

# HTML5 - CSS3

## 1. VOORKENNIS

Voor dit deel van de cursus is **geen enkele voorkennis** vereist. Iedereen kan instappen, ongeacht ervaring met webontwikkeling.

## 2. DOEL

Na het doorlopen van deze cursus bent u in staat om een volledige, professionele en responsive website te bouwen volgens de moderne standaarden.

De cursus bestaat uit drie delen:

- **HTML5** – structuur en inhoud van een webpagina
- **CSS3** – vormgeving en opmaak
- **Bootstrap 5 Framework** – een populair CSS-framework waarmee u sneller responsive websites maakt

Dit eerste deel behandelt **HTML5** en **CSS3**. We gaan stap voor stap door alle fundamentele concepten, geïllustreerd met duidelijke voorbeelden.

Daarna volgt het luik **Bootstrap 5.\***, waarin we leren hoe je met voorafgebouwde componenten en klassen vlot een moderne responsive website kunt realiseren.

Wanneer u deze onderdelen volledig hebt doorlopen, bouwen we samen een volledig webproject:

Vanaf het ontwerp → tot en met de online publicatie van de website.

## 3. INSTALLATIE TEXT EDITOR

Voor deze cursus gebruiken we **PhpStorm** van JetBrains als standaardomgeving. PhpStorm is een krachtige **IDE (Integrated Development Environment)** die alle nodige functies bevat om efficiënt te werken met **HTML5, CSS3 en later ook PHP, JavaScript en frameworks zoals Bootstrap en Laravel**.

Waarom PhpStorm?

- Volledige ondersteuning voor HTML5, CSS3, JavaScript en PHP
- Automatische code-aanvulling en foutdetectie
- Ingebouwde Git- en versiebeheerintegratie
- Live server en debuggingtools
- Integratie met databases en frameworks
- Cross-platform: werkt op Windows, macOS en Linux

### 3.1. INSTALLATIESTAPPEN

1. Surf naar <https://www.jetbrains.com/phpstorm>
2. Klik op **Download** en kies uw besturingssysteem (Windows, macOS of Linux).
3. Installeer de applicatie via de standaard wizard.
4. Start PhpStorm op en kies **30 dagen gratis trial** of activeer via een **educatieve licentie** (gratis voor studenten/docenten via JetBrains Education).

## EERSTE INSTELLINGEN

Bij de eerste opstart vraagt PhpStorm naar enkele voorkeuren:

- **Theme:** Kies “Darcula” (donkere modus) of “Light”.
- **Plugins:** Voor deze cursus zijn de standaard plugins voldoende. Optioneel kan je *Git, Database Tools of Markdown Support* activeren.
- **Project Setup:** Maak een nieuwe projectmap aan, bijvoorbeeld **C:\webcursus** of **/Users/naam/webcursus**.

## PRAKTISCH

- Elk oefenbestand dat we in de cursus maken, wordt bewaard in een eigen projectmap.
- PhpStorm zorgt automatisch voor syntax highlighting, code-formatting en foutmeldingen.
- Via de ingebouwde browser preview kan je je webpagina's meteen testen.

## 4. HTML 5

Een webpagina zichtbaar maken voor het publiek vergt een duidelijke **structuur**.

Die structuur heeft twee belangrijke componenten:

1. **HTML5 (HyperText Markup Language)**
  - Beschrijft de **inhoud** van een webpagina: titels, alinea's, lijsten, links, afbeeldingen, formulieren, enz.
  - HTML zegt *wat* de inhoud betekent, maar bepaalt **niet** hoe die eruitziet.
2. **CSS3 (Cascading Style Sheets)**
  - Zorgt voor de **opmaak**: kleuren, lettertypes, positionering, spacing, layout en responsiviteit.
  - CSS3 werkt altijd bovenop HTML: zonder HTML geen CSS.

Later komt er ook **JavaScript** bij: dat voegt interactiviteit en logica toe (bijvoorbeeld menu's die openklappen, sliders, of validatie van formulieren). In dit deel focussen we echter op **HTML5 en CSS3**.

### 4.1. EERSTE HTML 5 PAGINA

Elke website begint met een **basisbestand** dat de structuur van een pagina beschrijft.

#### Stappen

1. Open **PhpStorm** en maak een **nieuw HTML-bestand** aan.
  - Menu: **File → New → HTML File**
  - Geef het bestand de naam **index.html**
2. Controleer dat de extensie correct is:
  - **.html** (aanbevolen en standaard)
  - **.htm** (verouderd; komt bijna niet meer voor, enkel nog om compatibiliteit).

✓ Belangrijk: een webserver zoekt standaard naar **index.html** als startpunt. Dit is dus altijd je homepage.

