Hoofdstuk 4: CSS - de basis

Doel van CSS

Cascading Style Sheets of CSS heeft als doel de **opmaak van een webpagina te verzorgen**. Via CSS kan je aan de hand van stijlregels bijvoorbeeld de achtergrond van je pagina een andere kleur geven, het lettertype van een paragraaf wijzigen of de positie van verschillende blokken aanpassen.

Algemeen principe

CSS is in principe een eenvoudige taal om te gebruiken. Een CSS document bestaat uit een set van stijlregels of **CSS rules**. Elke regel bestaat uit twee delen:

1. Een **selector** om aan te geven welke elementen de regels moeten toepassen
2. Een of meerdere **declaraties** om aan te geven welke eigenschap(pen) er moet(en) aangepast worden. Elke declaratie bestaat uit:
    a. De **naam** van de eigenschap
    b. De **waarde** die de eigenschap moet krijgen

Als je bijvoorbeeld de tekst van alle <p> tags in het wit wil zetten met een rode achtergrond kan dit met de volgende CSS rule:

```
p{
    color: white;
    background-color: red;
}
```

- De **selector** is de <p> tag. De CSS rule zal dus op alle paragrafen toegepast worden.
- Er zijn **twee declaraties**. De eerste past de tekstkleur aan, de tweede de achtergrond kleur.
    - De namen van de properties zijn color en background-color.
    - De waarden worden ingesteld op white en red.
    - Merk dat er een dubbele punt (:) staat tussen de naam van de property en de waarde
    - Merk dat elke declaratie afgesloten wordt met een puntkomma (;).

Algemeen principe

Lorum Ipsum...

Lorum Ipusm...

Je kan dit voorbeeld ook bekijken op https://codepen.io/fdc/pen/NLqBKr.

Inline, intern en extern

Er zijn drie plaatsen waar je CSS code kan schrijven. Een eerste mogelijkheid is om dit **inline bij de HTML tags** te schrijven:

```
<p style="color: white; background-color: red">Lorum Ipsum...</p>
```

Je doet dit echter nooit aangezien je dezelfde code moet herhalen voor elke <p> tag in het document. Als de klant later vraagt om de kleur aan te passen moet je de code op verschillende plaatsen in het document aanpassen.

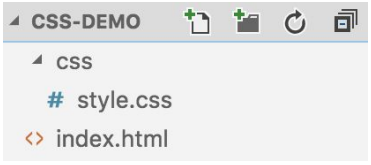
Een tweede mogelijkheid is om een **interne stylesheet** te voorzien. Je doet dit door een <style> tag te voorzien binnen de <head> tag.

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <style>
        p{
            color: white;
            background-color: red;
        }
    </style>
</head>
```

Deze methode is al beter dan inline CSS, maar aangezien een website meestal uit meerdere pagina's bestaat zal je de code op elke pagina moeten herhalen. Daarnaast streven we ook een **strikte scheiding** na tussen HTML (structuur) en CSS (opmaak).

De laatste mogelijkheid is een **extern CSS bestand** te voorzien. Alle stijlregels voor de website komen dan in dit externe bestand dat gelinkt wordt aan de HTML bestanden. Op die manier hoef je de code maar op één plaats aan te passen om het resultaat in alle pagina's te zien.

Net als een HTML file is een CSS file gewoon een tekstfile, maar dan met de extensie .css. Binnen deze cursus plaats je CSS bestanden altijd in een mapje "css" en noem het bestand "style.css". Je bekomt dan de volgende mappenstructuur:



Het CSS bestand moet wel nog gekoppeld worden aan het HTML bestand. Dit doe je met de <link> tag binnen de <head> tag.

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>CSS demo</title>
    <link rel="stylesheet" href="css/style.css">
</head>
```

Selectoren

Er zijn heel wat mogelijke selectoren in CSS. Binnen deze cursus wordt vooral met de class selector gewerkt. Hieronder vind je een overzicht van de minst specifieke selector tot de meest specifieke selector.. Je kan dit ook nalezen op MDN.

Universal selector

Deze selector gebruik je om alle HTML elementen aan te spreken.

```
*{
    text-transform: uppercase;
}
```

Bovenstaande code zorgt er voor dat alle tekst in hoofdletters komt te staan.

Type selector

Deze selector gebruik je om één of meerdere HTML elementen aan te spreken

```
h1{
    font-size: 16px;
}
```

Bovenstaande code zorgt er voor dat alle <h1> tags een lettergrootte van 16px krijgen.

