body>
</html>
```

□□ Extra uitdaging (optioneel):

- Gebruik `<strong>`, `<em>` en `<br>` om je pagina mooier te maken
- Geef je pagina een leuke kleur of lettertype met een inline CSS-regel

? Inleveren

1. Maak een screenshot van jouw volledige homepage

CSS - Phoenix

1 Wat is CSS?

datasource

In deze les leer je wat CSS is, waarom het belangrijk is, en hoe je het gebruikt om de opmaak van een webpagina aan te passen. Je gaat zelf aan de slag met eenvoudige kleuren en lettertypes.

? Leerdoelen

- Je weet wat CSS doet en waarom we het gebruiken
- Je kunt een eenvoudige opmaakregel toevoegen via de HTML-tag (inline)
- Je kunt kleuren en lettertypes toepassen met CSS

? Wat is CSS?

- **CSS** staat voor **Cascading Style Sheets**
- Met CSS bepaal je hoe HTML eruitziet: kleuren, lettertypes, ruimte, achtergrond, enzovoort
- Zonder CSS is een website saai: zwart-wit, standaardletters, geen stijl

? Hoe voeg je CSS toe?

- Je kunt CSS op drie manieren gebruiken:
  1. **Inline** – direct in het HTML-element (in deze les)
  2. **In een <style>-blok** – in de `<head>` (volgende les)
  3. **In een extern .css-bestand** – voor grote projecten (komt later)

Voorbeeld: CSS inline gebruiken

```
<p style="color: red;">Deze tekst is rood</p>
<h1 style="font-family: Arial;">Welkom</h1>
```

De `style="..."` zit in het HTML-element en verandert de stijl van dat element.

?CSS Introduction

In deze korte video wordt uitgelegd wat CSS is. Kijk de video en als je nog niet alles begrijpt dan is dan prima, want we gaan alles ook nog een keer in deze lessen uitleggen.

https://www.youtube.com/watch?v=AGDDdsiZ0Ko&list=PLP9IO4UYNF0UCaUSF3XNZ1U9f01E5h5PM

https://www.youtube.com/embed/AGDDdsiZ0Ko?list=PLP9IO4UYNF0UCaUSF3XNZ1U9f01E5h5PM

?? Opdracht – Pas kleur en stijl toe

Let op: gebruik Phoenix Code om deze opdracht te maken.

1. Maak een nieuw HTML-bestand aan, noem het `css-les1.html`
2. Plak onderstaande code in het bestand:

```
<!DOCTYPE html>
<html>
<head>
  <title>Mijn eerste CSS</title>
</head>
<body>

  <h1>Welkom op mijn pagina</h1>
  <p>Ik ben een beginnende webdeveloper.</p>
  <p>Ik leer nu hoe ik CSS kan gebruiken.</p>

</body>
</html>
```

3. Pas de HTML aan met **inline CSS**, zodat:
   - De `<h1>` een blauwe kleur krijgt en een ander lettertype (bijv. Arial)
   - De eerste `<p>` rood wordt en gecentreerd staat
   - De tweede `<p>` een andere kleur krijgt die jij mooi vindt

4. Bekijk je werk in Phoenix Code via **Run**

? Inleveren

1. Maak een screenshot van jouw pagina met de aangepaste kleuren en lettertypes en zorg dat de code zichtbaar is.

2 CSS in het style-blok

In deze les leer je hoe je CSS los van HTML schrijft in het <style>-blok boven in het document. Je gebruikt selectoren zoals `body`, `h1` en `p` om elementen op te maken.

? Leerdoelen

- Je begrijpt waarom het slim is om CSS apart te schrijven
- Je kunt stijlen toevoegen in het <style>-blok
- Je kunt opmaak toepassen op meerdere elementen tegelijk met selectors

? Wat is het <style>-blok?

- Het `<style>`-blok zet je bovenin de pagina, tussen de `<head>`-tags
- Je schrijft daar CSS die geldt voor de hele pagina
- Je gebruikt een **selector** om aan te geven welk element je wilt opmaken. Hieronder vind je de selectors **body h1** en **p**

Voorbeeld:

```
<style>
  body {
    background-color: lightyellow;
  }

  h1 {
    color: navy;
    font-family: Verdana;
  }
```

```
  p {
    color: darkgreen;
    text-align: center;
  }
</style>
```

> In dit voorbeeld zijn body, h1 en p de **element**-selectors en die verwijzen naar de HTML-**elementen**.

?? Opdracht – Stijl je pagina met het <style>-blok

1. Maak een nieuw HTML-bestand in Phoenix Code, noem het `css-les2.html`
2. Plak deze basiscode in het bestand:

```
<!DOCTYPE html>
<html>
<head>
  <title>Stijl oefenen</title>
  <style>

  /* Voeg hier je CSS toe */

  </style>
</head>
<body>

  <h1>Over mij</h1>
  <p>Ik ben een student die leert over HTML en CSS.</p>
  <p>Ik vind het leuk om websites te maken.</p>

</body>
</html>
```

3. Voeg in het `<style>`-blok CSS toe zodat:
   - De `body` een lichtgrijze achtergrondkleur krijgt

- Alle `<h1>`-koppen blauw worden en een ander lettertype krijgen (bijv. Georgia)
- De `<p>`-tekst donkergroen wordt en gecentreerd staat

? Reflectie

- Wat is het voordeel van een `<style>`-blok boven inline stijl?
- Welke stijlen gelden voor de hele pagina, ook als je later extra tekst toevoegt?
- Wat zou je veranderen om je pagina nog mooier te maken?

Opdracht

1. Voer alle stappen uit zoals hierboven beschreven.

? Inleveren

1. Maak een screenshot van jouw pagina waarin je aangepaste stijlen duidelijk zichtbaar zijn

3 Tekstopmaak en achtergronden

In deze les leer je hoe je tekst kunt opmaken met kleur, lettertype, uitlijning en hoe je een achtergrondkleur toevoegt aan je pagina.

? Leerdoelen

- Je kunt tekst centreren of links/rechts uitlijnen
- Je kunt de kleur en het lettertype van tekst aanpassen
- Je kunt de achtergrondkleur van de hele pagina of een specifiek element aanpassen
- Je weet wat een selector is en wat het verschil is tussen een **class-selector** en een **element-selector**.

Selectors; element-selector en class selector

Zoals we eersder zagen beginnen we een **CSS blok** met een **selector**.

Tot nu toe hebben we **element-selectors** gebruikt. Deze verwezen naar de HTML-**elementen**.

We kennen ook **class-selectors** deze verwijzen naar **classes**. Classes zien er in HTML als volgt uit:

```
<div class="blok"> ... </div>
```

In dit geval is blok de naam van de selector.

In CSS verwijzen we naar een class op de volgende manier.