## Voorbeeld: een eerste HTML5-skelet

```
<!DOCTYPE html>
<html lang="nl">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1">
  <title>Mijn eerste HTML5-pagina</title>
  <meta name="description" content="Oefening: basisstructuur van een
HTML5-document.">
</head>
<body>
  <h1>Welkom!</h1>
  <p>Dit is mijn eerste webpagina. De inhoud staat er, de opmaak
volgt later met CSS3.</p>
</body>
</html>
```

### Uitleg bij de onderdelen

- `<!DOCTYPE html>` Vertelt de browser dat dit een **HTML5-document** is. Zonder dit kan de browser in “quirks mode” terechtkomen, wat zorgt voor afwijkende weergave.
- `<html lang="nl"> ... </html>` De **root-tag**. Het attribuut `lang="nl"` geeft aan dat de inhoud Nederlandstalig is. Dit helpt bij toegankelijkheid (screenreaders) en SEO.
- `<head> ... </head>` Hier staat **metadata**: info over de pagina (titel, tekenset, beschrijving). Dit is niet zichtbaar voor de bezoeker.
- `<meta charset="UTF-8">` Tekenset die zorgt dat letters, accenten en emoji correct worden weergegeven.
- `<meta name="viewport" content="width=device-width, initial-
scale=1">` Essentieel voor mobiele weergave: dit zorgt dat de pagina zich automatisch aanpast aan de breedte van het scherm.
- `<title>` Verschijnt in het browser-tabblad en wordt gebruikt door zoekmachines.
- `<body> ... </body>` Alles wat zichtbaar is voor de bezoeker: tekst, afbeeldingen, formulieren, menu's, enz.

### Mogelijke uitbreidingen

- **Accessibility (toegankelijkheid):** vanaf het begin correcte lang, logische koppenstructuur (`h1-h6`), en later ook alt-teksten bij afbeeldingen.
- **SEO (zoekmachineoptimalisatie):** goede titel en beschrijving zijn een must, extra meta-tags zoals keywords zijn achterhaald en niet meer nodig.
- **Validatie:** controleer je code met de [W3C Validator](#) om fouten tijdig op te sporen.

✓ **Resultaat:** je hebt nu een eerste werkende HTML5-pagina die je kan openen in je browser. Voorlopig toont ze enkel platte tekst, maar dit vormt de basis waarop we verder bouwen.

## 4.2. STANDAARD TAGS

Hier zie je per lijn **TAGS** staan. TAGS worden gedefinieerd door de volgende opmaak <**tagnaam**>. In het voorbeeld hieronder zie je enkele voorbeelden staan: html, head, body, ...

De meeste tags hebben een begin- en een eindtag ook wel de closing-tag genoemd die wordt voorafgegaan door het slash-teken. Bijvoorbeeld: <body></body>.

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>HTML 5</title>
6  </head>
7  <body>|</body>
8  </html>
```

<!DOCTYPE html>

Document type ook wel de DOM (Domein object model) genoemd. Dit bepaalt de HTML versie van je pagina ook wel DOCTYPE DECLARATION genoemd. De notatie die u hierboven ziet definieert een **html 5** pagina.

<html lang="en">

Bovenstaande tag bestaat uit 2 delen:

**tag** = html

**attribuut of property** = lang.

Een attribuut of property heeft ook telkens een waarde die tussen quotes wordt geschreven. In dit geval is de waarde de taal van de pagina, nl. **en**. Zet dit op **nl** voor Nederlands.

Merk op dat de html tag ook dient te worden afgesloten met wat we noemen een closing tag = </html>

## <head>

Het <head> element is een container voor metagegevens en wordt geplaatst tussen de html tag en de body tag. Metagegevens geven meer informatie mee over de betreffende pagina.

Standaard geven we hier een titel mee aan het document en de characterset UTF-8.

### Characterset

In HTML 5 is de characterset UTF-8 die de standaard unicode is.

Een characterset gaat telkens om een verzameling van tekens zoals letters, symbolen en cijfers uit diverse talen.

### Meta tags

Met tags geven een beknopte beschrijving mee waarover de pagina gaat.

```
<meta charset="UTF-8">
<title>HTML 5</title>
<meta name="description" content="Klik per klik tutorials">
<meta name="keywords" content="HTML5,CSS3,BOOTSTRAP">
<meta name="author" content="Tom Vanhoutte">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Bovenstaande tags zijn duidelijk.

De title tag die hier wordt ingegeven zal weergegeven worden op het tabblad in je browser.

Description, keywords en author bepalen over wat de webpagina gaat. Content is het attribuut waar de beschrijving volgens hun respectievelijke meta tag wordt ingevuld.