Class selector

Dit is de belangrijkste selector om de BEM notatie (zie later) te gebruiken. Deze selector zal alle elementen aanspreken die een class attribuut hebben met dezelfde naam. Let wel: deze naam is hoofdlettergevoelig. Maak er een goede gewoonte van om alles in kleine letters te schrijven.

```
.menu{
    background-color: red;
}
```

Bovenstaande code zal alle elementen die een class attribuut hebben met de waarde menu (class=”menu”) een rode achtergrond geven.

Child selector

Deze selector zal een rechtstreeks child van een element aanspreken.

```
.menu>.menu-item{
    color: white;
}
```

Bovenstaande code zal alle elementen met een class attribuut “menu-item” aanspreken als het element een rechtstreeks child is van een element met als class attribuut “menu”.

Descendant selector

Deze selector zal een child element van een element aanspreken. Het child element hoeft echter geen rechtstreeks child te zijn van het parent element.

```
ul a{
    text-decoration: none;
}
```

Bovenstaande code zal de onderlijning van alle <a> tags verwijderen die ergens onder een <ul> tag zitten. De <a> tag mag gerust nog binnen een <li> tag en andere tags zitten.

Adjacent sibling selector

Deze selector zal het eerstvolgende element selecteren dat op hetzelfde niveau staat.

```
h1+p{
    color: blue;
}
```

Bovenstaande code zal de <p> tag selecteren die direct na een <h1> tag komt. Als er na een <h1> tag vijf <p> tags komen zal enkel de eerste aangesproken worden.

General sibling selector

Deze selector zal alle volgende elementen selecteren die op hetzelfde niveau staan.

```
h1~p{
    color: blue;
}
```

Bovenstaande code zal alle <p> tags selecteren die direct na een <h1> tag komen. Als er na een <h1> tag vijf <p> tags komen zullen deze allemaal geselecteerd worden.

Selectoren combineren

Je kan altijd selectoren combineren door er een komma tussen te plaatsen

```
.menu, .intro, .contact, p{
    font-size: 12px;
}
```

Bovenstaande code zal alle elementen die een class attribuut met de waarde “menu”, “intro” of “contact hebben een lettergrootte van 12px geven, net als alle elementen met een <p> tag.

Cascading

Cascading betekent letterlijk een kleine waterval. Wanneer je verschillende stijlregels hebt die hetzelfde element selecteren zal de meer specifieke selector voorrang krijgen op een minder specifieke. Bijvoorbeeld: een class selector zal altijd voorrang krijgen op een type selector.

Block Element Modifier

Een CSS bestand voor een professionele website kan al snel uit duizend regels code bestaan. Hoe groter het aantal stijlregels in een bestand, hoe groter de kans dat je dubbele namen gebruikt of dat stijlregels elkaar overschrijven zonder dat dit de bedoeling is.
BEM is - onder andere - een naming convention om je CSS code leesbaarder te houden.

Definities

De definities kan je nalezen op http://getbem.com/introduction/.
BEM bestaat uit drie onderdelen:

- **Block**: dit is een op zichzelf staand stuk HTML. Bijvoorbeeld: een header, een menu, een artikel,...
- **Element**: dit is een onderdeel van een block dat semantisch verbonden is met het block. Bijvoorbeeld: een menu-item, de titel van een header,...
- **Modifier**: dit is een "flag" op een element met als doel het uitzicht of gedrag van het element aan te passen. Bijvoorbeeld: een menu-item actief zetten, een knop disablen,...

Deze definities klinken waarschijnlijk erg vaag. Daarom een paar voorbeelden om ze te verduidelijken.

```
<ul class="menu">
    <li class="menu__item">Eerste</li>
    <li class="menu__item">Tweede</li>
    <li class="menu__item menu__item--active">Derde</li>
</ul>
```

- **Block**: het <ul> element staat op zichzelf
- **Element**: de <li> elementen zijn semantisch verbonden met het <ul> element
- **Modifier**: het laatste <li> element is het active menu-item en dient dus een extra opmaak te krijgen

```
<header class="header">
   <h1 class="header__title">Titel</h1>
   <p class="header__subtitle">
       <span class="subtitle__word">Sub</span>
       <span class="subtitle__word subtitle__word--size-big">titel</span>
   </p>
   <img class="header__logo" src="assets/img/logo.jpg" alt="logo">
</header>
```

- **Block**:
    - het element <header> staat op zichzelf.
    - Het <p> element kan op zichzelf staan, maar is in dit geval semantisch verbonden met het <header> element.
- **Element**:
    - de <p> en <img> elementen zijn semantisch verbonden met het <header> element.