```
.blok {
  /* stijlregels */
}
```

> Een class selector in CSS begint met een . (punt)

? Belangrijke CSS-eigenschappen in deze les

- `color` – de kleur van de tekst
- `font-family` – het lettertype
- `text-align` – de uitlijning van tekst (bijv. `center`)
- `background-color` – de achtergrondkleur van een element

Voorbeeld:

```
.my-header {
  color: darkred;
  font-family: 'Comic Sans MS';
  text-align: center;
  background-color: lightyellow;
}
```

?? Opdracht – Maak een kleurrijke poster

1. Maak een nieuw HTML-bestand in Phoenix Code, noem het `poster.html`
2. Plak deze HTML-code in het bestand:

```
<!DOCTYPE html>
<html>
<head>
  <title>Mijn poster</title>
  <style>

  /* Voeg hier jouw stijlen toe */

  </style>
</head>
<body>

  <h1>Mijn favoriete onderwerp</h1>
  <p class="eerste">Dit is een stukje tekst over iets wat ik leuk vind.</p>
  <p>Bijvoorbeeld een sport, hobby, film of spel.</p>
  <p>Of een vak op College Amstelland dat je heel leuk vind :)</p>

</body>
</html>
```

3. Geef je `body` een leuke `background-color`
4. Geef je `h1` een opvallende kleur, een ander lettertype en `text-align: center`
   Gebruik hiervoor de **class** met **class-selector** `.header`
5. Geef de `p`-teksten elk een andere kleur en uitlijning en gebruik hiervoor **classes** met **class-selectors**. Geef elke <p> hiervoor een aparte class.
6. Experimenteer met verschillende kleuren en lettertypes zoals: `Arial`, `Georgia`, `Courier New`, `Comic Sans MS`

? Inleveren

1. Maak een screenshot van jouw kleurrijke poster waarin de CSS code ook duidelijk te lezen is.

4 Box-model: randen, padding en margins

In deze les leer je hoe je ruimte kunt maken rondom en binnenin een element met CSS. Je leert ook hoe je randen toevoegt en de afstand tussen blokken regelt.

? Leerdoelen

- Je begrijpt het verschil tussen **margin**, **padding** en **border**
- Je kunt ruimte tussen tekstblokken instellen met CSS
- Je kunt randen toevoegen aan blokken om ze beter zichtbaar te maken

? Wat is het box-model?

Elk HTML-element is als een ‘doos’ opgebouwd uit vier lagen:

- **Content:** de tekst of afbeelding binnenin
- **Padding:** ruimte tussen de content en de rand
- **Border:** de rand om het element heen
- **Margin:** de ruimte tussen dit element en andere elementen

image.png

Voorbeeld:

```
.blok {
  margin: 15px;
  border: 2px solid black;
  padding: 20px;
  background-color: lightblue;
}
```

?? Opdracht – Drie gekleurde tekstblokken

1. Maak een nieuw HTML-bestand in Phoenix Code, noem het `boxmodel.html`
2. Plak onderstaande code in je bestand:

```
<!DOCTYPE html>
<html>
<head>
```

```
  <title>Box model test</title>
  <style>

  .blok {
    border: 2px solid black;
    padding: 20px;
    margin: 20px;
    background-color: lightyellow;
  }

  </style>
</head>
<body>

  <div class="blok">
    <h2>Blok 1</h2>
    <p>Dit is het eerste tekstblok met padding en marge.</p>
  </div>

  <div class="blok">
    <h2>Blok 2</h2>
    <p>Ook dit blok heeft een rand en achtergrondkleur.</p>
  </div>

  <div class="blok">
    <h2>Blok 3</h2>
    <p>Pas de kleuren en afstanden aan zoals jij het mooi vindt.</p>
  </div>

</body>
</html>
```

3. Pas de CSS aan zodat:
   - Ieder blok een andere `background-color` heeft
   - De `margin` tussen de blokken groter of kleiner is (proberen met `10px`, `30px`, enz.)
   - De `padding` meer of minder ruimte geeft binnen het blok
4. Test verschillende `border`-instellingen, zoals `dashed`, `double` of `4px solid red`

? Reflectie

- Wat is het verschil tussen `margin` en `padding`?
- Waarom is het handig om borders tijdelijk toe te voegen tijdens het bouwen?
- Wat gebeurt er als je `padding` op 0 zet?

? Inleveren

1. Geef antwoord op de drie reflectievragen in eigen woorden en lever dat in, in een txt- of PDF bestand.
2. Voeg een schermafdruk van jouw boxen toe aan de PDF.

5 Classes gebruiken in CSS

In deze les leer je hoe je een `class` gebruikt om meerdere elementen dezelfde opmaak te geven. Zo kun je stijlen hergebruiken zonder telkens dezelfde regels te schrijven.

? Leerdoelen

- Je weet wat een `class` is en hoe je die toepast in HTML
- Je kunt CSS-stijlen koppelen aan een class
- Je kunt verschillende klassen maken en gebruiken op één pagina

? Wat is een class?

- Een `class` is een naam die je aan een element geeft zodat je het in CSS kunt opmaken
- Je zet in HTML: `<div class="menu">`
- In CSS gebruik je een punt (.) om een class aan te roepen: `.menu { ... }`
- Je kunt dezelfde class meerdere keren gebruiken.
- Een HTML element kan ook meerdere classes bevatten: `<div class="menu speciaal">`
  De stylen met de class-selector menu én speciaal zijn beiden van teopassing op dit element.

Voorbeeld:

```
<div class="menu">Pizza Margherita</div>
<div class="menu speciaal">Spaghetti Bolognese</div>
```

```
.menu {
  background-color: lightyellow;
  padding: 10px;
  border: 1px solid gray;
  margin-bottom: 10px;
}

.speciaal {
  color: red;
}
```

?? Opdracht – Bouw een gestyleerde menukaart

1. Maak een nieuw HTML-bestand in Phoenix Code, noem het `menukaart.html`
2. Plak onderstaande code als startpunt:

```
<!DOCTYPE html>
<html>
<head>
  <title>Menukaart</title>
  <style>

  .menu {
    background-color: lightyellow;
    border: 2px solid #999;
    padding: 15px;
    margin-bottom: 15px;
    font-family: Georgia;
  }

  </style>
</head>
<body>

  <h1>Restaurant De Codepan</h1>
```

```
  <div class="menu">Pizza Margherita</div>
  <div class="menu">Truffelpasta met parmezaan</div>
  <div class="menu">Ravioli met spinazie en ricotta</div>
  <div class="menu">Chef’s surprise</div>

</body>
</html>
```

3. Pas zelf de kleuren, marges of lettertypes aan zoals jij het mooier vindt
4. Voeg minstens **twee** eigen gerechten toe aan de lijst
5. Geef **twee** menu-items de **class** `speciaal` en zorg ervoor dat deze speciale gerechten duidelijk opvallen door hun stijl aan te passen.

? Reflectie

- Waarom is het handig om stijlen te herhalen met een class?
- Wat gebeurt er als je meerdere klassen aan één element geeft?
- Kun je bedenken wanneer je liever een class gebruikt dan een tag-selector (zoals `p`)?

? Inleveren

1. Maak een screenshot van jouw gestyleerde menukaart in de browser.

6 Layout en positionering

In deze les leer je hoe je met behulp van blokken en eenvoudige CSS een duidelijke indeling (layout) maakt. Denk aan een header bovenaan, een middenstuk (main) en een footer onderaan.

? Leerdoelen

- Je weet hoe je de layout van een pagina opbouwt met blokken (divs)
- Je kunt tekst en blokken centreren met `text-align` en `margin`
- Je kunt blokken op een vaste breedte zetten

? Wat is een layout?

- Een layout is de structuur van je pagina – hoe dingen verdeeld zijn over het scherm
- Voor layout gebruiken we vaak het HTML-element `<div>`
- Een `div` is een onopvallend blok dat je zelf kunt opmaken met CSS

Belangrijke CSS-eigenschappen:

- `text-align: center;` – centreren van tekst
- `margin: auto;` – centreren van blokken
- `width:` – bepaalt hoe breed een blok is (bijv. `80%` of `600px`)

Voorbeeld:

```
.blok {
  width: 80%;
  margin: 20px auto;
  padding: 20px;
  background-color: #f0f0f0;
  text-align: center;
}
```

?? Opdracht – Bouw een eenvoudige homepage-layout

1. Maak een nieuw HTML-bestand in Phoenix Code, noem het `layout.html`
2. Plak onderstaande basiscode in het bestand:

```
<!DOCTYPE html>
<html>
<head>
  <title>Mijn layout</title>
  <style>

    .blok {
      width: 80%;
```

```
      margin: 20px auto;
      padding: 20px;
      background-color: #eeeeee;
      text-align: center;
    }

    .header {
      background-color: lightblue;
    }

    .main {
      background-color: white;
    }

    .footer {
      background-color: lightgray;
    }

  </style>
</head>
<body>

  <div class="blok header">
    <h1>Welkom op mijn site</h1>
  </div>

  <div class="blok main">
    <p>Hier komt de inhoud van je website.</p>
  </div>

  <div class="blok footer">
    <p>Gemaakt door [jouw naam] - 2025</p>
  </div>

</body>
</html>
```

3. Pas de kleuren aan zodat het bij jouw stijl past
4. Voeg een afbeelding toe in de main-blok en plaats jouw naam onderaan in de footer.

5. Geef de blokken afgeronde hoeken (`border-radius`) van bijvoorbeeld `15px`

? Reflectie

- Wat gebeurt er als je de `width` verandert naar `60%` of `400px`?
- Hoe werkt `margin: auto` precies?
- Wat zou je toevoegen als je een echte website maakt?

? Inleveren

1. Maak een screenshot van jouw homepage met drie blokken die door jouw zijn vormgegeven.

7 CSS-selectors: element, class en id

In deze les leer je wat een `id`-selector is en wanneer je die gebruikt. Je herhaalt ook wat een `element`-selector en een `class`-selector doet. Daarna pas je deze drie selectors toe in één pagina.

? Leerdoelen

- Je herkent het verschil tussen element-, class- en id-selectors
- Je weet wanneer je een id gebruikt in plaats van een class
- Je kunt deze drie selectors toepassen in één pagina

? Wat zijn selectors in CSS?

Een **selector** in CSS bepaalt op welk HTML-element de stijlregel toegepast wordt.

Selector	Voorbeeld	Toepassing
Element	`p { color: blue; }`	Geldt voor **alle** paragrafen
Class	`.belangrijk { color: red; }`	Geldt voor **meerdere** elementen met class="belangrijk"
ID	`#titel { font-size: 30px; }`	Geldt voor **één uniek element** met id="titel"

Belangrijk verschil:

- `class` gebruik je voor meerdere elementen
- `id` gebruik je maar één keer op een pagina – het is uniek

?? Opdracht – Gebruik drie soorten selectors

1. Maak een nieuw HTML-bestand in Phoenix Code, noem het `selectors.html`
2. Plak deze code als basis:

```
<!DOCTYPE html>
<html>
<head>
  <title>CSS Selectors</title>
  <style>

    /* Element-selector */
    p {
      color: green;
    }

    /* Class-selector */
    .special {
      font-weight: bold;
      background-color: lightyellow;
    }

    /* ID-selector */
    #titel {
      font-size: 30px;
      color: navy;
    }

  </style>
</head>
<body>

  <h1 id="titel">Welkom op mijn pagina</h1>

  <p>Dit is een normale paragraaf.</p>
```

```
  <p class="special">Dit is een speciale paragraaf.</p>
  <p>Nog een gewone paragraaf.</p>

</body>
</html>
```

3. Pas de stijlen aan zoals jij het mooi vindt:
   - Geef de `#titel` een andere kleur en een ander lettertype
   - Geef de class `.special` een rand of andere achtergrond
4. Voeg zelf nog een extra class toe voor een andere stijl (bijv. `.opvallend`)

? Reflectie

- Wanneer gebruik je een `class` en wanneer een `id`?
- Wanneer gebuik je een element-selector, bijvoorbeeld `body` of `h1`?
- Wat gebeurt er als je per ongeluk twee keer hetzelfde `id` gebruikt?

? Inleveren

1. De refectievragen en jouw eigen antwoorden (eigen woorden) in een txt- of PDF bestand.

8 Flexbox: moderne layouttechniek

In deze les leer je werken met `flexbox`. Dit is een moderne techniek waarmee je blokken makkelijk naast of onder elkaar zet, zonder ingewikkelde marges of floats.

? Leerdoelen

- Je weet wat `display: flex` doet
- Je kunt blokken naast elkaar zetten
- Je weet wat resonsive design is.
- Je kunt blokken centreren en de ruimte verdelen met `justify-content` en `align-items`

? Wat is flexbox?

Met flexbox kun je elementen binnen een container netjes ordenen, zowel horizontaal als verticaal.

- `display: flex` zet de container om in een flex-container
- `justify-content` bepaalt de horizontale uitlijning (zoals `center`, `space-between`)
- `align-items` bepaalt de verticale uitlijning (zoals `center`, `flex-start`)

Met felxboxes maak je een **responsive design**: dit betekent dat je web pagina op verschillende aparaten goed wordt getoond en dat de boxes worden aangepast als de breedte van het scherm wordt aangepast.

Voorbeeld CSS:

```
.container {
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.blok {
  width: 150px;
  height: 150px;
  background-color: lightblue;
  text-align: center;
  padding: 20px;
  border-radius: 10px;
}
```

> Een complete uitleg over alle mogelijkheden vind je hier: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

?? Opdracht – Maak een blokkenrij met Flexbox

1. Maak een nieuw bestand in Phoenix Code: `flex.html`
2. Gebruik deze HTML als basis:

```
<!DOCTYPE html>
<html>
<head>
  <title>Flexbox oefening</title>
  <style>

    .container {
      display: flex;
      justify-content: space-between;
      align-items: center;
      background-color: #f0f0f0;
      padding: 20px;
    }

    .blok {
      width: 150px;
      height: 150px;
      background-color: lightcoral;
      color: white;
      font-weight: bold;
      border-radius: 10px;
      display: flex;
      justify-content: center;
      align-items: center;
    }

  </style>
</head>
<body>

  <h1>Mijn Flexbox layout</h1>

  <div class="container">
    <div class="blok">Blok 1</div>
    <div class="blok">Blok 2</div>
    <div class="blok">Blok 3</div>
  </div>