De meta name = "viewport" zullen we later nodig hebben wanneer we over responsive design zullen praten. Meer hierover later in dit boek.

### Speciale Meta tags

Met onderstaande meta tag verplicht je de pagina om de 5 seconden te verversen.

```
<meta http-equiv = "refresh" content = "5" />
```

Wanneer we dit uitbreiden met een url als attribuut dan zal hij na 5 seconden richting een andere pagina verversen.

```
<meta http-equiv = "cookie" content = "userid = 2;
expires = Sunday, 10-Oct-21 23:59:59 GMT;" />
```

Deze pagina zet een vaste cookie op je pc. Wanneer er geen datum zou bijstaan wordt deze aanzien als een SESSION cookie. Later meer over cookies.

## <body>

Tussen deze tags wordt de eigenlijke pagina opgebouwd die de inhoud aan de gebruiker in een browser zal weergegeven.

Als webontwikkelaars zullen wij steeds gebruiken maken van Google Chrome omdat dit de meest complete browser is momenteel.

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>HTML 5</title>
6      <meta name="description" content="Klik per klik tutorials">
7      <meta name="keywords" content="HTML 5, CSS3, BOOTSTRAP">
8      <meta name="author" content="Tom Vanhoutte, Full Stack Web Developer">
9      <meta name="viewport" content="width=device-width, initial-scale=1.0">
10 </head>
11 <body>
12     Hier komt de inhoud van uw webpagina te staan!
13 </body>
14 </html>
```

Bewaar de pagina op je pc en geef deze de volgende bestandsnaam:  
**Klik\_per\_klik\_1.html**.

***OPMERKING: de naamgeving van de bestanden werd enkel gekozen in functie van de cursus.***

***DE HOMEPAGINA VAN IEDERE WEBSITE DIENT DE NAAMGEVING INDEX.HTML TE ZIJN. EEN WEBSERVER ZAL ALTIJD NAAR DEZE PAGINA ZOEKEN OM JE WEBSITE TE LADEN!***

Bekijk nu het resultaat in je webbrowser.

## RESULTAAT

Hier komt de inhoud van uw webpagina te staan!

Zoals je hierboven kan zien bevat de pagina alleen platte tekst momenteel.

### 4.3. GOOGLE CHROME: INSPECTEER ELEMENT

Klik met je rechtermuisknop op een blanco plaats in de pagina en selecteer inspecteer element.

Onderstaand scherm toont je een "developer" helpscherm. Aan de linkerkant kan je de html code zien die we zonet hebben geschreven. Dit scherm zal je in je loopbaan als webontwikkelaar altijd open hebben staan. Aan de rechterkant zie je de styles (css) die zorgen voor de opmaak van de pagina met daaronder een vierkant venster die we het boxmodel noemen. Het boxmodel zullen we in detail later in de cursus bespreken.



## 4.4. BLOCK LEVEL EN INLINE ELEMENTEN

Block level elementen zijn elementen (tags) die onmiddellijk naar de volgende lijn retourneren. Inline elementen blijven op dezelfde lijn.

## 4.5. BODY TAGS

De standaard tags die een html 5 pagina definiëren hebben we reeds gezien. In dit grootste gedeelte van de cursus zullen we het hebben over alle tags die binnen de body tag kunnen worden beschreven. Een tag heeft als doel om aan zoekrobots zoals die van google, duidelijk mee te geven **over wat een stuk tekst, afbeelding,... handelt**. Deze tags bepalen ook niet alleen welk soort van tekst er wordt weergegeven maar ook de structuur en opbouw van een webpagina en zijn uitermate belangrijk in o.a. zoekmachineoptimalisatie (SEO = Search Engine Optimization). Tags zijn een belangrijke schakel in de semantiek van een webpagina. Een slecht opgebouwde pagina zal minder goed ranken in uiteindelijke zoekresultaten op het web.

### 4.5.1. BODY TAGS

#### 4.5.1.1. TEKST TAGS

##### 4.5.1.1.1. COMMENTAARLIJNEN

COMMENTAAR	
Tag	<!-- -->
Beschrijving	Hiermee kun je <b>commentaarlijnen</b> toevoegen aan je HTML code. Commentaarlijnen zijn enkel zichtbaar in het inspect element van je webbrowser.

#### CODE

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>HTML 5</title>
6      <meta name="description" content="Klik per klik tutorials">
7      <meta name="keywords" content="HTML 5, CSS3, BOOTSTRAP">
8      <meta name="author" content="Tom Vanhoutte, Full Stack Web Developer">
9      <meta name="viewport" content="width=device-width, initial-scale=1.0">
10 </head>
11 <body>
12     <!-- Hieronder zullen we alle body tags bespreken -->
13 </body>
14 </html>
```

