

# Webdesign: HTML en CSS

Danny Devriendt

Sint-Jozefsinstituut Brugge - 2024-2025



# INHOUD

<b>1</b>	<b>INLEIDING</b>	<b>6</b>
<b>2</b>	<b>DE HEILIGE DRIEVULDIGHEID</b>	<b>7</b>
2.1	Inhoud en HTML	7
2.2	Structuur en opmaak via CSS	7
2.3	Javascript en JQuery	8
2.4	Surfen op het internet	8
<b>3</b>	<b>HULPMIDDELEN BIJ HET WEBSITE-BOUWEN</b>	<b>10</b>
3.1	Kant-en-klaar succes	10
3.2	Editor	10
3.3	Professionele software: Dreamweaver	10
3.4	CSS preprocessors	10
3.5	Grids en frameworks	10
3.6	CMS-systemen	11
3.7	Visual Studio Code: onze keuze	11
3.8	Hulpbronnen	12
<b>4</b>	<b>HTML5</b>	<b>13</b>
4.1	HTML tags	13
4.2	De opbouw van een HTML-pagina	14
4.3	De hiërarchie binnen HTML	15
4.4	HTML-pagina's opbouwen	15
4.5	Commentaar in je HTML-code toevoegen	16
<b>5</b>	<b>TEKST INTYPEN EN WERGEVEN</b>	<b>17</b>
5.1	Koppen	17
5.2	Opsommingen en lijsten	17
5.3	Hyperlinks	18
5.3.1	Links tussen pagina's in een website	18
5.3.2	Hyperlinks binnen een pagina	18
5.3.3	Hyperlinks naar andere websites	19
5.3.4	Hyperlink naar een e-mailadres	19
5.3.5	Weergave van de opbouw/opmaak van een webpagina	19
<b>6</b>	<b>OPMAAK MET CSS</b>	<b>20</b>
6.1	Scheiding tussen inhoud en opmaak	20
6.2	Verschillende niveaus CSS	20
6.3	Basisopmaak: lettertype en kleurgebruik	21
6.3.1	Lettertypes	21
6.3.2	Corpsgrootte: absolute en relatieve groottes	21
6.3.3	Kleurgebruik	22

6.4	Een eerste eenvoudig extern CSS-bestand	23
6.4.1	Links tussen een webpagina en een extern CSS-bestand	23
6.4.2	Over CSS-regels, selectors, declarations, properties en values	23
6.5	Het CSS BOX-model	25
6.5.1	CSS border	25
6.5.2	CSS margin	26
6.5.3	CSS padding	26
6.5.4	CSS border-radius	27
6.5.5	CSS width en height	29
6.6	CSS classes en IDs	29
6.7	Pseudo-classes	31
6.7.1	Pseudo-classes bij hyperlinks (A-tag)	31
6.7.2	Eerste/laatste element uit een reeks	31
6.7.3	Pseudo-elementen	31
6.8	Samengestelde selectoren	31
<b>7</b>	<b>WERKEN MET AFBEELDINGEN</b>	<b>33</b>
7.1	Het formaat van de afbeelding	33
7.2	Bronbestand en hyperlinks	34
7.3	Afbeeldingen omranden	34
7.4	Afbeeldingen als achtergrond	35
7.5	Afgeronde afbeeldingen	35
<b>8</b>	<b>CONTAINERS EN HUN PLAATSING</b>	<b>36</b>
8.1	Positionering van elementen	38
8.1.1	Box-types en CSS-display	38
8.1.2	De flow van de elementen	39
8.1.3	Floating elementen	43
8.2	Bootjes: banner en info	45
8.3	Bootjes: menu	45
8.4	Bootjes: content	47
8.5	Bootjes: footer	47
8.6	Menu's en submenu's	48
8.6.1	Verticaal hoofdmenu	48
8.6.2	Horizontaal hoofdmenu	49
<b>9</b>	<b>CODE INSPECTIE MET GOOGLE CHROME</b>	<b>51</b>
<b>10</b>	<b>FORMULIEREN</b>	<b>53</b>
10.1	Een formulier toevoegen	53
10.2	Basiselementen in een formulier	53
10.3	Formulier opmaken	55
10.4	HTML5 validatie	55
<b>11</b>	<b>RESPONSIVE WEBDESIGN</b>	<b>57</b>
11.1	Viewport	57
11.2	Mediaqueries	57
11.3	Grids	59

<b>12</b>	<b>FLEXBOX</b>	<b>60</b>
12.1	Eigenschappen op container-niveau	60
12.2	Eigenschappen op item-niveau	63
<b>13</b>	<b>CSS GRID</b>	<b>65</b>
13.1	CSS Grid container eigenschappen	65
13.1.1	Verdeling in kolommen en rijen	66
13.1.2	Uitlijning in de cellen	67
13.1.3	Uitlijning binnen de container	67
<b>14</b>	<b>BOOTSTRAP</b>	<b>69</b>
14.1	Het bootstrap grid-systeem	69
14.2	Extra ruimte tussen kolommen	72
14.3	Uitlijning	72
14.4	Volgorde van de elementen	73
14.5	Bootstrap typografie	73
14.6	Tabellen	73
14.7	Bootstrap en illustraties	74
14.8	Alerts	74
14.9	Bootstrap knoppen	75
14.10	Bootstrap icons	76
14.11	Bootstrap CSS variables	76
14.12	Uitklapbare tekst	76
14.13	Accordion	77
14.14	Carrousel	77
<b>15</b>	<b>INLEIDING OP PHP</b>	<b>79</b>
15.1	XAMPP installeren	80
15.2	Servertest	80
15.3	PHP-code in je webpagina opnemen	81
15.4	Include()	81
<b>16</b>	<b>OVERZICHT BELANGRIJKE HTML-ELEMENTEN</b>	<b>83</b>
<b>17</b>	<b>OVERZICHT BELANGRIJKE CSS-EIGENSCHAPPEN</b>	<b>85</b>
<b>18</b>	<b>BRONNEN</b>	<b>89</b>

# 1 INLEIDING

Voor u ligt de cursus **Inleiding tot Webdesign**. De opzet van deze cursus is het aanreiken van een aantal basisbouwstenen voor het uitbouwen van een eenvoudige website.

*Het maken van websites is een relatief jonge discipline waarin de laatste jaren een geweldig boeiende evolutie aan de gang is en afgewogen gekozen moet worden tussen verschillende mogelijkheden qua opbouw en beheer. In die zin past een papieren cursus niet zo best in een wereld die zo snel evolueert. We verwijzen daarom regelmatig naar externe digitale bronnen en benadrukken de beperkte houdbaarheid van een belangrijk gedeelte van de inhoud van de cursus.*

*Naast deze snelle evolutie is er evenwel een belangrijke basis die in de afgelopen jaren, decennia zelfs, quasi onveranderd is gebleven. Dat is ons uitgangspunt. De keuzes die we bovenop die basis maken, zijn keuzes die vanuit een bepaalde invalshoek zijn genomen, maar waarvan achteraf kan blijken dat ze niet de goede keuzes waren of snel achterhaald zijn omdat ze vervangen worden door steeds nieuwe en betere tools of frameworks. Flexibiliteit en gedrevenheid tot mee-evolueren zijn key-competenties in de wereld van het webdesign waar heel veel relatief is.*

*De uitdaging van de recentste jaren is ongetwijfeld een antwoord bieden op de correcte weergave van websites op verschillende devices. Naast gewone pc's worden websites steeds vaker op laptops, tablets, smart phones en andere devices in tal van formaten bekeken. Ervoor zorgen dat een website op die verschillende devices mooi en aangepast weergegeven wordt, is een belangrijke uitdaging die in het vakjargon **responsive design** genoemd wordt. We evolueren overigens sterk in de richting waarin mobile devices heel belangrijk worden: **mobile first design**.*

*Met deze insteek hopen we je een gezonde basis aan te bieden zodat je in staat bent om zelf doordacht een eenvoudige website op te zetten.*

*Alvast heel veel succes!*

*Danny Devriendt*

## 2 DE HEILIGE DRIEVULDIGHEID

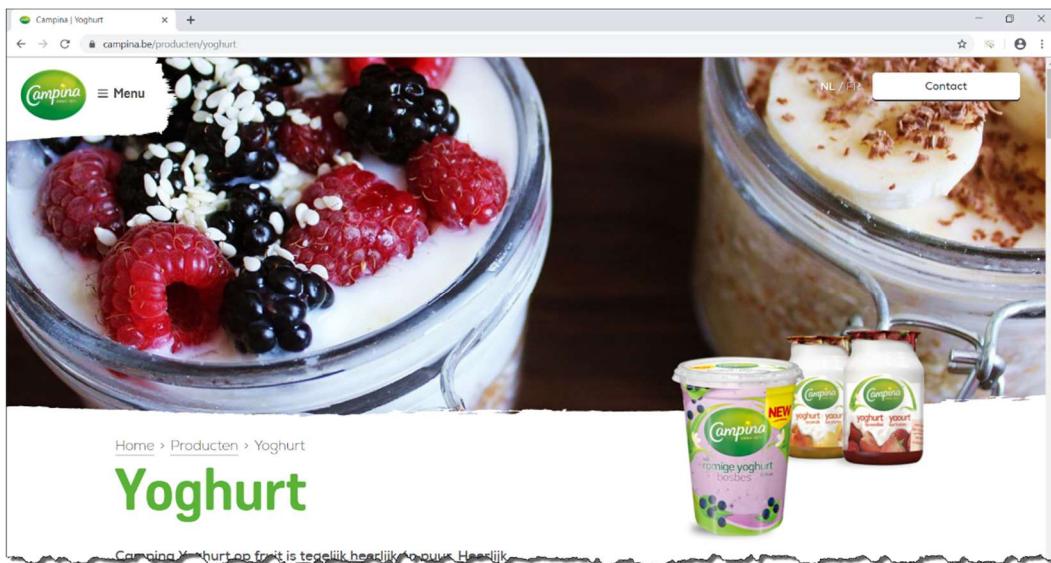
“Alle goede dingen bestaan uit drie” wordt wel eens gezegd en dat gaat mits enige vereenvoudiging ook voor webdesign: inhoud, opmaak en structuur.

Of een andere drievuldigheid: HTML, CSS en Javascript.

### 2.1 Inhoud en HTML

Een webpagina is eigenlijk een gewone tekstpagina die naast inhoud ook heel wat codes bevat die bepalen hoe de webpagina weergegeven wordt. De basistaal waarin een webpagina gecodeerd wordt, is **HTML** of *HyperText Markup Language*. Je browser interpreteert de HTML-code en zorgt ervoor dat je het eindresultaat op je scherm te zien krijgt.

HTML vormt de basis van elke webpagina en is altijd aanwezig. Hoe de webpagina effectief weergegeven wordt, hangt van de tweede speler af: **CSS**.



Meer en meer wordt inhoud niet zomaar ingetypt in webpagina's, maar uit databanken gehaald. Raadpleeg je bv. de programma's op een tv-station, de beschikbare treinverbindingen op een bepaald moment of boek je tickets voor een event: in al deze gevallen draait achter de website een **databank** die informatie aanlevert, verwerkt en stockeert. Hier is bijkomende software nodig, bv. **PHP** (programmeertaal) en **MySQL** (databank).

### 2.2 Structuur en opmaak via CSS

Aan HTML-webpagina's wordt **CSS**-code toegevoegd. CSS staat voor *Cascading Style Sheets* en bepaalt heel sterk hoe informatie in een webpagina weergegeven wordt. HTML en CSS zijn twee handen op één buik en werken naadloos samen.

Momenteel is **CSS3** de standaard.

## 2.3 Javascript en JQuery

Waar HTML en CSS statisch zijn, wordt vaak javascript gebruikt voor animaties, transities, effecten..., bv. een content rotator die je op vele websites ziet.

Veelal wordt de javascript bibliotheek **JQuery** gebruikt waarmee je heel gemakkelijk delen van je website kunt animeren. Heel veel voorbeelden kan je rechtstreeks van het Internet downloaden en aanpassen. JQuery werkt perfect samen met HTML en CSS.

We focussen in deze cursus niet verder op het gebruik van javascript/JQuery.

## 2.4 Surfen op het internet

Hoewel elk zichzelf respecterend bedrijf of organisatie vandaag een website heeft, is de opkomst van het Internet met het World Wide Web als belangrijk onderdeel met moeite een kwarteeuw jaar oud.

We nemen aan dat je vertrouwd bent met het surfen op het Internet. Maar wat gebeurt er nu als je bijvoorbeeld naar **www.delijn.be** surft?

- 1 Het webadres **www.DELIJN.BE** wordt op één van de DNS-servers (*Domain Name Server*) op het Internet omgezet in een IP-adres, bv. 108.92.11.26, waarmee de juiste **webserver** bereikt wordt waarop de site aanwezig is. Zo'n webserver is eigenlijk een krachtige computer die 24 op 24 uur via het Internet bereikbaar is en waarop specifieke software geïnstalleerd is. Op het moment dat je surft, ben jij de **client**.
- 2 Aan elke website is op de webserver een hoofdmap gekoppeld en een standaard startpagina. In ons voorbeeld is dat INDEX.HTML. Je kon dus ook **WWW.DELIJN.BE/INDEX.HTML** intypen.
- 3 De pagina INDEX.HTML wordt van de webserver naar je eigen computer gestuurd, *gedownload* via het TCP/IP-protocol voor het transport én het HTTP-protocol voor de communicatie. Je **browser** die de HTML-code uit de webpagina interpreteert, tovert een fraaie pagina op je scherm.



De site die je gerust op je eigen PC kunt maken, moet je dus uiteindelijk op een webserver plaatsen, wil je die voor het grote publiek toegankelijk maken.

Hiervoor heb je externe hulp nodig en moet je een plaatsje op een webserver huren via een *hosting* bedrijf. Een voorbeeld van zo'n bedrijf is **one.com** ([www.one.com](http://www.one.com)) waar je eenvoudig en snel online webruimte kunt huren en een domeinnaam kunt registreren. Bekijk aanbod en prijzen.

Aantrekkelijk				
Guru <small>Pro</small>	Expert <small>Pro</small>	Enthusiast	Explorer	Beginner
Perfect voor bureaus	Perfect voor bedrijven	Perfect voor ondernemers	Perfect voor blogs	Perfect voor kleine projecten
€ 6,99 /mnd.*	€ 6,99 /mnd.*	€ 0,99 /mnd.*	€ 2,99 /mnd.*	€ 2,99 /mnd.*
<small>Verlengt voor € 22,99 /mnd. Jaarlijks gefactureerd</small>	<small>Verlengt voor € 17,99 /mnd. Jaarlijks gefactureerd</small>	<small>Verlengt voor € 14,99 /mnd. Jaarlijks gefactureerd</small>	<small>Verlengt voor € 9,99 /mnd. Jaarlijks gefactureerd</small>	<small>Verlengt voor € 5,99 /mnd. Jaarlijks gefactureerd</small>
<a href="#">In het winkelmandje</a>	<a href="#">In het winkelmandje</a>	<a href="#">In het winkelmandje</a>	<a href="#">In het winkelmandje</a>	<a href="#">In het winkelmandje</a>
<b>Belangrijkste kenmerken</b>	<b>Belangrijkste kenmerken</b>	<b>Belangrijkste kenmerken</b>	<b>Belangrijkste kenmerken</b>	<b>Belangrijkste kenmerken</b>
10 ondersteunde websites <small>750 GB SSD-opslag</small> Onbeperkt e-mailaccounts met premium-functies <small>Dagelijkse back-up &amp; restore</small>	7 ondersteunde websites <small>500 GB SSD-opslag</small> Onbeperkt e-mailaccounts met premium-functies <small>Dagelijkse back-up &amp; restore</small>	5 ondersteunde websites <small>200 GB SSD-opslag</small> Onbeperkt aantal e-mailaccounts <small>Dagelijkse back-up &amp; restore</small>	1 website ondersteund <small>100 GB SSD-opslag</small> Onbeperkt aantal e-mailaccounts <small>Dagelijkse back-up</small>	1 website ondersteund <small>50 GB SSD-opslag</small> Onbeperkt aantal e-mailaccounts <small>Dagelijkse back-up</small>
<small>Ook inbegrepen</small>				

Meer informatie over de registratie van .be domeinen vind je op [WWW.DNS.BE](http://WWW.DNS.BE). Voor internationale domeinnamen kan je onder meer op [WWW.WHOIS.ORG](http://WWW.WHOIS.ORG) terecht.

## 3 HULPMIDDELEN BIJ HET WEBSITE-BOUWEN

### 3.1 Kant-en-klaar succes

Wil je snel en zonder enige voorkennis een website in elkaar boksen, dan kan je via een kant-en-klaar platform werken. Je kiest een model/sjabloon, past de inhoud aan en je mooi ogende webpagina/website is klaar en onmiddellijk gratis online beschikbaar. Gratis geldt evenwel soms enkel voor een basisversie met beperkte mogelijkheden. Wil je extra's, dan betaal je ervoor.

Populaire voorbeelden zijn **weebly** ([www.weebly.com](http://www.weebly.com)) en **wix** (<https://nl.wix.com/>).

Hoe verleidelijk kant-en-klaar succes ook is, in deze cursus bouwen we *from scratch* een website op zodat we die later met kennis van zaken kunnen aanpassen en uitbreiden.

### 3.2 Editor

Hoger hoorde je al dat een webpagina eigenlijk een pure platte tekstpagina is. In principe volstaat dus de Windows kladblok om webpagina's te maken.

Op de markt zijn evenwel heel wat andere *editors* met hoger gebruikerscomfort. Eentje ervan is **Notepad++** (<https://notepad-plus-plus.org/>) dat je gratis kunt downloaden. De moeite waard om te installeren, maar wij kiezen **Visual Studio Code** (zie 3.7).

### 3.3 Professionele software: Dreamweaver

Wil je professioneler aan de slag, dan kan je software als **Adobe Dreamweaver** gebruiken. Naast een volwaardige editor bevat Dreamweaver heel wat gebruikersvriendelijke faciliteiten om je website uit te bouwen en te beheren. Dreamweaver is evenwel niet gratis. Er bestaan educatieve licenties. Alles wat aan bod komt, kan ook zonder Dreamweaver gerealiseerd worden, soms mits wat extra moeite.

### 3.4 CSS preprocessors

We gaan in deze cursus elke regel CSS zelf schrijven. Op de markt vind je een aantal tools waarmee je CSS-code voor vooral grotere projecten sneller en met veel groter onderhoudsgemak kunt genereren. Bekende voorbeelden zijn **Sass** en **Less**.

We maken in deze basiscursus geen gebruik van CSS preprocessors.

### 3.5 Grids en frameworks

Om webpagina's responsive te ontwerpen voor verschillende devices zijn heel wat hulpmiddelen beschikbaar voor het positioneren van informatie in functie van de gebruikte device. **Flexbox**, **CSS Grid** en **Bootstrap** zijn bekende voorbeelden. Ze komen verder in deze cursus aan bod.

We kiezen er bewust voor om in een eerste fase de webpagina's geheel zelf te ontwerpen, zonder gebruik van hulpmiddelen bij het positioneren.

### 3.6 CMS-systemen

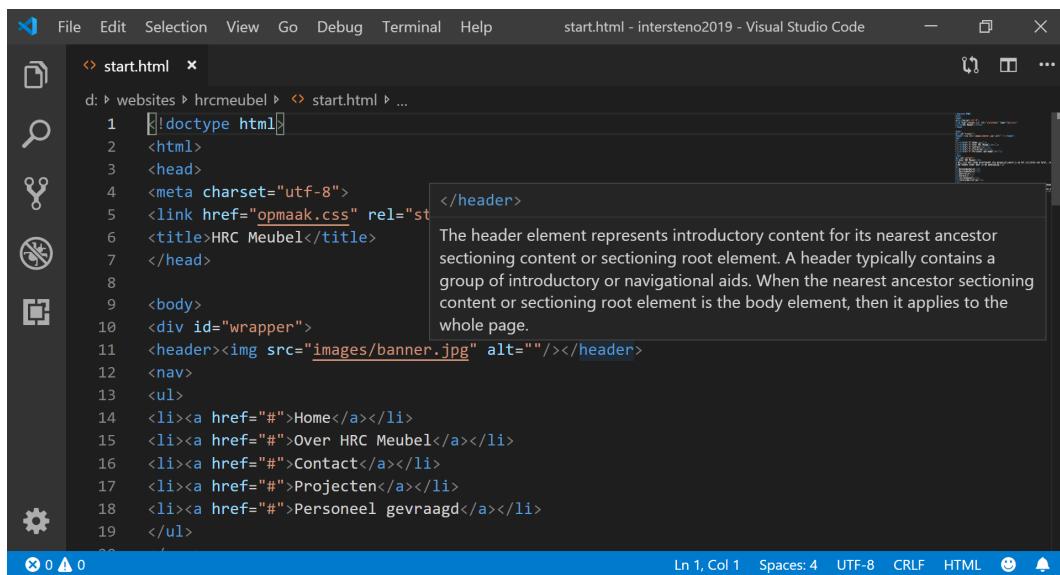
Vooral voor grotere websites die door verschillende personen beheerd worden zijn CMS-systemen beschikbaar. CMS staat voor *Content Management System*. Bekende voorbeelden zijn **Joomla**, **Drupal** en vooral **Wordpress**. In zo'n CMS systeem zijn tal van sjablonen en modules beschikbaar die je helpen om professioneel ogende websites op te zetten door een team. Zeker de moeite, maar tijd ontbreekt om hierop in te gaan.

### 3.7 Visual Studio Code: onze keuze

Uit de vele editors kiezen wij voor Visual Studio Code, gratis te downloaden via <https://code.visualstudio.com/>. Visual Studio Code ondersteunt vele programmeertalen en bevat ook support voor HTML en CSS. Er komen regelmatig extra extensies bij die je kunt activeren.

We voorzien geen cursus in het gebruik van Visual Studio Code. Gaandeweg leer je wel een en ander. Maar toch dit:

- Via **File > Open Folder** (Ctrl+K,Ctrl+O) kan je een (website)map openen en krijg je de inhoud van die map te zien.
- Met **Ctrl+spatie** roep je mogelijke suggesties bij een code/tag op.
- Tags worden automatisch afgesloten als je het >-teken van de openingstag intypt of als je de / in de sluitingstag intypt.
- Met **Ctrl+K,Ctrl+F** schakel je de formatting in of uit: code springt nu netjes in.
- Met **Shift+Alt+F** springt de code netjes hiërarchisch in.



```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<link href="opmaak.css" rel="st...
<title>HRC Meubel</title>
</head>
<body>
<div id="wrapper">
<header></header>
<nav>
<ul>
<li><a href="#">Home</a></li>
<li><a href="#">Over HRC Meubel</a></li>
<li><a href="#">Contact</a></li>
<li><a href="#">Projecten</a></li>
<li><a href="#">Personnel gevraagd</a></li>
</ul>
</div>
</body>
</html>
```

- Via **View > Command Palette (Ctrl+Shift+P)** krijg je een overzicht van allerlei handelingen. Typ gerust een trefwoord in om op te zoeken.
- Via **File > AutoSave** worden wijzigingen in je documenten automatisch opgeslagen.

Binnen Visual Studio Code kan je extra extensies installeren die je helpen bij je werk. Via het Extensions-pictogram ga je op zoek in de *Extensions Marketplace*.



