

Webdevelopment

- Introductie
 - Wat is webdevelopment?
 - Ontwikkelomgeving opzetten
 - Eerste stappen met HTML
 - Opdracht: Maak je eigen Webpagina
 - Opdracht: Uitbreiden van je Webpagina
 - Afronding Introductie
- HTML en Introductie tot CSS
 - HTML: Semantische Elementen
 - Inleiding tot CSS
 - Opdracht: Stijl je webpagina met CSS
 - Styling van Tekst, Kleuren en Achtergronden
 - Opdracht: Personaliseer je Webpagina
 - Afronding HTML en Introductie tot CSS
- CSS Layouts en Flexbox
 - CSS Box Model
 - Flexbox
 - Opdracht: Gebruik van Flexbox
 - Oefeningen
 - Afronding CSS Layouts en Flexbox
- CSS Grid en Responsive Design
 - Introductie tot CSS Grid
 - CSS Grid Eigenschappen
 - Opdracht: Layout Creëren met CSS Grid

- Responsive Design met Media Queries
- Oefeningen
- Afronding CSS Grid en Responsive Design

Introductie

Welkom bij het eerste hoofdstuk van webdevelopment! Deze week maak je kennis met de basisprincipes van het bouwen van websites. Je leert wat webdevelopment inhoudt, zet je eigen ontwikkelomgeving op, en maakt je eerste eenvoudige webpagina.

Wat is webdevelopment?

Webontwikkeling of te wel 'webdevelopment' is het proces van het creëren en onderhouden van websites. Het omvat alles van het ontwerpen van een eenvoudige statische pagina met tekst tot complexe webapplicaties.

Front-end vs. Back-end

- **Front-end ontwikkeling:** Dit betreft het deel van de website dat je in je browser ziet en waar je mee interacteert. Denk aan de layout, kleuren, knoppen, enzovoort. Hier gebruiken we **HTML**, **CSS** en **JavaScript**.
- **Back-end ontwikkeling:** Dit is het 'onzichtbare' deel dat op de server draait, zoals databases en server-side scripting. Denk hierbij aan bijvoorbeeld **PHP** of **C# .NET**

In deze module focussen we op **front-end ontwikkeling**.

Overzicht van HTML, CSS en JavaScript

HTML (HyperText Markup Language)

- **Functie:** Bepaalt de **structuur** en **inhoud** van een webpagina.
- **Voorbeeld:** Het maken van koppen, paragrafen, lijsten, links, enzovoort.

CSS (Cascading Style Sheets)

- **Functie:** Bepaalt de **stijl** en **opmaak** van een webpagina.
- **Voorbeeld:** Instellen van kleuren, lettertypen, layout, marges, enzovoort.

JavaScript

- **Functie:** Voegt **interactiviteit** en **dynamiek** toe aan webpagina's.
- **Voorbeeld:** Het maken van interactieve formulieren, animaties, reageren op gebruikersacties.

Ontwikkelomgeving opzetten

Stap 1: Installeer een Teksteditor

Een teksteditor is een programma waarmee je code kunt schrijven.

Aanbevolen teksteditors:

- [Visual Studio Code](#)
- [Atom](#)
- [Sublime Text](#)

Actie: Download en installeer een teksteditor naar keuze.

Stap 2: Kies een Webbrowser

Je hebt een webbrowser nodig om je webpagina's te bekijken en te testen.

Aanbevolen browsers:

- [Google Chrome](#)
- [Mozilla Firefox](#)

Deze browsers hebben ingebouwde ontwikkelaarstools die je later van pas zullen komen.

Eerste stappen met HTML

De Basisstructuur van een HTML-document

Alle HTML-documenten beginnen met een basisstructuur. Hier is een voorbeeld:

```
<!DOCTYPE html>
<html lang="nl">
<head>
  <meta charset="UTF-8">
  <title>Mijn Eerste Webpagina</title>
</head>
<body>
  <!-- Jouw inhoud komt hier -->
</body>
</html>
```

Uitleg:

- `<!DOCTYPE html>`: Geeft aan dat dit een HTML5-document is.
- `<html lang="nl">`: Het root-element met de taalinstelling (Nederlands).
- `<head>`: Bevat meta-informatie zoals de tekencodering en de titel van de pagina.
- `<title>`: De titel die in het browsertabblad wordt weergegeven.
- `<body>`: Bevat de daadwerkelijke inhoud die in de browser wordt getoond.

Basiselementen in HTML

Koppen (Headings)

Er zijn zes niveaus van koppen, van `<h1>` tot `<h6>`.

```
<h1>Dit is een Titel</h1>
<h2>Dit is een Subtitel</h2>
```

Paragrafen

Voor teksten en alinea's gebruik je het `<p>`-element.

```
<p>Dit is een alinea met tekst.</p>
```

Links

Met het `<a>`-element maak je hyperlinks.

```
<a href="https://www.voorbeeld.com">Bezoek Voorbeeld.com</a>
```

Afbeeldingen

Afbeeldingen voeg je toe met het `<img>`-element.

```

```

- `src`: De locatie van de afbeelding.
- `alt`: Alternatieve tekst als de afbeelding niet kan worden geladen.

Opdracht: Maak je eigen Webpagina

Stap 1: Maak een Nieuw HTML-bestand

1. Open je teksteditor.
2. Maak een nieuw bestand.
3. Sla het bestand op als `index.html` in een nieuwe map voor je project.