#### 4.5.1.1.2.

#### HEADINGS <h1> t.e.m. <h6>

Headings	
Tag	<h1> t.e.m <h6>
Beschrijving	Headings worden gedefinieerd met de tags <h1> t.e.m. <h6> in volgorde van belangrijkheid. OPMERKING: een <h1> tag zal meestal de titel dragen van het onderwerp van een webpagina. Een misvatting die veelal wordt meegegeven is dat een webpagina niet meer dan één h1 tag mag bevatten. Google toont duidelijk in hun documentatie dat die NIET verkeerd is, maar draagt niet bij tot duidelijkheid over wat de pagina handelt. In de praktijk zullen we echter in 99% van de gevallen één h1 tag per pagina gebruiken.
Type	Block level

#### CODE

```
13  <body>
14      <!-- Hieronder zullen we alle body tags bespreken -->
15      <!-- HEADINGS -->
16      <h1>HEADING 1</h1>
17      <h2>HEADING 2</h2>
18      <h3>HEADING 3</h3>
19      <h4>HEADING 4</h4>
20      <h5>HEADING 5</h5>
21      <h6>HEADING 6</h6>
22  </body>
23
24  </html>
```

#### RESULTAAT

**HEADING 1**

**HEADING 2**

**HEADING 3**

**HEADING 4**

**HEADING 5**

**HEADING 6**

#### 4.5.1.1.3. <p>

p tag	
Tag	<p>
Beschrijving	Paragraph tag. Dit wordt gebruikt om gewone tekst weer te geven. De p-tag erft de grootte van lettertype standaard van de browser. Deze is 16px (pixels). Later in css meer.
Type	Block level

#### CODE

```
13<body>
14    <!-- Hieronder zullen we alle body tags bespreken -->
15    <!-- HEADINGS -->
16    <h1>HEADING 1</h1>
17    <h2>HEADING 2</h2>
18    <h3>HEADING 3</h3>
19    <h4>HEADING 4</h4>
20    <h5>HEADING 5</h5>
21    <h6>HEADING 6</h6>
22    <!-- PARAGRAPH TAG -->
23    <p>paragraph</p>
24</body>
25
26</html>
```

#### RESULTAAT

# HEADING 1

## HEADING 2

### HEADING 3

#### HEADING 4

##### HEADING 5

###### HEADING 6

paragraph

#### 4.5.1.1.4. LIJSTEN: <ol> en <ul>

HTML lists	
Tag	<ol> of <ul>
Beschrijving	<ol> = geordende lijst (Bijv. 1,2,3,...) <ul> = ongeordende lijst (Bijv bullets) <ul> of <ol> bepalen welk soort van lijst. <li> wordt gebruikt om de lijst items weer te geven en ook voor opmaak van een navigatiestructuur.
Type	Block level

#### CODE

```
24      <!-- HTML LISTS -->
25      <!-- ONGEORDENDE LIJST -->
26<ul>
27      <li>HTML5</li>
28      <li>CSS3</li>
29      <li>BOOTSTRAP</li>
30</ul>
31      <!-- GEORDENDE LIJST TAG -->|
32<ol>
33      <li>HTML5</li>
34      <li>CSS3</li>
35      <li>BOOTSTRAP</li>
36</ol>
37</body>
38
39</html>
```

## RESULTAAT

- HTML5
  - CSS3
  - BOOTSTRAP
1. HTML5
  2. CSS3
  3. BOOTSTRAP

#### 4.5.1.1.5. <blockquote>

Blockquote	
Tag	<blockquote>
Beschrijving	De tag <blockquote> geeft een sectie aan afkomstig van een ander bron. Standaard wordt een blockquote ook ingesprongen vanaf de linkermarge en wordt het attribuut cite meegegeven.
Type	Block level

#### CODE

```
37      <!-- BLOCKQUOTE TAG -->
38      <blockquote cite="https://nl.wikipedia.org/wiki/Stephen_Hawking">
39          Stephen Hawking werd geboren op 8 januari 1942. Zijn ouders,
40          Frank en Isobel Hawking, woonden in Londen, maar wegens de
41          bombardementen op Londen tijdens de Tweede Wereldoorlog waren
42          ze naar het veiligere Oxford verhuisd.
43      </blockquote>
```

#### RESULTAAT

The screenshot shows a browser's developer tools with the 'Elements' tab selected. At the top, there is a preview window displaying a blockquote containing text about Stephen Hawking. Below the preview, the DOM tree is visible, showing the original code with line numbers and the rendered HTML. The rendered HTML includes the opening and closing <blockquote> tags, the cite attribute, and the inner text of the quote.

```
37      <!-- BLOCKQUOTE TAG -->
38      <blockquote cite="https://nl.wikipedia.org/wiki/Stephen_Hawking">
39          Stephen Hawking werd geboren op 8 januari 1942. Zijn ouders,
40          Frank en Isobel Hawking, woonden in Londen, maar wegens de
41          bombardementen op Londen tijdens de Tweede Wereldoorlog waren
42          ze naar het veiligere Oxford verhuisd.
43      </blockquote>
```