    - de elementen zijn semantisch verbonden met het <p> element.
- **Modifier**: het tweede element heeft extra opmaak nodig.

Naamgeving

Zoals je in bovenstaande voorbeelden merkt zijn er afspraken rond de naam van het class attribuut voor elk onderdeel. De regels rond naamgeving kan je nalezen op: http://getbem.com/naming/.

- **Block**: een **korte bondige naam** die het block omschrijft.
- **Element**: de naam van het block gevolgd door een **dubbele underscore** met daarna de naam voor het element.
- **Modifier**: de naam van het block, gevolgd door een **dubbele underscore**, gevolgd door de naam van het element, gevolgd door een **dubbel koppelteken**, gevolgd door de naam van de modifier.

Meerdere classes instellen

Zoals je misschien al gemerkt hebt kan een element meerdere class attributen hebben. Deze worden gescheiden door een spatie. Bijvoorbeeld:

```
<li class="menu__item menu__item--active">Derde</li>
```

Op dit element worden zowel de stijlregels van menu__item als menu__item--active toegepast. Dit is een heel krachtig systeem om je code zo compact mogelijk te houden. Alles wat menu items gemeenschappelijk hebben kan in menu__item. Zaken die specifiek zijn voor een actief menu item komen dan in menu__item--active. Zo hoef je niet alle declaraties te herhalen.

OPMERKING: het combineren van verschillende classes is niet specifiek iets van BEM, maar gewoon een goede best practice om CSS code zo optimaal mogelijk te schrijven.

BEM in de praktijk

Een uitgewerkt voorbeeld van een menustructuur waar de BEM notatie gebruikt wordt kan je vinden op https://codepen.io/fdc/pen/WgvWyY.

Inheritance

Overerving of inheritance zorgt er voor dat je niet op elk element alle properties hoeft in te stellen. Bekijk het voorbeeld op https://codepen.io/fdc/pen/yxNWyy. Pas gerust enkele waarden aan en bekijk het resultaat.

```
.topic{
 font-size: 14px;
 background-color: red;
}
.topic__title{
 font-size: 38px;
}
.topic__content--intro{
 font-size: 18px;
```

```
}
```

Op het <article> element (het block) wordt een font-size van 14px ingesteld. Je merkt dat alle elementen binnen het <article> die size **overnemen** tenzij deze expliciet wordt overschreven door een andere waarde. Op het element topic__content hoef je dus de font-size niet meer op te geven. Hetzelfde principe is van toepassing op de rode achtergrondkleur.

Weet wel dat niet elke CSS property overgeërfd wordt. De documentatie op MDN geeft telkens aan of dit voor een bepaalde property al dan niet het geval is.

CSS properties

Er zijn enorm veel verschillende CSS properties beschikbaar. Je kan deze terugvinden op MDN. Het is onbegonnen werk om deze één voor één te bespreken. Je leert de meeste properties kennen door ze gewoon te gebruiken. Onderstaande secties bespreken enkele veel voorkomende properties.

Tekstkleur en achtergrondkleur

Dit kan ingesteld worden via color en background-color. Als waarde kan je zowel de **naam** van een **kleur** als een **hex waarde** opgeven.

Lettertype

De meeste browsers kiezen standaard Times New Roman als lettertype. Je kan dit uiteraard ook met CSS aanpassen via de property font-family.
Hou er wel rekening mee dat het lettertype op het toestel van de gebruiker moet geïnstalleerd zijn om te gebruiken in de browser. Gelukkig zijn er een aantal lettertypes die op zo goed als elk toestel geïnstalleerd zijn. De meest bekende zijn Times New Roman en Arial. Een overzicht kan je vinden op https://www.cssfontstack.com/. Sowieso voorzie je **altijd een fallback naar een generische font** zoals serif of sans-serif. Bijvoorbeeld:

```
body{
    font-family: Arial, Verdana, sans-serif;
}
```

Bovenstaande code stelt het lettertype voor het volledige document in op Arial. Als dit font niet op het toestel van de gebruiker staat wordt gebruik gemaakt van Verdana. Als dit lettertype ook niet aanwezig is valt de browser terug op zijn standaard sans-serif font.