</body>
</html>
```

3. Experimenteer met andere waarden voor `justify-content` zoals `center`, `space-around` of `flex-end`
4. Voeg eventueel nog een vierde blok toe
5. Pas de kleuren en tekst aan zodat het jouw stijl heeft

??Opdracht

Maak het volgende ontwerp zo goed mogelijk na en gebruik daarbij Flexboxen zoals je dat net hebt geleerd.:

image.png

De drie blokken staan op het midden van de webpagina en je design is **responisve**.

? Inleveren

1. Maak een screenshot van je resultaat (van de *laatste* opdracht).
2. Indien je AI hebt gebruikt, stuur dan de AI-Chat log mee en zorg ervoor dat je alleen zaken gebruikt die we hebben behandeld en die je kan uitleggen.

> Je kan worden gevraagd om jouw ontwerp te komen toe lichten / uitleggen.

--

Intro JS

1 Wat is JavaScript?

datasource

In deze les leer je wat JavaScript is, waarom we het gebruiken en hoe je jouw eerste stukje code uitvoert in de browser.

? Leerdoelen

- Je weet wat JavaScript is en waarvoor het wordt gebruikt
- Je kunt een script uitvoeren in de browserconsole
- Je kunt een script toevoegen aan een HTML-pagina met <script>

? Wat is JavaScript?

- **HTML** zorgt voor de inhoud (tekst, koppen, knoppen...)
- **CSS** zorgt voor de opmaak (kleuren, marges, lettertypes...)
- **JavaScript** zorgt voor de interactie (reageren op klikken, invullen, bewegen...)

  Bijvoorbeeld, een **knop**

- **HTML**: een **kale knop** zonder opmaak
- **CSS**: dezelfde knop, maar nu **blauw** met **afgeronde hoeken**
- **JavaScript**: als je op de knop klikt, **verschijnt er een melding** ("popup")

Wat kun je met JavaScript doen?

- Een knop laten reageren als je erop klikt
- Een rekenmachine bouwen
- Invoer van een formulier controleren

- Spelletjes maken in de browser

?Video

In deze video wordt goed uitgelegd waarom je asl Web Developer, JavaScript moet leren.

Niet alles wat hier in de voorbeelden wordt getoond wordt niet allemaal in deze lessen behandeld. In blok 5 waarneer we het over DOM gaan hebben zal dit allemaal uitvoerig worden behandeld. We houden het voor nu even bij de basics.

https://www.youtube.com/watch?v=zofMnllkVfI

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

? Oefenen in de browserconsole

1. Open je browser (bijv. Chrome of Firefox)
2. Rechtermuisklik op een lege plek op een website > Kies **Inspecteren** > Ga naar het tabblad **Console**
3. Typ daar:

   ```
   console.log("Hallo wereld!");
   ```

   en druk op Enter
4. Je ziet in de console de tekst `Hallo wereld!`

? JavaScript in een HTML-bestand gebruiken

1. Open Phoenix Code
2. Maak een nieuw HTML-bestand aan, noem het `script.html`
3. Typ of plak deze code in het bestand:

```
<!DOCTYPE html>
<html>
<head>
  <title>Eerste script</title>
```

```
</head>
<body>

  <h1>Welkom op mijn site</h1>
  <p>Open de console om het bericht te zien.</p>

  <script>
    console.log("Dit bericht komt uit de HTML!");
  </script>

</body>
</html>
```

4. Sla op en open het bestand in de browser
5. Open opnieuw de **console** en zie het bericht verschijnen

> Zie je meer berichten of fouten?
> Verwijder alle bestanden uit je project zodat je alleen script.html overhoud.

?? Opdracht – Jouw eerste script

1. Maak een HTML-bestand in Phoenix Code met een eigen titel en één paragraaf
2. Voeg onderaan het bestand een script toe met console.log()-regels
3. Laat in de console bijvoorbeeld jouw naam en je favoriete dier zien

Voorbeeld:

```
console.log("Ik heet Yasmin");
console.log("Mijn lievelingsdier is een rode panda");
```

? Reflectie

- Wat is het verschil tussen wat je in de browser ziet en wat je in de console ziet?
- Waarom gebruiken programmeurs de console?

? Inleveren

1. Maak een screenshot van jouw HTML-bestand én de console-output.

2 Variabelen en de console

In deze les leer je hoe je informatie kunt opslaan in een variabele, hoe je dat zichtbaar maakt in de console, en waarom de console zo belangrijk is bij het programmeren.

? Leerdoelen

- Je begrijpt wat een variabele is
- Je kunt een variabele maken met `let` of `const`
- Je gebruikt `console.log()` om informatie weer te geven
- Je weet wat debuggen is en waarom de console daarbij helpt

? Wat is een variabele?

Een variabele is een soort doosje waarin je iets bewaart, zoals tekst of een getal.

```
let naam = "Jasper";
const leeftijd = 17;
```

- `let` = variabele die later nog mag veranderen
- `const` = variabele die niet meer mag veranderen

Je gebruikt `console.log()` om de inhoud van een variabele te bekijken:

```
console.log(naam);
```

? Wat is debuggen?

Debuggen betekent: je code controleren op fouten (bugs) en begrijpen wat er gebeurt.

- De **console** is een soort gereedschapskist voor programmeurs.
- Je kunt er controleren of variabelen kloppen, of een functie wordt uitgevoerd, of ergens een fout zit.
- Een programmeur kijkt vaak in de console tijdens het schrijven van code.

Als je in de console iets ziet wat je niet verwacht – dan kun je makkelijker de fout vinden. Daarom gebruiken we `console.log()` vaak tijdens het debuggen.

?? Opdracht – Variabelen gebruiken en debuggen

1. Maak een nieuw HTML-bestand in Phoenix Code: `variabelen.html`
2. Typ de volgende basiscode in je bestand:

```
<!DOCTYPE html>
<html>
<head>
  <title>Variabelen oefenen</title>
</head>
<body>

  <h1>Bekijk de console!</h1>

  <script>
    let voornaam = "Ali";
    const leeftijd = 18;

    console.log("Naam: " + voornaam);
    console.log("Leeftijd: " + leeftijd);
  </script>

</body>
</html>
```

3. Open je bestand in de browser en bekijk de console (rechtermuisklik → Inspecteren → Console)
4. Verander de naam en leeftijd naar jouw eigen gegevens
5. Voeg een derde variabele toe: `favorietDier` en toon die ook met `console.log()`

Voorbeeld:

```
let favorietDier = "koala";
console.log("Mijn favoriete dier is: " + favorietDier);
```

? Reflectie

- Wat gebeurt er als je de waarde van `const` probeert te veranderen nadat je deze eerst een waarde hebt gegeven?
- Waarom is het handig om even iets te loggen in de console?
- Heb je fouten gezien? Wat hielp je om ze op te lossen?

? Inleveren

1. Beantwoord de vragen uit de reflectie en lever die in (.txt. of .pdf bestand).

3 Functies

In deze les leer je wat een functie is, waarom functies handig zijn en hoe je zelf een functie maakt in JavaScript. Je oefent met functies die tekst tonen of berekeningen uitvoeren.

? Leerdoelen

- Je weet wat een functie is en wat het doel ervan is
- Je kunt een functie schrijven met `function`
- Je kunt een functie aanroepen (laten uitvoeren)
- Je begrijpt wat een parameter is en hoe je die meegeeft

? Wat is een functie?

Een **functie** is een blokje code dat je een naam geeft, zodat je het later makkelijk opnieuw kunt gebruiken.

Stel, je wilt iets drie keer doen, zoals een bericht tonen. Dan is het dus handig om dat in een functie te stoppen.