Een handige extensie is de **HTML5 Boilerplate** waarmee je heel snel de basiscode in een nieuwe pagina HTML-pagina toevoegt. Sla wel eerst de pagina met extensie HTML op, zoniet werkt de extensie niet. Typ bovenaan de pagina gewoon **html** in en kies **html:5** uit het lijstje. Deze basiscode wordt nu automatisch aan de pagina toegevoegd:

```
<!DOCTYPE html>
<html lang="en">
<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>Document</title>
</head>
<body>

</body>
</html>
```

Als alternatief kan je bovenaan een leeg opgeslagen HTML-document een **uitroepTeken** (!)gevolgd door de **tab**-toets intypen. Ook dan wordt basiscode voor een HTML-pagina toegevoegd.

Typ je bovenaan de extensielijst **@installed** in, dan krijg je een overzicht van je geïnstalleerde extensies. Er zijn tal van interessante extensies, bv. Live Preview, Live Server.

Via **Manage > Themes > Color theme** kan je je werkomgeving aanpassen qua kleuren: achtergrondkleur, tekstkleur...



### 3.8 Hulpbronnen

Gelukkig vind je op het Internet over alle aangehaalde thema's uitgebreide informatie, voorbeelden, tutorials allerhande.

Heel vaak verwijzen we naar **w3schools** ([www.w3schools.com](http://www.w3schools.com)) die zeer bevattelijk uitlegt en voorbeelden geven in hun talrijke tutorials, o.a. over HTML, CSS, Bootstrap, JavaScript/JQuery en nog zoveel meer...

Ook genAI zoals **Chat GPT** kan je zeker helpen!

## 4 HTML5

HTML, *HyperText Markup Language*, is dus de taal die gebruikt wordt om webpagina's te coderen. HTML is een codetaal, geen programmeertaal en bepaalt de indeling en weergave van je webpagina's.

Het is in dit korte hoofdstuk helemaal niet de bedoeling een cursus HTML aan te bieden waarin alle HTML-codes aan bod komen, maar wel je een aantal basisprincipes van HTML uit de doeken te doen. In de volgende hoofdstukken verwijzen we dan heel regelmatig naar de achterliggende HTML-codes, **tags** genoemd.

Momenteel is HTML5 de meest actuele versie die door alle belangrijke browsers ondersteund wordt.

### 4.1 HTML tags

Een HTML-pagina – een standaard webpagina zeg maar – bevat naast de eigenlijke inhoud heel wat HTML-tags die de structuur en ook de weergave van het document bepalen. De extensie van een HTML-pagina is **.html** of **.htm**.

HTML-tags worden altijd tussen **punthaken** weergegeven en worden zo door de webbrowser herkend en geïnterpreteerd.

HTML-tags komen **meestal per paar** voor: een start-tag en een eind-tag. De eindtag bevat een **/**. Tussen beide tags staat de inhoud waarop die tags betrekking hebben.

```
<p>Professioneel <strong>webdesign</strong> met html en  
css.</p>
```

Het voorbeeld hierboven toont een paragraaf (**<p>**-tag) met daarin het woordje 'webdesign' dat vet weergegeven wordt (**<strong>**-tag). Het resultaat in de browser wordt:

Professioneel **webdesign** met html en css.

Bij enkele tags is geen eindtag voorzien. Ze eindigen op een slash.

```
html<br>css
```

Met de **<br>**-tag – *break* – neem je een nieuwe regel zonder een nieuwe paragraaf te beginnen. Het resultaat in de browser ziet er zo uit:

```
html  
css
```

Aan de meeste tags kan je, soms verplicht, één of meer **attributen** toevoegen. Attributen worden altijd aan de begintag toegevoegd. Bijvoorbeeld:

```

```

De `<img>`-tag voegt een figuur aan de webpagina toe en heeft geen eind-tag. Hier is het attribuut SRC (source) essentieel. De andere attributen geven bijkomende informatie. Als ze weggelaten worden, wordt de figuur in zijn originele afmetingen weergegeven.

Let op de syntaxis bij het gebruik van attributen:

- gelijkheidsteken en waarde die veelal tussen aanhalingstekens staat
- spaties tussen de verschillende attributen
- alles binnen de punthaken van de begin-tag.

HTML-tags en attributen worden normaal in kleine letters geplaatst.

Niet alle browsers interpreteren alle tags en hun attributen op dezelfde manier. Op [www.w3.org](http://www.w3.org), de website van het World Wide Web consortium, vind je alle officiële informatie over HTML.

## 4.2 De opbouw van een HTML-pagina

Een HTML-pagina bestaat uit twee grote delen:

- het **head**-gedeelte bevat allerlei informatie over het document, maar geen eigenlijke inhoud. Naarmate de cursus vordert, zullen we steeds meer nuttige informatie in dat head-gedeelte onderbrengen. HTML: `<head></head>`.
- het **body**-gedeelte bevat de eigenlijke inhoud van de pagina en dus van wat je in de browser te zien krijgt. HTML: `<body></body>`.

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Naamloos document</title>
</head>

<body>
</body>
</html>
```

De volledige HTML-pagina wordt tussen `<html></html>`-tags weergegeven. Alle inhoud komt tussen de `<body>` en `</body>` tags.

Je kunt gerust returns en tabs gebruiken om je HTML-pagina visueel overzichtelijk voor te stellen. Deze returns en tabs hebben geen effect op de weergave in de browser, aangezien het geen HTML-tags zijn.

Elke webpagina start evenwel met de declaratie `<!doctype html>`. Dit is geen echte HTML-tag maar een instructie voor de browser over de gebruikte HTML-versie. In HTML5 is die instructie altijd `<!doctype html>`.

Verder vind je nog informatie over de gebruikte tekencodering (UTF-8), van belang voor de correcte weergave van accentletters.

Met de `<title>`-tags geef je de titel van de webpagina op die bovenin je browser-tabblad verschijnt en ook van belang is voor het weergeven van zoekresultaten in

zoekmachines, het markeren van favorieten... Vervang dus *Naamloos document* door een passende titel.

### 4.3 De hiërarchie binnen HTML

Er is een belangrijke hiërarchische structuur binnen de tags. In principe zijn alle tags binnen de pagina ondergeschikt aan de `<body>`-tag en nemen ze ook eigenschappen van die tag over.

```
<body>
<p>De <strong>hiërarchie</strong> binnen HTML.</p>
</body>
```

In het voorbeeld hierboven is de `<p>`-tag een **child** van de `<body>`-tag en is de `<body>`-tag de **parent** van de `<p>`-tag. De `<strong>`-tag is op zijn beurt child van de `<p>`-tag.

De tekstkleur die je bv. op het niveau van de `<body>`-tag instelt, wordt normaal overgenomen door alle onderliggende tags. Je kunt natuurlijk in de child-tag uitdrukkelijk afwijken van de opmaak van de parent-tag door voor die child-tag een specifieke kleur op te geven. Hoe dat moet, leer je verder.

### 4.4 HTML-pagina's opbouwen

HTML5 biedt een aantal structuurtags die je rechtstreeks kunt gebruiken om bepaalde informatie in een webpagina aan te duiden: `<article>`, `<aside>`, `<details>`, `<summary>`, `<main>`, `<header>`, `<footer>`, `<hgroup>`, `<mark>`, `<nav>`, `<meter>`, `<progress>`, `<section>`, `<time>`...

Zo begrijp je zeker dat informatie tussen `<header></header>` tags wellicht bedoeld is om bovenaan de webpagina weer te geven.

Algemeen kan je zoveel informatieblokken – vaak **containers** genoemd – als je wilt in je webpagina opnemen door `<div></div>` tags te gebruiken binnen de `<body>`-tags. Geef elke `<div>`-tag een ID-mee zodat je die later nog kunt aanspreken.

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Naamloos document</title>
</head>

<body>
<div id="wrapper">
    <header>hier komt de header-informatie</header>
    <nav>hier komt het menu</nav>
    <main>hier komt de eigenlijke inhoud</main>
    <aside>hier komt de inhoud van een zijbalk</aside>
    <footer>hier komt de inhoud van de voettekst</footer>
</div>
</body>
</html>
```

In het voorbeeld hiervoor worden zowel specifieke HTML5-tags als algemene DIV-tags gebruikt.

In het hoofdstuk *Containers en hun plaatsing* bespreken we de opbouw van een webpagina meer in detail.

#### 4.5 Commentaar in je HTML-code toevoegen

Wil je commentaar aan HTML-code toevoegen, dan plaats je die tussen `<!--` en `-->` afbakening.

```
...
<body>
<div id="wrapper">
<!-- de wrapper omsluit de gehele inhoud van de webpagina
-->
...
...
```

## 5 TEKST INTYPEN EN WEERGEVEN

### HTML

<p></p>	begrenst een paragraaf in een webpagina
---------	---

Als je een begin tag intypt, wordt automatisch de eind tag toegevoegd en kan je tussen beide tags inhoud intypen.

<p></p>
---------

### HTML

 	Break. Nieuwe regel zonder nieuwe paragraaf
------	---

### 5.1 Koppen

In een webpagina komen natuurlijk ook koppen op verschillende niveaus voor. In plaats van de `<p>` tags, gebruik je hier de tags `<h1>` tot `<h6>` om alinea's tot op zes niveaus als titel te markeren.

### HTML

<hx></hx>	markeert een titel op het x-niveau (van 1 tot 6)
-----------	--

### 5.2 Opsommingen en lijsten

Heel vaak vind je in webpagina's lijsten met opsommingen terug. In HTML onderscheiden we vooral:

- niet-genummerde lijsten (*unordered lists*): opsommingen die met bullets weergegeven worden
- genummerde lijsten (*ordered lists*): genummerde opsommingen.

Zowel bij het intypen als achteraf kan je van een reeks items die in afzonderlijke alinea's voorkomen, een lijst maken.

Let voorlopig nog niet verder op de opmaak van de lijsten en onthoud dat we vaker dan je denkt van opsommingen gebruik zullen maken.

### HTML

<ul></ul>	bakent een volledige opsommingslijst met bullets af
-----------	---

<ol></ol>	bakent een volledige genummerde opsomming af
-----------	--

<li></li>	markeert elk individueel opsommingsitem in de lijst
-----------	---

```

<p>Het personeel is als volgt ingedeeld:</p>
<ul>
    <li>Business-to-consumer: 328</li>
    <li>OmegaSoft: 206</li>
    <li>Omega Medical: 213</li>
    <li>Omega Dental: 444</li>
</ul>

```

Bekijk de hiërarchie in het voorbeeld hierboven: de `<li>`-tags zijn children van de hogere `<ul>`-tag. Dit is heel belangrijk wat betreft het verder toekennen van opmaak!

## 5.3 Hyperlinks

Het gebruik van **hyperlinks** of **links** is essentieel: via een klik op een woord, tekstdeel of illustratie spring je naar een andere plaats in de pagina, naar een andere webpagina binnen of buiten de site. Dit van de ene plaats naar een andere plaats 'verspringen' is alom bekend onder de naam **surfen** op het Internet.

### 5.3.1 Links tussen pagina's in een website

Om een link naar een andere pagina in je site te leggen, voeg je volgende tags aan de tekst of illustratie waarop de gebruiker moet klikken toe om de link te activeren.

#### HTML

<code>&lt;a&gt;&lt;/a&gt;</code>	bakent de tekst/illustratie af waarop geklikt kan worden
<code>HREF</code>	bevat de informatie waarnaar de link gelegd wordt
<code>TARGET</code>	geeft aan waar de hyperlink moet geopend worden

```

<p>De <a href="aardbei.html">aardbei</a> is één van de meest gesmaakte fruitsoorten bij kinderen.</p>

```

De geselecteerde tekst wordt standaard blauw en onderstreept weergegeven. Let voorlopig niet op die opmaak.

De is child van de `<p>`-tag in het voorbeeld hierboven.

Binnen de `<a>`-tag kan je met het **target** attribuut opgeven waar het gelinkte document geopend moet worden, bv.:

- **\_blank**: in een nieuw venster/tabblad, bv.  
`<a href="bestel.html" target="_blank">`
- **\_top**: in het volledige actieve venster, bv.  
`<a href="historiek.html" target="_top">`.

### 5.3.2 Hyperlinks binnen een pagina

Lange webpagina's zijn niet zo gebruikersvriendelijk, maar als ze toch voorkomen kan je ook binnen zo'n webpagina hyperlinks gebruiken om naar een bepaalde plaats in die webpagina te springen.

Vooraf moet je op de plaats(en) waar je een sprong naartoe wilt voorzien een markering aanbrengen, een *named anchor*. Gebruik de `<a>`-tag met het **id** attribuut.

## HTML

< a > / a > plaats een markering in een webpagina

ID geef een unieke naam aan de markering

```
< h2 > < a id="bebat" > </ a > Batterijen recycleren < / h2 >
```

Named anchor

```
< p > < a href="#bebat" > Link naar Batterijen recycleren < / a > < / p >
```

Link naar named anchor

Verwijs naar zo'n named anchor door vóór de naam van de id een #-teken te plaatsen:

```
< p > < a href="recycleren.html#bebat" > Link naar Batterijen  
recycleren < / a > < / p >
```

Link naar named anchor in andere webpagina 'RECYCLEREN.HTML'.

Named anchors zijn uniek en hoofdlettergevoelig.

### 5.3.3 Hyperlinks naar andere websites

Natuurlijk kan je ook een link naar een andere website voorzien. Dat gebeurt net zoals je een link binnen je site legt, maar als link geef je nu een andere website op, bv.

<HTTPS://WWW.ENERGIESPAREN.BE>.

```
< p > < a href="https://www.energiesparen.be" > Bespaar energie! < / a >  
< / p >
```

Vergeet niet het protocol (<https://>) in de link op te nemen!

Je kunt binnen een andere website al direct naar een bepaalde pagina linken, bv.

[HTTPS://WWW.ELECTRABEL.BE/HOME PAGE/CAREER/INDEX\\_NL.HTML](HTTPS://WWW.ELECTRABEL.BE/HOME PAGE/CAREER/INDEX_NL.HTML).

### 5.3.4 Hyperlink naar een e-mailadres

Heel frequent worden e-maillinks in websites gebruikt. Bij het klikken op een dergelijke link wordt je e-mailprogramma geopend en kan je direct een bericht intypen waarin automatisch het opgegeven e-mailadres als geadresseerde is opgenomen.

Je gebruikt het href attribuut binnen de [a](#)-tag en voegt vóór het e-mailadres **mailto:** toe (geen spatie na het :-teken).

```
< p > < a href="mailto:an.baes@abw.be" > Mail voor meer info! < / a > < / p >
```

### 5.3.5 Weergave van de opbouw/opmaak van een webpagina

Hoe je opbouw/opmaak van je webpagina eruit ziet, bepaal je niet via HTML. Binnen HTML 'vertel' je gewoon welk soort tekst je weergeeft: een gewone paragraaf, een item uit een opsomming, een titel van een bepaald niveau, een hyperlink... Hoe die informatie uiteindelijk weergegeven wordt, geef je via CSS op.

## 6 OPMAAK MET CSS

### 6.1 Scheiding tussen inhoud en opmaak

Het begrip CSS is al enkele keren gevallen. CSS bepaalt hoe je webpagina eruit ziet via een aantal stijlregels die gekoppeld zijn aan HTML-elementen.

De opmaak die je via CSS instelt, wordt gescheiden van de inhoud in *stijlregels* opgeslagen zodat latere aanpassingen vlot en centraal kunnen uitgevoerd worden.

Waar je die opmaak terugvindt en hoe het allemaal werkt, leer je in dit belangrijke hoofdstuk.

### 6.2 Verschillende niveaus CSS

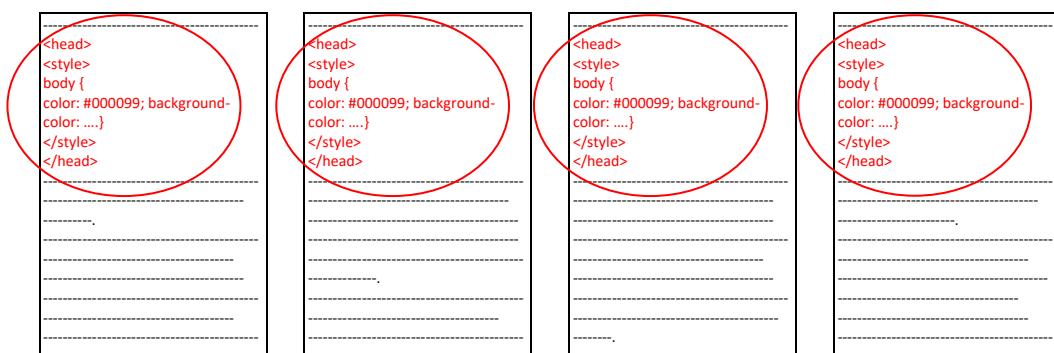
Je kunt CSS-opmaak op verschillende niveaus toevoegen:

- als **inline style**: op een heel specifieke plaats in een document om verfijnde opmaak in te stellen die met HTML niet mogelijk is. Deze mogelijkheid gebruiken we zelden.
- als **embedded style**: de CSS-regels worden in het HEAD-deel van je pagina opgeslagen en hebben enkel betrekking op deze pagina.

```
<head>
<style type="text/css">
body {
color: #009; background-color: ...
...
</style>
</head>
```

#### Embedded style sheet

In de webpagina wordt in het <HEAD>-gedeelte de volledige CSS-opmaak opgenomen. Die opmaak geldt telkens voor de betrokken pagina.



- als **externe stylesheet**: de CSS-opmaak wordt in een afzonderlijk bestand opgeslagen en heeft betrekking op alle pagina's die voor hun opmaak naar dat afzonderlijk bestand verwijzen.

Het spreekt vanzelf dat een externe stylesheet de meest krachtige oplossing is waarbij de opmaak in één centraal bestand bijgehouden wordt: snel, gemakkelijk aan te passen en gegarandeerd dezelfde opmaak in alle pagina's.



In pagina's die bewust van die standaardopmaak afwijken, kan je alsnog een (extra) embedded style toevoegen die dan enkel voor de betrokken pagina geldt of verwijzen naar een andere externe stylesheet die ook weer voor verschillende pagina's kan gebruikt worden.

Een combinatie van embedded style en externe stylesheet is ook mogelijk.

### 6.3 Basisopmaak: lettertype en kleurgebruik

#### 6.3.1 Lettertypes

De keuze van lettertypes is belangrijk en bepaalt in hoge mate de *look and feel* – het uitzicht – van de webpagina's.

Niet alle lettertypes zijn op elke computer beschikbaar. Om die reden kan je verschillende lettertypes opgeven, bv. Arial, Helvetica, *sans-serif*. Is op de computer van je bezoeker Arial aanwezig – standaard op elke PC maar misschien niet op Mac! –, dan wordt Arial gebruikt. Zoniet, wordt Helvetica gebruikt. Helvetica lijkt sterk op Arial en is normaal standaard op Mac-aanwezig. Is ook Helvetica niet beschikbaar, dan kiest je browser zelf een lettertype van het type *sans-serif*.

*Sans-serif* is de groep lettertypes zonder schreefjes aan de letteruiteinden. *Serif*-lettertypes hebben wel schreefjes aan de uiteinden. Bekijk de verschillen hieronder:

**Webdesign**

Arial (*sans-serif*)

**Webdesign**

Times New Roman (*serif*)

Zorg best ervoor dat er altijd een algemeen, generiek lettertype op het einde van je lijstje voorkomt, *serif* of *sans-serif* bijvoorbeeld. Elke browser heeft immers een *serif* en *sans-serif* lettertype aan boord.

#### 6.3.2 Corpsgrootte: absolute en relatieve groottes

In CSS kan je de **corpsgrootte** uitdrukken in allerlei eenheden: pixels, punten, inches, cm, mm, em, rem... We maken in eerste instantie een onderscheid tussen absolute en relatieve eenheden:

- pt (punt), mm, cm, in (inch)... zijn absolute eenheden. Ze hebben een vaste grootte die op elk moment dezelfde is.

- %, em, rem, vw (viewport width), vh (viewport height) zijn relatieve waarden die afhangen van andere waarden of instellingen.

Kies je voor weergave in **punten**, dan werk je met een vaste grootte: 12 pt is altijd 12 pt = 12/72 of 1/6<sup>e</sup> van een inch (2,54 cm). In je webpagina's is het dan moeilijk om aan te passen aan bv. de gebruikte device: van smart phone tot groot beeldscherm: alle informatie wordt met dezelfde lettergrootte weergegeven, wat minder interessant is.

Vaak wordt met relatieve groottes gewerkt in **%** of **em**. Daarbij wordt vertrokken van een grootte van 100% op de BODY-tag.

1em is dus de normale lettergrootte en komt overeen met 100% bij lettergrootte. Stel dat je de lettergrootte op 12pt instelt – geen goed idee trouwens! – dan komt 1em met die 12pt overeen.

Naast em wordt ook **rem** als grootte gebruikt: em is relatief t.o.v. het dichtste parent element, rem is enkel relatief t.o.v. de lettergrootte van het hoogste element, het HTML-element (root). Rem is root em. De standaard lettergrootte in de browser is normaal 16px.

Bij lettertypes hangt de relatieve grootte van het bovenliggend element af. Bij een kop h1 met een *font-size* van 1.5em of 150%, is het lettertype dus anderhalf keer zo groot als dat in het bovenliggende element, de BODY.

Stel je geen lettergrootte in de BODY in, dan bepaalt de browser van de gebruiker hoe groot tekst wordt weergegeven.

Je kunt de lettergrootte ook in beeldpunten of pixels (**px**) uitdrukken, maar dan mis je de voordelen van rem en em voor lettergroottes.

Opgelet: kies in het algemeen je corps **niet te groot**.

Door ook marges en (wit)ruimtes relatief in te stellen (in *em* bv.), past de witruimte zich ook relatief aan de lettergrootte aan. Die eenheden worden dus voor veel meer dan enkel de lettergrootte gebruikt.

Met de eenheden **vw** (viewport width) en **vh** (viewport height) werk je t.o.v. de viewport. 1vw bv. is 1 % van de viewport width, de breedte van je scherm in je browser.

### 6.3.3 Kleurgebruik

Basiskleuren op een beeldscherm zijn normaal rood, groen en blauw (RGB). Elk van deze drie kleuren heeft een waarde die varieert van 0 tot 255. Theoretisch zijn hier meer dan 16 miljoen combinaties mogelijk.

Kleurwaarden worden hexadecimaal uitgedrukt, voor elke kleur 2 waarden van 0 tot F (0 1 2 3 4 5 6 7 8 9 A B C D E F).

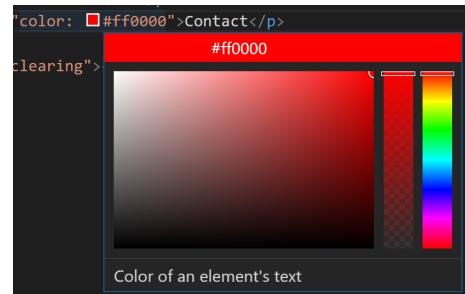
Zo komt het hexadecimale FF met de decimale waarde 255 overeen, zijnde  $(15 * 16^1) + (15 * 16^0) = 240 + 15 = 255$  en is het hexadecimale 75 decimaal 117:  $(7 * 16^1) + (5 * 16^0) = 112 + 5 = 117$ .

De hexadecimale kleurencode wordt door een # teken voorafgegaan. Zo is #FF0000 zuiver rood en #0000FF zuiver blauw.

Verkort wordt #FF0000 als #F00 en bv. #CC33BB als #C3B weergegeven.

Gelukkig hoeft je je hoofd niet te breken op deze kleurberekeningen. Je kunt bv. de extensie **VS Color Picker** installeren.

Heel handig is het **kleurenpiet** waarmee je een kleur van je scherm kan opzuigen. Zo kan je bv. kleuren uit illustraties verder als kleur in je opmaak gebruiken voor tekst of achtergrond.



Voor een aantal kleuren (17 standaard + 130 extra) kan je ook kleurnamen gebruiken in CSS: *aqua, black, blue, fuchsia...* Zie [www.147COLORS.COM](http://www.147COLORS.COM) voor meer informatie.