Tekstopmaak

Enkele properties die veel gebruikt worden om tekst op te maken zijn:

- font-size: instellen van de lettergrootte
- font-weight: tekst in het vet zetten of niet
- font-style: tekst cursief zetten of niet
- text-transform: tekst in hoofdletters, kleine letters of capitalize
- text-decoration: tekst onderlijnen of doorstrepen
- line-height: de lijnhoogte van een stuk tekst
- text-align: tekst links, rechts of centraal uitlijnen

CSS units

In CSS kan je verschillende eenheden gebruiken om bijvoorbeeld een font in te stellen, afstand tussen twee blokken te bepalen, dikte van een rand op te geven,...

Pixels (px)

Tot nu toe hebben we enkel pixels (px) gebruikt om een font-size in te stellen. Het probleem is echter dat pixels een **absolute eenheid** zijn. Als je wil dat een tekst dubbel zo groot is als een andere tekst moet je dit telkens berekenen. Een absolute eenheid zorgt er ook voor dat een font-size niet mee zal schalen als de gebruiker de standaard font-size van de browser aanpast.

Percentages (%)

De standaard lettergrootte in een browser is 16px wat overeenkomt met 100%. Als je de font-size instelt op 75% zal die dus 12px groot zijn. Het probleem met percentages is echter dat een font-size altijd wordt **berekend ten opzichte van de parent**. Onderstaande code illustreert dit probleem:

```
body{
    font-size: 75%;
}
.content{
    font-size: 50%;
}
```

De <body> tag stelt de size in op 75% (12px). Een element content stelt de size in op 50% van die 12px, dus op 6px. Zo is het moeilijk bij te houden hoe groot een font nu precies zal zijn.

EM

Een em waarde is gelijk aan de **breedte van de letter M**. Deze eenheid werkt net als percentages relatief **ten opzichte van de parent**. Bij het nesten van bijvoorbeeld lijsten kan dit dezelfde problemen opleveren als bij het werken met percentages.

REM

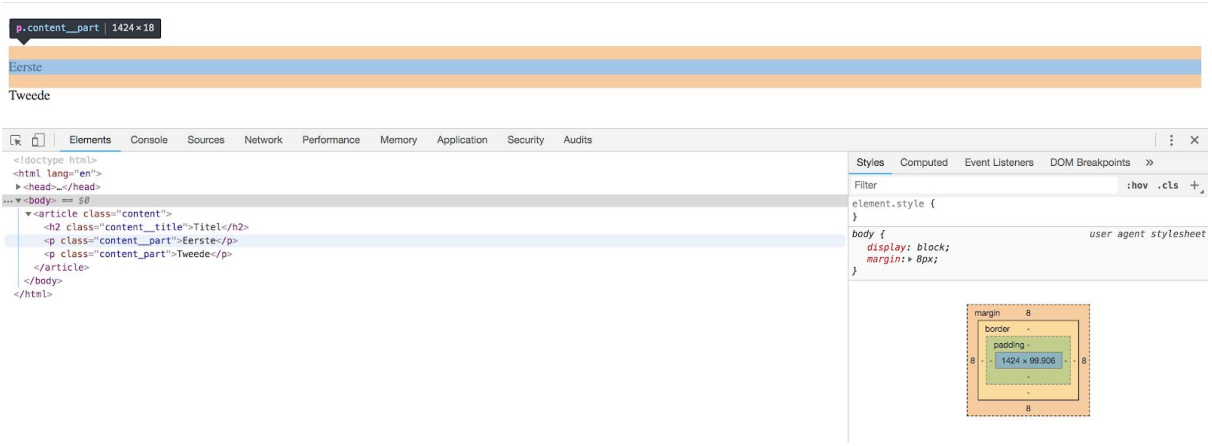
Bij rem **bepaalt het root element de grootte** in plaats van het parent element.
De standaard font-size van de browser is 16px. Door 10 te delen door 16 (10/16 = 62,5%) stel je 1rem gelijk aan 10 pixels, 1.4rem is dan gelijk aan 14px, enz,... Dit rekent een stuk gemakkelijker. In CSS doe je dit als volgt:

```
html{
    font-size: 62.5%;
}
.content{
    font-size: 1.2rem;
}
```

De font-size van het block content zal nu 12 pixels zijn. Tijdens de rest van de cursus zal je **meestal met rem waarden werken**.

Box model

Als je met de inspector over een block element beweegt zie je dat er in feite een **onzichtbare box rond het element** staat. Deze box kan je gebruiken om verschillende properties in te stellen zoals de breedte, hoogte, borders, afstand tot content en andere elementen. Merk op dat inline elementen zoals of zo geen box hebben.