Stap 2: Voeg de Basisstructuur Toe

Kopieer de basisstructuur van het HTML-document in je `index.html`-bestand.

```
<!DOCTYPE html>
<html lang="nl">
<head>
  <meta charset="UTF-8">
  <title>Over Mij</title>
</head>
<body>
  <!-- Jouw inhoud komt hier -->
</body>
</html>
```

Stap 3: Voeg Inhoud Toe aan de Body

1. Voeg een Titel Toe met je Naam

```
<h1>Hallo, ik ben [Jouw Naam]!</h1>
```

Vervang `[Jouw Naam]` door je eigen naam.

2. Schrijf een Korte Introductie

```
<p>Ik ben een student met interesse in webontwikkeling. In mijn vrije tijd hou ik van lezen, sporten en programmeren.</p>
```

Pas de tekst aan zodat deze over jou gaat.

3. Voeg een Subtitel Toe voor je Hobby's

```
<h2>Mijn Hobby's</h2>
```

4. Maak een Lijst van je Hobby's

Je kunt een ongeordende lijst maken met `<ul>` en `<li>`.

```
<ul>
  <li>Programmeren</li>
  <li>Muziek luisteren</li>
  <li>Buiten sporten</li>
</ul>
```

5. Voeg een Afbeelding Toe

Zorg ervoor dat je een afbeelding hebt (bijvoorbeeld `profiel.jpg`) en plaats deze in dezelfde map als je `index.html`-bestand.

```

```

Stap 4: Sla het Bestand op en Bekijk je Pagina

1. Sla je `index.html`-bestand op.
2. Open de map waar je bestand is opgeslagen.
3. Dubbelklik op `index.html` om het in je webbrowser te openen.

Je zou nu je eerste webpagina moeten zien met je eigen inhoud!

Opdracht: Uitbreiden van je Webpagina

Wat te doen?

- **Voeg een extra paragraaf toe** waarin je vertelt over je favoriete vakken op school.
- **Maak links** naar je favoriete websites.

```
<a href="https://www.youtube.com/" target="_blank">Bezoek YouTube</a>
```

De `target="_blank"` zorgt ervoor dat de link in een nieuw tabblad opent.

- **Experimenteer met verschillende koppen** (`<h3>`, `<h4>`, etc.) om je pagina te structureren.
- **Voeg meer afbeeldingen toe** die relevant zijn voor jouw interesses.

Uitdaging

- **Gebruik lijsten** om een opsomming te maken van plaatsen die je hebt bezocht of wilt bezoeken.
- **Maak een tabel** met je lesschema of iets anders wat je interessant vindt.

Afronding Introductie

In dit hoofdstuk heb je het volgende geleerd:

- Wat webontwikkeling is en het verschil tussen front-end en back-end.
- De rol van HTML, CSS en JavaScript in webontwikkeling.
- Hoe je je eigen ontwikkelomgeving kunt opzetten met een teksteditor en een webbrowser.
- De basisstructuur van een HTML-document.
- Hoe je basiselementen zoals koppen, paragrafen, links en afbeeldingen gebruikt om een webpagina te maken.

Extra Hulpbronnen

- **W3Schools HTML Tutorial:** <https://www.w3schools.com/html/>
- **MDN Web Docs - Leer HTML:** <https://developer.mozilla.org/nl/docs/Learn/HTML>
- **Codecademy - Gratis HTML-cursus:** <https://www.codecademy.com/learn/learn-html>

Veelgestelde Vragen

Waarom zie ik mijn afbeelding niet op mijn webpagina?

- **Controleer of de bestandsnaam klopt**, inclusief de extensie (bijv. `.jpg`, `.png`).
- **Zit de afbeelding in dezelfde map** als je `index.html`-bestand? Zo niet, pas dan het pad aan in de `src`-attribuut.
- **Let op hoofdletters en kleine letters;** `Afbeelding.jpg` is niet hetzelfde als `afbeelding.jpg`.

Hoe kan ik mijn webpagina delen met anderen?

- Nu draait je webpagina lokaal op je eigen computer. Mocht je het willen delen met anderen dan zou je jouw HTML en CSS bestanden moeten plaatsen op een hosting omgeving. Let op! Dit kost wel geld.

Vragen of Hulp Nodig?

Als je ergens niet uitkomt of vragen hebt, voel je vrij om contact op te nemen:

- **Stel je vraag in de klas.**
- **Zoek online naar antwoorden** op forums zoals Stack Overflow.
- **Gebruik de extra hulpbronnen** die hierboven zijn vermeld.

Tips voor Succes

- **Blijf oefenen:**
Programmeervaardigheden verbeteren door regelmatig te oefenen.
- **Wees nieuwsgierig:**
Experimenteer met code en ontdek wat er gebeurt als je dingen verandert.
- **Stel vragen:**
Als je iets niet begrijpt, aarzel dan niet om hulp te vragen aan je docent of klasgenoten.
- **Gebruik bronnen:**
Maak gebruik van de beschikbare online bronnen en documentatie.

HTML en Introductie tot CSS

Je hebt je eerste stappen gezet in webontwikkeling en je eigen webpagina gemaakt met HTML. In dit hoofdstuk gaan we dieper in op HTML en beginnen we met het toevoegen van stijl met CSS om je webpagina er nog beter uit te laten zien.