Ook via externe online tools als **IMAGECOLORPICKER** (<https://imagecolorpicker.com>) kan je kleurencodes uit afbeeldingen ophalen.

## 6.4 Een eerste eenvoudig extern CSS-bestand

Een extern CSS-bestand is een gewoon bestand dat je in je website plaatst en de extensie **.CSS** meegeeft. Ook hier volstaat een gewone teksteditor.

### 6.4.1 Links tussen een webpagina en een extern CSS-bestand

Maak je bv. een CSS-pagina OPMAAK.CSS, dan voeg je aan de webpagina's waarop je de CSS-opmaak wilt toepassen in het HEAD-gedeelte volgende code toe:

```
<link href="opmaak.css" rel="stylesheet" type="text/css">
```

Alle opmaak uit OPMAAK.CSS wordt voortaan op de webpagina toegepast.

### 6.4.2 Over CSS-regels, selectors, declarations, properties en values

Een CSS-bestand bestaat uit CSS-regels met instructies (declarations) die toegepast worden op een bepaald type informatie, de selector. Een declaratie bestaat uit een CSS-eigenschap en een toegekende waarde.

In een eerste eenvoudig voorbeeld is de selector een HTML-element, maar er is veel meer mogelijk.

**h1 {color: red; font-size: 120%;}**

In bovenstaand voorbeeld is h1 de selector en zijn er twee declarations: kleur en lettergrootte krijgen elk een bepaalde waarde. CSS-regels worden tussen {} (accooldades) geplaatst en een declaratie eindigt op een ;. Je kunt het voorbeeld hierboven ook overzichtelijker als volgt noteren.

```
h1 {  
    color: red;  
    font-size: 120%;  
}
```

In CSS worden geen punthaken gebruikt om HTML-tags als selector weer te geven.

Je hebt wellicht al door dat alle koppen van het hoogste niveau (`<h1>`-tags) met deze CSS-regel rood zullen kleuren en 20 % groter zullen zijn dan de tekstgrootte die in de parent tag gebruikt wordt. De hiërarchische structuur blijft haar rol spelen.

```
body {  
color: #BBB;  
font-size: 16px;  
}  
  
h1 {  
color: red;  
font-size: 120%;  
}
```

Aangezien `<h1>` een child van de `<body>`-tag is, erven de titels in principe de kleur en lettergrootte van de body over, tenzij ze voor de `<h1>`-tag anders zijn ingesteld, wat hier het geval is.

De `<h1>`-tags kleuren hier dus rood en zijn 19.2px groot: 120% van de 16px uit de parent tag.



Plaats nooit spaties bij CSS-eenheden: **16px** en niet 16 px!

Het komt er dus op neer:

- telkens zorgvuldig de juiste selector te kiezen. Hier zijn vele combinaties mogelijk zoals verder zeker blijkt.
- de juiste eigenschappen aan te passen. Het is niet de bedoeling dat je de vele honderden eigenschappen van buiten kent. Veel gebruikte eigenschappen zal je sowieso snel kennen. Alle eigenschappen kan je ook opzoeken. In de volgende hoofdstukken leer je telkens nieuwe CSS-eigenschappen gebruiken.

Werk zo hiërarchisch mogelijk van algemeen naar meer specifiek. Algemene instellingen die voor de gehele webpagina's gelden, stel je best op de `<body>`-tag in.

```
body {  
background-color: #EEE;  
font-size: 16px;  
}  
  
h1 {  
color: red;  
font-size: 120%;  
}
```

Je voegt commentaar in je CSS-code tussen /\* en \*/ afbakening in, bv.:

```
h1 {  
color: red;  
/*hoofdtitels 20% groter dan normale tekst*/  
font-size: 120%;  
}
```

CSS-bestanden kunnen vele pagina's lang zijn. Het komt erop aan ze goed te structureren en van voldoende commentaar te voorzien.

In CSS kan je kleurwaarden met de `rgb()` functie ook decimaal opgeven. Rood is dan:

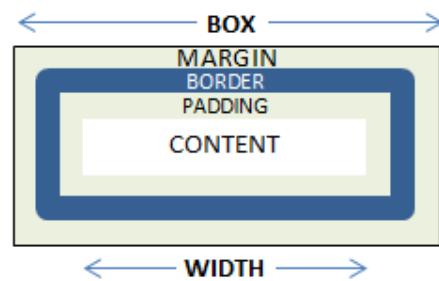
```
h2 {  
color: rgb(255,0,0);  
}
```

## 6.5 Het CSS BOX-model

De meeste HTML-elementen worden in een denkbeeldige *box* weergegeven die horizontaal standaard alle beschikbare breedte, bepaald door de parent tag, inneemt.

Via CSS kunnen we met een aantal eigenschappen spelen:

- de **border** (rand) die rond zo'n box zit;
- de **padding**, dit is de ruimte tussen de rand en de inhoud van de box, de ruimte aan de *binnenkant* van de box dus;
- de **margin**, dit is de ruimte rond de box, aan de *buitenkant* van de box, tussen de elementen dus.
- de **width** en **height**, de breedte en hoogte die we aan de box kunnen meegeven.



### 6.5.1 CSS border

Enkele voorbeelden met de `<p>`-tag maken één en ander duidelijk.

"Ik zag een schokkende documentaire over de enorme hoeveelheid plastic afval die zich een weg weet te banen naar onze oceanen. Daar verandert al die troep onze wateren in een plastic soep.

Tegelijk viel het me op hoeveel plastic flesjes voor bronwater mensen dagelijks weggooien, terwijl het beste drinkwater in veel landen gewoon uit de kraan komt. Je kunt wel zeggen dat ik me hierbij ongemakkelijk voelde. Ik moest er iets aan doen!"

Merijn besluit in actie te komen: hij schrijft een design competitie uit om op zoek te gaan naar 'de perfecte herbruikbare fles voor kraanwater'. Tussen de 100 inzendingen die Merijn ontvangt, zit een opvallend ontwerp van TU Delft alumnus Rinke van Remortel. Zijn minimalistische design is niet alleen esthetisch, maar ook praktisch.

Dankzij de unieke vormgeving is zijn fles goed schoon te houden waardoor je 'm jarenlang kan

```
p {  
border: solid 1px #Foo;  
}
```

Voeg je een rand aan de `<p>`-tag toe, dan zie je snel dat de afstand tussen de paragrafen buiten de box zit en dus margin moet zijn.

"Ik zag een schokkende documentaire over de enorme hoeveelheid plastic afval die zich een weg weet te banen naar onze oceanen. Daar verandert al die troep onze wateren in een plastic soep."

Tegelijk viel het me op hoeveel plastic flesjes voor bronwater mensen dagelijks weggooien, terwijl het beste drinkwater in veel landen gewoon uit de kraan komt. Je kunt wel zeggen dat ik me hierbij ongemakkelijk voelde. Ik moest er iets aan doen!"

Merijn besluit in actie te komen: hij schrijft een design competitie uit om op zoek te gaan naar 'de perfecte herbruikbare fles voor kraanwater'. Tussen de 100 inzendingen die Merijn ontvangt, zit een opvallend ontwerp van TU Delft alumnus Rinke van Remortel. Zijn minimalistische design is niet alleen esthetisch, maar ook praktisch.

Dankzij de unieke vormgeving is zijn fles goed schoon te houden waardoor je 'm jarenlang kan gebruiken."

## 6.5.2 CSS margin

Door de margin op 0 in te stellen, verdwijnt de ruimte tussen de verschillende elementen:

```
p {  
border: solid 1px #F00;  
margin: 0px;  
}
```

"Ik zag een schokkende documentaire over de enorme hoeveelheid plastic afval die zich een weg weet te banen naar onze oceanen. Daar verandert al die troep onze wateren in een plastic soep."

Tegelijk viel het me op hoeveel plastic flesjes voor bronwater mensen dagelijks weggooien, terwijl het beste drinkwater in veel landen gewoon uit de kraan komt. Je kunt wel zeggen dat ik me hierbij ongemakkelijk voelde. Ik moest er iets aan doen!"

Merijn besluit in actie te komen: hij schrijft een design competitie uit om op zoek te gaan naar 'de perfecte herbruikbare fles voor kraanwater'. Tussen de 100 inzendingen die Merijn ontvangt, zit een opvallend ontwerp van TU Delft alumnus Rinke van Remortel. Zijn minimalistische design is niet alleen esthetisch, maar ook praktisch.

Dankzij de unieke vormgeving is zijn fles goed schoon te houden waardoor je 'm jarenlang kan gebruiken."

## 6.5.3 CSS padding

We gaan nog een stapje verder en voegen padding toe. Je merkt nu dat de vrije ruimte telkens aan de binnenkant van de box komt.

```
p {  
border: solid 1px #F00;  
margin: 0px;  
padding: 1em;  
}
```

"Ik zag een schokkende documentaire over de enorme hoeveelheid plastic afval die zich een weg weet te banen naar onze oceanen. Daar verandert al die troep onze wateren in een plastic soep.

Tegelijk viel het me op hoeveel plastic flesjes voor bronwater mensen dagelijks weggooien, terwijl het beste drinkwater in veel landen gewoon uit de kraan komt. Je kunt wel zeggen dat ik me hierbij ongemakkelijk voelde. Ik moest er iets aan doen!"

Merijn besluit in actie te komen: hij schrijft een design competitie uit om op zoek te gaan naar 'de perfecte herbruikbare fles voor kraanwater'. Tussen de 100 inzendingen die Merijn ontvangt, zit een opvallend ontwerp van TU Delft alumnus Rinke van Remortel. Zijn minimalistische design is niet alleen esthetisch, maar ook praktisch.

Dankzij de unieke vormgeving is zijn fles goed schoon te houden waardoor je 'm

De eigenschappen border, margin, padding kan je op nagenoeg alle selectoren toepassen.

Bij **border: solid 1px #FOO;** gebruikten we de border-eigenschap die eigenlijk een *shorthand* is voor drie individuele eigenschappen:

- **border-style:** waarmee je het type rand opgeeft: naast solid zijn dotted, dashed, double, none... en andere keuzes mogelijk
- **border-width:** de dikte van de rand, in alle mogelijke eenheden op te geven
- **border-color:** de kleur van de rand

Als je wilt, kan je nog verder gaan en zijde per zijde een rand instellen:

- **border-top-style:** stelt enkel de bovenrand in;
- **border-style: dotted solid double dashed;** stelt de omranding bovenaan *dotted*, rechts *solid*, onderaan *double* en links *dashed* in.
- **border-left:** solid 3px #00F; stelt enkel een linkerrand in.
- **border-right-color:** ...

Heel wat combinaties zijn dus mogelijk!

#### 6.5.4 CSS border-radius

Sinds kort kan je – in de meeste browsers – de omranding ook afronden met de **border-radius** eigenschap:

```
p {  
border: solid 2px #00F;  
border-radius: 5px;  
margin-bottom: 1em;  
padding: 0.5em;  
}
```

Je kunt de border-radius per hoek verschillend instellen, bv.:

**Border-radius: 50px 25px 30px 10px**, stelt achtereenvolgens de afronding linksboven, rechtsboven, rechtsonder en linksonder in.

Even goed kan je één bepaalde hoek aanspreken: **border-top-left-radius** en **border-bottom-right-radius** zijn voorbeelden.

"Ik zag een schokkende documentaire over de enorme hoeveelheid plastic afval die zich een weg weet te banen naar onze oceanen. Daar verandert al die troep onze wateren in een plastic soep."

Tegelijk viel het me op hoeveel plastic flesjes voor bronwater mensen dagelijks weggooien, terwijl het beste drinkwater in veel landen gewoon uit de kraan komt. Je kunt wel zeggen dat ik me hierbij ongemakkelijk voelde. Ik moest er iets aan doen!"

Merijn besluit in actie te komen: hij schrijft een design competitie uit om op zoek te gaan naar 'de perfecte herbruikbare fles voor kraanwater'. Tussen de 100 inzendingen die Merijn ontvangt, zit een opvallend ontwerp van TU Delft alumnus Rinke van Remortel. Zijn minimalistische design is niet alleen esthetisch, maar ook praktisch.

Dankzij de unieke vormgeving is zijn fles goed schoon te houden waardoor je 'm jarenlang

De witruimte tussen bv. een titel en een alinea of tussen twee alinea's is dus niets anders dan de margin rond de normaal niet-zichtbaar omrande boxen.

De ruimte tussen paragraaf 1 en paragraaf 2 is de optelsom van volgende elementen:

- de padding-bottom van paragraaf 1 als die paragraaf niet van een rand voorzien is.
- de margin-bottom van paragraaf 1 **of** de margin-top van paragraaf 2. Beide marges schuiven ineen: van beide marges wordt de **grootste** marge genomen; niet de optelsom van beide marges!
- de padding-top van paragraaf 2 als die paragraaf niet van een rand voorzien is.

Ook **margin** en **padding** kan je per zijde afzonderlijk instellen:

- margin of padding : alle zijden zelfde waarde
- margin-left, margin-right, padding-top, padding-bottom om per zijde een waarde in te stellen.

Door de margin op **auto** in te stellen en een breedte mee te geven die lager is dan de volle breedte (**width**), wordt een element binnen de beschikbare ruimte, bepaald door de parent container, gemiddeld.

```
p {  
margin: auto;  
margin-top: 0.6em;  
margin-bottom: 0.6em;  
width: 50%;  
padding: 0.5em;  
}
```

Begrijp je volgend resultaat?

"Ik zag een schokkende documentaire over de enorme hoeveelheid plastic afval die zich een weg weet te banen naar onze oceanen. Daar verandert al die troep onze wateren in een plastic soep."

Tegelijk viel het me op hoeveel plastic flesjes voor bronwater mensen dagelijks weggooien, terwijl het beste drinkwater in veel landen gewoon uit de kraan komt. Je kunt wel zeggen dat ik me hierbij ongemakkelijk voelde. Ik moest er iets aan doen!"

Merijn besluit in actie te komen: hij schrijft een design competitie uit om op zoek te gaan naar 'de perfecte herbruikbare fles voor kraanwater'. Tussen de 100 inzendingen die Merijn ontvangt, zit een opvallend ontwerp van TU Delft alumnus Rinke van Remortel. Zijn minimalistische design is niet alleen esthetisch, maar ook praktisch.

Dankzij de unieke vormgeving is zijn fles goed schoon te houden waardoor ie 'niereplastic'

### 6.5.5 CSS width en height

Hoogte en breedte worden normaal automatisch ingesteld in functie van beschikbare ruimte en inhoud. Nagenoeg elk element kan je een eigen breedte/hoogte meegeven.

De width/height eigenschappen houden geen rekening met margin, padding en borders. Die komen er eventueel nog bovenop.

Naast **width** kan je ook een maximum/minimum breedte/hoogte instellen: **max-width**, **min-width**, **max-height**, **min-height**.

### 6.6 CSS classes en IDs

In alle voorbeelden hierboven hebben we enkele HTML-tags als selector gebruikt. Heel vaak echter gaan we verfijnder te werk. Neem nu het voorbeeld hieronder waarbij de eerste alinea als inleiding een ietwat andere opmaak krijgt.

Laat ons je iets vertellen over de oerknal die leidde tot het bestaan van Dopper. Net als alle goede verhalen, begint die van ons op de bank. We schrijven 2009. Merijn Everaarts zapt met 't bord op schoot langs een documentaire die zijn leven verandert...

"Ik zag een schokkende documentaire over de enorme hoeveelheid plastic afval die zich een weg weet te banen naar onze oceanen. Daar verandert al die troep onze wateren in een plastic soep.

Tegelijk viel het me op hoeveel plastic flesjes voor bronwater mensen dagelijks weggooien, terwijl het beste drinkwater in veel landen gewoon uit de kraan komt. Je kunt wel zeggen dat ik

De `<p>`-tag als selector gebruiken, zou alle paragrafen van die opmaak voorzien. We gaan dus die eerste paragraaf speciaal moeten aanspreken en dat kunnen we op twee manieren: via een **class** of via een **ID**:

- **class:** `<p class="inleiding">`... door aan een tag een class toe te voegen, naam naar keuze, kan je die paragraaf afzonderlijk benaderen in CSS.
- **ID:** `<p id="inleiding">`... door aan een tag een ID toe te voegen, naam naar keuze, kan je die paragraaf afzonderlijk benaderen in CSS.

Classes en ID's worden in de praktijk vaak dooreen gebruikt. Je zou een ID als uniek te gebruiken kunnen beschouwen, terwijl een class herhaald gebruikt kan worden, bv. bij elke eerste paragraaf in een webpagina.

In CSS spreek je classes en ID's verschillend aan: je verwijst naar een class door de naam met een **punt** te beginnen (`.inleiding`), naar een ID door de naam met een **spoorwegteken** te starten (`#inleiding`).

```
.inleiding {  
    font-size: 1.1em;  
    font-weight: bold;  
    color: #444;  
    line-height: 1.5em;  
}  
  
#inleiding {  
    font-size: 1.1em;  
    ...  
}
```

In het voorbeeld hierboven leer je twee nieuwe eigenschappen gebruiken: **font-weight** en **line-height**. Die eigenschappen spreken ongetwijfeld voor zich.

De class *inleiding* kan je overal gebruiken, maar je kunt je CSS-selector nog verder specifiëren:

- **p.inleiding** geldt enkel bij paragrafen die van de class *inleiding* zijn voorzien.
- **h1.inleiding** kan een andere opmaak instellen voor koppen `<h1>` die van de class *inleiding* zijn voorzien.

Je kunt aan één element gerust verschillende classes koppelen, bv.:

```
<p class="inleiding geenkader">...</p>
```

Hier worden zowel de eigenschappen uit de class *inleiding* als uit de class *geenkader* toegepast.

## 6.7 Pseudo-classes

In CSS zijn ook enkele zogenaamde *pseudo-classes* beschikbaar om een speciale toestand weer te geven. Pseudo-classes worden via een dubbelpunt toegevoegd.

### 6.7.1 Pseudo-classes bij hyperlinks (A-tag)

Bij de `<a>`-tag, de tag die je gebruikt bij hyperlinks, horen vier zogenaamde *pseudo-classes* die de verschillende weergaves van een hyperlink regelen.

- **a:link**: weergave van een gewone hyperlink
- **a:visited**: weergave van een bezochte hyperlink
- **a:hover**: weergave als je met de muis over de hyperlink beweegt
- **a:active**: weergave van een hyperlink waarop laatst geklikt werd.

Zo kan je hyperlinks verschillend weergeven als ze al bezocht zijn, als je erover beweegt...

### 6.7.2 Eerste/laatste element uit een reeks

- **:first-child**: heeft enkel betrekking op het eerste element binnen de parent tag.
- **:last-child**: heeft enkel betrekking op het laatste element binnen de parent tag.

Hierboven hebben we voor de eerste paragraaf uit een pagina, de inleiding, een specifieke class *inleiding* voorzien. Even goed had je hier **:first-child** als selector kunnen gebruiken om de eerste paragraaf een specifieke opmaak te bezorgen.

### 6.7.3 Pseudo-elementen

Een CSS pseudo-element gebruik je in heel specifieke gevallen. Bij pseudo-elementen plaats je twee ::-tekens.

```
p::first-letter {  
    font-weight: bold;  
    font-size: 2em;  
    color: #FOO;  
}
```



In dit voorbeeld geef je de eerste letter van elke paragraaf een specifieke opmaak.

## 6.8 Samengestelde selectoren

Je kunt CSS-selectoren van heel eenvoudig tot complex samengesteld uitwerken.

Eigenlijk is **p.inleiding** al zo'n samengestelde selector die enkel werkt in paragrafen die van de class *inleiding* zijn voorzien.

Enkele voorbeelden:

- **footer p**: werkt enkel op paragrafen die in de `<footer>`-container zitten.
- **#content ul, #content ol**: werkt op opsommingen binnen de container met ID `content`, zowel op genummerde opsommingen, als op opsommingen met bullets.
- **#content.inleiding**: werkt op een element met ID `content` én class `inleiding`.
- **#content .inleiding**: werkt op een element met de class `inleiding` binnen een container met de ID `content` (spatie)
- **h1, h2**: van toepassing op zowel `<h1>` als `<h2>` tags.
- **h1 i**: van toepassing op `<i>`-tags binnen `<h1>`-tags.
- **div > p**: selecteert `<p>`-tags waarvan de parent een `<div>`-tag is.
- **div + p**: selecteert `<p>`-tags die onmiddellijk na een `<div>`-tag volgen.
- **p ~ ul**: selecteert elke `<ul>`-tag voorafgegaan door een `<p>`-tag.

## 7 WERKEN MET AFBEELDINGEN

Afbeeldingen zijn onontbeerlijk in webpagina's. Een goede keuze aan afbeeldingen draagt in hoge mate bij tot de aantrekkelijkheid van een website.

In websites kan je niet alle formaten afbeeldingen gebruiken, maar deze formaten zijn wel geschikt:

- **JPEG of JPG** (*Joint Photographic Experts Group*). Miljoenen kleuren. Goed voor foto's. Goede compressie zodat de afbeeldingen niet te veel bestandsruimte innemen. Heel vaak gebruikt.
- **GIF** (*Graphics Interchange Format*): maximaal 256 kleuren. Transparante kleur is mogelijk. Toepassing: lijnen, logo's met tekst, achtergronden.
- **PNG** (*Portable Network Graphics*): compact, miljoenen kleuren, transparantie mogelijk. Goed voor foto's.

Organiseer je website zo dat alle afbeeldingen in een afzonderlijke map, bv. **IMAGES** geplaatst worden.

De HTML-code voor een afbeelding:

```

```

<IMG>	voegt een illustratie aan je webpagina toe
SRC	source, verwijzing naar het gebruikte afbeeldingsbestand
ALT	documenteert de afbeelding met tekst
HEIGHT	bepaalt de hoogte van de afbeelding
WIDTH	bepaalt de breedte van de afbeelding

### 7.1 Het formaat van de afbeelding

Probeer zoveel als mogelijk met originele formaten te werken.

Het aanpassen van breedte en hoogte vermindert vaak de kwaliteit van de afbeelding. Zorg altijd ervoor dat breedte en hoogte gelijkmatig aangepast worden, zoniet vervormt de afbeelding.

Je kunt ook het formaat van de afbeelding aanpassen aan de beschikbare ruimte. Haal in dat geval de html-attributen *width* en *height* gewoon weg en gebruik CSS:

```
img {  
width: 100%;  
height: auto;  
}
```

Hier neemt de afbeelding de volledige beschikbare breedte in en past de hoogte zich proportioneel aan. In plaats van de **width** eigenschap, kan je ook **max-width** gebruiken. De illustratie kan nu wel verkleinen als dat nodig is, maar nooit groter worden dan haar eigen oorspronkelijke breedte.

```
img {  
    max-width: 100%;  
    height: auto;  
}
```

## 7.2 Bronbestand en hyperlinks

Via het **src**-attribuut haal je het achterliggende grafische bestand op. Dat moet dus van de webserver getransporteerd worden en mag niet te groot van volume zijn om snel beschikbaar te zijn. Web-afbeeldingen hoeven sowieso niet de kwaliteit van afbeeldingen die voor print bedoeld zijn te hebben. Zoek de juiste verhoudingen tussen kwaliteit en downloadsnelheid.

Aan een afbeelding kan je ook een tekst koppelen (**ALT**-attribuut). Deze tekst wordt weergegeven als je met de muis over de illustratie beweegt. De Alt-tekst wordt ook wel eens gebruikt om verduidelijkingen of instructies aan een illustratie toe te voegen. Ook voor personen met een visuele beperking is die Alt-informatie heel belangrijk in de gebruikte hulpsoftware.