#### 4.5.1.1.6. <pre>

pre tag	
Tag	<pre >
Beschrijving	De tekststopmaak tussen deze tags blijft behouden. Hier wordt er ingesprongen vanaf de linkermarge.
Type	Block level

#### CODE

```
<!-- PRE TAG -->
<pre>
    De pre tag      zal spaties en opmaak behouden zoals
    deze werden ingetikt door      d e   gebruiker
</pre>
    |
```

#### RESULTAAT

```
De pre tag      zal spaties en opmaak behouden zoals
deze werden ingetikt door      d e   gebruiker
```

#### 4.5.1.1.7. <abbr>

abbr tag	
Tag	<abbr>
Beschrijving	De abbreviation tag, of de afkorting tag wordt gebruikt om de afkorting in een webpagina weer te geven. Wanneer we er met de muis overheen gaan wordt de volledige benaming weergegeven in een tool-tip. Standaard wordt de afkorting ook licht onderlijnd.
Type	Block level

#### CODE

```
<!-- ABBR TAG -->
<abbr title="Hypertext Markup Language">HTML</abbr>
```

#### RESULTAAT

HTML

#### 4.5.1.1.8. <b> <strong>

b tag, strong tag	
Tag	<b> <strong>
Beschrijving	De tag b drukt een tekst in het vet op het scherm. De strong tag doet hetzelfde maar legt meer nadruk op het woord. Dit is interessant voor 'screenreaders' die dan de nadruk zullen leggen tijdens het voorlezen van de tekst op het vetgedrukte woord.
Type	Inline level elementen

#### CODE

```
<!-- B TAG, STRONG TAG -->
<b>Vetgedrukt</b><br>
<strong>Vetgedrukt</strong>
```

#### RESULTAAT

HTML Vetgedrukt Vetgedrukt

OPMERKING: Hierboven hebben we voor de eerste keer inline level elementen gebruikt. D.w.z. dat de verschillende tags op één lijn naast elkaar worden weergegeven. Om de inline elementen toch op een volgende lijn te krijgen kunnen we gebruik maken van de <br> tag.

#### CODE

```
<!-- B TAG, STRONG TAG -->
<b>Vetgedrukt</b><br>
<strong>Vetgedrukt</strong>
```

#### RESULTAAT

HTML Vetgedrukt  
Vetgedrukt

#### 4.5.1.1.9. <i> <em> <var>

i tag, em tag, var tag	
Tag	<i> <em> <var>
Beschrijving	De tag i drukt een tekst schuin op het scherm. De em tag doet hetzelfde maar legt meer nadruk op het woord. Dit is interessant voor 'screenreaders' die dan de nadruk zullen leggen tijdens het voorlezen van de tekst op het schuingedrukte woord. De var tag wordt ook schuin weergegeven maar gaat over een variabele die wordt weergegeven. De semantiek is hier van belang.
Type	Inline level elementen

#### CODE

```
<!-- I TAG, EM TAG, VAR TAG -->
<i>Schuingedrukt</i>
<em>Schuingedrukt</em>
<var>Schuingedrukt, definieert een variabele, bijv. x = 1</var>
```

#### RESULTAAT

Schuingedrukt Schuingedrukt Schuingedrukt

#### 4.5.1.1.10. <u> <mark>

u tag, mark tag	
Tag	<u> <mark>
Beschrijving	Beiden zorgen voor een markering. De <u> tag zorgt ervoor dat woorden of delen van een zin onderlijnd kunnen worden. De <mark> tag zorgt voor een fluo markering binnen in de tekst. Standaard in google chrome is deze geel.
Type	Inline level elementen

#### CODE

```
<!-- U TAG, MARK TAG -->
<p>This <mark>is</mark> a <u>paragraph</u>. </p>
```

#### RESULTAAT

This **is** a paragraph.