Wat ga je leren?

- **Structureren van content met semantische HTML-elementen.**
- **Inleiding tot CSS en het koppelen van CSS aan HTML.**
- **Gebruik van CSS-selectors, eigenschappen en waarden.**
- **Stylen van tekst, kleuren en achtergronden.**

HTML: Semantische Elementen

Wat zijn Semantische HTML-elementen?

Semantische elementen geven betekenis aan je content. Ze beschrijven de rol van het inhoudselement in de pagina.

Veelgebruikte Semantische Elementen

Header

Gebruikt voor de koptekst van een pagina of sectie.

```
<header>
  <h1>Mijn Website</h1>
</header>
```

Nav

Gebruikt voor navigatielinks.

```
<nav>
  <ul>
    <li><a href="#home">Home</a></li>
    <li><a href="#over">Over Mij</a></li>
    <li><a href="#contact">Contact</a></li>
  </ul>
</nav>
```

Main

Bevat de hoofdinhoud van de pagina.

```
<main>
  <!-- Hoofdinhoud hier -->
</main>
```

Section

Definieert een sectie in het document.

```
<section>
  <h2>Mijn Hobby's</h2>
  <p>Ik hou van programmeren en sporten.</p>
</section>
```

Article

Gebruikt voor zelfstandige content zoals blogposts.

```
<article>
  <h2>Mijn Eerste Blogpost</h2>
  <p>Welkom bij mijn blog!</p>
</article>
```

Aside

Bevat inhoud die gerelateerd is aan de hoofdinhoud, zoals zijbalken.

```
<aside>
  <h3>Gerelateerde Links</h3>
  <ul>
    <li><a href="#link1">Link 1</a></li>
  </ul>
</aside>
```

Footer

Bevat de voettekst van de pagina of sectie.

```
<footer>
  <p>&copy; 2023 Mijn Naam</p>
</footer>
```

Waarom Semantische HTML Gebruiken?

- **Betere toegankelijkheid:** Hulpmiddelen zoals schermlezers kunnen de pagina beter interpreteren.
- **Betere SEO:** Zoekmachines begrijpen de structuur van je pagina beter.
- **Betere onderhoudbaarheid:** Je code is overzichtelijker en makkelijker te begrijpen.

Inleiding tot CSS

Wat is CSS?

CSS (Cascading Style Sheets) is een stylesheet-taal die wordt gebruikt om de presentatie van een document geschreven in HTML te beschrijven. Met CSS kun je de stijl van je webpagina's bepalen, zoals kleuren, lettertypen en layout.

CSS Koppelen aan HTML

Er zijn drie manieren om CSS aan je HTML te koppelen:

1. **Inline CSS:** Stijlregels direct in HTML-elementen

```
<p style="color: blue;">Dit is een blauwe tekst.</p>
```

2. **Interne CSS:** Stijlregels in de `<head>` van je HTML-document binnen `<style>`-tags.

```
<head>
  <style>
    p {
      color: blue;
    }
  </style>
</head>
```

3. **Externe CSS:** Een apart `.css`-bestand dat je linkt in je HTML-document.

```
<head>
  <link rel="stylesheet" href="styles.css">
</head>
```

Aanbevolen methode: Gebruik externe CSS-bestanden voor een betere organisatie en herbruikbaarheid van je code.

CSS Basisconcepten

CSS Selectors

CSS-selectors bepalen op welke HTML-elementen de stijlregels worden toegepast.

- **Element Selector:** Selecteert alle elementen van een bepaald type.

```
p {  
  color: red;  
}
```

- **Class Selector:** Selecteert elementen met een bepaalde class-attribuut.

- HTML:

```
<p class="belangrijk">Dit is belangrijk.</p>
```

- CSS

```
.belangrijk {  
  font-weight: bold;  
}
```

- **ID Selector:** Selecteert een element met een uniek id-attribuut.

- HTML:

```
<p id="intro">Welkom op mijn website.</p>
```

- CSS

```
#intro {  
  font-size: 20px;  
}
```

Eigenschappen en Waarden

CSS bestaat uit eigenschappen (properties) en waarden (values).

```
selector {  
  eigenschap: waarde;  
}
```

Voorbeeld:

```
h1 {  
  color: blue;  
  font-size: 24px;  
}
```

Er zijn ongelooflijk veel verschillende CSS eigenschappen. Geen zorgen je hoeft ze niet allemaal uit je hoofd te leren! Gaan de weg tijdens het programmeren leer je de meest voorkomende CSS eigenschappen kennen.

Opdracht: Stijl je webpagina met CSS

Deze opdracht gaat verder met de bestanden van het project uit het vorige hoofdstuk.