Width en height

De breedte van de box zal standaard de volledige breedte van de browser innemen. De hoogte is ingesteld op de inhoud van de box. Via de properties width en height kun je deze afmetingen aanpassen in CSS. Deze waarden kunnen pixels, procenten, em's of rems zijn. Meestal werk je met rem, maar voor de afmetingen van een box kan het ook handig zijn om procenten te gebruiken. Een voorbeeld kan je vinden op https://codepen.io/fdc/pen/EeVEgL.

Je kan via de properties min-width, min-height, max-width en max-height ook een minimale en maximale breedte en hoogte instellen.

Overflow

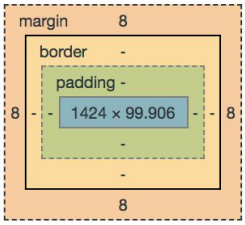
Als de content binnen een box groter is dan de afmetingen van de box is er sprake van overflow. Je kan overflow op verschillende manier aanpakken:

- visible: de content buiten de box wordt getoond
- hidden: de content buiten de box wordt niet getoond
- scroll: een scrollbar tonen aan de rand van de box

Een voorbeeld kan je vinden op https://codepen.io/fdc/pen/ZMbxea. Pas de waarde voor de overflow property eens aan.

Margin, border, padding

Als je in de inspector een element selecteert zie je aan de rechterkant een kader met verschillende kleuren.



Deze kleuren duiden de margin, border en padding van het element aan. Samen bepalen ze hoeveel afstand er zal zijn tussen verschillende elementen.

- Margin: de afstand tussen het element en andere elementen
- Border: de rand van het element
- Padding: de afstand tussen de content van het element en de rand van het element

Een voorbeeld dat het gebruik van margin, border en padding illustreert kan je vinden op https://codepen.io/fdc/pen/WgQzBZ.

Margin

Je kan de marge aan de linkerkant, bovenkant, rechterkant en onderkant afzonderlijk instellen via margin-left, margin-top, margin-right en margin-bottom. Je kan ook de shorthand property margin gebruiken.

- 1 waarde: de margin is voor alle zijden gelijk
- 2 waarden: top/bottom en left/right
- 3 waarden: top, left/right en bottom
- 4 waarden: top, right, bottom, left (volgt de wijzers van de klok)

Om margins in te stellen maak je meestal gebruik van rem als eenheid.

Border

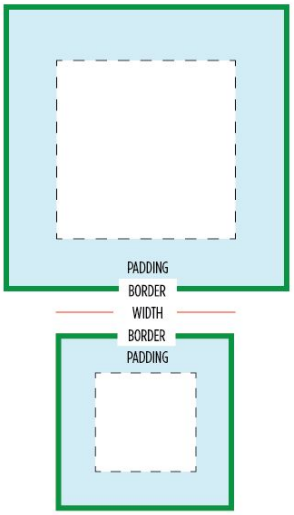
De property border is ook een shorthand property waarmee je verschillende eigenschappen van de border kan instellen. Deze zijn de border-width, border-style en border-color.

Padding

Net als bij de margin kan je voor de padding een shorthand property gebruiken. Deze werkt op dezelfde manier als bij de margin.

Box-sizing

De box-sizing property wordt gebruikt om aan de browser aan te geven wat er inbegrepen is in de width en height property van een block. Standaard staat deze waarde op content-box wat betekent dat de properties width en height enkel over de content gaan. De padding en border zijn niet inbegrepen. Dit kan voor onverwachte resultaten zorgen en is moeilijk om mee te designen.



Daarom stel je best deze property in op border-box voor alle elementen door gebruik te maken van de universal selector en de root tag.

```
*{
    box-sizing: border-box;
}

html{
    box-sizing: inherit;
}
```

Maak er - net als het instellen van de font-size - een goede gewoonte van om dit bij elk document te doen.

Content centreren binnen een vaste breedte

Een goede website zorgt er voor dat alle content binnen een vaste breedte valt. Meestal wordt gekozen voor een breedte van 960 pixels omdat alle content dan nog goed leesbaar is op een resolutie van 1024 pixels en 960 is een getal dat deelbaar is door veel getallen zodat je een evenwichtig design kan maken.

Veel websites centreren de content binnen 960 pixels. Dit wordt gedaan door de margin aan de linkerzijde en rechterzijde op auto te plaatsen . Volgend voorbeeld illustreert deze techniek die je veel zal gebruiken: https://codepen.io/fdc/pen/XPmYYM.