#### 4.5.1.1.11. <code> <kbd><samp>

code tag, kbd tag, samp tag	
Tag	<code> <var> <samp>
Beschrijving	<code>computer code</code> <samp>computer output of resultaat</samp> <kbd>Keyboard input, bijvoorbeeld CTRL+Z</kbd>
Type	Inline level elementen

#### CODE

```
<!-- CODE TAG, SAMP TAG, KBD TAG -->
<code>computer code</code><br>
<samp>computer output of resultaat</samp><br>
<kbd>Keyboard input, bijvoorbeeld CTRL+Z</kbd><br>
```

#### RESULTAAT

```
computer code
computer output of resultaat
Keyboard input, bijvoorbeeld CTRL+Z
```

#### 4.5.1.1.12. <a>

a tag	
Tag	<a>
Beschrijving	De a tag zorgt ervoor dat we links op de pagina kunnen gebruiken. Het attribuut <b>href</b> bevat de uiteindelijke link naar een interne of externe html pagina. Bijvoorbeeld: <a href="http://www.google.be">http://www.google.be</a>
Type	Inline level element

#### CODE

```
<!-- A TAG -->
<a href="http://www.google.be">Google</a><br>
<!-- BDO TAG -->
```

#### RESULTAAT

```
Google
```

Bij de a tag kunnen we gebruik maken van het target attribute. Die zal bepalen waar de aangeklikte link (href tag) geopend zal worden. Wanneer je target niet zou gebruiken als attribuut dan wordt standaard "target=\_self" gebruikt. target mogelijkheden:

- \_blank
  - opent een nieuw tabblad
- \_parent
  - wordt gebruikt om het volgende iframe te openen (komt later aan bod).
- \_self
  - nieuwe link (pagina) wordt geladen in hetzelfde tabblad van de webbrowser.
- \_top
  - breekt buiten bestaande iframes (komt later aan bod).

Voorbeeld:

```
<a href="http://www.syntrawest.be" target="_blank" title="site van syntrawest">Klik op mij</a>
```

Bovenstaand voorbeeld zorgt ervoor dat de href link in een nieuw tabblad wordt geopend in uw webbrowser.

De title tag geeft zoekmachines bijkomende informatie over de link. Je kan dit vergelijken met de alt tag die we gebruiken als bijkomende informatie bij de img tag.

We kunnen ook linken binnen eenzelfde pagina, daarvoor gebruiken we een **anchor tag**.

We kunnen terug de anchor tag en het href attribuut hiervoor gebruiken met een klein verschil in de link, nl. #

Het **#-teken** zorgt ervoor dat er gezocht wordt in de pagina naar een section met dezelfde naam als de link zelf.

Voorbeeld:

```
<a href="#een" title="section een">Link naar section een</a>
<a href="#twee" title="section een">Link naar h3 met de naam twee</a>
...
<section id="een"></a>
` 
...
```
<h3 id="twee">dit is het begin van twee</h3>
```

#### 4.5.1.1.13. <bdo>

bdo tag	
Tag	<bdo>
Beschrijving	De bdo tag in combinatie met het attribuut dir="rtl" zorgt ervoor dat tekst omgekeerd op het scherm wordt weergegeven.
Type	Inline level element

#### CODE

```
<!-- BDO TAG -->
<bdo dir="rtl">
    moordnilap nee si lepel
</bdo><br>
```

#### RESULTAAT

Lepel is een palindroom

#### 4.5.1.1.14. <sub> <sup> <small> <q>

sub tag, sup tag, small tag, q tag	
Tag	<sub> <sup> <small> <q>
Beschrijving	Elk van deze tags zorgt voor een wijziging aan standaard tekst. De sub tag wordt iets lager gepositioneerd t.o.v. van andere tekst op eenzelfde lijn. De sup tag werkt omgekeerd en wordt iets hoger gepositioneerd. De small tag verkleind de standaard grootte van de tekst. De q tag zorgt voor afdruk van aanhalingstekens.
Type	Inline level element

#### CODE

```
<!-- SUB TAG, SUP TAG, SMALL TAG, Q TAG !-->
<sub>Sub tag</sub>
<sup>sup tag</sup>
<small>small tag</small><br>
<small><q>small tag tussen aanhalingstekens</q></small><br>
```

#### RESULTAAT

Sub tag <sup>sup tag</sup> small tag  
“small tag tussen aanhalingstekens”

#### 4.5.1.1.15. <address>

address tag	
Tag	<address>
Beschrijving	De address tag kan tweeledig worden gebruikt. Het definieert de auteur van een artikel of wanneer het om het contact informatie gedeelte gaat tussen de body tags van een pagina.
Type	Block level

#### CODE

```
<!-- ADDRESS TAG -->
<address>
    Adres:<br>
    www.mijnsite.be<br>
    Straat, huisnummer, postcode, plaats
</address>
```

#### RESULTAAT

Adres:  
www.mijnsite.be  
Straat, huisnummer, postcode, plaats

#### 4.5.1.1.16. <dfn>

dfn tag	
Tag	<dfn>
Beschrijving	De dfn of ook de definition tag genoemd wordt gebruikt bij de afkorting van een woord waarvan de uitleg wordt gegeven ernaast. Je kan het vergelijken met de <abbr> tag die we reeds hebben besproken. Het verschil is de uitleg ervan in plaats dat de afkorting voluit wordt beschreven zoals bij de <abbr> tag.
Type	Block level

#### CODE