Stap 1: Maak een Extern CSS-bestand

1. Maak een nieuw bestand in je teksteditor.
2. Sla het bestand op als `styles.css` in dezelfde map als je `index.html`.

Stap 2: Koppel je CSS-bestand aan je HTML

Voeg in de `<head>` van je `index.html` het volgende toe:

```
<head>
  <meta charset="UTF-8">
  <title>Over Mij</title>
  <link rel="stylesheet" href="styles.css">
</head>
```

Stap 3: Voeg CSS-stijlregels toe

1. Stijl de Body

In `styles.css`:

```
body {
  background-color: #f0f0f0;
  font-family: Arial, sans-serif;
  color: #333;
```

```
margin: 0;
padding: 0;
}
```

2. Stijl de Koppen

```
h1 {
  color: #00539C;
  text-align: center;
  padding-top: 20px;
}

h2 {
  color: #027373;
  margin-left: 20px;
}
```

3. Stijl de Paragrafen

```
p {
  line-height: 1.6;
  margin: 0 20px 20px 20px;
}
```

4. Stijl de Lijst

```
ul {
  list-style-type: square;
  margin: 0 20px 20px 40px;
}
```

5. Stijl Afbeeldingen

```
img {
  display: block;
  max-width: 200px;
  height: auto;
  margin: 20px auto;
  border-radius: 50%;
}
```

6. Stijl Links

```
a {  
  color: #00539C;  
  text-decoration: none;  
}  
  
a:hover {  
  text-decoration: underline;  
}
```

Stap 4: Bekijk je Webpagina

1. Sla zowel `index.html` als `styles.css` op.
2. Vernieuw je webpagina in de browser.

Je zou nu een opgemaakte versie van je webpagina moeten zien!

Styling van Tekst, Kleuren en Achtergronden

Hieronder zijn een aantal voorbeelden aangegeven van de basis mogelijkheden van CSS opmaak.

Tekst

- **Lettertype instellen:**

```
body {  
  font-family: 'Helvetica', sans-serif;  
}
```

- **Lettergrootte:**

```
p {  
  font-size: 16px;  
}
```

- **Letterdikte:**

```
h1 {  
  font-weight: bold;  
}
```

- **Tekstkleur:**

```
h2 {  
  color: #00539C;  
}
```

Kleuren

Kleuren kunnen worden opgegeven met:

- **Naam:**

```
p {  
  color: red;  
}
```

- **Hexadecimale waarde:**

```
p {  
  color: #ff0000;  
}
```

- **RGB-waarde:**

```
p {  
  color: rgb(255, 0, 0);  
}
```

Achtergronden

- **Achtergrondkleur:**

```
body {  
  background-color: #f0f0f0;  
}
```

- **Achtergrondafbeelding:**

```
body {  
  background-image: url('achtergrond.jpg');  
  background-size: cover;  
}
```

Opdracht: Personaliseer je Webpagina

Wat te doen?

- **Gebruik Classes en IDs** om specifieke elementen te stylen.
- **Experimenteer met Kleuren** om je webpagina een thema te geven.
- **Pas het Lettertype aan** voor verschillende elementen.
- **Voeg een Achtergrondafbeelding toe** aan je pagina of specifieke secties.
- **Stijl je Navigatie** als je een menu hebt toegevoegd.

Uitdaging

- **Google Fonts:** Bezoek [Google Fonts](#) en kies een lettertype dat je mooi vindt. Voeg dit toe aan je webpagina.

Hoe te doen:

1. Kies een lettertype op Google Fonts.
2. Klik op 'Select this style' en **volg de instructies** om het lettertype in je HTML te laden.
3. Pas het toe in je CSS:

```
body {  
    font-family: 'Open Sans', sans-serif;  
}
```

Afronding HTML en Introductie tot CSS

In dit hoofdstuk heb je het volgende geleerd:

- Het belang van semantische HTML-elementen en hoe je deze gebruikt om je pagina te structureren.
- Wat CSS is en hoe je het koppelt aan je HTML-document.
- De basis van CSS-selectors, eigenschappen en waarden.
- Hoe je tekst, kleuren en achtergronden kunt stylen om je webpagina aantrekkelijker te maken.

Extra Hulpbronnen

- **W3Schools CSS Tutorial:** <https://www.w3schools.com/css/>
- **MDN Web Docs - Leer CSS:** <https://developer.mozilla.org/nl/docs/Learn/CSS>
- **CSS-Tricks:** <https://css-tricks.com/>

Veelgestelde Vragen

Waarom wordt mijn CSS niet toegepast op mijn HTML?

- **Controleer of je CSS-bestand correct is gekoppeld** in de `<head>` van je HTML.
- **Controleer op typfouten** in je bestandsnamen en paden.
- **Zorg ervoor dat je CSS-selectors kloppen** en overeenkomen met je HTML-elementen.

Hoe kan ik zien welke CSS-regels op een element worden toegepast?

- Gebruik de **ontwikkelaarstools** van je browser (F12) om elementen te inspecteren en de toegepaste CSS te bekijken.

Vragen of Hulp Nodig?

Als je ergens tegenaan loopt of vragen hebt:

- **Bespreek het in de klas.**
- **Zoek online naar oplossingen** of stel vragen op forums zoals Stack Overflow.
- **Raadpleeg de extra hulpbronnen** die hierboven zijn vermeld.

CSS Layouts en Flexbox

Je hebt nu een goede basis in HTML en CSS, en je webpagina begint vorm te krijgen. In dit hoofdstuk gaan we een grote stap zetten in het verbeteren van de layout van je website door gebruik te maken van het **CSS Box Model** en **Flexbox**. Dit zal je helpen om je webpagina professioneler en responsiever te maken.

Wat ga je leren?