De opgegeven ALT-tekst wordt door zoekmachines ' gevonden' en hiermee houd je dus best rekening bij het bepalen van je ALT-tekst.

Net zoals tekst, kan je ook rond een afbeelding een **hyperlink** voorzien die geactiveerd wordt als je op de afbeelding klikt.

## 7.3 Afbeeldingen omranden

Wil je één of meer afbeeldingen omranden, doe dit dan via de **border**-eigenschap in CSS.

```
img {  
    border: solid 1px #00F  
}
```

Hiermee worden alle afbeeldingen in je website omrand. Bij uitzonderingen kan je aan die afbeeldingen dan bv. de class **noborder** toevoegen: `<img class="noborder" src=...>`. Via die class schakel je dan de omranding uit.

```
img.noborder {  
    border: none;  
}
```

## 7.4 Afbeeldingen als achtergrond

Afbeeldingen kunnen ook als achtergrond in een container gebruikt worden via de eigenschap **background-image**. Met **background-repeat** kan je een kleine afbeelding herhaald naast/onder elkaar kopiëren tot de beschikbare ruimte gevuld is.

```
#banner {  
width: 100%;  
height: 400px;  
background-image: url('images/fles.jpg');  
background-repeat: no-repeat;  
background-size: contain;  
border: 1px solid red;  
}
```

Nog even aandacht voor de eigenschap **background-size**:

- **contain**: schaalt de afbeelding maar houdt de breedte/hoogte verhouding. De container wordt hier enkel volledig gevuld als breedte/hoogte van de container overeenkomen met breedte/hoogte van de afbeelding.
- **100% 100%**: de container wordt volledig gevuld maar dat kan leiden tot een vervormde ('uitgerokken') afbeelding als de verhoudingen breedte/hoogte niet matchen.
- **cover**: de container wordt volledig gevuld én de breedte/hoogte verhouding wordt gerespecteerd. Hierdoor zal een gedeelte van de achtergrondillustratie misschien niet zichtbaar zijn en buiten de container vallen.

## 7.5 Afferonde afbeeldingen

De eerder besproken **border-radius** eigenschap kan je ook mooi op afbeeldingen toepassen. Enkele voorbeelden.



border-radius: 50%

border-radius: 25% 0%

In de vorige hoofdstukken hebben we al heel wat basiselementen van HTML en CSS besproken. In dit hoofdstuk komen we – stap voor stap – tot een volledige webpagina.

Vertrekend vanuit het besproken box-model gaan we ook dieper in op het plaatsen van informatiecontainers in een webpagina. Hierin speelt CSS opnieuw een belangrijke rol.

In het HTML-gedeelte bouwen we de webpagina uit verschillende containers op. Algemeen maken we hier vaak gebruik van de `<div>`-tag, een tag waarin we alle mogelijke informatie kwijt kunnen. Door aan die `<div>`-tags telkens een ID toe te voegen, kunnen we elke DIV-tag afzonderlijk aanspreken binnen CSS.

Als je het voorbeeld hieronder bekijkt, dan onderscheid je al snel een aantal containers:

- de gehele website heeft een witte achtergrond en de ruimte buiten de website is groen. Afhankelijk van de gebruikte device zal die groene ruimte kleiner/groter zijn. Ze dient als een soort *bufferruimte*.  
Om de website in haar geheel, gemakkelijk te kunnen aanspreken, voorzien we een container die we de ID *wrapper* meegeven. Op die manier kunnen we gemakkelijk achtergrondkleur, omranding, positie... van de eigenlijke webinhoud instellen.
- Binnen die *wrapper* verfijnen we nu de informatie in een aantal containers. De namen kies je in principe volledig zelf:
  - *banner*: de ruimte die de foto bevat.
  - *info*: de ruimte met adres en telefoonnummers
  - *menu*: de ruimte links met het menu.
  - *main*: de ruimte rechts met de eigenlijke inhoud die pagina per pagina verschilt.
  - *copyright*: de ruimte onderaan met de copyright-informatie.



In HTML wordt onze code:

```
<body>
  <div id="wrapper">
    <div id="banner"></div>
    <div id="info"></div>
    <nav></nav>
    <main></main>
    <div id="copyright"></div>
  </div>
</body>
```

Die code gaan we nu stap voor stap verfijnen en via CSS opmaken en positioneren.

Met de \* als selector stellen we opmaak in die toegepast wordt op alle elementen. De startpositie van de margin en padding op alle elementen bij de start op 0 instellen is een goed idee. Sommige elementen hebben immers standaard een margin en/of padding, andere niet. Door padding en margin bij alle elementen vooraf op 0 in te stellen, stel je zelf verder alle vrije ruime (margin en padding) in.

```
* {
  margin: 0;
  padding: 0;
}
```

Voor een aantal algemene instellingen die voor de volledige website gelden, haal je de `<body>`-tag als selector boven:

```
body {
  font-family: "Trebuchet MS", Arial, Helvetica, sans-serif;
  font-size: 0.8em;
  background-color: #9C0;
  color: #990;
}
```

Zo geldt de groene achtergrond in principe voor de gehele website. Elk child-element erft die eigenschap, tenzij je de achtergrondkleur in het child-element zelf aanpast, wat we hier doen op de wrapper door daar een witte achtergrond in te stellen.

```
#wrapper {
  width: 960px;
  margin-left: auto;
  margin-right: auto;
  border: solid 2px #999900;
  background-color: #FFF;
  padding: 10px;
}
```

In dit eenvoudige voorbeeld werken we met een vaste breedte voor de gehele website (960 px). Door margin-left/right op auto in te stellen wordt de restruimte mooi links en rechts verdeeld en staat de webpagina dus horizontaal in het midden van de browser.

De padding zorgt voor witruimte langs de binnenkant van de *wrapper*-container.

De *banner*-container heeft geen behoefte aan CSS-instructies. In de *info*-container moeten we wél aan de slag. Hier moet de tekst immers in witte letters op groene achtergrond komen:

```
#info {  
padding: 10px;  
background-color: #990;  
color: #FFF;  
}
```

## 8.1 Positionering van elementen

Voor we verder gaan, moeten we eerst begrijpen hoe elementen in een webpagina geplaatst worden. Opnieuw komen we bij CSS terecht.

### 8.1.1 Box-types en CSS-display

Elk HTML-element wordt in principe standaard op één van volgende manieren weergegeven, ingesteld via **display**.

- **block**: de meeste elementen nemen de volledige beschikbare breedte in, bv. bij een `<p>`-tag, een `<hx>`-tag... Een element dat op zo'n block weergave volgt, start in principe onder het vorige element, nooiternaast. Verder leer je dat ook dat relatief is.
- **inline**: bij bv. de `<strong>`-tag, de `<img>`-tag... die niet altijd de volledige beschikbare breedte innemen. Inline-elementen sluiten normaal naast elkaar aan en volgen de gewone tekststroom van links naar rechts.

Bekijk het ietwat absurde voorbeeld hieronder:

```
<body>  
<p>België</p>  
<p>Nederland</p>  
<p>Luxemburg</p>  
De <strong>Benelux</strong> bestaat uit  
<strong>België</strong>, <strong>Nederland</strong> en  
<strong>Luxemburg</strong>.  
</body>
```

Niets speciaals, tenzij je de display via CSS verandert. Bestudeer de weergave hieronder rechts met onderstaande CSS-regels:

```
p {  
display: inline;  
}  
strong {  
display: block;  
}
```

België

Nederland

België Nederland Luxemburg De  
**Benelux**  
bestaat uit  
**België**

Luxemburg

De Benelux bestaat uit **België, Nederland en Luxemburg**

*normale weergave*

,  
**Nederland**  
en  
**Luxemburg**

*weergave met CSS-stijl*

Met de CSS **display** eigenschap kan je elementen die normaal onder elkaar weergegeven worden gemakkelijk naast elkaar weergeven en omgekeerd. De belangrijkste waarden van de display eigenschap zijn voorlopig:

- **display: block**  
element neemt de volledige beschikbare breedte in. Standaard bij o.a. `<p>`, `<h1>`...tags. Block-elementen komen dus normaal **onder** elkaar.
- **display: inline**  
element volgt de gewone tekststroom. Standaard bij o.a. `<strong>` tag. Inline-elementen sluiten dus normaal **naast** elkaar aan.
- **display: inline-block**  
element volgt de gewone tekststroom (naast elkaar), maar je kunt er een breedte (`width`) aan meegeven en eigen margin/padding instellen.
- **display: none**  
element wordt niet weergegeven en de voorziene ruimte voor het element wordt opgeheven alsof er niets staat. Alternatief: **visibility: hidden**, maar hier wordt de ruimte van het element nog vrijgehouden (lege plaats).

### 8.1.2 De flow van de elementen

Met **flow** bedoelen we hoe de elementen geplaatst worden: naast elkaar, onder elkaar... altijd binnen de beschikbare ruimte van de parent container. Om elementen naast elkaar te plaatsen, moet er natuurlijk voldoende ruimte beschikbaar zijn.

Elementen hoeven evenwel niet altijd de normale flow te volgen. Dat wordt dan weer door de **position**-eigenschap in CSS bepaald. Die eigenschap heeft volgende mogelijke waarden:

- **static** (standaardwaarde): het element volgt de normale flow.
- **relative**: het element wordt relatief t.o.v. de normale positie geplaatst (bv. `left: 20px`: het element schuift 20 px naar rechts op t.o.v. haar normale positie).
- **absolute**: het element wordt t.o.v. het eerste parent element dat niet *static* gepositioneerd is geplaatst.
- **fixed**: het element wordt t.o.v. het browser venster gepositioneerd. Scrollen heeft geen invloed op de positie van het element.

Via de opties **left**, **right**, **top** en **bottom** (ver)plaats je de elementen.

Er zijn dus heel wat instellingen die meespelen bij het plaatsen van elementen. Geen nood, in de voorbeelden die volgen wordt één en ander duidelijk.

Een **eerste voorbeeld**. Bij het openen van de startpagina verschijnt een tijdelijke boodschap 'bovenop' de eigenlijke inhoud van de pagina.



In HTML plaatsen we een lege `<div>`-tag met de ID *boodschap* direct onder de `<body>`-tag.

```
<body>
<div id="bodschap"></div>
```

Positie en grootte van de `<div>`-tag worden via CSS-geregegd. Zelfs de inhoud – een afbeelding – kan je in CSS-opgeven. In de HTML-code blijft het bij een lege `<div>`-tag.

```
#bodschap {
background-image: url(images/win_kaasplank.jpg);
width: 447px;
height: 217px;
position: fixed;
top: 150px;
left: 50%;
margin-left: -224px;
opacity:0.8;
```

- de inhoud van de `<div>`-tag is een afbeelding die als CSS-achtergrond gebruikt wordt. Het formaat van de `<div>`-tag (width/height) stemt uiteraard met de grootte van de afbeelding overeen.
- door de position op *fixed* in te stellen krijgt de `<div>`-tag een vaste plaats t.o.v. het browser-venster: 150 px van de bovenrand (top) en horizontaal gemiddeld. Met *left: 50 %* komt de linkerrand van de DIV-tag met het midden van het venster overeen. De helft van de breedte wordt naar links gecorrigeerd (*margin-left: -224px*) zodat de bodschap in het midden verschijnt.
- de css-eigenschap **opacity** zorgt voor enige transparantie zodat je achter de bodschap nog een glimp van de achterliggende inhoud kunt zien.
- **position: fixed** zorgt ervoor dat de bodschap niet mee scrollt als je door de pagina scrollt. Bij position **absolute** zou de bodschap mee op-en-neer scrollen. ☒

Het is de bedoeling dat na het lezen van de pop-up bodschap, de informatie kan afgesloten worden. Voeg binnen de `<div>`-tag *bodschap* een sluit-knop toe (bv. een afbeelding SLUITEN.JPG) die je rechtsboven positioneert. Met een javascript onclick-event kan je ervoor zorgen dat de bodschap bij klikken op de knop verborgen wordt. Maar dit leidt ons al (te) ver.

```

<body>
<div id="boodschap">


```

**Tweede voorbeeld.** Bovenop de bannerfoto plaatsen we 'Rederij De Gentenaer':



In HTML voeg je gewoon een container met bv. **#rederij** als ID toe, met daarin de tekst 'Rederij De Gentenaer'. Bemerk dat die container child is van de **#banner**-container.

```

<div id="banner">
  
  <div id="rederij">Rederij De Gentenaer</div>
</div>

```

In CSS halen we de container **#rederij** uit zijn normale flow met **position: absolute**. We positioneren de container in de rechterbovenhoek van de parent-container **#banner** met 0.5em afstand tussenruimte. Verder passen we de weergave van de tekst aan: lettergrootte, kleuren, uitlijning...

```

#rederij {
  position: absolute;
  top: 0.5em;
  right: 0.5em;
  font-size: 2em;
  background-color: blueviolet;
  text-align: center;
  color: #FFF;
  font-style: italic;
  padding: 0.3em;
}

```

Om dit te laten lukken, mag de parent-container **#banner** geen static position hebben.

```

#banner {
  position: relative;
}

```

Door de **position** op **relative** in te stellen – wat op zich niets wijzigt – wordt de child **#rederij** nu netjes bovenop haar parent **#banner** geplaatst met het gewenste resultaat.

Laat ons nu even focussen op de informatie in de website **bootjes**. Die informatie bestaat uit drie blokken, die we als 3 containers kunnen plaatsen.

Rederij De Gentenaer Hoogpoort 39 9000 Gent	T + 32 (0)9 269 08 69 maandag-vrijdag: 09.00-17.00 u.	T + 32(0)473 48 10 36 zaterdag & zondag: 09.00-17.00 u.
---	--	--

Bekijk de HTML-code waarin we binnen de *info*-container drie containers nesten:

```
<div id="info">
  <div id="adres">
    <p>Rederij De Gentenaer</p>
    <p>Hoogpoort 39</p>
    <p>9000 Gent</p>
  </div>
  <div id="week">
    <p>T + 32 (0)9 269 08 69 </p>
    <p>maandag-vrijdag: 09.00-17.00 u.</p>
  </div>
  <div id="weekend">
    <p>T + 32(0)473 48 10 36 </p>
    <p>zaterdag & zondag: 09.00-17.00 u.</p>
  </div>
</div>
```

Om die drie containers naast elkaar te krijgen, moeten we ingrijpen. **<div>**-tags hebben standaard een *block* display en komen dus onder elkaar.

```
#adres, #week, #weekend {
  display: inline-block;
  width: 32%;
}
```

In één CSS-regel kunnen we de drie ID's aanspreken en hun display op **inline-block** instellen. Ze worden nu alvast naast elkaar weergegeven, elk binnen dezelfde breedte, 32%.

Rederij De Gentenaer Hoogpoort 39 9000 Gent	T + 32 (0)9 269 08 69 maandag-vrijdag: 09.00-17.00 u.	T + 32(0)473 48 10 36 zaterdag & zondag: 09.00-17.00 u.
---	--	--

De gezamenlijke breedte moet natuurlijk minder dan 100% zijn. Zoniet, dan 'duw' je het derde element onder de eerste twee.

De inhoud van de containers verschilt. Bemerk dat de inhoud onderaan uitgelijnd wordt, wat niet altijd de bedoeling is, maar in dit voorbeeld wel uitkomt.

Door aan de containers *week* en *weekend* nog een linkerrand toe te voegen, krijg je de verticale lijnen, evenwel nog niet over de volle hoogte. Hiervoor moeten we anders werken.

```
#week, #weekend {
  border-left: solid 1px #FFF;
```

```
padding-left: 5px;  
}
```

Rederij De Gentenaer Hoogpoort 39 9000 Gent	T + 32 (0)9 269 08 69 maandag-vrijdag: 09.00-17.00 u.	T + 32(0)473 48 10 36 zaterdag & zondag: 09.00-17.00 u.
---	--	--

Conclusie: gebruik de display inline-block bij voorkeur enkel voor containers die je naast elkaar wilt weergeven en waarvan de inhoud uit verschillende regels bestaat die per container dezelfde hoogte hebben. Er wordt sowieso onderaan uitgelijnd.

### 8.1.3 Floating elementen

Een andere manier om elementen naast elkaar weer te geven, is te werken met zogenaamde *floating* elementen. Eigenlijk haal je hiermee elementen uit hun normale flow en leg je ze zwevend in een laag bovenop de basisinhoud van je webpagina.

Als we #adres, #week en #weekend *floating* maken, heeft de parent container #info eigenlijk geen echte inhoud meer. Die zal dus geen hoogte meer hebben en ook geen achtergrondkleur meer bezorgen aan de tekst uit de drie floating containers. Vandaar dat we de container #info een hoogte meegeven, bv. 4em, waardoor de ingestelde achtergrondkleur voor 4em hoogte telt.

```
#info {  
...  
height: 4em;  
}
```

Met **float:left** plaats je de containers van links naar rechts naast elkaar: de eerste uiterst links, de tweede ernaast... altijd voor zover er nog ruimte is binnen de parent container. Elk floating element krijgt altijd een breedte.

```
#adres, #week, #weekend {  
float: left;  
width: 32%;  
height: 4em;  
}
```

Door ook de floating containers een hoogte mee te geven, neemt de linkerrand die hoogte aan.

Rederij De Gentenaer Hoogpoort 39 9000 Gent	T + 32 (0)9 269 08 69 maandag-vrijdag: 09.00-17.00 u.	T + 32(0)473 48 10 36 zaterdag & zondag: 09.00-17.00 u.
---	--	--

Met **float:right** start je rechts en worden bijkomende containers links toegevoegd. De verticale randen moet je dan natuurlijk aanpassen (niet gebeurd hieronder):

T + 32(0)473 48 10 36 zaterdag & zondag: 09.00-17.00 u.	T + 32 (0)9 269 08 69 maandag-vrijdag: 09.00-17.00 u.	Rederij De Gentenaer Hoogpoort 39 9000 Gent
--	--	---

Belangrijk is dat je na floating elementen die naast elkaar staan altijd eronder een container plaatst die het naast elkaar 'zweven' van elementen stopt. In de CSS van die container neem je dan de eigenschap **clear** op:

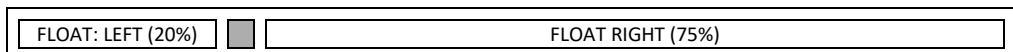
- **clear: both** betekent dat langs beide zijden van het element geen floating element kan geplaatst worden.
- met **clear:left** of **clear:right** kan je links of rechts floating elementen uitsluiten.

Terug naar de bootjes-website, betekent dit dat de volgende container onder #info de regel **clear:left** meekrijgt. Zo kan het menu nooit omhoog naast de info springen.

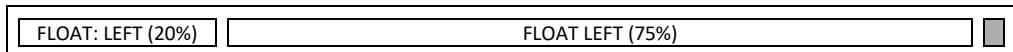
**clear:both** kan hier niet want ook het menu maken we floating zodat naast het menu een container met de eigenlijke inhoud van de webpagina kan geplaatst worden.

```
nav {
  clear: left;
  float: left;
  width: 20%;
}
```

Alhoewel we bij **main** ook **float:left** kunnen kiezen, is **float:right** hier beter. De overige ruimte tussen beide containers wordt dan in het midden gebufferd:



Bij twee keer **float:left**, zit de eventuele bufferruimte op het einde, wat visueel minder goed uitkomt.



```
main {
  float: right;
  width: 75%;
}
```

Aangezien nav en main floating elementen zijn, komt in het element eronder, de footer, de regel **clear:both**. De footer neemt de volle breedte in en dus worden langs beide zijden floating elementen uitgesloten.

```
#footer {
  clear: both;
  padding: 4px;
  background-color: #990;
  color: #FFF;
}
```

Hiermee is de basis-CSS voor de webpagina *bootjes* klaar. Nu kan je stap voor stap de verschillende elementen verder invullen en opmaken.

Nog even herhalen dat voor bepaalde containers specifieke HTML-tags bestaan. Zo hadden we:

- i.p.v. `<div id="banner">...` de tag `<header>` kunnen gebruiken.
- i.p.v. `<div id="footer">...` de tag `<footer>` kunnen gebruiken.

## 8.2 Bootjes: banner en info

In de #banner container voeg je gewoon de illustratie in. Niet toevallig is die illustratie even breed dan de voorziene ruimte binnen de #wrapper. 960 is niet zomaar gekozen: het past netjes op de meeste tablets. Verwijder gerust/best de width/height in de `<img>`-tag!

Bij PC's met hoge resolutie zal er evenwel een aanzienlijke bufferruimte ontstaan en op de smart phone zal je moeten scrollen. In een verdere fase zoeken we een oplossing.

```
<div id="banner">
  
</div>
<div id="info">
  <div id="adres">
    <p>Rederij De Gentenaer</p>
    <p>Hoogpoort 39</p>
    <p>9000 Gent</p>
  </div>
```

Wil je ietwat ruimte tussen de informatie in de #info-container, spreek dan de `<p>`-tag aan binnen #info. Zo pas je de afstand tussen de paragrafen enkel in de #info container aan en niet in andere containers.

```
#info p {
  margin-top: 2px;
  margin-bottom: 2px;
}
```

## 8.3 Bootjes: menu

Nagenoeg alle menu's worden gewoon als opsommingen ingetypt, of de menu-items nu horizontaal of verticaal weergegeven worden, maakt niet uit.

Binnen een opsomming kan je immers gemakkelijk een tweede, derde niveau nesten en dat is handig voor submenu's.

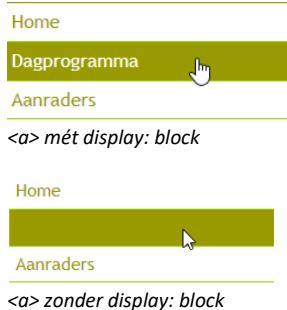
```
<nav>
<ul>
  <li><a href="index.html">Home</a></li>
  <li><a href="dagprogramma.html">Dagprogramma</a></li>
  <li><a href="aanraders.html">Aanraders</a></li>
  <li><a href="scholen.html">Scholen</a></li>
  <li><a href="bedrijven.html">Bedrijven</a></li>
</ul>
</nav>
```

Zonder enige CSS ziet het menu er inderdaad als een gewone opsomming uit (links).

Met CSS wordt het snel ietsje fraaier (rechts).



```
nav ul {
list-style:none;
}
nav li {
margin-left: 0px;
padding-left: 5px;
padding-top: 5px;
padding-bottom: 5px;
border-bottom: solid 1px #9C0;
}
nav li:hover {
color: #FFF;
}
nav a
{
text-decoration: none;
color: #990;
display: block;
}
nav a:hover {
background-color: #990;
color: #FFF;
}
```



Met **ul** spreek je de opsomming in haar geheel aan.

Door **list-style** op **none** te plaatsen, verdwijnt alvast het opsommingsbullet in de opsomming.

List-style is een *shorthand* voor **list-style-type**, **list-style-position** (binnen of buiten de opsommingstekst) en **list-style-image** (eventueel figuurtje als opsommingsteken).

**list-style-type: disc, circle, decimal, square, none...**

**list-style-position: inside, outside**

**list-style-image: url(...)**

Met **li** spreek je elk individueel opsommingsitem aan. Onder elk item komt een horizontaal lijtje.

De pseudoclass **:hover** zorgt voor inversie (witte letters op groene achtergrond) als je over een item beweegt met de cursor.

De menu-items zijn natuurlijk elk van een hyperlink voorzien. Met **text-decoration:none** schakel je de standaard onderstreping van de hyperlink uit. Stel ook de kleur van de hyperlink in: gewoon weergave en bij **:hover**.