```
<!-- DFN TAG-->
<p><dfn>HTML</dfn> is de taal gebruikt binnen webpagina's</p>
```

#### RESULTAAT

HTML is de taal gebruikt binnen webpagina's

#### 4.5.1.2. CONTAINER TAGS

Om container tags uit te leggen beginnen we een nieuw document.

##### klik\_per\_klik\_2.html

Zorg ervoor dat je eenzelfde hoofding hebt zoals in het vorige document klik\_per\_klik\_1.html

```
<!DOCTYPE html>
<html lang="nl">

<head>
    <meta charset="UTF-8">
    <title>HTML 5</title>
    <meta name="description" content="Klik per klik tutorials">
    <meta name="keywords" content="HTML 5, CSS3, BOOTSTRAP">
    <meta name="author" content="Tom Vanhoutte, Full Stack Web Developer">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>

</body>
</html>
```

Tot nu toe hebben we enkel tags gezien die teksten of woorden op je scherm een betekenis geven. Er zijn bijvoorbeeld teksten die bij elkaar horen en die we graag samen zouden **groeperen**. Hiervoor gebruiken we enkele "containers" met elk hun eigen betekenis die we hier zullen overlopen.

Zonder het te beseffen heb je reeds te maken gehad met enkele van die containers die bijdragen tot de semantiek of ook wel de betekenis van de onderdelen van een site genoemd. In de bovenstaande schermafbeelding zie je bijvoorbeeld de `<head>` en de `<body>` tags staan die containers zijn en die andere tags bevatten. We hebben ook reeds gezien dat enkel tags met teksten binnen de body tags worden weergegeven op het scherm. Dit is dan ook onze werkomgeving.

De bekendste en oudste container in HTML is de `<div>` tag. In vorige versies van HTML was dit de belangrijkste tag. Om betekenis en semantiek te geven aan deze `<div>` tag gebruiken we het **id** attribuut.

#### 4.5.1.2.1. <div>

div	
Tag	<div>
Beschrijving	Het div element wordt vaak gebruikt als een container voor andere HTML-elementen om ze dan als geheel later te stylen met CSS en/of JAVASCRIPT.
Type	Block level

#### CODE

```
<!-- DIV TAG -->
<div id="webontwikkelaars">
    <p><dfn>Webontwikkelaars</dfn> zijn in staat applicaties te ontwikkelen voor het web!</p>
    <p><dfn>HTML</dfn> is de taal van het web</p>
    <p><dfn><abbr title="Cascading Style Sheets">CSS</abbr> wordt gebruikt om HTML te gaan stylen volgens design</dfn></p>
</div>
```

#### RESULTAAT

*Webontwikkelaars zijn in staat applicaties te ontwikkelen voor het web!*

*HTML is de taal van het web*

*CSS wordt gebruikt om HTML te gaan stylen volgens design*

Tot voor de opkomst van HMTL5 werd bovenstaande gebruikt om bepaalde blokken tekst te groeperen met een div attribuut als naamgeving.

OPMERKING: het attribuut id van een div tag is **UNIEK** binnen een webpagina, d.w.z. dat zoals in ons voorbeeld id="webontwikkelaars" slechts 1x mag voorkomen.

Bij grotere pagina's werd dit soms een wirwar van id's binnen HTML. Met de komst van HMTL5 werd hier een oplossing voor geboden d.m.v. "**section elementen**".

#### **4.5.1.2.2. SECTION ELEMENTEN**

Onderstaande zijn de basis elementen die iedere html5 pagina kan bevatten.  
De eerste hiervan is reeds gekend nl. de headings. De headings zijn ook de enige elementen die in alle voorgaande versies van html op identieke manier werken.  
De andere elementen zijn nieuw sinds html5.

- <h1> .. <h6>
- <article></article>
- <aside></aside>
- <nav></nav>
- <section></section>
- <header></header>
- <main></main>
- <footer></footer>

OPMERKING: iedere HTML5 tag kan ook een id attribuut bevatten.

##### **4.5.1.2.2.1. NAV TAG**

De navigatie tag wordt gebruikt om alle navigatie elementen weer te geven.

##### **4.5.1.2.2.2. MAIN TAG**

De main tag bevat alle inhoud van een pagina behalve aside, nav, header en footer. De main tag kan ook GEEN kind zijn van een article of een section.

##### **4.5.1.2.2.3. SECTION EN ARTICLE TAG**

Een section en article kunnen onafhankelijk van elkaar in een pagina komen te staan, maar afhankelijk van de inhoud kunnen deze ook genest worden.

Bijvoorbeeld: sport artikelen kunnen in de sport section worden gezet.

Een section kan dus meerdere article tags bevatten.

Maar je kan ook een article hebben over een bepaald onderwerp die je in verschillende sections kan onderverdelen.

Voorbeeld 1:

section = jupilerproleague. Deze bevat 2 articles over een nieuwsbericht van Anderlecht en één van Cercle Brugge. De section KAN een header bevatten en de articles HEBBEN een header. Dit voorbeeld zullen we straks uitwerken.

Voorbeeld2:

article = een sportartikel over Real Madrid. Dit article kan meerdere secties hebben zoals: video section, spelers section, ...

Zoals je ziet is NESTEN van sections en articles mogelijk.

#### **4.5.1.2.2.4. HEADER TAG**

Een header tag bevat een inleiding of intro voor een bepaalde inhoud. Bijv. de article tag kan als kind een header hebben. Deze header heeft dan als kinderen bijvoorbeeld een heading tag.