Display

Met de property display kan je een block element omzetten naar een inline element en vice versa. Dit kan handig zijn om verschillende block elementen naast elkaar te plaatsen, bijvoorbeeld om een horizontaal menu te maken: https://codepen.io/fdc/pen/oPjMgP.
OPMERKING: In het volgende hoofdstuk zie je dat het beter is om hier flexbox voor te gebruiken.

Je kan de property display ook op none plaatsen. Dit zal ervoor zorgen dat het element niet meer zichtbaar is en geen ruimte meer inneemt op het scherm. Indien je de ruimte toch wil behouden kan je gebruik maken van de property visibility.

CSS reset

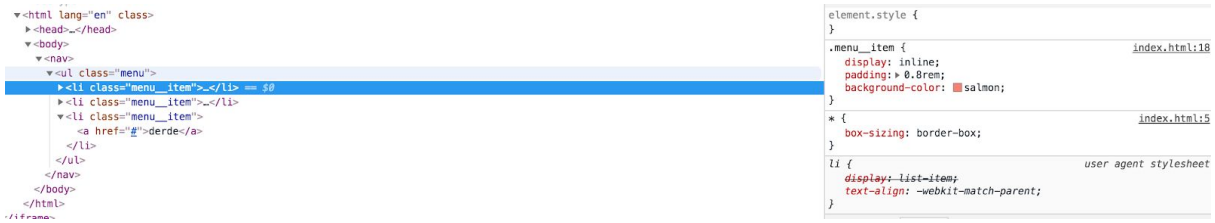
Elke browser heeft zo zijn eigen standaarden wat betreft margins, padding, border,... Dit is in feite niet wenselijk aangezien je website er altijd hetzelfde moet uitzien bij verschillende browsers. Daarom wordt er gebruik gemaakt van een reset.css bestand. Dit bestand zal de standaardwaarden van de browser resetten zodat je met een propere lei start. Je kan het reset bestand downloaden op https://meyerweb.com/eric/tools/css/reset/ en voor elk project opnieuw gebruiken.

In de HTML file link je naar dit bestand. Zorg wel dat je dit bestand aanspreekt voor je eigen style.css bestand. Anders worden al je stijlregels overschreven!