```
function zegHallo() {
  console.log("Hallo!");
```

```
}
```

Om de functie uit te voeren (aan te roepen), typ je:

```
zegHallo();
```

? Functies met parameters

Je kunt een functie ook informatie meegeven. Bijvoorbeeld een naam:

```
function begroet(naam) {
  console.log("Hoi " + naam + "!");
}

begroet("Fatima"); // toont: Hoi Fatima!
begroet("Jesse");  // toont: Hoi Jesse!
```

?? Opdracht – Bouw je eigen functies

1. Maak een nieuw HTML-bestand in Phoenix Code: `functies.html`
2. Typ deze code als basis:

```
<!DOCTYPE html>
<html>
<head>
  <title>Functies oefenen</title>
</head>
<body>

  <h1>Bekijk de console</h1>

  <script>

    function begroet(naam) {
      console.log("Hallo " + naam + "!");
    }

    begroet("Maya");
    begroet("Koen");
```

```
  </script>

</body>
</html>
```

3. Maak nog een functie die een getal verdubbelt:

```
function verdubbel(getal) {
  console.log(getal * 2);
}
```

4. Roep die functie minstens twee keer aan met verschillende getallen
5. Voeg daarna zelf een nieuwe functie toe met een eigen idee, bijvoorbeeld:
   - Een functie die een leeftijd in hondenjaren berekent (× 7)
   - Een functie die een bericht toont in jouw stijl.

? Reflectie

- Waarom is het handig om herhaalbare code in een functie te zetten?
- Wat gebeurt er als je een functie oproept zonder de juiste parameter?

? Inleveren

1. Maak een screenshot van je code én de console-output

4 Voorwaardes met `if` en `else`

In deze les leer je hoe je met JavaScript keuzes kunt maken. Je gebruikt `if` en `else` om bepaalde code alleen uit te voeren als aan een voorwaarde is voldaan.

? Leerdoelen

- Je begrijpt wat een voorwaarde is
- Je kunt werken met `if`, `else if` en `else`

- Je kunt eenvoudige beslissingen laten uitvoeren op basis van een getal

? Wat is een voorwaarde?

Met een voorwaarde bepaal je of een stukje code wél of niet uitgevoerd moet worden. Je gebruikt een `if`-statement:

```
let leeftijd = 17;

if (leeftijd >= 18) {
  console.log("Je bent volwassen");
} else {
  console.log("Je bent nog geen 18");
}
```

Je kunt meerdere keuzes maken met `else if`:

```
let punt = 6;

if (punt >= 8) {
  console.log("Super goed!");
} else if (punt >= 5.5) {
  console.log("Voldoende");
} else {
  console.log("Onvoldoende");
}
```

?? Opdracht – Leeftijdscontrole met voorwaarden

1. Maak een nieuw HTML-bestand in Phoenix Code: `ifelse.html`
2. Gebruik deze code als basis:

```
<!DOCTYPE html>
<html>
<head>
  <title>If-statement oefenen</title>
</head>
```

```
<body>

  <h1>Bekijk de console</h1>

  <script>

    let leeftijd = 15;

    if (leeftijd >= 18) {
      console.log("Je mag stemmen.");
    } else {
      console.log("Je bent nog te jong om te stemmen.");
    }

  </script>

</body>
</html>
```

3. Pas de variabele `leeftijd` aan naar jouw eigen leeftijd en test het resultaat
4. Voeg een `else if`-blok toe dat controleert of iemand precies 17 is
5. Maak daarna zelf een voorbeeld met een `punt` (bijvoorbeeld een toetscijfer) en geef een boodschap: onvoldoende / voldoende / goed

? Inleveren

1. Maak een screenshot van je code én de console-output bij jouw leeftijd en punt
2. Lever dit screenshot in via Teams of Canvas

? Reflectie

- Wat gebeurt er als je geen `else` gebruikt?
- Wanneer gebruik je `else if` in plaats van meerdere `if`-regels?

5 Gebeurtenissen (events)

*In deze les leer je hoe je iets kunt laten gebeuren als een gebruiker op een knop klikt. Je leert werken met `onclick` waarmee je een **functie** kan aanroepen bij een gebeurtenis.*

? Leerdoelen

- Je weet wat een event is in JavaScript
- Je kunt een knop laten reageren op een klik met `onclick`
- Je kunt een functie aanroepen wanneer een event plaatsvindt

? Wat is een event?

Een **event** is een gebeurtenis, zoals:

- een klik op een knop (`click`)
- iets typen in een tekstvak (`input`)
- de muis bewegen over een element (`mouseover`)

Met JavaScript kun je ervoor zorgen dat er iets gebeurt als zo'n event plaatsvindt. Het meest gebruikte event is `onclick`.

Voorbeeld: klik op een knop

```
<button onclick="groet()">Klik hier</button>

<script>
  function groet() {
    console.log("Hoi! Je hebt op de knop geklikt.");
  }
</script>
```

Wordt je 'gek' van console.log? Je kunt ook alert gebruiken:

```
 alert("Je hebt geklikt!");
```

?? Opdracht – Laat een knop iets doen

1. Maak een nieuw HTML-bestand in Phoenix Code: `knop.html`

2. Typ deze code als basis:

```
<!DOCTYPE html>
<html>
<head>
  <title>Gebeurtenis oefenen</title>
</head>
<body>

  <h1>Druk op de knop!</h1>

  <button onclick="toonBericht()">Klik hier</button>

  <script>
    function toonBericht() {
      console.log("Je hebt geklikt!");
    }
  </script>

</body>
</html>
```

3. Verander de tekst in de functie naar je eigen boodschap
4. Maak daarna nog een tweede knop met een andere functie, bijvoorbeeld: `toonLeeftijd()`
5. Laat die tweede functie een leeftijd of getal tonen in de console (of via alert())

Voorbeeld:

```
function toonLeeftijd() {
  console.log("Ik ben 16 jaar");
}
```

? Reflectie

- Wat is er nodig om een knop iets te laten doen?
- Wat gebeurt er als je een functie aanroept zonder dat die bestaat?

? Inleveren

1. Maak een screenshot van je HTML én console na het klikken op de knoppen

6 Formulierinvoer gebruiken met JavaScript

? Leerdoelen

- Je weet hoe je met JavaScript input uit een formulier ophaalt met `getElementById()`.
- Je kunt een `if`-statement gebruiken op basis van invoer.
- Je toont het resultaat op het scherm, niet in de console.

Start Code

```
<!DOCTYPE html>
<html>
<head>
  <title>Leeftijd check</title>
  <style>
    body { font-family: sans-serif; }
    #output { font-weight: bold; margin-top: 15px; }
  </style>
</head>
<body>

  <h1>Voer je leeftijd in</h1>

  <input type="number" id="leeftijdInput" placeholder="Typ je leeftijd">
  <button>Check mijn leeftijd</button>

  <p id="output"></p>

  <script>
    function checkLeeftijd() {
```

```
      let leeftijd = Number(document.getElementById("leeftijdInput").value);
      let boodschap = "";

      if (leeftijd >= 18) {
        boodschap = "Je bent volwassen.";
      } else {
        boodschap = "Je bent een kind.";
      }

      document.getElementById("output").innerText = boodschap;
    }
  </script>

</body>
</html>
```

? Uitleg

(lees goed door!)

Op regel **14** wordt een leeftijd gevraagd.

Op regel **15** staat een knop. Deze doet nog niets maar die moet via het `onlclick` event gekoppeld worden aan de JS functie `checkLeeftijd()` .

Op regel **17** staat een legel paragraaf. Deze paragraaf wordt (later) gevuld door de JS functie.

Op regel **21** wordt de waarde van het input veld met het id leeftijdInput gevraagd. De waarde wordt in de JS variabele leeftijd bewaard.

Op regel **30** wordt `document.getElementById("id").value` gebuikt om de waarde uit een inputveld op te halen. Het HTML element dat waarvan de tekst wordt opgehaald heeft als id `output`.

?? Opdracht – Leeftijd analyseren en tonen

1. Zorg ervoor dat de knop wordt gekoppeld aan de JavaScript functie checkLeeftijd().
2. Bereid de if-then-else structuur uit: als je ouder dat 65 bent dan ben je senior, ben je jonger dan 2 dan ben je een baby, jonger dan 6 een peuter, jonger dan 12 een jong kind, en jonger dan 18 een kind.
3. Test op een leeftijd <0 en laat dan een boodscahp zien dat de onvoer niet geldig is.

? Reflectie

- Wat gebeurt er als je niets invult of een negatieve waarde gebruikt?
- Waarom is `Number(...)` nodig?

? Inleveren

- Lever de geteste en werkende code als html bestand met daarin de JS code in.

7 Bereken een prijs met korting

In deze les gebruik je JavaScript om een prijs te berekenen na korting. Je gebruikt een formulier met invoervelden voor het bedrag en de korting in procenten. Daarna laat je in de console zien wat de nieuwe prijs is.

? Leerdoelen

- Je kunt meerdere waarden ophalen uit een formulier
- Je gebruikt JavaScript om een berekening te maken met die waarden
- Je toont het resultaat in de console

? Formule: prijs met korting

Als je een korting in procenten hebt, dan gebruik je de volgende formule:

```
kortingsbedrag = bedrag * (korting / 100);
nieuwePrijs = bedrag - kortingsbedrag;
```

Code

```
<!DOCTYPE html>
<html>
<head>
  <title>Korting berekenen</title>
```

```
</head>
<body>

  <h1>Bereken de korting</h1>

  <p>Vul een bedrag en een kortingspercentage in:</p>

  <input type="number" id="bedragInput" placeholder="Bedrag in euro's">
  <input type="number" id="kortingInput" placeholder="Korting in %">
  <button onclick="berekenKorting()">Bereken</button>

  <script>
    function berekenKorting() {
      let bedrag = Number(document.getElementById("bedragInput").value);
      let korting = Number(document.getElementById("kortingInput").value);

      let kortingBedrag = bedrag * (korting / 100);
      let nieuwePrijs = bedrag - kortingBedrag;

      console.log("Origineel bedrag: €" + bedrag.toFixed(2));
      console.log("Korting: " + korting + "%");
      console.log("Nieuwe prijs: €" + nieuwePrijs.toFixed(2));
    }
  </script>

</body>
</html>
```

?? Opdracht

1. Test het formulier met verschillende bedragen en percentages
2. Controleer of het resultaat klopt in de console
3. In plaats van console.log maak je net als bij de vorige opdracht een lege paragraaf aan en plaats daaarin het resultaat.
   De drie regels (regel 24-26 van de code) wordt dus op de pagina getoond, net als bij de vorige pagina.
4. Maak CSS stijl aanpassingen naar eigen inzicht zodat de pagina er netjes uitziet.

? Reflectie

- Wat gebeurt er als je niks invult? Wat kun je daaraan doen?
- Wat gebeurt er als je een korting invult die hoger is dan 100%?

? Inleveren

1. De complete geteste en werkende HTML/JS/CSS in één HTML bestand.

--

Portfolio Challenge

1 Persoonlijke website – HTML & vormgeving

datasource

? Leerdoelen

- Je kunt een persoonlijke website opbouwen met HTML en CSS.
- Je maakt meerdere pagina’s die met elkaar verbonden zijn via een werkend menu.
- Je past opmaak toe met een externe stylesheet.

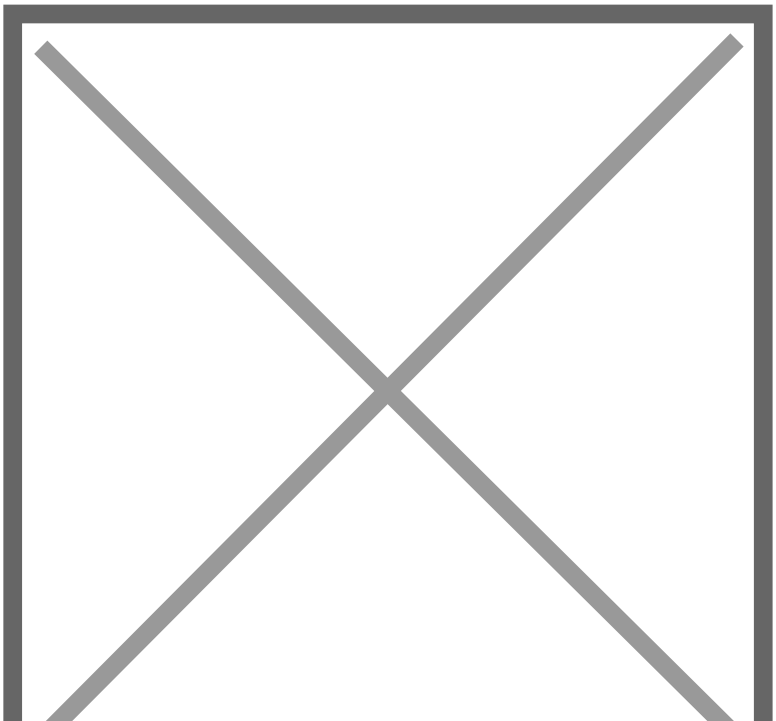
?? Bouw jouw website

Bouw een website die over jezelf gaat, de teksten van je website ga je tijdens de ***persoonlijk profileren*** lessen opstellen.

De opdracht:

- De website moet bestaan uit een **homepage** en **3 vervolgpagina's.**

  voorbeeld home pages:



Screenshot 2023-09-27 152429.png

Elke pagina bevat:

- Een **header** met jouw logo, naam en een menu naar alle pagina’s.
- Een **footer** met een copyrightregel.
- Opmaak via een externe `style.css` bestand met classes.
- Elke pagina moet een header blok hebben met **jouw logo** , **jouw naam** en een **menu** waar **alle menu-items** in staan.
  Let op, het menu moet werken!
- Elke pagina moet een **footer** hebben waar een copyright in staat.

  voorbeeld footer:

- Alle pagina's moeten worden vormgegeven met **css classes** , maak gebruik van een **externe stylesheet** .

- De website heeft de volgende pagina's / menu-items:

    1. **Home (homepage)** (index.html)
       Je homepagina is je etalage, een homepagina moet bezoekers nieuwsgierig maken naar de rest van de website.
       Zorg dat je homepage er aantrekkelijk uitziet en dat de bezoeker in een oogopslag kan zien welke informatie er op de website te vinden is.
       Gebruik op de home page afbeeldingen en links naar de vervolgpagina's

    2. **Mijn profiel (1e vervolgpagina)** (profiel.html)
       Hierin plaats je een foto met een klein verhaaltje over jezelf.
       De exacte inhoud staat in de volgende opdracht,

    3. **Over mij (2e vervolgpagina)** (overmij.html)
       Plaats hier de 6 lesopdrachten van persoonlijk profileren *(tijdlijn, normen &waarden, DISC, kernkwadranten, SWOT analyse, 360 graden feedback)*

    4. **Mijn visie (3e vervolgpagina)** (mijnvisie.html)
       Voorlopig nog leeg, deze wordt pas vele later gevuld.

?? Opdracht

1. Bouw een pagina voor de *homepage* (index *.html*).
2. Bouw een pagina voor je *profiel* (profiel.html).
3. Bouw een pagina voor je *overmij* (overmij.html).
4. Bouw een pagina voor je *visie (* visie *.html).*

Zorg dat alle pagina's hetzelfde menu hebben. Menu's moeten ook werken!
Maak van je website iets persoonlijks, geef het een mooie vorm, gebruik een achtergrond plaatje, doe iets met de font keuze.

Leef je uit en gebruik de technieken die je hebt geleerd!

? Reflectie

- Wat maakt een homepage aantrekkelijk voor een bezoeker?

- Wat is het voordeel van een apart CSS-bestand gebruiken voor styling?
- Wat vond je lastig bij het koppelen van menu's tussen meerdere pagina’s?

? Inleveren

- De **4 HTML**-bestanden: `index.html`, `profiel.html`, `overmij.html`, `visie.html`.
- Een **externe** `style.css` bestand.
- **Screenshot** van **alle** 4 pagina's in de browser (inclusief menu en footer zichtbaar).
- De **reflectie**; beantwoord de drie vragen i jouw eigen woorden en lever dit in (txt-bestand of pdf-bestand)

Je hebt in totaal dus **9 bestanden** ingeleverd.

2 Persoonlijke website – content

Plaats jouw opdrachten van Persoonlijk Profileren op jouw website. De vorm mag jezelf bepalen.

De vervolgpagina's vul je met onderstaande

Vervolgpagina Mijn profiel (profiel.html)

- Plaats hier een foto van jezelf
  Schrijf hier je naam, leeftijd, de stad waar je woont

Vervolgpagina Over mij (overmij.html)

- Plaats hier de **6 lesopdrachten** van persoonlijk profileren
  - *tijdlijn*
  - *normen &waarden*
  - *DISC*
  - *Kernkwadranten*
  - *SWOT analyse*
  - *360 graden feedback*

Vervolgpagina Mijn visie (visie.html)

- Deze laat je (nog) leeg

? Inleveren

- **Screenshot** van **alle** 4 pagina's in de browser. Zorg dat ale content goed leesbaar is. Evnetueel maak je extra screenshots zodat alles leesbaar is.

--

Extra: Design & Veilig werken

(Optionele verdieping om je project naar een hoger niveau te tillen)

? Pimp je design

Je PHP-code werkt nu, maar de kans is groot dat je pagina er nog wat "kaal" uitziet (witte achtergrond, zwarte tekst). Om je website er professioneel uit te laten zien, gebruiken developers vaak **CSS Frameworks**.

Wat zijn CSS Frameworks?

Je kunt alle CSS zelf schrijven, maar dat kost veel tijd. Een framework is als een doos met LEGO-steentjes die al klaar liggen. Je hoeft alleen de juiste "class" aan je HTML-element te geven en het ziet er direct goed uit.

- **Bootstrap:** De klassieker. Heel handig voor beginners. Je kopieert een stukje code in je `<head>` en je hebt direct toegang tot kant-en-klare knoppen, navigatiebalken en formulieren.
  Kenmerk: Je gebruikt componenten (bijv. een 'card' of 'navbar').
- **Tailwind CSS:** Modern en populair. Hierbij heb je heel veel kleine classes (zoals `text-center`, `bg-blue-500`, `p-4`) waarmee je zelf je ontwerp "bouwt" in de HTML.
  Kenmerk: Je hebt volledige vrijheid, maar je HTML krijgt wel veel classes.

□□ Hulp vragen aan AI voor design

Je hoeft geen designer te zijn. Je kunt AI vragen om de CSS of de classes voor je te schrijven. Hier zijn goede **prompts** die je kunt gebruiken:

>

Voorbeeld Prompts:

- "*Ik gebruik Bootstrap 5 via een CDN. Kun je een mooie responsive navigatiebalk maken met links naar Home, Over Mij en Contact?*"
- "*Ik heb een HTML tabel met scores. Kun je deze stylen met Tailwind CSS zodat de rijen om-en-om grijs zijn en de header donkerblauw?*"
- "*Maak een simpel CSS bestand voor mijn formulier. Zorg dat de inputs ronde hoeken hebben en de verzendknop groen wordt als je er met de muis overheen gaat.*"

Wat is een CDN?

? De basis

CDN staat voor **Content Delivery Network**. Het klinkt heel ingewikkeld, maar eigenlijk is het een slimme truc om websites sneller te laten laden voor gebruikers, waar ze ook zijn.

Je kunt een CSS framework op twee manieren toevoegen aan jouw website.

Manier 1, downloaden

Je downloadt de bestanden (`bootstrap.css`) en zet ze in je eigen mapje op je computer. Je linkt ernaar in je HTML:

```
<link rel="stylesheet" href="css/bootstrap.min.css">
```

Manier 2, via CDN

Je downloadt **niets**. Je zet gewoon een linkje in je code naar een server van het CDN. De browser van de gebruiker haalt het bestand daar vandaan.

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
```

Waarom is Optie B (CDN) vaak handiger (of niet)?

- **Snelheid:** Het CDN is vaak sneller dan jouw eigen server.

- **Cache:** Veel mensen hebben Bootstrap al eens geladen op een *andere* website. De browser heeft het bestand dan al opgeslagen ("in de cache"). Jouw site laadt dan direct!
- **Gemak:** Je hoeft geen bestanden te downloaden en in mapjes te slepen. Kopiëren, plakken, klaar.
- **Nadeel**: Als de server(s) waar de CDN staat niet werken of niet bereikbaar zijn dan werkt jouw website niet meer.

?? Veilig werken met Git & VS Code

Tijdens het programmeren (zeker als je AI code laat genereren!) gaat er wel eens iets stuk. Je wilt dan eigenlijk een "Ctrl+Z" doen, maar dan voor je hele project. Daarvoor gebruiken we **VCS**.

Wat is VCS en Git?

VCS staat voor *Version Control System*. Het is een systeem dat wijzigingen in bestanden bijhoudt. De bekendste versie daarvan heet **Git**.

Zie Git als een tijdmachine of het maken van een "Savegame" in een spel. Voordat je een moeilijke missie gaat doen (grote code wijziging), sla je het spel op. Als je 'game over' gaat, laad je gewoon je oude savegame weer in.

Wat is VSC?

VSC is gewoon de afkorting voor **Visual Studio Code**, de editor waarin je typt. Het mooie is: Git zit *ingebouwd* in VSC!

Wat is een Commit?

In Git-termen noem je het opslaan van je werk een **Commit**.
Een commit is een momentopname van al je bestanden op dat moment. Je geeft elke commit een bericht mee, bijvoorbeeld: "*Inlogformulier werkt nu*" of "*Navigatiebalk toegevoegd*".

□□ Waarom is dit handig met AI?

Stel, je vraagt ChatGPT: "*Herschrijf mijn hele PHP-code zodat het veiliger is.*" Je plakt de nieuwe code en... **BOEM**. Niets werkt meer. Error meldingen overal.

- **Zonder Git:** Paniek! Je moet met Ctrl+Z proberen terug te gaan of hopen dat je nog een backup had.
- **Met Git:** Geen probleem. Je klikt op "Ongedaan maken" (Discard changes) in VS Code en je bent direct terug bij je laatste *Commit* (je laatste savepoint).

> **Pro Tip:** Maak altijd een commit voordat je een groot stuk code van AI kopieert en plakt!

--

Kennis Check Blok 3

Kennis Check Blok 3

datasource

HTML

1. Wat is HTML?

HTML betekent **HyperText Markup Language**. Het is de **taal waarmee je een webpagina maakt**. HTML vertelt de browser wat er op een website moet staan, zoals:

- tekst (bijvoorbeeld een titel of paragraaf),
- afbeeldingen (foto’s of plaatjes),
- knoppen en formulieren,
- en links naar andere pagina’s.

Je gebruikt HTML om de **inhoud en structuur** van je website te bepalen. Denk aan het skelet van een huis: HTML zorgt ervoor dat alles op de juiste plek staat.
Later kun je met **CSS** de kleuren en opmaak toevoegen, en met **JavaScript** de pagina interactief maken.