- **Begrijpen van het CSS Box Model: margins, borders, padding en content.**
- **Introductie tot Flexbox voor geavanceerde lay-outs.**
- **Hoe je flex containers en items opzet.**
- **Elementen uitlijnen en ordenen met Flexbox.**
- **Je webpagina herstructureren voor een betere layout.**

CSS Box Model

Wat is het CSS Box Model?

In CSS wordt elk element op een webpagina beschouwd als een rechthoekige doos. Het **CSS Box Model** beschrijft deze doos en bestaat uit:

1. **Content:** De inhoud van het element (tekst, afbeeldingen, enz.).
2. **Padding:** Ruimte rondom de content, binnen de rand.
3. **Border:** De rand om de padding en content heen.
4. **Margin:** Ruimte buiten de rand, scheidt het element van andere elementen.

Visualisatie van het Box Model

box-model-1358225844.png

Eigenschappen van het Box Model

- **Content:** `width`, `height`
- **Padding:** `padding-top`, `padding-right`, `padding-bottom`, `padding-left`
- **Border:** `border-width`, `border-style`, `border-color`
- **Margin:** `margin-top`, `margin-right`, `margin-bottom`, `margin-left`

Voorbeeld

```
.box {  
  width: 200px;  
  height: 100px;  
  padding: 10px;  
  border: 5px solid black;  
  margin: 20px;  
}
```

Uitleg:

- `width` en `height` bepalen de grootte van de content.
- `padding` voegt ruimte toe binnen de border.

- `border` definieert de rand rondom het element.
- `margin` voegt ruimte toe buiten de border, tussen dit element en anderen.

Flexbox

Wat is Flexbox?

Flexbox, of de **Flexible Box Layout Module**, is een CSS3-layoutmodel dat het eenvoudiger maakt om responsive layouts te maken. Het is ontworpen om ruimte te verdelen en items uit te lijnen binnen een container, zelfs als hun grootte onbekend of dynamisch is.

Waarom Flexbox Gebruiken?

- **Eenvoudiger layoutbeheer:** Geen floats of complexe marges meer.
- **Responsiviteit:** Elementen passen zich automatisch aan verschillende schermgroottes aan.
- **Verticale en horizontale uitlijning:** Makkelijk items centreren en positioneren.

flexbox.png

Basisconcepten

- **Flex Container:** Het element dat `display: flex;` krijgt.
- **Flex Items:** De directe kinderen van de flex container.

Opzetten van een Flex Container

```
.container {  
  display: flex;  
}
```

Belangrijke Flexbox Eigenschappen

Voor de Container

- **flex-direction:** Bepaalt de richting van de flex items.

```
.container {  
  flex-direction: row; /* opties: row, row-reverse, column, column-reverse */  
}
```

- **justify-content:** Horizontale uitlijning van flex items.

```
.container {  
  justify-content: center; /* opties: flex-start, flex-end, center, space-between, space-around */  
}
```

- **align-items:** Verticale uitlijning van flex items.

```
.container {  
  align-items: center; /* opties: stretch, flex-start, flex-end, center, baseline */  
}
```

- **flex-wrap:** Bepaalt of flex items op meerdere regels kunnen staan.

```
.container {  
  flex-wrap: wrap; /* opties: nowrap, wrap, wrap-reverse */  
}
```

Voor de Items

- **flex-grow:** Bepaalt hoe een flex item groeit om beschikbare ruimte op te vullen.

```
.item {  
  flex-grow: 1; /* standaard is 0 */  
}
```

- **flex-shrink:** Bepaalt hoe een flex item krimpt als er minder ruimte is.
- **flex-basis:** De basisgrootte van een element voordat ruimte wordt verdeeld.

Opdracht: Gebruik van Flexbox

Stap 1: Structureren van je HTML

Pas je `index.html` aan om een flex container en flex items te hebben.

```
<body>
  <header>
    <h1>Mijn Website</h1>
  </header>
  <nav>
    <ul class="nav-list">
      <li><a href="#home">Home</a></li>
      <li><a href="#over">Over Mij</a></li>
      <li><a href="#contact">Contact</a></li>
    </ul>
  </nav>
  <main>
    <section class="intro">
      <!-- Inhoud -->
    </section>
    <section class="hobbies">
      <!-- Inhoud -->
    </section>
  </main>
  <footer>
    <p>&copy; 2023 Mijn Naam</p>
  </footer>
</body>
```

Stap 2: Maak je Navigatie Responsief met Flexbox

HTML

Zorg ervoor dat je navigatie-items binnen een `<ul>` staan met de class `nav-list`.

```
<nav>
  <ul class="nav-list">
    <li><a href="#home">Home</a></li>
    <li><a href="#over">Over Mij</a></li>
    <li><a href="#contact">Contact</a></li>
  </ul>
</nav>
```

CSS

```
/* Maak de nav-list een flex container */
.nav-list {
  display: flex;
  justify-content: center;
  list-style: none;
  padding: 0;
  background-color: #00539C;
}

/* Stijl de navigatielinks */
.nav-list li {
  margin: 0 15px;
}

.nav-list a {
  color: white;
  text-decoration: none;
  font-weight: bold;
}

.nav-list a:hover {
```

```
text-decoration: underline;
}
```

Stap 3: Layout van de Hoofdinhoud met Flexbox

HTML

In je `main`-element heb je twee secties: `.intro` en `.hobbies`.

```
<main>
  <section class="intro">
    <!-- Inhoud -->
  </section>
  <section class="hobbies">
    <!-- Inhoud -->
  </section>
</main>
```