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="css/reset.css">
    <link rel="stylesheet" href="css/style.css">
    <title>Reset demo</title>
</head>
```

Debugging

CSS schrijven is - zeker in het begin - niet altijd even eenvoudig. Soms zal je onverwachte resultaten krijgen en niet weten hoe dit komt. Maak je daar geen zorgen in, zelfs de meest ervaren web developer heeft dit soms voor.
Belangrijk is echter om te weten hoe dit onverwachte resultaat tot stand gekomen is en hoe je het kan oplossen. De inspector van de browser is daar een heel handige tool voor.



Als je de inspector opent en een element selecteert zie je aan de rechterkant alle CSS properties die op het element ingesteld zijn en welke er overschreven werden. Zo kan je direct zien hoe een waarde tot stand gekomen is.

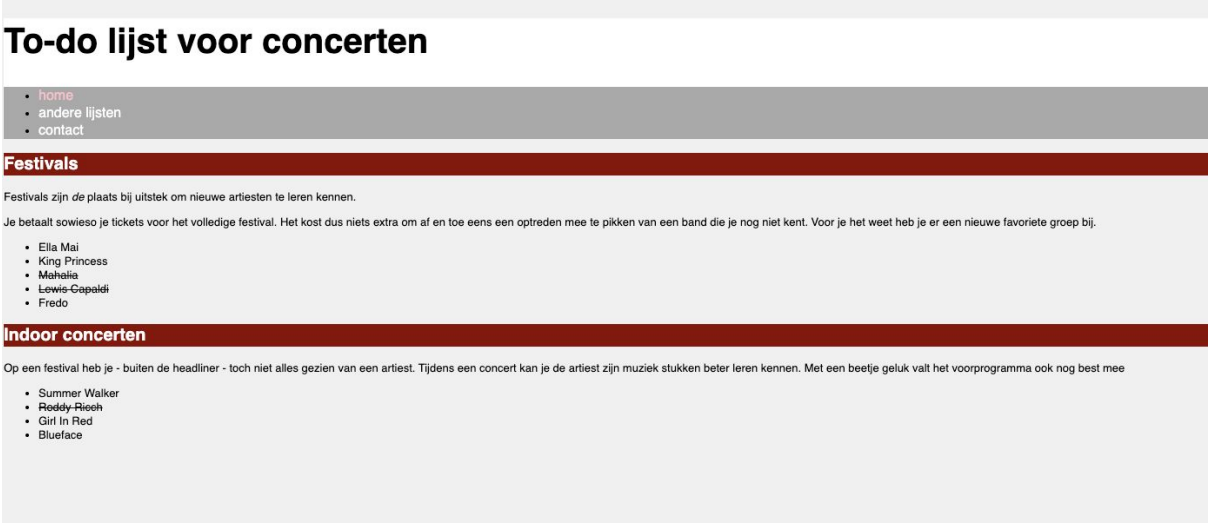
Er is echter meer. Je kan elke property ook aan- en uitvinken en je kan ze ook aanpassen. Op die manier kan je je CSS snel en efficiënt debuggen. Let wel: als je de pagina refresht zijn alle wijzigingen verdwenen.

Het is een goede gewoonte om je inspector tijdens het developen altijd open te laten staan.

Oefeningen 4: CSS - de basis

Oefening 1: lijsten

Maak de pagina na op basis van de waarden die opgegeven zijn. In het HTML document zijn al de classes gegeven. Let hierbij op de BEM notatie die gebruikt wordt. Je hoeft ze enkel in het CSS document op te roepen. Gebruik ook de inspector om te zien wat er precies wordt toegepast.



Oefening 2: Lil' Kleine

Maak de pagina na op basis van onderstaande screenshot. De HTML structuur is gegeven. De kleuren en font-sizes die je dient te gebruiken staan bij het bronmateriaal. Je dient in het HTML document wel nog de nodige classes te voorzien. Denk hierbij aan de BEM notatie.

Toppotch

LIL' KLEINE'S WERELDVEROVERING

Lil' Kleine, artiestennaam van Jorik Scholten, (Amsterdam, 15 oktober 1994) is een Nederlands muzikant en acteur. Met Danny de Munk bereikte hij de Top 100 met het nummer ZO VERDOMD ALLEEN. Samen met Ronnie Flex kwam hij met het nummer DRANK & DRUGS in 2015 op nummer 1 in de Top 100 en verdiende hij een dubbelplatina plaat. Verder is hij lid van de formatie Giocatori en werd hij in 2014 presentator van het radioprogramma Hip Hop NL. Scholten maakte in 2005 zijn acteerdebuut, toen hij een klein rolletje had in de film Diep. Hierna had hij gastrollen in enkele televisieseries en speelde hij Franky in de speelfilm Prins. Voor meer info zie Wikipedia

LEVENSLOOP

Lil Kleine's jeugd

Een jaar vol ontdekking en verkenning.
Lil' Kleine groeide op in Amsterdam en begon zijn carriere op TV. Hij werd gefeatured in:

- Het kinderprogramma PRAATJESMAKERS
- De film DIEP
- De politieserie VAN SPEIJK (2007)

Verdere carriere

Niet alleen was zijn TV-CARRIERE een succes, ook als ARTIEST scoort hij. Samen met andere rappers verbleef hij op het Nederlandse eiland Schiermonnikoog om zijn nieuwe album te produceren.

Het album NEW WAVE features de welbekende hits DRANK & DRUGS en ZEG DAT NIET.

Internationale bekendheid

Onze favoriete rapper blijft niet alleen in NL

- Het nieuwe album ALLEEN scoort in Vlaanderen
- De hits van Lil Kleine worden vertaald in het CHINEES.
- Het nummer Zeg Dat Niet wordt gebruikt in de trailer van de nieuwe DIE HARD moviefranchise.

In juli 2017 opende hij in het centrum van Amsterdam zijn eigen frietzaak. Hij wordt naast zijn goede vriend Daniel Lam mede-eigenaar van de zaak, die De Beig heet.

LIL' KLEINE - CV

- Film
  - Praatjesmakers
  - Diep
  - Van Speijk
  - TBA: Die Hard
- Muziek
  - Alleen
  - Patsergedrag
- Andere skills:
  - Uitstekende kok
  - Aansteller
  - Vader

INTERVIEW MET LIL'

Lil' Kleine / Artiest

Je echte naam is Jorik Scholten. Waarom koos je Lil Kleine als artiestennaam?

Wel, ik werd altijd 'Kleine' genoemd bij ons in de buurt in Amsterdam. Op een gegeven moment kwam er iemand die voor elke naam 'Lil' zette. Hij noemde me dan ook voor de eerste keer 'Lil Kleine'. Ik vond dat eigenlijk wel grappig en besloot dat als artiestennaam te gebruiken.