Let bij de hyperlink nog even op de **display:block** regel. De **<a>**-tag is standaard een inline-tag. Normaal kan je dus enkel op de tekst van de hyperlink klikken en door ernaast te bewegen zou je zelfs niets meer merken (groen op groen). Met **display:block** loopt de hyperlink nu over de volle breedte, bepaald door de parent **<li>**-tag die van nature een block-display heeft. Nu kan je over de volle breedte bewegen en klikken. Details maken het verschil!

## 8.4 Bootjes: content

### Rederij De Gentenaer

Wandelen langs kunst, cultuur en geschiedenis.

Ten volle avonturen beleven, open uw zintuigen en laat u verleiden door de eindeloze mogelijkheden die Gent en België u te bieden hebben. Reserveer een één- of meerdagse excursie met Rederij De Gentenaer en uw trip kan niet meer stuk.

Op deze manier bent u altijd direct bij de juiste persoon.

Wees tijdig met uw reservatie van kamers.

Wij staan steeds ter beschikking voor vragen en info, service en kwaliteit staan bij ons voorop.

Wat wij voor u kunnen reserveren:

- Hotelkamers
- Boottochten

```
main h1 {  
margin-bottom: 1em;  
}  
  
main h2 {  
margin-top: 0.8em;  
margin-bottom: 0.8em;  
}  
  
main p {  
margin-top: 0.4em;  
margin-bottom: 0.4em;  
}  
  
main ul {  
padding-left: 0px;  
}  
  
main li {  
margin-left: 17px;  
}  
  
.lijntje {  
border-bottom: solid 1px #FC3;  
}
```

Bij de content bepaal je natuurlijk de witruimte bij koppen, paragrafen...

Als je bij **h1** geen **font-size** opgeeft, wordt een ingebouwde standaardgrootte toegepast, die vrij groot is. De grootte wordt dus niet van de parent-tag overgenomen.

Bij de opsommingen hier is wél een opsommings-teken nodig. Dat opsommingsteken staat standaard buiten de box. Vandaar dat we bij elk element een kleine correctie maken (**margin-left: 17px**) om het opsommingsteken mooi uit te lijnen.

Alternatief zonder **margin-left** op **li**:

```
main ul {  
position: relative;  
left: 1.2em;}
```

De class **.lijntje** plaatst horizontale lijntjes, bv in het programma:



## 8.5 Bootjes: footer

In deze oefening is de footer een eitje.

```
#footer {  
clear: both;  
padding: 4px;  
background-color: #990;  
color: #FFF;  
}
```

Gewoon tekstkleur en achtergrondkleur instellen en zorgen dat de tekst niet helemaal tegen de achtergrondrand kleeft (**padding**).

De **clear**-eigenschap hebben we eerder besproken.

## 8.6 Menu's en submenu's

Bestudeer aandachtig de HTML-code hieronder en bekijk hoe de hoofdmenu's *Boottochten* en *Scholen* een submenu onder zich hebben. Een submenu is een autonome opsomming binnen één opsommingselement.

Typisch aan een submenu is dat het een `<ul>`-tag betreft waarvan de parent tag een hoger `<li>`-element is. Daarvan maken we in CSS gebruik.

```
<nav>
<ul>
    <li><a href="index.html">Home</a></li>
    <li><a href="boottochten.html">Boottochten</a>
        <ul>
            <li><a href="#">Standaard formule</a></li>
            <li><a href="#">Formule Gourmand</a></li>
            <li><a href="#">VIP-aanbod</a></li>
        </ul>
    </li>
    <li><a href="dagprogramma.html">Dagprogramma</a></li>
    <li><a href="aanraders.html">Aanraders</a></li>
    <li><a href="scholen.html">Scholen</a>
        <ul>
            <li><a href="#">Basisonderwijs</a></li>
            <li><a href="#">Secundair onderwijs</a></li>
            <li><a href="#">Hoger onderwijs</a></li>
        </ul>
    </li>
    <li><a href="bedrijven.html">Bedrijven</a></li>
</ul>
</nav>
```

### 8.6.1 Verticaal hoofdmenu

```
nav li ul {
    display: none;
    background-color: #990;
    width: 100%;
}

nav li:hover ul {
    display: block;
    position: absolute;
    left: 100%;
    top: 0px;
}

nav li {
    position: relative;
}

nav li ul a {
    color: #FFF;
```

Standaard zijn de submenu's niet zichtbaar. Aangezien ze – indien zichtbaar – boven de content verschijnen, geven we ze zeker een gekleurde achtergrond. De 100% breedte betekent dat het submenu even breed is dan de parent-tag, een item uit het hoofdmenu.

Van zodra we over een hoofdmenu-item bewegen (`li:hover`), maken we het bijhorend submenu zichtbaar (`display:block`). Het submenu moet netjes naast het hoofdmenu komen. Daarvoor gebruik je `position:absolute` waarmee je het submenu t.o.v. het hoofdmenu plaatst. `Position:absolute` betekent dat t.o.v. het eerste parent-element dat niet static is gepositioneerd wordt.

Probleem is dat de parent `<li>` juist wél static is, de standaard position. Daarom voegen we aan de `<li>`-tag `position:relative` toe. Dat verandert op

}

zich niets, maar is nodig om het submenu netjes te positioneren.

Met **left:100%** schuift het submenu 100% naar rechts en komt zo juist naast het hoofdmenu terecht. Met **top:0px** start het submenu verticaal ter hoogte van het parent hoofdmenu.



Het is héél belangrijk dat hoofd- en submenu naadloos op elkaar aansluiten. Bij een **gap** tussen hoofd- en submenu, hoe klein ook, vervalt het hover-effect op het hoofdmenu waardoor het submenu plots zal 'verdwijnen' (**display:none** wordt dan opnieuw actief).



## 8.6.2 Horizontaal hoofdmenu

Van het verticale menu van daarnet kan je met enkele CSS-aanpassingen snel een horizontaal menu maken:



```
nav {  
margin-top: 0.3em;  
background-color: #990;  
}  
nav ul {  
list-style:none;  
padding-left: 3em;  
}  
nav li {  
display: inline-block;  
width: 12%;  
border-left: solid 1px #FFF;  
color: #FFF;  
position: relative;  
margin-left: 0;  
padding-left: 0;  
}  
nav li ul {  
display: none;
```

De **nav** container is nu een gewone container die standaard de volledige ruimte onder de banner inneemt. Geen **float** instellingen meer dus.

Met de **margin-top** voorzien we hier een kleine witruimte tussen **#info** container en **nav**.

De **padding-left** op de **ul**-selector zorgt ervoor dat het eerste menu item niet geheel tegen de linker-rand start. Met **display: inline-block** komen de menu-items naast elkaar i.p.v. onder elkaar. Hier is gekozen voor een vaste breedte van 12% voor elk menu-item. Aangezien elk menu-item even hoog is (één regel), kan je hier gerust met een inline-block display werken.

Het submenu wordt opnieuw eerst verborgen (**display:none**) om het weer te geven van zodra over het hoofdmenu met de muis bewogen wordt.

Het submenu wordt breder weergegeven dan het hoofdmenu (**width: 150%**) omdat bij enkele items in

```
background-color: #990;
margin-left: 0;
padding-left: 0em;
border: solid 1px #990;
}

nav li:hover ul {
display: block;
position: absolute;
left: 0;
top: 100%;
width: 100%;
}

nav li li {
display: block;
width: 100%;
border-left: none;
padding-left: 0;
padding-right: 0;
}

nav a {
text-decoration: none;
color: #FFF;
display: block;
padding-top: 5px;
padding-bottom: 5px;
padding-left: 5px;
padding-right: 5px;
}

nav a:hover {
background-color: #FFF;
color: #990;
}
```

het submenu een langere menutekst voorzien is die best niet over twee regels gespreid wordt.

Bestudeer aandachtig de volledige CSS-code van dit horizontale menu.

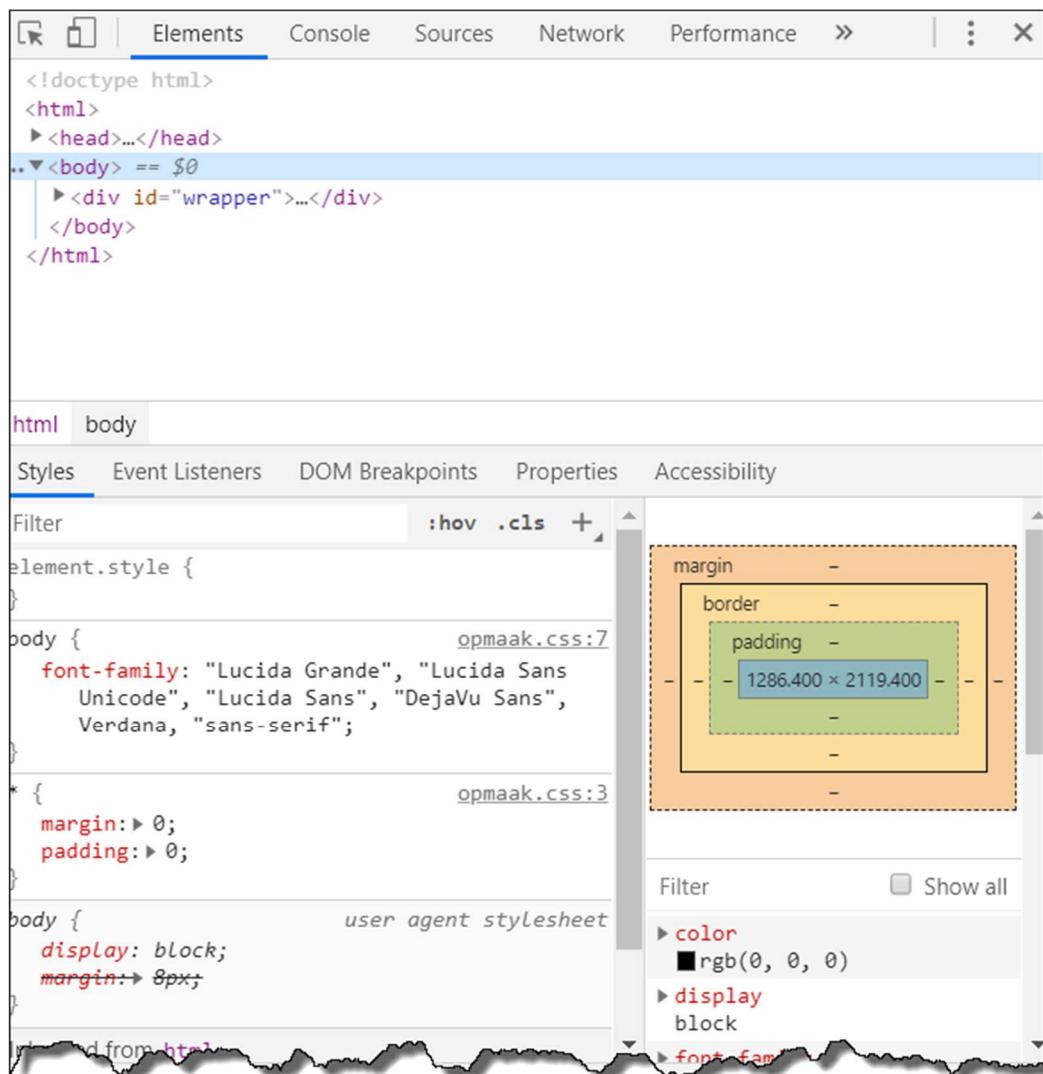
## 9 CODE INSPECTIE MET GOOGLE CHROME

Een heel handig hulpmiddel is de inspectiemogelijkheid die je in verschillende browsers hebt. We bespreken hier de inspectie bij **Google Chrome**, één van de meest gebruikte browsers.

Vanuit een webpagina, kan je via het snelmenu **Inspecteren (Ctrl+Shift+I)** het inspectievenster openen.

In het **Elements** tabblad kan je bovenaan doorheen de html-code scrollen, in- en uitklappen. Ook met de pijltjestoetsen → en ← klap je in- en uit.

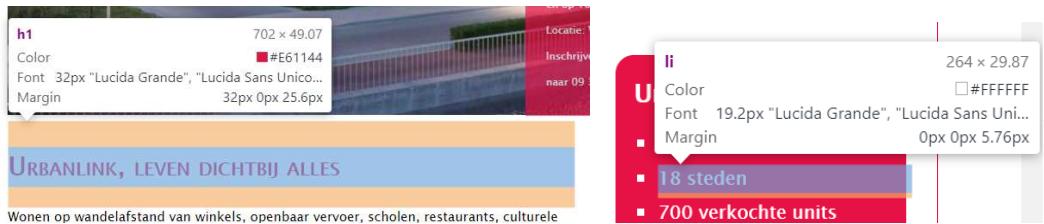
Bij **Styles** eronder zie je dan de verschillende opmaak die van toepassing is en een overzicht van de *box* waarin gewerkt wordt met margin, padding, border. Wijzig je ter plekke instellingen, dan krijg je direct het aangepaste resultaat in je browser.



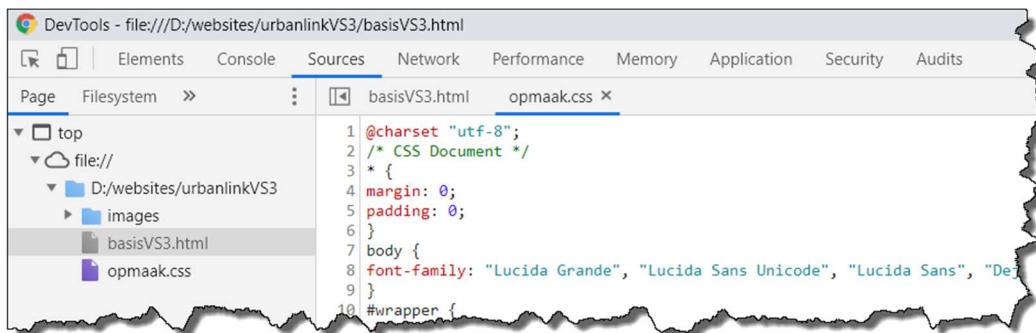
Je kunt gelijk welk individueel element in de webpagina selecteren en met **Ctrl+Shift+C** het inspectievenster openen of vanuit de linkerbovenhoek de element inspector activeren.



Bij de selectie van een element krijg je nu detailinformatie te zien.



In het tabblad Sources kan je de bron van de code raadplegen.



Via inspecteren kan je dus heel veel bijleren over onder meer de werking van HTML en CSS.

## 10 FORMULIEREN

De meeste websites verstrekken niet enkel heel wat informatie aan de bezoeker maar vragen ook informatie van de bezoeker terug. Hiervoor vind je in vele websites invul-formulieren om informatie aan te vragen, een bestelling te plaatsen, een enquête in te vullen, een test af te leggen, keuzes te maken, in te loggen... Interactie met de bezoeker is dus heel belangrijk.

De informatie die via formulieren verstrekt wordt, laten we natuurlijk niet verloren gaan. Heel vaak komen de gegevens uit een ingevuld formulier in een databank terecht die achter de website zit of worden ze op de server verwerkt en wordt een resultaat teruggestuurd.

Maar ook zonder achterliggende databank kan je formulieren gebruiken en de informatie bijvoorbeeld via e-mail 'verwerken'.

### 10.1 Een formulier toevoegen

Een formulier wordt altijd tussen `<form>`-tags geplaatst en normaal via een webserver verwerkt als je het ingevulde formulier meestal via een knop stuurt.

Voor de verwerking speelt de ID geen rol, maar wel de **name** die je aan de verschillende objecten in je formulier meegeeft.

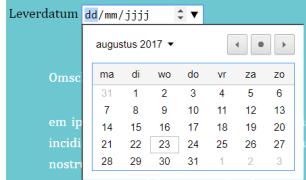
```
<form id="frmform1" name="form1" method="post">  
</form>
```

Alle formulierobjecten moeten binnen het formulier – tussen `<form>` en `</form>` dus – opgenomen worden.

### 10.2 Basiselementen in een formulier

De lijst met basiselementen is lang. We pikken de voornaamste elementen uit.

Tekstvak	Familienaam: <input name="txtNaam" type="text"/>	Tekstvak met één regel invoer <code>&lt;input type="text" name="txtNaam"&gt;</code>
Wachtwoord	Wachtwoord: <input name="txtWachtwoord" type="password"/>	Tekstvak met niet-zichtbare invoer <code>&lt;input type="password" name="txtWachtwoord"&gt;</code>
E-mail	E-mail: <input name="txtEmail" type="email"/>	Tekstvak voor e-mailadres <code>&lt;input type="email" name="txtEmail"&gt;</code>
Numeriek	Aantal: <input max="5" min="1" name="txtAantal" type="number"/>	Tekstvak voor numerieke waarden <code>&lt;input type="number" name="txtAantal" min="1" max="5"&gt;</code>

Tekstgebied	Omschrijving <input type="text"/>	Tekstgebied voor uitgebreide informatie <textarea name="txtOmschrijving"> ... </textarea>						
Datum	Leverdatum <input type="text"/> dd/mm/jjjj 	Tekstvak met date-picker <input type="date">						
Keuzelijst	Betaling <input type="button" value="Credit card"/> Credit card <input type="button" value="Overschrijving"/> Overschrijving <input type="button" value="PayPal"/> PayPal	<select name=".cboBetalung"> <option value="CC">Credit card </option> <option value="OV"> Overschrijving</option> <option value="PP"> PayPal</option> </select>						
Keuzerondje	Keuze dessert <input type="radio" value="Tiramisu"/> Tiramisu <input checked="" type="radio" value="Dame blanche"/> Dame blanche <input type="radio" value="Tarte tatin"/> Tarte tatin	Slechts één keuze mogelijk <input type="radio" name="optDessert" value="tiramisu">Tiramisu  <input type="radio" name="optDessert" value="ijs">Dame blanche  <input type="radio" name="optDessert" value="tartetatin">Tarte tatin						
Selectievakje	Gesproken talen <input checked="" type="checkbox"/> Nederland <input type="checkbox"/> Frans <input checked="" type="checkbox"/> Engels <input type="checkbox"/> Duits	Verschillende keuzes mogelijk <input type="checkbox" name="optNL" value="NL"> Nederland   <input type="checkbox" name="optFR" value="FR">Frans   <input type="checkbox" name="optEN" value="EN">Engels   <input type="checkbox" name="optDE" value="DE">Duits						
Knop	<input type="button" value="Verzenden"/>	Verstuurt het formulier (type: submit) <input type="submit" name="btnVerzenden" value="Verzenden">						
Label	Gemeente 	<label for="txtGemeente">Gemeente </label>						
Veldenset	Adresgegevens <table border="1"><tr><td>Familienaam</td><td>Voornaam</td></tr><tr><td>Straat</td><td>Nr en bus</td></tr><tr><td>Postcode</td><td>Gemeente</td></tr></table>	Familienaam	Voornaam	Straat	Nr en bus	Postcode	Gemeente	Groepeert informatie visueel <fieldset> <legend>Adresgegevens</legend> ... </fieldset>
Familienaam	Voornaam							
Straat	Nr en bus							
Postcode	Gemeente							
Schuifregelaar	Documentatie <input type="range"/> Catering <input type="range"/> Parkeergelegenheid <input type="range"/>	<input name="Documentatie" type="range" min="0" max="10"/>						

Verborgen veld	Informatie die verborgen wordt meegestuurd zonder dat de gebruiker die invult.	<code>&lt;input name="txtBron" type="hidden" value="knack"&gt;</code>
----------------	--	---

### 10.3 Formulier opmaken

Nu je de gebruikte tags kent, kan je die gebruiken om je formulier via CSS mooi op te maken.

Aandachtspunt is misschien de `<input>`-tag die naast tekstvakken ook voor selectievakjes, keuzerondjes en knoppen gebruikt wordt. Wat als je bv. de verzendknop wil aanpakken. Dat doe je als volgt:

```
input[type="submit"]  
{  
padding: 0.5em;  
border: solid 2px rgba(10,126,162,1.00);  
color: rgba(10,126,162,1.00);  
font-weight: bold;  
font-size: 1.2em;  
}
```

Met vierkante haakjes beperk je de selectie van een element tot een bepaald attribuut uit dat element.

Hiernaast worden enkel `<input>`-tags met het attribuut `type="submit"` aangesproken, knoppen dus.

**Verzenden**

### 10.4 HTML5 validatie

Bij het gebruik van formulieren zijn er enkele interessante opties beschikbaar, vooral bij de `<input>`-tag.



Zo kan je het versturen van lege tekstvakken vermijden door het attribuut **required** aan de input-tag toe te voegen.

```
<input name="txtGemeente" type="text" required />
```

Er is ook een type **url**, waarmee je het ingeven van een geldige hyperlink afdwingt en een type **tel** voor telefoonnummers. Geef je de website bv. op een smartphone weer, dan past het toetsenbord zich onmiddellijk aan zodat je cijfers kunt intypen.

Verder is er nog het type **color** waarmee je zeer gebruikersvriendelijk een kleur kunt kiezen.

Ook de eigenschappen **placeholder** en **pattern** zijn heel interessant:

- via **placeholder** geef je een tekst op die in het lege tekstvak verschijnt maar automatisch verdwijnt van zodra je iets intypt.
- met **pattern** geef je een patroon op waaraan de invoer moet beantwoorden. In het voorbeeld hieronder moet de artikelcode uit 2 hoofdletters en 3 cijfers bestaan.

```
<input name="txtArt" type="text" pattern="[A-Z]{2}[0-9]{3}"  
placeholder="Geef artikelcode in" />
```

Artikelcode:

Artikelcode:

 Zorg dat de indeling voldoet aan de gevraagde indeling.

Tenslotte kan je ook een combinatie maken van een tekstvak en een keuzelijst. In het tekstvak neem je met de eigenschap **list** een verwijzing op naar een **datalist** waarin je de verschillende keuzes opneemt. Van zodra het tekstvak de focus krijgt, kan je uit de aangeboden keuzes kiezen. Je kunt evenwel ook een waarde intypen buiten het aanbod.

```
<input type="text" name="vervoer" list="transport">
<datalist id="transport">
<option value="auto">
<option value="bus">
<option value="fiets">
<option value="te voet">
<option value="trein">
</datalist>
```

Verdere interessante opties bij de `<input>` tag:

- **autofocus**: om één bepaald vak standaard actief te maken bij het openen van het formulier
- **autocomplete**: op *off* instellen als je niet wilt dat eerder ingevulde gegevens als mogelijke standaardinput weergegeven worden.

# 11 RESPONSIVE WEBDESIGN

In dit hoofdstuk leggen we de nadruk op **responsive** webdesign waarbij dezelfde website goed oogt op verschillende devices: desktop, tablet, smartphone...



## 11.1 Viewport

De **viewport** is de zichtbare ruimte van een website in je browservenster en verschilt natuurlijk sterk van device tot device.

Voortaan voegen we in het *head*-gedeelte van elke webpagina een meta-tag toe die de viewport instelt:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

De breedte van de viewport wordt hier dynamisch ingesteld op de schermbreedte van het gebruikte toestel en de *initial-scale* zet het zoom-niveau in bij het laden van de pagina.

Gebruikers zijn gewoon om verticaal te scrollen. Horizontaal scrollen vermijden we.

## 11.2 Mediaqueries

Afhankelijk van de beschikbare breedte kan je via CSS de containers verschillend weergeven. Eigenlijk maken we voor verschillende breedtes verschillende CSS-regels. Elke groep CSS-regels is dan gekoppeld aan een bepaalde breedte. Zo'n groep samenhangende regels is een **mediaquery**.

Je externe CSS-bestand start dan met algemene instellingen en instellingen voor mobile devices met een kleinere *viewport*, gevolgd door verschillende media-queries die instellingen bevatten die enkel gelden voor devices die aan de voorwaarden voldoen.