CSS

Maak van `main` een flex container en pas de layout aan.

```
main {
  display: flex;
  flex-direction: row;
  justify-content: space-around;
  align-items: flex-start;
  flex-wrap: wrap;
  margin: 20px;
}

/* Stijl de secties */
main section {
  flex: 1 1 300px; /* flex-grow, flex-shrink, flex-basis */
  margin: 10px;
  padding: 20px;
  background-color: #f9f9f9;
```

```
border: 1px solid #ccc;
}
```

Uitleg:

- `flex-direction: row;` zorgt ervoor dat de secties naast elkaar staan.
- `flex-wrap: wrap;` zorgt ervoor dat de secties naar een nieuwe regel gaan als er niet genoeg ruimte is.
- `flex: 1 1 300px;` betekent:
 - `flex-grow: 1;` – het item kan groeien als er ruimte is.
 - `flex-shrink: 1;` – het item kan krimpen als er minder ruimte is.
 - `flex-basis: 300px;` – de startbreedte is 300px.

Stap 4: Responsieve Afbeeldingen met Flexbox

HTML

Zet je afbeeldingen binnen een container.

```
<section class="gallery">
  
  
  
</section>
```

CSS

Maak van `.gallery` een flex container.

```
.gallery {
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
  margin: 20px;
}

.gallery img {
  flex: 1 1 200px;
  max-width: 300px;
```

```
margin: 10px;
```

```
height: auto;
```

```
}
```

Oefeningen

Oefening 1: Maak een Flexbox Layout voor je Footer

Wat te doen?

- Maak van je `<footer>` een flex container.
- Voeg drie `<div>`-elementen toe binnen je footer met informatie zoals contactgegevens, sociale media links, en copyright.

Oefening 2: Centraal Uitlijnen van Inhoud

Wat te doen?

- Centreer de inhoud van je `.intro` sectie zowel horizontaal als verticaal.

Uitdaging: Maak een Responsief Navigatiemenu

Wat te doen?

- Zorg ervoor dat het navigatiemenu zich aanpast op kleinere schermen.
- Gebruik media queries om het menu te veranderen in een dropdown op mobiele apparaten.

Hint

Gebruik een media query om de flex-direction te veranderen:

```
@media (max-width: 600px) {  
  .nav-list {  
    flex-direction: column;
```

align-items: center;

}

}

Afronding CSS Layouts en Flexbox

In dit hoofdstuk heb je het volgende geleerd:

- **Het CSS Box Model:** Begrip van margins, borders, padding en content en hoe deze de layout beïnvloeden.
- **Introductie tot Flexbox:** Hoe je flex containers en flex items opzet en stylet.
- **Flexbox Eigenschappen:** Gebruik van `flex-direction`, `justify-content`, `align-items`, `flex-wrap`, en `flex`.
- **Praktische Toepassing:** Je webpagina herstructureren met Flexbox voor een responsieve en flexibele layout.

Extra Hulpbronnen

- **W3Schools Flexbox Tutorial:** https://www.w3schools.com/css/css3_flexbox.asp
- **MDN Web Docs - Flexbox:**
https://developer.mozilla.org/nl/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox
- **Flexbox Froggy:** <https://flexboxfroggy.com/> – Een leuke manier om Flexbox te oefenen.

Veelgestelde Vragen

Mijn flex items overlappen of worden niet correct weergegeven. Wat kan ik doen?

- **Controleer je HTML-structuur:** Zorg ervoor dat je de juiste elementen als flex container hebt ingesteld.
- **Bekijk je CSS-selectors:** Zijn ze correct toegepast op de juiste elementen?
- **Gebruik de ontwikkelaarstools:** Inspecteer de elementen om te zien welke CSS-regels worden toegepast.

Hoe maak ik mijn layout responsief met Flexbox?

- **Gebruik flex-wrap:** Dit zorgt ervoor dat items naar een nieuwe regel gaan als er niet genoeg ruimte is.
- **Gebruik percentages of flex-grow:** Hierdoor passen items zich aan de beschikbare ruimte aan.
- **Gebruik media queries:** Pas de layout aan op basis van schermgrootte.

Vragen of Hulp Nodig?

Als je vragen hebt of ergens tegenaan loopt:

- **Stel je vraag in de klas** of aan je docent.
- **Oefen met online tools** zoals Flexbox Froggy om je begrip te verbeteren.
- **Raadpleeg de extra hulpbronnen** voor meer informatie en voorbeelden.

CSS Grid en Responsive Design

Je hebt nu een solide basis in HTML en CSS, en je bent bekend met Flexbox voor het creëren van flexibele layouts. In dit hoofdstuk gaan we een stap verder door **CSS Grid** te introduceren, een krachtig layout-systeem dat je nog meer controle geeft over de opmaak van je webpagina's. Daarnaast duiken we in **responsief ontwerp** of te wel 'Responsive Design', zodat je website er op elk apparaat geweldig uit ziet.

Wat ga je leren?

- **Introductie tot CSS Grid Layout.**
- **Het creëren van complexe layouts met CSS Grid.**
- **Gebruik van media queries voor responsief ontwerp.**
- **Je website aanpassen voor verschillende schermgroottes: desktop, tablet en mobiel.**

Introductie tot CSS Grid

Wat is CSS Grid?