Hoe was het voor jou om in Amsterdam op te groeien?

Fijn! Nu ik wat ouder ben, ben ik heel blij dat ik daar ben opgegroeid. Ik heb daar echt veel geleerd.'

Wat heb je dan zoal geleerd daar?

Een mentaliteit, hoe je met mensen moet omgaan, hoe de wereld in elkaar zit, hoe mooi de wereld kan zijn, hoe hard de wereld kan zijn, enzoverder. Ik leerde ook met verschillende culturen om te gaan. Voor al die dingen ben ik heel erg dankbaar.'

Waar komt je passie om te rappen vandaan?

Ik heb meer een passie voor muziek, dan echt voor het rappen op zich. Ik kwam er snel achter dat ik niet goed was in 'instrumenten bespelen' en ook niet heel goed kan zingen. Toen kwam ik uit bij rappen en dan dacht ik 'dan zal ik dat maar gaan doen'.

CONTACT

Adres:
Budafabriek, Dam 2A - Kortrijk
E-mail:
lil.kleine@howest.be
Tel:
+31 6 83547209

Oefening 2 bis: Li'l Kleine - reset

Voeg het bestand reset.css toe aan je project en link het aan je html bestand. Bekijk het resultaat. Je zal merken dat alle standaard opmaak verdwenen is. Ook het automatisch inspringen van lijsten. Vanaf

nu maak je in elke oefening gebruik van de reset.css. Je zal hier in de toekomst dus moeten op anticiperen.

Oefening 3: Tupac

In deze oefening hoef je niets aan te passen. Alle code is gegeven. Bekijk aandachtig volgende zaken aangezien ze veel zullen voorkomen in de volgende oefeningen:

- Het gebruik van reset.css
- Het instellen van de property box-sizing
- Het instellen van de font-size op 62.5%
- Het centreren van een container met een breedte van 960px
- Het toepassen van **image replacement**. Dit betekent dat de tekst binnen het <h1> element verborgen wordt en vervangen wordt door een header beeld.

Open de pagina in de browser en bekijk deze met de inspector. Je kan de verschillende properties die ingesteld werden telkens bekijken en eventueel aanpassen.

Oefening 4: Colors

I am a nightmare **WALKIN'**, psychopath **TALKIN'**
King of my jungle, just a gangster **STALKIN'**
Livin' life like a firecracker, quick is my fuse
Vendettas of death back; the **COLORS** I choose
RED or **BLUE**, Cuz or Blood, it just don't matter
Sucka dive for your life when my shotgun scatters
The gangs of L.A. will never die - just multiply

Maak de pagina op basis van bovenstaande screenshot. Je kan je voor een groot deel baseren op de code van de vorige oefening.

- Bekijk de HTML, denk na over de namen van je classes
- Stel de box-sizing en font-size in
- Gebruik image replacement voor het header beeld
- Centreer de container met een breedte van 960 pixels
- Zorg voor opmaak in de tekst:
    - pas de lettergrootte en lijnhoogte aan
    - zorg dat sommige woorden een accent krijgen: in hoofdletters, in het vet, extra padding en andere kleuren
- Denk na hoe je de CSS zo optimaal mogelijk kan structureren

Oefening 5: CYA

In deze oefening is de volledige html structuur gegeven. Ook het class attribuut is telkens al ingesteld.
Schrijf de code voor style.css zodat je de opmaak van onderstaande screenshot bekomt.
De fonts en kleuren zijn niet opgegeven voor deze oefening. Probeer deze zelf eens in te schatten.

HLN.BE

NIEUWS

REGIO

SPORT

SHOWBIZZ

Bieber kan weer een aanklacht wegstrepen

Justin Bieber © photo news.

Justin Bieber kan weer een aanklacht van zijn lijstje strepen. De Canadese zanger wordt niet vervolgd voor een uithaal naar een fan ...

De zanger, die regelmatig met justitie in aanraking komt, werd woedend toen de man hem met zijn gsm filmde tijdens een etentje in Hollywood. Hij zou hebben uitgehaald naar de fan en geprobeerd hebben diens telefoon af te pakken. Alles werd op camera vastgelegd, maar dat lijkt nu zijn redding.

Nadat de politie de beelden heeft bekeken, is besloten de zaak te laten rusten. Op de beelden is namelijk te zien dat het tieneridool inderdaad uithaalt maar de fan nooit aan heeft geraakt. En dat heeft hij waarschijnlijk te danken aan zijn bodyguards, meldt TMZ. Zij wisten de 'Baby'-zanger ervan te weerhouden de fan te lijf te gaan.