In het voorbeeld hieronder – BIVV is ondertussen wel VIAS geworden – voorzien we twee *breakpoints*: op 480px en op 768px:

### [algemene instellingen + smartphone]

**@media only screen and (min-width: 481px) {**

[instellingen voor tablet]...

}

**@media only screen and (min-width: 769px) {**

[instellingen voor desktop]

...

The screenshot shows a desktop view of the BIVV website. At the top, there's a navigation bar with links for Overheden, Pers, Bedrijven, Particulieren, Jeugd & Verenigingen, and Vrijwilligers. Below the navigation is a large green header graphic. The main content area is organized into several columns: 'Rijgeschiktheid' (with an image of two men in a car), 'Onderzoeken & statistieken' (with an image of a ruler on a road map), 'Vrijwilligers' (with an image of people in high-visibility vests and text 'word vrijwilliger bij het BIVV'), 'Verkeersveiligheidsbarometer' (with an image of a highway), 'Via Secura' (with a large green banner), and 'Recente artikelen' (listing articles like 'Uw vakantieherinneringen kunnen u veel kosten' and 'Resultaten scholenbevraging BIVV').

The screenshot shows a mobile or tablet view of the BIVV website. The layout is more condensed. The navigation bar is at the top. Below it is a green header graphic. The main content area contains the same sections as the desktop version: Rijgeschiktheid, Onderzoeken & statistieken, Vrijwilligers, Verkeersveiligheidsbarometer, Via Secura, and Recente artikelen. The images and text are scaled down to fit the smaller screen.

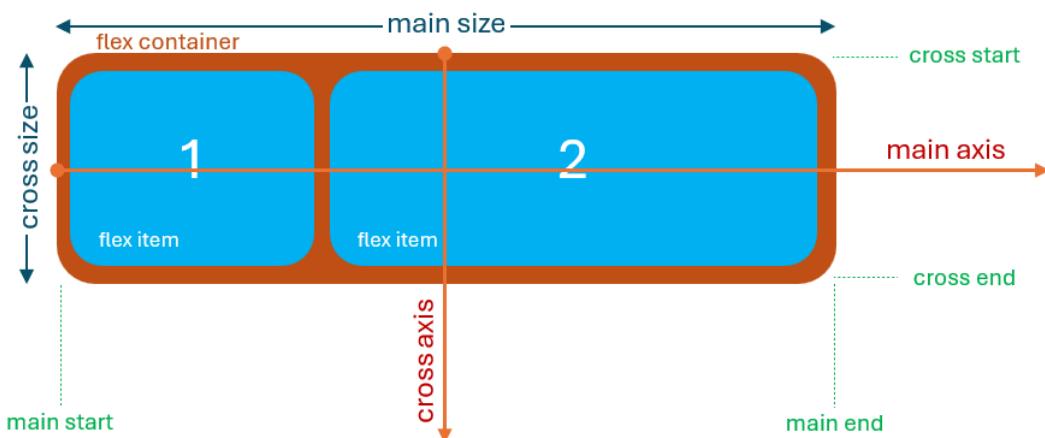
### 11.3 Grids

Het schrijven van CSS voor responsive webdesign kan heel uitgebreid en complex zijn. Op de markt zijn er gelukkig heel wat hulpmiddelen (sjablonen, grids) die je via eigen ingebouwde classes flink vooruit helpen. In de volgende hoofdstukken komen **Flexbox**, **css grid** en **Bootstrap** aan bod.

## 12 FLEXBOX

Flexbox is een opmaakmodel voor het vlot positioneren van verschillende items binnen een container in functie van de beschikbare ruimte.

Een flex **container** bevat flex **items** die tekst, illustraties... bevatten. In de illustratie hieronder zie je zo'n container, met daarin 2 items naast elkaar. Met een reeks CSS-parameters kan je die items (her)positioneren, afhankelijk van de gebruikte device.



### 12.1 Eigenschappen op container-niveau

De vogels hiernaast zitten elk in een container die van de class **item** voorzien is. Alle items/vogels samen zitten in een container met de class **container**:

```
<div class='container'>
    <div class='item'><img src='vogel1.jpg'></div>
    <div class='item'><img src='vogel2.jpg'></div>
    <div class='item'><img src='vogel3.jpg'></div>
    <div class='item'><img src='vogel4.jpg'></div>
    <div class='item'><img src='vogel5.jpg'></div>
</div>
```

Het is normaal dat de vogels onder elkaar komen. **<div>** tags verschijnen immers standaard onder elkaar. Je weet waarom?

Plaats je nu in CSS de class **container** op **display: flex**, dan komen de items in die container standaard naast elkaar (zie hieronder).

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



Dit komt door de **flex-direction** die standaard op **row** is ingesteld. Hierdoor verschijnen de items naast elkaar. Met **column** als flex-direction worden de items onder elkaar weergegeven.

```
.container {
  display: flex;
  flex-direction: column;
}
```

Naast **row** en **column**, kan je ook **row-reverse** en **column-reverse** als flex-direction instellen. Hieronder het resultaat met **row-reverse**.



De items worden zoveel mogelijk naast elkaar geforceerd. De eigenschap **flex-wrap** brengt daar verandering in en schuift vogel 4 onder vogel 1 als er onvoldoende ruimte in de container overblijft om vogel 4 naast vogel 3 te plaatsen. Zie hiernaast. (**flex-wrap: wrap**).



Je hebt bij flex-wrap keuze uit:

- **nowrap** (standaard);
- **wrap**: zie hoger.
- **wrap-reverse**: zie voorbeeld hiernaast.



De eigenschap **flex-flow** combineert de direction en de wrap instellingen, bv.:

```
.container {
  flex-flow: row wrap;
}
```

Met **justify-content** verdeel je de vrije ruimte rond de items. Flex-wrap staat hier op wrap ingesteld.

- **justify-content: flex-start** (standaard). De elementen worden aan het begin van de container geplaatst.
- **justify-content: flex-end**. De elementen worden aan het einde van de container geplaatst.
- **justify-content: center**. De elementen worden gemiddeld binnen de container

In geen van deze drie gevallen, wordt ruimte tussen de items toegevoegd. Dat gebeurt wel met de opties hieronder die de vrije ruimte verdelen.



Naast **justify-content** is er ook **align-content**. Waar justify-content de vrije ruimte horizontaal verdeelt, doet align-content dat verticaal. Dit werkt uiteraard enkel als er voldoende items zijn die via wrap over verschillende *rijen* verdeeld worden.

Een andere eigenschap van een flex container is **align-items**. In het voorbeeld hieronder zie je drie items naast elkaar; het derde item met tekst neemt meer ruimte in (width: 40%).

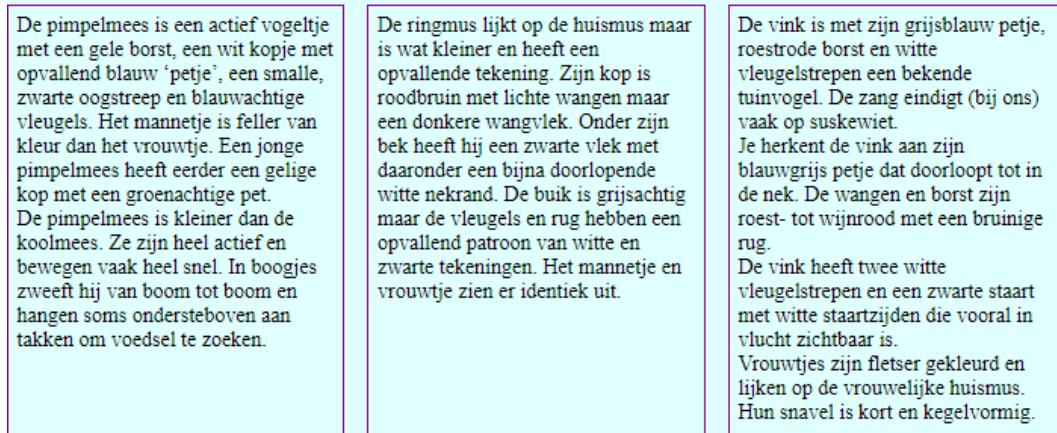
Bekijk nu de verticale uitlijning:



In het voorbeeld dat nu volgt worden drie items met verschillende lengte tekst naast elkaar weergegeven.

```
.container {  
    display: flex;  
    flex-direction: row;  
    flex-wrap: wrap;  
    align-items: stretch;  
    justify-content: space-between;  
}  
.item {  
    width: 30%;  
    padding: 0.4em;  
    border: solid 1px darkmagenta;  
}
```

De **align-items: stretch** zorgt ervoor dat de drie items dezelfde hoogte krijgen. Met **justify-content: space-between** wordt de resterende ruimte horizontaal gelijkmatig tussen de items verdeeld.



Naast **justify-content** en **align-content** die ruimte verdelen tussen items, kan je met de **gap** eigenschap uitdrukkelijk ruimte tussen items instellen.

- **gap: 10px;** zorgt voor 10px ruimte tussen de items, zowel naast als onder elkaar.
- **gap: 15px 30px;** zorgt voor 15px ruimte tussen de rijen met items en 30px ruimte tussen de kolommen met items.
- **row-gap: 20px;** zorgt voor 20px ruimte tussen items onder elkaar.

- **column-gap: 25px;** zorgt voor 25px ruimte tussen items naast elkaar.



Het gaat hier telkens om minimumruimte. De ruimte kan groter zijn via de instellingen bij justify-content of align-content.

## 12.2 Eigenschappen op item-niveau

De items verschijnen normaal in hun logische volgorde, hieronder links van vogel 1 tot vogel 5. Met de item-eigenschap **order** kan je aan de volgorde sleutelen. Standaard hebben alle items 0 als order. Hoe hoger je de order instelt, hoe later het item wordt weergegeven.



Standaard is de order 0 bij alle items.

```
.vogel2 {order: 1;}  
.vogel3 {order: 3;}
```

Met **align-self** kan je voor individuele items afwijken van de uitlijning die bv. ingesteld is via align-items.

In het voorbeeld hiernaast is op de container **align-items: flex-start;** ingesteld. Voor het tweede item wijken we hiervan af via **align-self: flex-end;** ingesteld via een class op dat item.

De **flex-grow** eigenschap bepaalt hoeveel meer ruimte een flex item zal innemen binnen een container. In het voorbeeld hiernaast is de breedte van de items via **flex-basis: 25%;** ingesteld. Op het tweede item is **flex-grow: 2;** ingesteld. Dat verklaart waarom dit item dubbel zo breed is.

Wil je bv. het tweede item in een container aanspreken, dan kan je daar een specifieke class voor voorzien. Nog beter is hier de pseudoclass nth-of-type() te gebruiken:

```
.container div:nth-of-type(2)
```

Bekijken we met een voorbeeld nu ook nog de werking van **flex-shrink**.

**De pimpelmees** is een actief vogeltje met een gele borst, een wit kopje met opvallend blauw 'petje', een smalle zwarte oogstreep en blauwachtige vleugels. Het mannetje is feller van kleur dan het vrouwtje. Een jonge pimpelmees heeft eerder een gele kop met een groenachtige pet.

**De ringmus** lijkt op de huismus maar is wat kleiner en heeft een opvallende tekening. Zijn kop is roodbruin met lichte vingers maar een donkernekkend. Onder zijn bek heeft hij een zwarte vlek met daaronder een bijna doorlopende witte nekrand. De buik is grijsachtig maar de vleugels en rug hebben een opvallend patroon van witte en zwarte tekeningen. Het mannetje en vrouwtje zien er identiek uit.

**De vink** is met zijn grijtblauw petje, roestrode borst en witte vleugelstrepen een bekende tuinvogel. De zang eindigt (bij ons) vaak op suskeviet.

Je herkent de vink aan zijn blauwgrauw petje dat dooloopt tot in de nek. De vingers en borst zijn roest- tot wijnrood met een bruinige rug.

De vink heeft twee witte vleugelstrepen en een zwarte staart met witte staartzijden die vooral in vlucht zichtbaar is.

Vrouwvinkjes zijn flester gekleurd en lijken op de vrouwelijke huismus. Hun snavel is kort en kegelvormig.

**De pimpelmees** is een actief vogeltje met een gele borst, een wit kopje met opvallend blauw 'petje', een smalle zwarte oogstreep en blauwachtige vleugels. Het mannetje is feller van kleur dan het vrouwtje. Een jonge pimpelmees heeft eerder een gele kop met een groenachtige pet.

**De ringmus** lijkt op de huismus maar is wat kleiner en heeft een opvallende tekening. Zijn kop is roodbruin met lichte vingers maar een donkernekkend. Onder zijn bek heeft hij een zwarte vlek met daaronder een bijna doorlopende witte nekrand. De buik is grijsachtig maar de vleugels en rug hebben een opvallend patroon van witte en zwarte tekeningen. Het mannetje en vrouwtje zien er identiek uit.

**De vink** is met zijn grijtblauw petje, roestrode borst en witte vleugelstrepen een bekende tuinvogel. De zang eindigt (bij ons) vaak op suskeviet.

Je herkent de vink aan zijn blauwgrauw petje dat dooloopt tot in de nek. De vingers en borst zijn roest- tot wijnrood met een bruinige rug.

De vink heeft twee witte vleugelstrepen en een zwarte staart met witte staartzijden die vooral in vlucht zichtbaar is.

Vrouwvinkjes zijn flester gekleurd en lijken op de vrouwelijke huismus. Hun snavel is kort en kegelvormig.

```

<div class="container">
  <div class="item item1">Item 1</div>
  <div class="item item2">Item 2</div>
  <div class="item item3">Item 3</div>
</div>

.container {
  display: flex;
  width: 800px;
  border: 2px solid black;
}
.item {
  flex-basis: 400px;
  padding: 10px;
  color: white;
}
.item1 {
  background-color: darkred;
  flex-shrink: 1;
}
.item2 {
  background-color: darkviolet;
  flex-shrink: 2;
}
.item3 {
  background-color: darkblue;
  flex-shrink: 0;
}

```

Zonder flex-shrink krimpen de items op een gelijke manier. De flex-basis is 400px en binnen een container van 800px is krimpen de items gelijkmatig tot 266,66px. De standaard waarde van flex-shrink ik immers 1.



Met de flex-shrink instellingen hierboven krijgen we deze weergave:

- Item 3 behoudt haar volle 400px doordat flex-shrink op 0 is ingesteld.
- Item 2 krimpt dubbel t.o.v. item 1. Item 1 neemt hier 266,66px van de resterende 400px in. Item 2 neemt 133,33px van die 400px in.



Mocht de container 1200px of meer breed zijn, dan hebben de flex-shrink instellingen geen effect omdat de drie items met flex-basis 400px netjes in de container passen.

De eigenschappen flex-grow, flex-shrink en flex-basis worden in één eigenschap **flex** gebundeld:

#### **flex: flex-grow [flex-shrink] [flex-basis]**

**flex: 2 1 50%** is dus een flex-grow van 2, een flex-shrink van 1 en 50% flex-basis.

## 13 CSS GRID

Naast Flexbox is ook CSS Grid dat zeer geschikt is voor meer complexe lay-out omdat je hier tegelijk rijen en kolommen kunt aansturen. Bij Flexbox werk je ofwel met rijen ofwel met kolommen. Bij CSS Grid werk je echt in een raster. Nog anders gezegd: Flexbox is één-dimensioneel, CSS Grid is twee-dimensioneel.

Je kunt in dezelfde pagina flexbox en CSS Grid door elkaar gebruiken.

In dit hoofdstuk focussen we op CSS Grid. Ook hier werken we met **grid containers** die **grid items** bevatten.

```
<div class="container">
  <div class="item item-1"> </div>
  <div class="item item-2"> </div>
  <div class="item item-3"> </div>
</div>
```

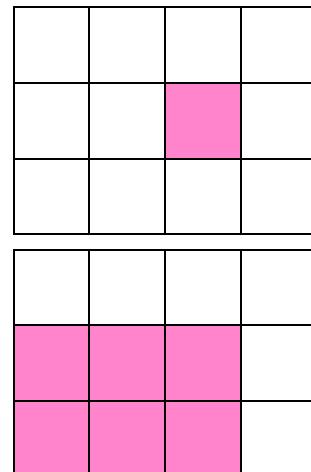
Binnen zo'n grid verdelen scheidingslijnen – **Grid Lines** – de ruimte zowel horizontaal (row grid lines) als verticaal (column grid lines). De ruimte tussen twee aaneengrenzende column grid lines en twee aaneengrenzende row grid lines vormt een **grid cell**.

In het voorbeeld hiernaast wordt de gemaakte cel afgebakend door row grid line 2 en 3 en door column grid line 3 en 4.

De ruimte tussen twee aaneengrenzende grid lines is de **grid track**. De **grid area** is de ruimte die afgebakend wordt door 4 grid lines. Een grid area kan één tot veel cellen bevatten.

In het voorbeeld hiernaast wordt de grid area afgebakend door column grid lines 1 en 4 en row grid lines 2 en 4. De grid area bevat 6 grid cells.

Zowel voor de grid container als voor de grid items in die container zijn er heel wat CSS-eigenschappen die je kunt instellen om de inhoud van je webpagina deftig te positioneren.



### 13.1 CSS Grid container eigenschappen

Voor het maken van een grid container gebruik je de display eigenschap:

- **display: grid**
- **display: inline-grid**

### 13.1.1 Verdeling in kolommen en rijen

Met de **grid-template-columns** en **grid-template-rows** eigenschappen geef je de track-size op. Dat kan in verschillende eenheden, o.a. een vast pixel aantal, een percentage of met de interessante **fr**-eenheid (fraction), waarover verder meer.

De **column-gap** en **row-gap** eigenschappen regelen de ruimte tussen de items. Ze worden niet toegevoegd vóór het eerste item noch na het laatste item.

Je kan ook de **gap** eigenschap gebruiken en in één eigenschap row gap en column gap opgeven, bv. **gap: 10px 15px** zorgt voor 10px row gap en 15px column gap.

```
.container {  
    display: grid;  
    grid-template-columns: 150px 300px 100px 1fr;  
    grid-template-rows: 100px 200px 300px;  
    column-gap: 10px;  
    row-gap: 10px;  
}
```

In een container met 12 items is dit het resultaat. De breedte van de eerste 3 kolommen staat vast. De vierde kolom neemt de resterende ruimte in.

1	2	3	4
5	6	7	8
9	10	11	12

```
.container {  
    display: grid;  
    grid-template-columns: 100px repeat(3, 1fr);  
    grid-template-rows: repeat(3, 100px);  
    column-gap: 10px;  
    row-gap: 10px;  
}
```

1	2	3	4
5	6	7	8
9	10	11	12

Bemerк hier het gebruik van de **repeat()** functie. De eerste kolom is 100px breed. De overige ruimte wordt over de andere 3 kolommen gelijkmatig verdeeld. Idem voor de rijhoogte. De twee statements hieronder hebben hetzelfde resultaat.

- `grid-template-columns: 1fr 1fr 1fr 1fr;`
- `grid-template-columns: repeat(4, 1fr);`

Stel even dat je vier kolommen van gelijke breedte wilt, dan zijn er twee opties:

- `grid-template-columns: repeat(4, 1fr);`
- `grid-template-columns: repeat(4, 25%);`

De eerste optie is hier de betere omdat die rekening houdt met de ingestelde column-gap, terwijl de %-eenheid dat niet doet en er dus nog horizontaal gescrelld moet worden.

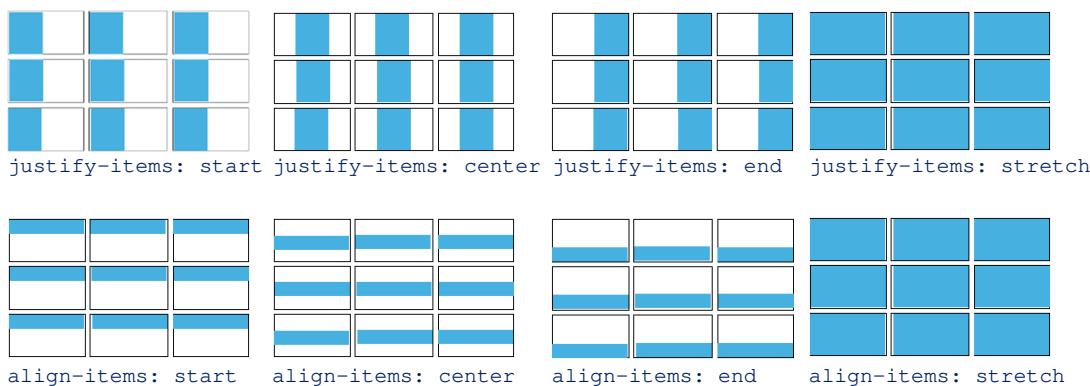
Dat zie je hiernaast duidelijk met de achtergrond die de beschikbare breedte arceert. De tweede container met 4 x 25% loopt uit door de column-gap die niet in die 25% zit.

1	2	3	4
5	6	7	8
9	10	11	12
<hr/>			
1	2	3	4
5	6	7	8
9	10	11	12

Te onthouden dus, die **fr** eenheid!

### 13.1.2 Uitlijning in de cellen

De uitlijning die je op de container instelt, geldt in principe voor alle items in die container. Met **justify-items** regel je de horizontale uitlijning. **Align-items** zorgt voor de verticale uitlijning binnen de cellen.

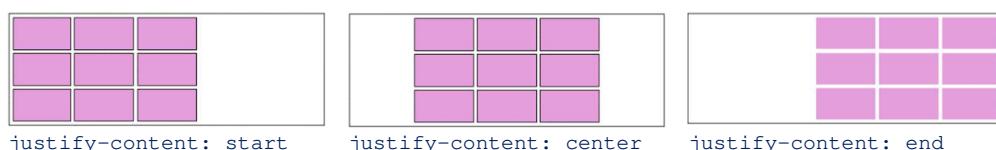


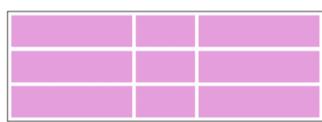
Met **place-items** combineer je align-items en justify-items, bv. **place-items: center end**.

Met **justify-self** en **align-self** kan je in individuele cellen van de ingestelde uitlijning afwijken.

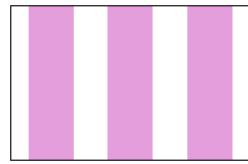
### 13.1.3 Uitlijning binnen de container

Als de totale ruimte die je grid inneemt kleiner is dan de beschikbare ruimte in de container, dan kan je die grid uitlijnen binnen de container. Met **justify-content** doe je dat horizontaal.





`justify-content: stretch`



`justify-content: space-around`   `justify-content: space-between`



`justify-content: space-evenly`

**Align-content** werkt parallel met dezelfde parameters, maar dan verticaal.

Met **place-content** combineer je align-content en justify-content, bv. **place-items: center end**.

## 14 BOOTSTRAP

**Bootstrap** is een vaak gebruikt gratis framework voor het maken van responsive websites op basis van HTML, CSS en javascript.

Installeer bootstrap van de website [getbootstrap.com](http://getbootstrap.com) en volg de instructies. We gaan er verder vanuit dat bootstrap geïnstalleerd is. Als alternatief kan je bootstrap ook vanuit een CDN gebruiken (Content Delivery Network) en hoef je het niet te downloaden.

Is bootstrap geïnstalleerd, dan kan je de vele classes en scripts gebruiken die bootstrap toevoegt in de mappen css en js. Wij gebruiken **Bootstrap 5.2** in deze cursus.