CSS Grid Layout, vaak gewoon **Grid** genoemd, is een tweedimensionaal grid-systeem dat je in staat stelt om elementen zowel in de rij- als in de kolomrichting te plaatsen. Het is ontworpen om complexe weblayouts eenvoudiger te maken.

grid.png

Waarom CSS Grid Gebruiken?

- **Tweedimensionale Controle:** In tegenstelling tot Flexbox, dat voornamelijk eendimensionaal is (rij of kolom), kun je met Grid zowel rijen als kolommen tegelijk beheren.
- **Complex Layouts:** Creëer geavanceerde en responsieve layouts met minder code.
- **Flexibiliteit:** Pas de layout aan op basis van de inhoud en beschikbare ruimte.

Basisconcepten

- **Grid Container:** Het element dat `display: grid;` krijgt.
- **Grid Items:** De directe kinderen van de grid container.
- **Grid Lines:** De lijnen die de rijen en kolommen scheiden.
- **Grid Tracks:** De rijen en kolommen zelf.
- **Grid Cells:** De ruimtes tussen de lijnen, vergelijkbaar met cellen in een tabel.

Opzetten van een Grid Container

```
.container {  
  display: grid;  
}
```

CSS Grid Eigenschappen

Definiëren van Kolommen en Rijen

Gebruik de eigenschappen `grid-template-columns` en `grid-template-rows`.

Voorbeeld

```
.container {  
  display: grid;  
  grid-template-columns: 200px 200px 200px;  
  grid-template-rows: 100px 100px;  
}
```

Uitleg:

- **Kolommen:** Drie kolommen van elk 200px breed.
- **Rijen:** Twee rijen van elk 100px hoog.

Gebruik van de `fr` Eenheid

De `fr` (fraction) eenheid verdeelt beschikbare ruimte in fracties.

```
.container {  
  display: grid;  
  grid-template-columns: 1fr 2fr 1fr;  
}
```

Uitleg:

- De totale beschikbare ruimte wordt verdeeld in 4 fracties (1 + 2 + 1).
- De eerste kolom krijgt 1/4 van de ruimte.
- De tweede kolom krijgt 2/4 (of 1/2) van de ruimte.
- De derde kolom krijgt 1/4 van de ruimte.

Grid Gap

Voeg ruimte toe tussen de grid items met `grid-gap`.

```
.container {  
  display: grid;  
  grid-template-columns: repeat(3, 1fr);  
  grid-gap: 20px;  
}
```

Plaatsen van Grid Items

Gebruik `grid-column` en `grid-row` om items te positioneren.

Voorbeeld

```
.item1 {  
  grid-column: 1 / 3; /* Van kolomlijn 1 tot 3 */  
  grid-row: 1 / 2; /* Van rijlijn 1 tot 2 */  
}
```

Opdracht: Layout Creëren met CSS Grid

Stap 1: HTML Structuur Aanmaken

Maak een HTML bestand genaamd `grid.html` aan om de grid layout toe te passen. Vergeet niet een bijbehorend CSS bestand aan te maken en te koppelen.

```
<body>
  <header>
    <h1>Mijn Website</h1>
  </header>
  <nav>
    <!-- Navigatie-inhoud -->
  </nav>
  <main class="grid-container">
    <section class="item item1">
      <!-- Inhoud Sectie 1 -->
    </section>
    <section class="item item2">
      <!-- Inhoud Sectie 2 -->
    </section>
    <aside class="item item3">
      <!-- Zijbalk of extra informatie -->
    </aside>
  </main>
  <footer>
    <!-- Footer-inhoud -->
  </footer>
</body>
```

Stap 2: Aanpassen van je Inhoud

- **item1:** Plaats hier je hoofdinhoud, zoals een introductie.
- **item2:** Gebruik dit voor aanvullende inhoud, zoals hobby's of iets anders.
- **item3:** Gebruik de aside voor extra informatie of links.

Stap 3: CSS Grid Implementeren

CSS voor de Grid Container

```
.grid-container {  
  display: grid;  
  grid-template-columns: 2fr 1fr;  
  grid-template-rows: auto;  
  grid-gap: 20px;  
  margin: 20px;  
}
```

Uitleg:

- **grid-template-columns: 2fr 1fr;** Twee kolommen waarbij de eerste tweemaal zo breed is als de tweede.
- **grid-template-rows: auto;** Rijen worden automatisch aangepast aan de inhoud.
- **grid-gap: 20px;** Ruimte tussen de grid items.

CSS voor de Grid Items

```
.item1 {  
  grid-column: 1 / 2;  
  grid-row: 1;  
}  
  
.item2 {  
  grid-column: 1 / 2;  
  grid-row: 2;  
}  
  
.item3 {  
  grid-column: 2 / 3;  
  grid-row: 1 / 3;  
}
```

Uitleg:

- **item1** en **item2** staan in de eerste kolom.
- **item3** beslaat de tweede kolom en strekt zich uit over twee rijen.

Als je de webpagina bekijkt dan zouden de elementen gepositioneerd moeten zijn volgens het CSS Grid principe.

Responsive Design met Media Queries

Wat zijn Media Queries?

Media queries in CSS stellen je in staat om stijlregels toe te passen op basis van kenmerken van het apparaat, zoals schermgrootte, resolutie, oriëntatie, enz.