2. Wat gebeurt er als je in "Phoenix Code" op "Run" of "Live Server" klikt?

Dan wordt je HTML-bestand zichtbaar als een echte webpagina in de browser. Je ziet meteen het resultaat van jouw code.

3. Waarom gebruiken we <!DOCTYPE html> aan het begin van elke HTML-pagina?

Deze regel vertelt de browser dat het om een moderne HTML5-pagina gaat. Zo weet de browser hoe hij de pagina moet weergeven.

4. Waarom staat er tekst tussen <h1>...</h1> of ... ?

HTML werkt met tags. Die tags geven betekenis aan de tekst: <h1> betekent een kop (grote titel), betekent een paragraaf (stukje tekst). Het zorgt voor structuur op je pagina.

5. Wat is het verschil tussen en in HTML?

 maakt tekst vet en betekent: dit is belangrijk. maakt tekst schuin en betekent: leg hier nadruk op. Beide zorgen dat de tekst opvalt, maar op een andere manier.

6. Waarom gebruik je
 in plaats van ?

Met
 breek je alleen de regel af, bijvoorbeeld in een adres of opsomming. is voor een heel nieuw stukje tekst, paragraaf (met ruimte ervoor en erna).

 staat voor " break", en staat voor "paragraaf".

7. Wat doet de `alt`-tekst bij een afbeelding?

Als de afbeelding niet wordt geladen, verschijnt de alt-tekst. Die beschrijft wat er op de afbeelding staat. Dat helpt ook mensen met een visuele beperking.

8. Wat is het verschil tussen een <ul> en een <ol> lijst?

 is een opsomming zonder volgorde (met **bolletjes**).

 is een **genummerde** lijst, waarbij de volgorde belangrijk is (zoals een top 5).

9. Waarom moet elke rij in een tabel evenveel kolommen hebben?

Anders weet de browser niet hoe hij de tabel netjes moet weergeven. Dan worden kolommen scheef of verdwijnen cellen. Elke rij moet dus evenveel <td> of <th> hebben.

10. Wat zou een formulier kunnen doen op een echte website?

Een formulier kan gegevens van een gebruiker opslaan of verzenden. Denk aan: aanmelden, iets bestellen, contact opnemen of een vraag stellen. In de les doet het formulier nog niets, maar met PHP of JavaScript kun je het echt laten werken.

CSS

1. Waarom gebruiken we CSS op een webpagina?

CSS zorgt ervoor dat een website er mooi uitziet. Je kunt kleuren, lettertypes en indeling aanpassen. Zonder CSS ziet een site er saai uit: alleen zwart-wit tekst zonder stijl.

2. Wat is het verschil tussen inline CSS en een style-blok?

Inline CSS staat direct in het HTML-element. Een style-blok zet je bovenin de pagina in de <head>. Daarmee kun je stijlen herhalen en overzichtelijker werken.

3. Waarom is het handig om een style-blok te gebruiken in plaats van inline CSS?

Met een style-blok kun je meerdere elementen tegelijk opmaken. Je hoeft dan niet elke keer dezelfde stijlregels te herhalen. Dat maakt je code netter en makkelijker aan te passen.

4. Wat is een class in CSS en waarom gebruik je dat?

Een class is een naam die je aan meerdere HTML-elementen kunt geven. Zo kun je dezelfde opmaak toepassen op verschillende plekken. Handig als je bijvoorbeeld meerdere blokken dezelfde kleur wilt geven.

5. Wat is het verschil tussen een element-selector, een class-selector en een id-selector?

Een element-selector (zoals p) geldt voor alle paragrafen. Een class-selector (zoals .belangrijk) geldt voor elementen met die class. Een id-selector (zoals #titel) is uniek en gebruik je maar één keer.

6. Wat is het box-model en waarom is het belangrijk?

Het box-model laat zien hoe ruimte om een element werkt: de tekst zit in het midden (content), dan padding, een border en buitenste ruimte heet margin. Hiermee bepaal je hoe groot een blok is en hoeveel ruimte ertussen zit.

7. Wat is het verschil tussen margin en padding?

Padding is de ruimte binnenin een element, tussen de tekst en de rand. Margin is de ruimte buiten een element, tussen het blok en andere blokken.

8. Waarom gebruik je flexbox voor layout?

Flexbox maakt het makkelijk om blokken netjes naast of onder elkaar te zetten. Je kunt ze makkelijk centreren of ruimte tussen de blokken maken, zonder moeilijke trucjes.

9. Wat betekent responsive design en hoe helpt flexbox daarbij?

Responsive design betekent dat je website er goed uitziet op elk scherm, groot of klein. Flexbox zorgt ervoor dat blokken automatisch aanpassen aan de schermgrootte.

10. Wat gebeurt er als je een element meerdere classes geeft?

Dan worden de stijlen van alle classes toegepast op dat element. Zo kun je stijlen combineren, bijvoorbeeld .menu en .speciaal. Dit geeft je veel controle over de opmaak.

JavaScript

1. Wat doet JavaScript op een webpagina?

JavaScript zorgt ervoor dat de webpagina dingen kan doen, zoals reageren op klikken of iets laten zien als je op een knop drukt. Zonder JavaScript is een pagina stil. Met JavaScript kun je hem interactief maken.

2. Wat is het verschil tussen HTML, CSS en JavaScript?

HTML is de inhoud, CSS is de opmaak (zoals kleuren en lettertypes) en JavaScript zorgt voor actie (zoals iets doen als je klikt of typt).

3. Waarom gebruiken programmeurs vaak de console?

De console laat zien wat er in de code gebeurt. Je kunt er fouten opsporen en controleren of je code werkt zoals je denkt. Het is een soort gereedschapskist.

4. Wat is een variabele en waarom gebruik je die?

Een variabele is een soort doosje waarin je iets bewaart, zoals een naam of een getal. Zo kun je er later nog iets mee doen of het laten zien.

5. Wat is het verschil tussen `let` en `const`?

Met `let` kun je het later nog veranderen, met `const` niet. Dus `const` gebruik je als iets altijd hetzelfde moet blijven.

6. Waarom zijn functies handig in je code?

Een functie is een stukje code dat je een naam geeft. Je kunt het steeds opnieuw gebruiken. Zo hoef je niet elke keer alles opnieuw te typen.

7. Wat is een parameter in een functie?

Een parameter is informatie die je aan een functie meegeeft. Zo kun je de functie flexibeler maken, bijvoorbeeld iemand begroeten met zijn eigen naam.

8. Wat is een voorwaarde (if-statement) en waar gebruik je die voor?

Een voorwaarde kijkt of iets waar is. Bijvoorbeeld: "Ben je 18 of ouder?" Als ja, dan doe je iets. Als nee, dan iets anders. Zo kun je keuzes maken in je code.

9. Wat is een event in JavaScript?

Een event is iets dat gebeurt, zoals een klik of iets typen. Met JavaScript kun je zeggen: als dit gebeurt, voer dan deze code uit.

Voorbeelden van events zijn: 'click', 'mouseover', 'keydown', ...

10. Hoe haal je invoer uit een formulier met JavaScript?

Je gebruikt `document.getElementById("id").value` om op te halen wat iemand in een vakje heeft getypt. Zo kun je iets met die invoer doen, zoals berekenen of controleren.

?? Opdracht

Maak de kennis-check.

? Inleveren

Aan het einde van de kennis-check ontvang je een certificaat. Maak een schermafdruk en lever deze in.