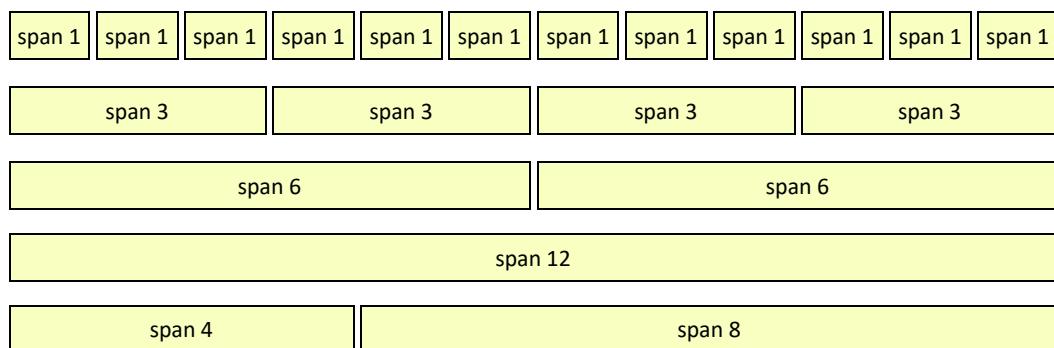
Binnen je `<body>`-tag plaats je alvast een `<div>`-tag met één van volgende classes:

- `<div class="container">`: je werkt met een container met vaste breedte
- `<div class="container-fluid">`: je gebruikt de volledige viewport

### 14.1 Het bootstrap grid-systeem

Binnen een container komen rijen (`class="row"`) voor die elk uit één of meer kolommen bestaan. De classes die voor de kolommen gebruikt worden, bepalen hoe de informatie weergegeven wordt op de verschillende apparaten.

Bootstrap werkt met een denkbeeldig grid (raster) dat uit 12 kolommen naast elkaar bestaat. Die kolommen kan je via classes op verschillende manieren groeperen:



Het bootstrap grid systeem werkt met classes die elk hun *breakpoint* hebben. Er zijn in Bootstrap 5 zes niveaus (*tiers*): van extra small tot extra extra large. Het breakpoint tussen extra small en small bv. ligt op 576px.

Voor de verschillende *tiers* kan dus een andere weergave van de kolommen ingesteld worden, zoals je verder zult merken via de `.col` classes.

Bootstrap <b>5.2</b>	Extra small XS <576px	Small SM >=576px	Medium MD >=768px	Large LG >=992px	Extra large XL >=1200px	Extra extra Large XXL >=1400px
class prefix	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-	.col-xxl-

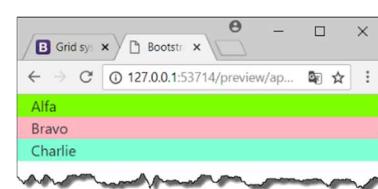
Je kunt gewoon de class `col` gebruiken zoals in het voorbeeld hieronder. Op elk apparaat krijg je hier drie kolommen van gelijke breedte, onafhankelijk van de totaal beschikbare breedte. Op een smart phone krijg je hier dus hele smalle kolommetjes:

```
<div class="container">
<div class="row">
    <div class="col">Alfa</div>
    <div class="col">Bravo</div>
    <div class="col">Charlie</div>
</div>
</div>
```



In het voorbeeld hieronder krijg je met de class `col-sm` vanaf 576px beschikbare breedte (**sm**) drie gelijke kolommen naast elkaar. Is de breedte kleiner, dan geldt de class niet en verschijnen de kolommen onder elkaar omdat ze elk de volle beschikbare breedte innemen.

```
<div class="container">
<div class="row">
    <div class="col-sm">Alfa</div>
    <div class="col-sm">Bravo</div>
    <div class="col-sm">Charlie</div>
</div>
</div>
```



In het voorbeeld hieronder hebben de kolommen niet langer dezelfde breedte. De eerste kolom is dubbel zo breed (6) dan de twee volgende kolommen (elk 3). Samen 12, het maximum binnen het grid dat gebruikt wordt. Bij extra kleine devices spelen de classes niet en komen de kolommen onder elkaar omdat ze dan elk de volle breedte innemen.

```
<div class="container">
<div class="row">
    <div class="col-sm-6">Alfa</div>
    <div class="col-sm-3">Bravo</div>
    <div class="col-sm-3">Charlie</div>
</div>
</div>
```



De zes niveaus (xs tot xxl) kolom-classes zijn voorgedefinieerd in de bootstrap-CSS-bestanden aanwezig natuurlijk. Met de class `.col-sm-3` neemt je kolom één/vierde van de beschikbare breedte (12 kolommen) in.

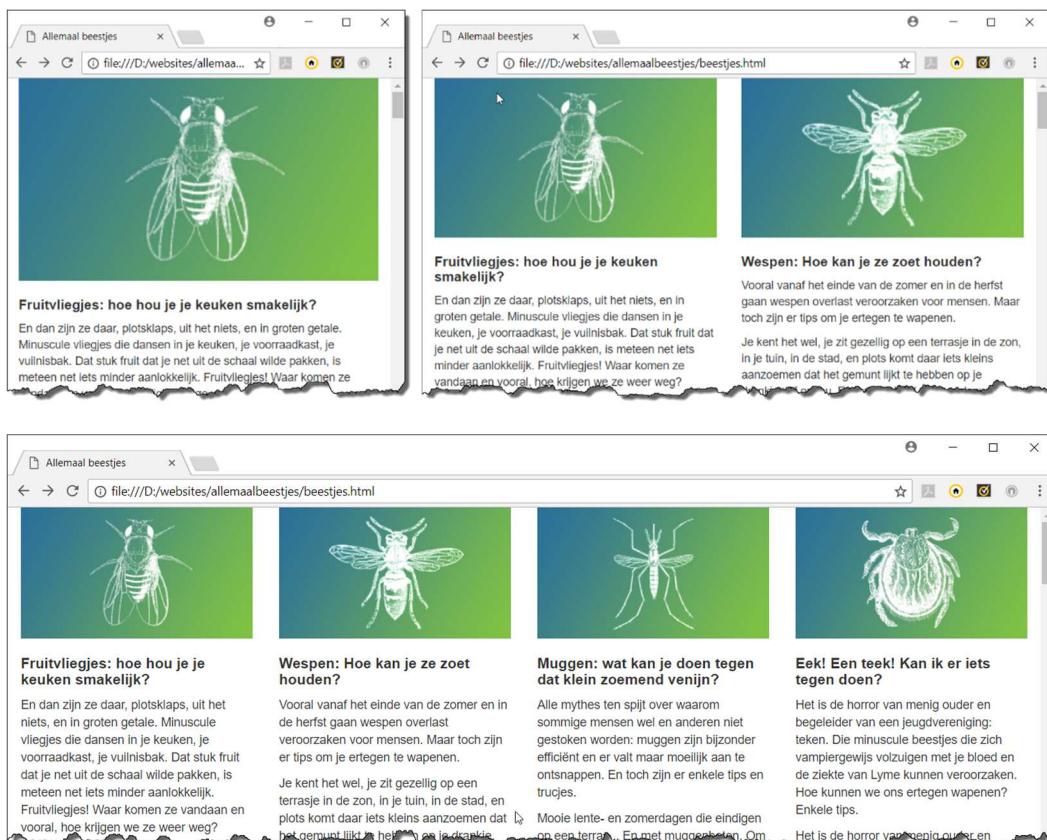
Binnen de kolommen is standaard links en rechts 15 px ruimte voorzien.

Je kunt verschillende classes in één container combineren, bv.:

```
<div id="fruitvlieg" class="col-xs-12 col-sm-6 col-md-3">
```

Op een device met beperkte breedte neemt de container de volle breedte in: alle 12 kolommen (xs-12), zie hieronder links. Is wat meer ruimte, dan komen twee containers naast elkaar (sm-6) en bij een bredere viewport komen 4 kolommen naast elkaar (md-3). Dit laatste geldt dan ook voor bredere viewports omdat geen specifieke **col-lg** of **col-xl** class is ingesteld.

Zorg dus altijd ervoor dat je in totaal 12 kolommen verdeelt. Als je meer dan 12 kolommen in één rij opneemt, dan komen de extra kolommen in één geheel onder de vorige kolommen terecht in plaats vanernaast.



In **BOOTSTRAP.CSS**, een bestand met 9000 (!) regels CSS-code, wordt met volgende breakpoints gewerkt die elk een vaste breedte geven aan de container waarbinnen gewerkt wordt.

```
@media (min-width: 576px) {
    .container {max-width: 540px; }
}

@media (min-width: 768px) {
    .container {max-width: 720px; }
}

@media (min-width: 992px) {
    .container {max-width: 960px; }
} ...
```

Door de class **.container** in **.container-fluid** te wijzigen, schakel je meteen op weergave over de volle beschikbare breedte over.

## 14.2 Extra ruimte tussen kolommen

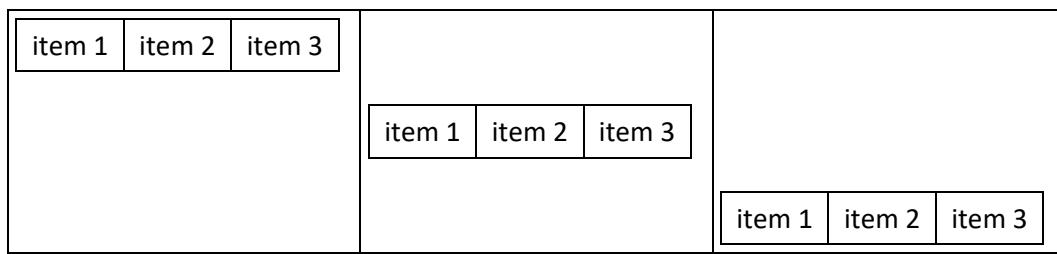
Met de class `.offset-sm-*` kan je de linkermarge van een kolom met het bij \* opgegeven aantal kolommen vergroten. Zie illustratie hieronder waarbij links ruimte vrij blijft aan de rechterkant en rechts die ruimte tussen beide kolommen voorzien wordt.



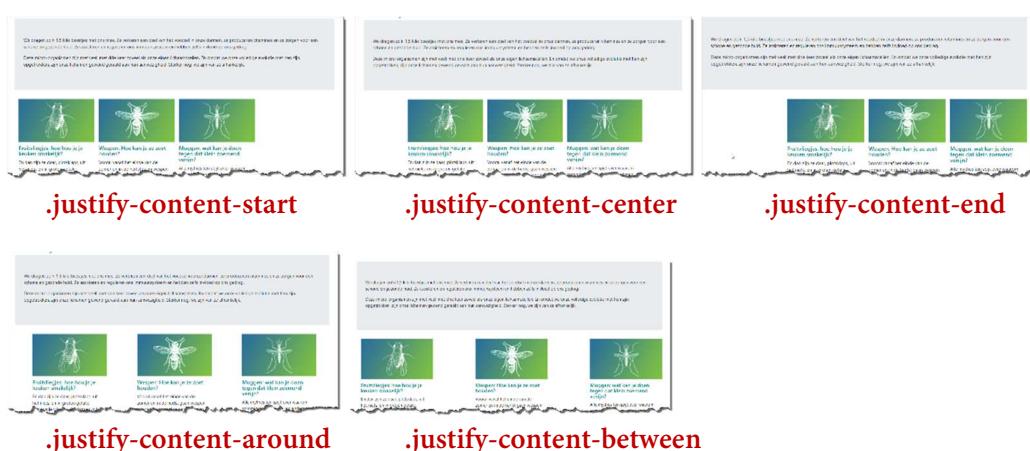
## 14.3 Uitlijning

Op basis van volgende bootstrap-classes kan je binnen een container verticaal uitlijnen. Je voegt de class op de container met `.row` class toe, bv.:

```
<div id="info" class="row align-content-center">
```



Ook horizontaal beschik je over verschillende classes die je aan de container met `.row` class kunt toevoegen om niet-benutte ruimte te verdelen. Hieronder een voorbeeld van de verschillende classes.



## 14.4 Volgorde van de elementen

De volgorde waarin bv. kolommen naast elkaar geplaatst worden, is normaal hun fysieke volgorde, maar met de `.order` class kan je dat beïnvloeden, algemeen of afhankelijk van de breedte van je viewport:

- Je kunt bv. `.order-1` gebruiken om een kolom vóór `.order-2` te plaatsen, ook al staat de kolom met `.order-1` fysiek verder/lager dan de kolom met `.order-2`. Je kiest van 1 tot 5.
- Je kunt de volgorde per type device instellen: `.order-sm-1`, `.order-md-1`...

## 14.5 Bootstrap typografie

Bootstrap werkt standaard met een lettergrootte van 16px en een regelhoogte van 1.5. De tekstkleur is niet volledig zwart (#212529) en als lettertype familie geldt "Helvetica Neue", Helvetica, Arial, sans-serif.

Boven paragrafen is geen witruimte (`margin-top: 0`). Onder paragrafen hebben we een `margin-bottom: 1rem`, dus standaard 16px.

De grootte van de koppen varieert van 2.5rem (h1) tot 1rem (h6). Binnen een kop kan de `<small>`-tag gebruikt worden om bijkomende titeltekst iets bescheidener weer te geven.

Verder zijn er verschillende classes ingebouwd om tekst een bepaalde weergave (kleur) mee te geven: `.text-muted`, `.text-primary`, `.text-success`, `.text-info`, `.text-warning` en `.text-danger` of van een bepaalde achtergrondkleur te voorzien: `.bg-primary`, `.bg-success`, `.bg-info`, `.bg-warning`, en `.bg-danger`.

Ook volgende classes, waarvan je meestal wel vermoedt wat ze doen, zijn beschikbaar: `.small`, `.text-left`, `.text-center`, `.text-right`, `.text-justify`, `.text-lowercase`, `.text-uppercase`, `.text-capitalize`...

## 14.6 Tabellen

Binnen bootstrap zijn verschillende classes ingebouwd om tabellen snel een fraaie look te geven. De basis-class `.table` zorgt voor horizontale tussenlijnen, terwijl `.table-striped` oneven rijen van een lichtgrijze achtergrond voorziet.

	Vlaanderen	Wallonië	België
Runderen	706.282	205.088	911.370
Varkens	10.480.902	700.432	11.181.334
Schapen	126.577	14.528	141.105
Geiten	12.719	82	12.801
Paarden	2.252	3.802	6.054

`.table`

	Vlaanderen	Wallonië	België
Runderen	706.282	205.088	911.370
Varkens	10.480.902	700.432	11.181.334
Schapen	126.577	14.528	141.105
Geiten	12.719	82	12.801
Paarden	2.252	3.802	6.054

`.table-striped`

Verder zijn er **.table-bordered**, **.table-hover**, **.table-condensed** en **.table-responsive**. Deze laatste class zorgt ervoor dat tabellen op devices onder 992px horizontaal scrollen als dat nodig is.

Met de **.table-dark** class voeg je een zwarte achtergrond aan de tabel toe, zoals hiernaast geïllustreerd.

	Vlaanderen	Wallonië	Brussel
Runderen	706 282	205 088	911 370
Varkens	10 480 902	700 432	11 181 334
Schapen	126 577	14 528	141 105
Geiten	12 719	82	12 801
Paarden	2 252	3 802	6 054

## 14.7 Bootstrap en illustraties

Bij afbeeldingen kan je enkele extra classes toepassen:



gewone afbeelding

.rounded

.rounded-circle

.img-thumbnail

Voeg je de **.float-left** of **.float-right** class aan een afbeelding toe, dan wordt de afbeelding netjes links of rechts geplaatst en loopt de tekst er mooi rond.

## 14.8 Alerts

Je kunt bepaalde tekstblokken extra laten opvallen door de **.alert** class te gebruiken in combinatie met één van volgende classes: **.alert-success**, **.alert-info**, **.alert-warning**, **.alert-danger**, **.alert-primary**, **.alert-secondary**, **.alert-light** of **.alert-dark**.

**Opgelet.** Vermijd zwaar werk in volle hitte en drink voldoende. Zorg dat je heel goed ingesmeerd bent met een zonnercrème van goede kwaliteit als je toch in de zon loopt.

Verder kan je aan de container rechtsboven een 'sluit-knop' x toevoegen (&times; code voor de x) door de class **.alert-dismissible** aan de container toe te voegen en door in de container een knop toe te voegen met **class="close"** en **data-bs-dismiss="alert"**. In plaats van een knop kan je ook een hyperlink gebruiken. Klik op de x en de container verdwijnt.

```
<div class="alert alert-warning alert-dismissible">
<button type="button" class="close" data-bs-dismiss=
"alert">&times;</button>
<strong>Opgelet</strong>. Vermijd ...</div>
```

## 14.9 Bootstrap knoppen

Ook hier verschillende classes om knoppen snel op te maken. De classes kunnen toegepast worden op `<a>`, `<input>` en `<button>`-tags.

Enkele classes: `.btn`, `.btn-primary`, `.btn-success`, `.btn-info`, `.btn-warning`, `.btn-danger`, `.btn-link`, `.btn-dark`...

Gecombineerd met de `.btn` class krijgen we in de geïllustreerde voorbeelden mooie afgelonde knoppen.

```
<input type="submit" class="btn btn-success"  
value="success">
```

Je kunt ook nog de classes `.btn-lg`, `.btn-md`, `.btn-sm` en `.btn-xs` toevoegen om de grootte van de knoppen te bepalen.



En wat te denken van knoppen die van een *outline* voorzien zijn, omranding zonder achtergrondkleur, zoals hieronder.



Hier voor beschik je over een aantal `.btn-outline-*` classes zoals de `.btn-outline-primary` class. Beweeg je over de knop, dan krijgt die een mooie achtergrond.

```
<button type="button" class="btn btn-outline-  
primary">Bestel hier</button>
```

Verder ook nog even de classes `.active` en `.disabled` vermelden waarmee je knoppen als respectievelijk geactiveerd of niet-beschikbaar kunt weergeven.

Met de class `.btn-group` kan je groepen knoppen als één geheel weergeven.

```
<div class="btn-group">  
<input type="submit" class="btn btn-info btn-xs"  
value="Cursusaanbod">  
<input type="submit" class="btn btn-info btn-xs"  
value="Praktische informatie">  
<input type="submit" class="btn btn-info btn-xs"  
value="Contact">  
</div>
```

Wil je de knoppen onder elkaar, gebruik dan de class `.btn-group-vertical`.



## 14.10 Bootstrap icons

Er zijn verschillende manieren om icons in je webpagina's te gebruiken. Bootstrap zelf beschikt over meer dan 1000 icons die op verschillende manieren te gebruiken zijn.

Het makkelijkst is door deze link die verwijst naar een online css-bestand in de header van je webpagina's op te nemen:

```
<link rel="stylesheet"
      href="https://cdn.jsdelivr.net/npm/bootstrap-
      icons@1.9.1/font/bootstrap-icons.css">
```

Voor elke icon is nu een class voorzien: .bi-\* waarbij \* de naam van de class is.  
Je kunt o.a. via css de grootte en kleur aanpassen:

```
<i class="bi bi-sign-stop-fill" style="font-size:
5rem; color: red;"></i>
```



## 14.11 Bootstrap CSS variables

Binnen Bootstrap kan je ook een reeks variabelen gebruiken die altijd met de prefix **bs-** beginnen om ze te onderscheiden van andere externe variabelen. Elke CSS variabele begint sowieso met --.

Om die variabelen in te stellen gebruik je de `:root` sectie bovenaan in je CSS-bestand.

```
:root {
--bs-pink: #D63384;
--bs-border-width: 1px;
...
}

h1 {
color: var(--bs-pink);
}
input {
background-color: var(--bs-pink);
}
```

Wil je nu de kleur van én de koppen én de achtergrond van tekstvakken aanpassen, dan volstaat het in de root-sectie de kleur van de variabele --bs-pink aan te passen.

## 14.12 Uitklapbare tekst

Als startvoorbeeld voorzien we een knop *Meer info*. Van zodra je op de knop klikt, verschijnt de achterliggende inhoud.

Je gebruikt de `.collapse` class in de informatietekst die afwisselend verborgen of getoond wordt.

Aan de knop voeg je twee attributen toe: `data-toggle="collapse"` en een verwijzing met `data-target` naar de uit te klappen tekst (via ID). Bij `<a>`-tags kan je het `href`-attribuut gebruiken i.p.v. `data-target`.

**Allemaal beestjes**

[Meer info](#)

We dragen zo'n 1,5 kilo beestjes met ons mee om ons te beschermen en reguleren ons immuunsysteem.

Deze micro-organismen zijn niet veel: met de hand geraakt aan hun aanwezigheid, werken nog...

```

<button data-bs-toggle="collapse" data-bs-target="#info">Meer info</button>
<div id="info" class="collapse">
<p>We dragen ....</p>
</div>
</div>

```

### 14.13 Accordion

Met een **accordion** geef je op een beknopte plaats toch veel informatie weer omdat slechts één panel tegelijk zichtbaar is. Door een panel te activeren, sluit het vorige.

Bestudeer aandachtig de HTML-code hieronder en je zult ongetwijfeld snel door hebben hoe je zo'n accordion in elkaar puzzelt.

The screenshot shows a user interface with three tabs or panels. The first tab, 'De Fruitvlieg', is highlighted with a grey background and white text. It contains a sub-section titled 'Fruitvliegjes: hoe hou je je keuken smakelijk?' with a short text about fruit flies. Below this are two other tabs: 'De wesp' and 'De mug', both in a standard grey font.

```

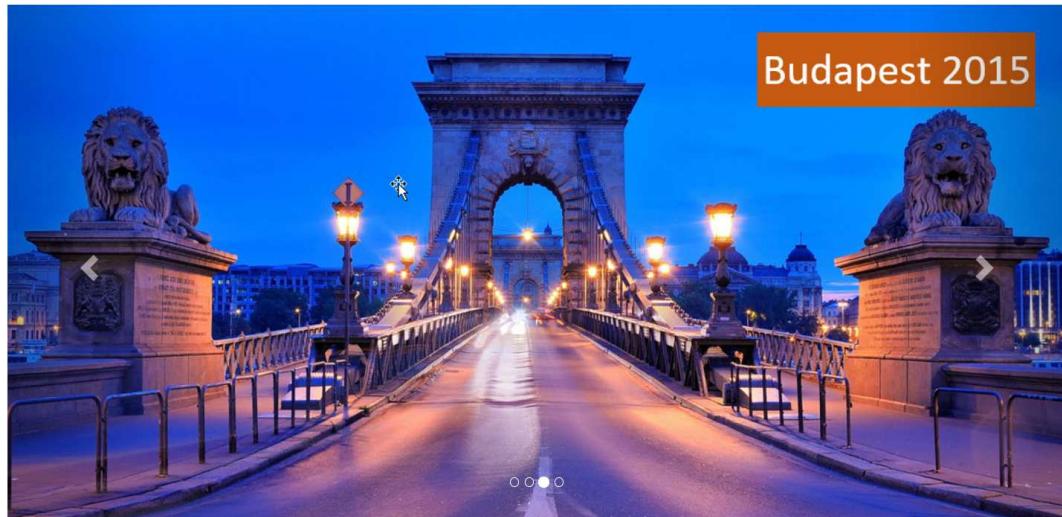
<div class="panel-group" id="accordion">
  <div class="panel panel-default">
    <div class="panel-heading">
      <h4 class="panel-title">
        <a data-bs-toggle="collapse" data-bs-parent=
          "#accordion" href="#collapse1">De Fruitvlieg</a>
      </h4>
    </div>
    <div id="collapse1" class="panel-collapse collapse in">
      <div class="panel-body">... inhoud die uitklapt ...</div>
    </div>
    <div class="panel panel-default">
      <div class="panel-heading">
        <h4 class="panel-title">
          <a data-bs-toggle="collapse" data-bs-parent=
            "#accordion" href="#collapse2">De wesp</a>
        </h4>
      </div>
      <div id="collapse2" class="panel-collapse collapse">
        <div class="panel-body">... inhoud die uitklapt ...</div>
      </div>
    </div>
  </div>
</div>

```

### 14.14 Carrousel

De **Carousel-plugin** zorgt voor handige carrousels (slideshows doorheen foto's) die vaak als banner in websites te zien zijn en waar je vlot doorheen kunt scrollen. Meestal krijg je om de x seconden automatisch telkens een volgende illustratie te zien.