Basis Syntax

```
@media (media-type) and (expressie) {  
  /* CSS-regels hier */  
}
```

Voorbeeld

```
@media screen and (max-width: 768px) {  
  /* Stijlen voor schermen kleiner dan of gelijk aan 768px */  
}
```

Toepassen op je Website

Stap 1: Mobile-First Benadering

Begin met het ontwerpen voor mobiele apparaten en schaal op naar grotere schermen.

Stap 2: Aanpassen van de Grid Layout

Voor kleinere schermen wil je wellicht dat je layout verandert naar één kolom.

```
/* Basis layout voor mobiel */  
.grid-container {  
  display: grid;
```

```
    grid-template-columns: 1fr;
}

/* Voor grotere schermen */
@media (min-width: 769px) {
    .grid-container {
        grid-template-columns: 2fr 1fr;
    }
}
```

Stap 3: Aanpassen van Navigatie

Voor mobiel kun je de navigatie vereenvoudigen.

```
/* Mobiele navigatie */
nav {
    background-color: #00539C;
}

.nav-list {
    display: flex;
    flex-direction: column;
    align-items: center;
}

.nav-list li {
    margin: 10px 0;
}

/* Desktop navigatie */
@media (min-width: 769px) {
    .nav-list {
        flex-direction: row;
    }

    .nav-list li {
        margin: 0 15px;
    }
}
```


Oefeningen

Oefening 1: Maak je Website Volledig Responsief

Wat te doen?

- **Pas media queries toe** om je website aan te passen voor verschillende schermgroottes.
- **Controleer je website op verschillende apparaten** of gebruik de ontwikkelaarstools van je browser om te simuleren.

Tips

- Begin met een **mobile-first** benadering.
- Gebruik **flexibele eenheden** zoals percentages of `fr` in Grid.
- Test op **breekpunten** zoals 480px (mobiel), 768px (tablet), en 1024px (desktop).

Uitdaging 1: Gebruik van Grid Areas

Wat te doen?

- Maak gebruik van **grid-template-areas** om je layout te definiëren.

Uitdaging 2: Voeg Een Responsieve Afbeeldingsgalerij Toe

Wat te doen?

- Maak een sectie met afbeeldingen die zich aanpassen aan de schermgrootte.
- Gebruik CSS Grid om de afbeeldingen in een raster te plaatsen dat verandert op basis van het scherm.

Tips

- Gebruik `grid-template-columns` met de `repeat()` en `auto-fill` of `auto-fit` functies.

Afronding CSS Grid en Responsive Design

Je hebt geleerd hoe je CSS Grid kunt gebruiken om geavanceerde en responsieve layouts te maken. Met deze vaardigheden kun je ervoor zorgen dat je website er op elk apparaat geweldig uitziet.

Je hebt in dit hoofdstuk het volgende geleerd:

- **CSS Grid Layout:** Hoe je Grid kunt gebruiken om complexe en flexibele layouts te maken.
- **Grid Eigenschappen:** Gebruik van `grid-template-columns`, `grid-template-rows`, `grid-gap`, `grid-area`, enz.
- **Responsief Ontwerp:** Hoe je media queries gebruikt om je website aan te passen aan verschillende schermgroottes.
- **Mobile-First Benadering:** Eerst voor mobiele apparaten ontwerpen en vervolgens uitbreiden voor grotere schermen.

Extra Hulpbronnen

- **W3Schools CSS Grid Tutorial:** https://www.w3schools.com/css/css_grid.asp
- **MDN Web Docs - CSS Grid:** https://developer.mozilla.org/nl/docs/Web/CSS/CSS_Grid_Layout
- **A Complete Guide to Grid:** <https://css-tricks.com/snippets/css/complete-guide-grid/>
- **Grid Garden:** <https://cssgridgarden.com/> – Oefen met CSS Grid op een leuke manier.

Veelgestelde Vragen

Wat is het verschil tussen Flexbox en Grid?

- **Flexbox** is ontworpen voor eendimensionale layouts (rij of kolom). Het is ideaal voor het uitlijnen van items in een enkele rij of kolom.
- **Grid** is tweedimensionaal en stelt je in staat om zowel rijen als kolommen te beheren. Het is geschikt voor complexe layouts die in beide richtingen variëren.

Mijn media queries werken niet. Wat kan ik doen?

- **Controleer de syntax:** Zorg ervoor dat je media queries correct zijn geformatteerd.
- **Plaatsing in CSS:** Media queries moeten na de standaardstijlen komen of specifiekere selectors hebben.
- **Cache legen:** Soms moet je de cache van je browser legen om de nieuwste wijzigingen te zien.

Hoe test ik mijn website op verschillende apparaten?

- **Browser Ontwikkelaarstools:** Gebruik de responsieve modus (meestal toegankelijk via F12).
- **Online Emulators:** Websites zoals BrowserStack kunnen helpen.
- **Fysieke Apparaten:** Als je toegang hebt tot verschillende apparaten, test dan daarop.

Vragen of Hulp Nodig?

Als je vragen hebt of ergens tegenaan loopt:

- **Stel je vraag in de klas** of aan je docent.
- **Gebruik online hulpbronnen** en communities zoals Stack Overflow.
- **Experimenteer:** Probeer verschillende CSS Grid en media query technieken uit om te zien wat ze doen.