We illustreren met een concreet voorbeeld.



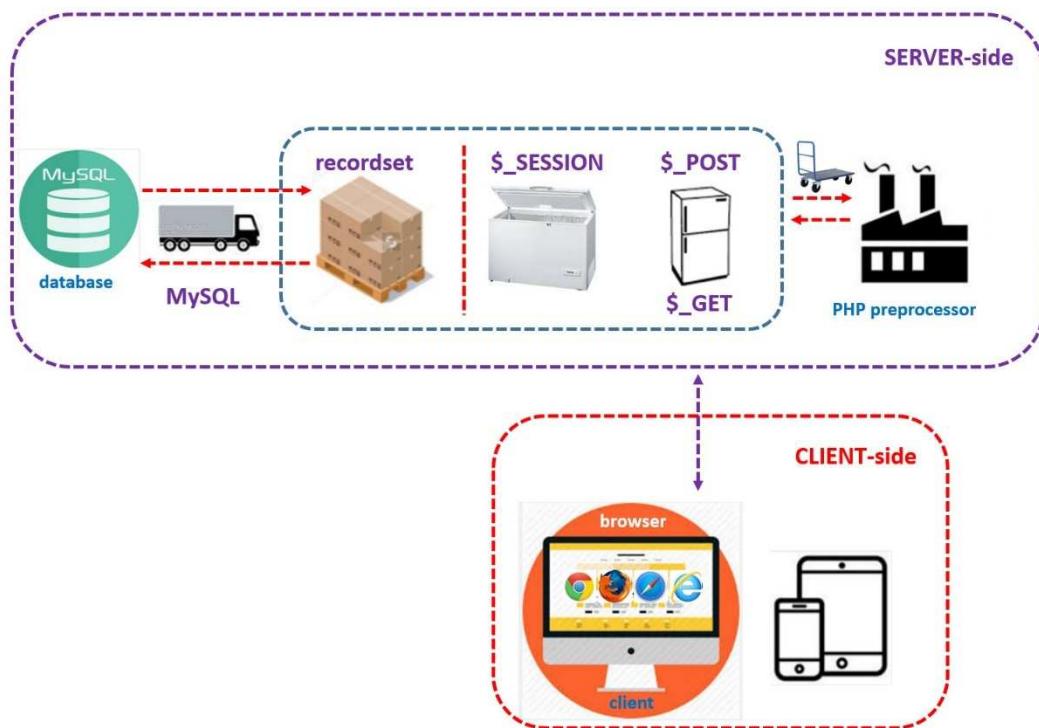
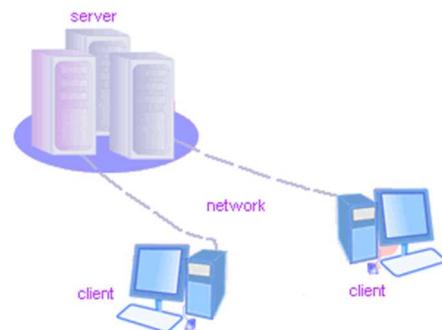
```
<div id="myCarousel" class="carousel slide" data-bs-  
ride="carousel">  
    <!-- Indicators -->  
    <ol class="carousel-indicators">  
        <li data-bs-target="#myCarousel" data-bs-slide-to="0" class="active"></li>  
        <li data-bs-target="#myCarousel" data-bs-slide-to="1"></li>  
        <li data-bs-target="#myCarousel" data-bs-slide-to="2"></li>  
        <li data-bs-target="#myCarousel" data-bs-slide-to="3"></li>  
    </ol>  
  
    <!-- Wrapper for slides -->  
    <div class="carousel-inner">  
        <div class="carousel-item active">  
              
        </div>  
        <div class="carousel-item">  
              
        </div>  
        ...  
    </div>  
    <!-- Left and right controls -->  
    <a class="carousel-control-prev" href="#myCarousel" data-slide="prev">  
        <span class="carousel-control-prev-icon"></span>  
    </a>  
    <a class="carousel-control-next" href="#myCarousel" data-slide="next">  
        <span class="carousel-control-next-icon"></span>  
    </a></div>
```

## 15 INLEIDING OP PHP

Het Internet is een mooi voorbeeld van een client-server model. Alle informatie is op webservers opgeslagen en via je webbrowser kopieer je die informatie naar je client-PC die met het Internet verbonden is, al dan niet draadloos (wireless).

Bij een statische webpagina is de webserver niet veel meer dan een altijd toegankelijke opslagplaats van webpagina's. Als je op het Internet surft, dan worden de opgeroepen webpagina's naar je client PC gekopieerd. Het is je webbrowser die ervoor zorgt dat je iets te zien krijgt: de HTML-codes worden omgezet in een tastbaar resultaat. Maar dat wist je al!

Met PHP is er op zijn minst één stap meer. Bij het aanroepen van een dynamische webpagina vanop je client PC, wordt de servercode in je webpagina eerst op de webserver (server-side) uitgevoerd door de PHP preprocessor. Het resultaat van die uitvoer – HTML-code – wordt naar je client-browser gestuurd die ervoor zorgt dat je de uitvoer op je scherm te zien krijgt. Al dan niet kunnen ook gegevens daarbij uit een database gehaald worden en/of in een database opgeslagen worden.



In deze basiscursus zijn we eigenlijk maar in één PHP-functie geïnteresseerd: **include()**. Om die te kunnen gebruiken moeten we PHP installeren. Dat kan ondermeer gebeuren door XAMPP te installeren. Daarmee heb je meteen ook de beschikking over MySQL voor het aanspreken van je database en verschillende andere tools.

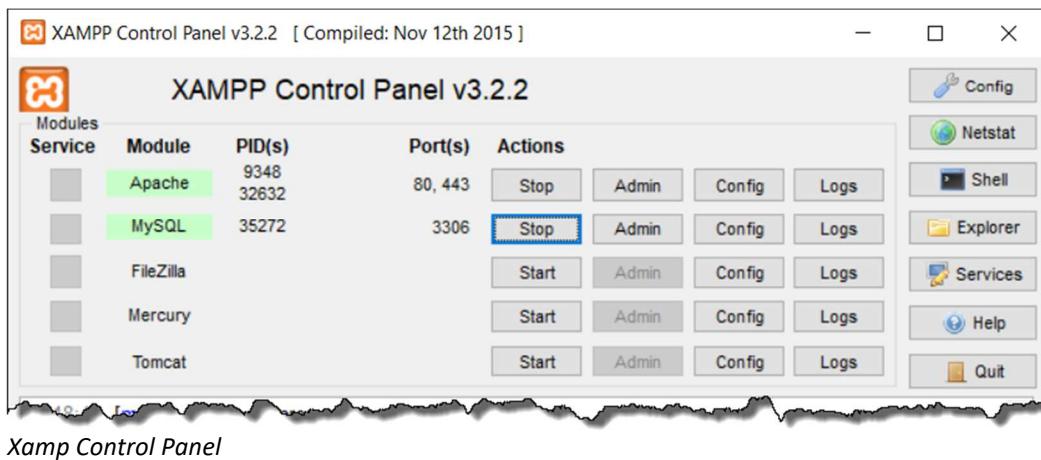
## 15.1 XAMPP installeren

Om te vermijden dat we onze pagina's telkens naar een externe webserver die PHP ondersteunt, moeten uploaden, installeren we zo'n webserver op onze eigen PC. We spelen dus server en client op één en hetzelfde toestel.

In plaats van PHP, MySQL en een webserver afzonderlijk te installeren, maak je best gebruik van een geïntegreerde oplossing als XAMPP. XAMPP bestaat zowel voor Linux, Mac als Windows en omvat onder meer volgende tools:

- Apache, een webserver
- MySQL, onze database
- PHP, de servertaal waarmee we instructies geven
- Perl, een programmeertaal (die gebruiken we niet in deze cursus).

Na de installatie beschik je over een XAMPP Control Panel waarin je kunt controleren of Apache en MySQL draait. Standaard vind je op de C-schijf de map XAMPP met daarin de map HTDOCS, de hoofdmap waarin je webpagina's standaard terecht moeten komen. Vergeet niet eventueel Apache en MySQL te starten vooraleer je aan de slag gaat!



Het installeren van XAMPP is op zich eenvoudig. De installatiebestanden kan je vinden via [www.apachefriends.org](http://www.apachefriends.org). Bekende alternatieven voor XAMPP zijn Easy PHP, WAMP...

Je kan ook voor XAMPP portable opteren en XAMPP USB Lite installeren zodat je op een USB-stick kan werken, maar zorg er zeker voor dat je ook lokaal op je laptop kunt werken.

Normaal gebruikt je webserver poort 80, wat soms tot conflicten leidt als je bv. programma's als Skype of TeamViewer gebruikt. Sluit eventueel die programma's af of probeer Apache via een andere poort te draaien.

## 15.2 Servertest

Je kunt op je toestel nu zowel server als client spelen. Open je webbrowser en surf naar <http://localhost> of <http://127.0.0.1>.

Hiermee kom je in de map HTDOCS terecht en wordt een standaardpagina geopend. Als dat zo is, dan werkt je webserver prima.

### 15.3 PHP-code in je webpagina opnemen

Webpagina's die PHP-code bevatten moet **.php** als extensie meekrijgen.

PHP-code wordt gewoon tussen de HTML-code geschreven. Elk blokje PHP-code begint met de tag **<?php** en eindigt met **?>**.

Binnen de PHP-*delimiters* kan je returns en tabs gebruiken om je code te structureren. Die returns en tabs hebben geen effect op de uitvoering van de code zelf.

De webserver leest de PHP-pagina in, gaat op zoek naar de PHP-code aan de hand van die delimiters en voert de instructies uit. Het resultaat is HTML-code die naar de client computer gestuurd wordt en door de client-browser gelezen wordt.

De klassieke test. Typ onderstaande instructie in de body van een lege webpagina TEST.PHP in.

```
<body>
<?php
echo 'Hello world!';
?>
</body>
```

**Echo** is een ingebouwde functie waarmee je een tekenreeks uitvoert. Die komt tussen de HTML-code terecht. De argumenten kunnen tussen enkele of dubbele aanhalings-tekens staan.

Elke instructie-regel eindigt op een ;-teken.

Sla de webpagina op in c:\xampp\htdocs en open de pagina in je browser:  
<http://localhost/test.php>. (Je hoeft HTTP:// niet in te typen).

### 15.4 Include()

De enige functie die we nu gebruiken is **include()** om informatie uit een externe pagina op een bepaalde plaats toe te voegen, bv.

```
<?php include("header.php"); ?>
```

Je maakt een basisstramien/basispagina die je dan telkens voor alle andere pagina's gebruikt via gewoon kopiëren.

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Bootjes van Gent</title>
</head>

<body>
<div id="wrapper">
    <header><?php include("header.php"); ?></header>
    <nav><?php include("nav.php"); ?></nav>
    <main>hier komt de eigenlijke inhoud</main>
```

```
<aside><?php include("aside.php");?></aside>
<footer><?php include("footer.php");?></footer>
</div>
</body>
</html>
```

In de containers met gemeenschappelijke inhoud (`<header>`, `<nav>`, `<aside>`, `<footer>`) neem je geen inhoud op maar verwijst je telkens naar een afzonderlijke externe pagina. Met de PHP-include functie neem je op die plaats telkens de inhoud uit die afzonderlijke pagina's op. Zo bevat de pagina **NAV.PHP** het volledige menu als HTML-code.

```
<ul>
<li><a href="index.html">Home</a></li>
<li><a href="boottochten.html">Boottochten</a>
    <ul>
        <li><a href="#">Standaard formule</a></li>
        <li><a href="#">Formule Gourmand</a></li>
        <li><a href="#">VIP-aanbod</a></li>
    </ul>
</li>
<li><a href="dagprogramma.html">Dagprogramma</a></li>
<li><a href="aanraders.html">Aanraders</a></li>
<li><a href="scholen.html">Scholen</a>
    <ul>
        <li><a href="#">Basisonderwijs</a></li>
        <li><a href="#">Secundair onderwijs</a></li>
        <li><a href="#">Hoger onderwijs</a></li>
    </ul>
</li>
<li><a href="bedrijven.html">Bedrijven</a></li>
</ul>
```

Pas je het menu in **NAV.PHP** aan, dan wordt het menu automatisch in de gehele website aangepast.

Je webpagina's moeten dan wel via een webserver uitgevoerd worden die de PHP-functie `include()` uitvoert. Nagenoeg alle webhosting platforms ondersteunen PHP. Om lokaal op je toestel te testen moet je wel PHP installeren. Dat wordt verder uitgelegd.

## 16 OVERZICHT BELANGRIJKE HTML-ELEMENTEN

Element	Toelichting
<!--...-->	HTML commentaar
<!DOCTYPE>	starttag in webpagina. Bepaalt type document voor browser. <!DOCTYPE html>
<a>	hyperlink. Belangrijke attributen: HREF en TARGET.
<area>	bakent een aanklikbaar gebied in een illustratief af. Zie ook <map>.
<article>	bakent een stuk bijeenhorende inhoud in een wepgagina af
<aside>	bakent inhoud in een zijbalk af
<body>	bakent de volledige inhoud van de webpagina af
 	neemt een nieuwe regel zonder nieuwe paragraaf te starten
<button>	aanklikbare knop. Gebruik <input> voor knoppen in een formulier!
<datalist>	biedt een aantal keuzes aan bij invoer in een tekstvak
<div>	container die een groep elementen groepeert om die via css op te maken en te positioneren. Werkt op block-niveau.
<em>	beklemtoont tekst (wordt meestal cursief weergegeven)
<fieldset>	groepeert een groep bijeenhorende elementen in een formulier visueel door een kader eromheen te plaatsen. Zie ook <legend>.
<footer>	bakent onderaan in de webpagina informatie af, bv. contact-informatie, copyright, info over auteur, sitemap...
<form>	formulier. Kan volgende elementen bevatten: <input>, <textarea>, <button>, <select>, <option>, <optgroup>, <fieldset> en <label>.
<h1> – <h6>	bakent koppen in een webpagina af, hiërarchisch van 1 tot 6
<head>	informatie over je webpagina, geen eigenlijke inhoud. Kan volgende tags bevatten: <title> (verplicht!), <style>, <base>, <link>, <meta>, <script> en <noscript>
<header>	bakent bovenaan in de webpagina inleidende inhoud of navigatielinks af
<html>	container voor alle HTML-tags, behalve <!DOCTYPE>
<img>	voegt een afbeelding toe. Verplichte attributen: SRC en ALT.
<input>	formulierobject. TYPE attribuut bepaalt soort object, bv.: BUTTON, CHECKBOX, COLOR, DATE, EMAIL, HIDDEN, NUMBER, PASSWORD, RADIO, RANGE, SUBMIT, TEXT...

<code>&lt;label&gt;</code>	bepaalt een titel voor een tekstvak in een formulier. Attribuut FOR moet naar ID van betrokken tekstvak verwijzen.
<code>&lt;legend&gt;</code>	geeft een titel aan een <code>&lt;fieldset&gt;</code> -groep. Zie ook <code>&lt;fieldset&gt;</code> .
<code>&lt;li&gt;</code>	bakent één item uit een lijst af. Zie ook <code>&lt;ul&gt;</code> en <code>&lt;ol&gt;</code> .
<code>&lt;link&gt;</code>	link naar externe bron, vooral externe style sheet
<code>&lt;map&gt;</code>	client-side image map met verschillende aanklikbare gebieden binnen één illustratie. Zie ook <code>&lt;area&gt;</code> .
<code>&lt;meta&gt;</code>	informatie over de webpagina zelf, geen inhoud. Attributen: NAME en CONTENT. Mogelijke NAME-waarden: charset, keywords, description, viewport, refresh... Binnen <code>&lt;head&gt;</code> -gedeelte van de website plaatsen!
<code>&lt;nav&gt;</code>	bakent menu met navigatielinks af
<code>&lt;ol&gt;</code>	bakent een genummerde lijst af. Zie ook <code>&lt;li&gt;</code> .
<code>&lt;option&gt;</code>	element uit keuzelijst in formulier. Zie ook <code>&lt;select&gt;</code> .
<code>&lt;p&gt;</code>	bakent een paragraaf af
<code>&lt;section&gt;</code>	bakent een sectie in een document af, een bijeenhorend geheel aan elementen
<code>&lt;select&gt;</code>	keuzelijst in formulier. Zie ook <code>&lt;option&gt;</code> .
<code>&lt;span&gt;</code>	groepeert inhoud om die via CSS aan te spreken. Werkt op inline niveau.
<code>&lt;strong&gt;</code>	bakent belangrijke tekst af (wordt meestal vet weergegeven)
<code>&lt;style&gt;</code>	voor embedded CSS-regels. Attribuut TYPE="TEXT/CSS". Binnen <code>&lt;head&gt;</code> -gedeelte van de website plaatsen!
<code>&lt;table&gt;</code>	bakent een tabel af
<code>&lt;td&gt;</code>	bakent een cel in een tabelrij af ( <b>table data</b> )
<code>&lt;textarea&gt;</code>	tekstvak dat verschillende regels tekst toelaat. Onbeperkt aantal tekens mogelijk.
<code>&lt;th&gt;</code>	bakent een cel in een titelrij van een tabel af ( <b>table head</b> )
<code>&lt;title&gt;</code>	titel van de webpagina (verplicht), verschijnt in browser tab. Binnen <code>&lt;head&gt;</code> -gedeelte van de website plaatsen!
<code>&lt;tr&gt;</code>	bakent een rij in een tabel af ( <b>table row</b> )
<code>&lt;ul&gt;</code>	bakent een opsommingslijst af. Zie ook <code>&lt;li&gt;</code> .

## 17 OVERZICHT BELANGRIJKE CSS-EIGENSCHAPPEN

Eigenschap	Omschrijving
background	stelt alle achtergrondeigenschappen in één declaratie in
background-attachment	bepaalt of een achtergrondillustratie vast staat of mee schuift bij het scrollen
background-color	bepaalt de achtergrondkleur van een element
background-image	stelt één of meer achtergrondillustraties in
background-position	bepaalt de plaatsing van een achtergrondillustratie
background-repeat	stelt in hoe een achtergrondillustratie al dan niet herhaald wordt om de ruimte te vullen
background-clip	bepaalt waarop de achtergrond toegepast wordt (al dan niet op de padding, border)
border	stelt alle randeigenschappen in één declaratie in
border-bottom	stelt de onderrand in één declaratie in
border-bottom-color	stelt de kleur van de onderrand in
border-bottom-left-radius	bepaalt de afronding van de rand linksonder
border-bottom-right-radius	bepaalt de afronding van de rand rechtsonder
border-bottom-style	stelt het soort onderrand in
border-bottom-width	stelt de dikte van de onderrand in
border-color	stelt de kleur van de omranding in
border-left	stelt de linkerrand in één declaratie in
border-left-color	stelt de kleur van de linkerrand in
border-left-style	stelt het soort linkerrand in
border-left-width	stelt de dikte van de linkerrand in
border-radius	stelt de afronding van het volledige kader in
border-right	stelt de rechterraand in één declaratie in
border-right-color	stelt de kleur van de rechterraand in
border-right-style	stelt het soort rechterraand in
border-right-width	stelt de dikte van de rechterraand in
border-style	stelt het soort omranding in
border-top	stelt de bovenrand in één declaratie in
border-top-color	stelt de kleur van de bovenrand in

Eigenschap	Omschrijving
border-top-left-radius	stelt de afronding van de linkerbovenrand in
border-top-right-radius	stelt de afronding van de rechterbovenrand in
border-top-style	stelt het soort bovenrand in
border-top-width	stelt de dikte van de bovenrand in
border-width	stelt de dikte van de volledige omranding in
box-shadow	voegt schaduw aan de omranding toe
clear	bepaalt aan welke zijde(n) van een element geen floating elementen kunnen geplaatst worden
display	bepaalt de plaatsing/weergave van een element
float	bepaalt of een element al dan niet zwevend is
height	stelt de hoogte van een element in
left	bepaalt de linker positie van een gepositioneerd element
margin	stelt alle marges in één declaratie in
margin-bottom	stelt de ondermarge van een element in
margin-left	stelt de linkermarge van een element in
margin-right	stelt de rechtermarge van een element in
margin-top	stelt de bovenmarge van een element in
max-height	stelt de maximale hoogte van een element in
max-width	stelt de maximale breedte van een element in
min-height	stelt de minimale hoogte van een element in
min-width	stelt de minimale breedte van een element in
overflow	bepaalt wat gebeurt als de volledige inhoud niet in een box past
padding	stelt de ruimte binnen een box in één declaratie in
padding-bottom	stelt de ruimte onderaan binnen een box in
padding-left	stelt de ruimte links binnen een box in
padding-right	stelt de ruimte rechts binnen een box in
padding-top	stelt de ruimte bovenaan binnen een box in
position	bepaalt op welke manier een element gepositioneerd wordt (static, relative, absolute of fixed)
right	bepaalt de rechtse positie van een gepositioneerd element
top	bepaalt de verticale startpositie van een gepositioneerd element
bottom	bepaalt de verticale eindpositie van een gepositioneerd element

<b>Eigenschap</b>	<b>Omschrijving</b>
visibility	bepaalt of een element al dan niet zichtbaar is
width	stelt de breedte van een element in
vertical-align	stelt de verticale uitlijning van een element in
z-index	bepaalt de stapelvolgorde van een element
color	stelt de tekstkleur in
opacity	stelt de transparantie van een element in

<b>Eigenschap</b>	<b>Omschrijving</b>
font	stelt het lettertype in één declaratie in
font-family	bepaalt de lettertypekeuze van de tekst
font-size	bepaalt de lettergrootte van de tekst
font-stretch	bepaalt welk sublettertype eventueel gekozen wordt (normaal, gecondenseerd, uitgerokken)
font-style	bepaalt de stijl van de tekst
font-weight	bepaalt de dikte van de tekst (vet)

<b>Eigenschap</b>	<b>Omschrijving</b>
list-style	stelt de eigenschappen van een opsomming in één declaratie in
list-style-image	bepaalt een figuur als opsommingsteken
list-style-position	bepaalt of het opsommingsteken in of uit de opsommingszone geplaatst wordt
list-style-type	bepaalt het soort opsommingsteken (NONE om opsommingsteken te verwijderen)

<b>Eigenschap</b>	<b>Omschrijving</b>
text-decoration	bepaalt alle eigenschappen van de tekst decoratie in één declaratie
text-decoration-color	bepaalt de kleur van de lijn
text-decoration-line	bepaalt de plaats van de lijn (boven, onder, door tekst)
text-decoration-style	bepaalt de stijl van de lijn
text-shadow	voegt een schaduw effect aan de tekst toe

<b>Eigenschap</b>	<b>Omschrijving</b>
letter-spacing	verhoogt of verlaagt de afstand tussen de letters
line-height	stelt de regelhoogte in
text-align	bepaalt de horizontale uitlijning van tekst

Eigenschap	Omschrijving
text-indent	bepaalt de insprong van de eerste regel van een tekstblok
text-transform	bepaalt de weergave in hoofdletters/kleine letters
word-wrap	lange woorden kunnen al dan niet over twee regels gesplitst worden

## **18 BRONNEN**

Interessante bronnen

- <http://getbootstrap.com/>
- <https://css-tricks.com/>
- <https://www.w3schools.com/>



Sint-Jozefsinstituut Brugge – 2024-2025