```
<article>
    <header>
        <h2>mijn blogbericht</h2>
    </header>
    <p>dit is de inhoud van mijn blogbericht</p>
</article>
```

#### **4.5.1.2.2.5. ASIDE TAG**

De aside tag bevat randinformatie die je op de pagina in de kijker wil stellen. Dit kan bijvoorbeeld een belangrijk blog bericht zijn. Daarnaast wordt de aside tag ook gebruikt om bijvoorbeeld een zijmenu weer te geven.

#### **4.5.1.2.2.6. FOOTER TAG**

De footer tag wordt gebruikt om items in de kijker te stellen onderaan de pagina. Elementen hier kunnen zijn:

- copyright
- contact informatie
- sitemap
- links

We zullen dit in de praktijk even toetsen en de pagina nadien laten VALIDEREN. Inderdaad, je kan je HTML5 code laten valideren op juistheid naar semantiek en structuur.

Hieronder een schematische voorbeeld over hoe je HTML5 elementen binnen een pagina kan structureren.

<nav></nav>	
<h1>Onderwerp van de pagina</h1>	
<main> <section> <header> <h2>subtitel van het onderwerp</h2> </header> <article> <header> <h3>artikel 1 van de subtitel</h3> </header> </article> <article> <header> <h3>artikel 2 van de subtitel</h3> </header> </article> </section> </main>	<aside>zijmenu of randinformatie aangaande het onderwerp.</aside>
<footer></footer>	

Pas **klik\_per\_klik2.html** als volgt aan:  
Wis de div tag van webontwikkelaars volledig zodat je een blanco pagina hebt.

## CODE

```
<body>
  <nav>
    <ul>
      <li><a href="http://www.google.be">GOOGLE</a></li>
      <li><a href="http://www.bing.com">BING</a></li>
    </ul>
  </nav>
  <h1>Voetbalcompetitie</h1>
  <main>
    <section id="jupilerproleague">
      <header>
        <h2>Jupiler Pro League</h2>
      </header>
      <article>
        <header>
          <h3>Anderlecht, 3de trainer in 3 maanden!</h3>
        </header>
        <p>Het gaat van kwaad naar erger bij Anderlecht.  

          Zal de 3de trainer verandering kunnen brengen in deze jonge ploeg!
        </p>
      </article>
      <article>
        <header>
          <h3>Cercle door het oog van de naald!</h3>
        </header>
        <p>In de allerlaatste minuut van de wedstrijd, sleept Cercle nog een gelijkspel uit de brand!
        </p>
      </article>
    </section>
    <section id="proximusleague">
      <header>
        <h2>Proximus League</h2>
      </header>
      <article>
        <header>
          <h3>KV Mechelen wint de beker van België!</h3>
        </header>
        <p>Het is van de jaren 70 geleden dat een tweedeklasser nog eens de beker van België won!
        </p>
      </article>
      <article>
        <header>
          <h3>Wie promoveert er mee naar de Proximus League</h3>
        </header>
        <p>Beerschot en KV Mechelen vechten een nek aan nek race uit. Benieuwd wie volgend jaar
          in de hoogste klasse zal spelen!
        </p>
      </article>
    </section>
    <main>
      <footer>Hier komt meestal informatie zoals telefoon, adres, locatie, copyright, ... te staan</footer>
    </main>
  </body>
```

## RESULTAAT

- [GOOGLE](#)
- [BING](#)

### Voetbalcompetitie

#### Jupiler Pro League

##### Anderlecht, 3de trainer in 3 maanden!

Het gaat van kwaad naar erger bij Anderlecht. Zal de 3de trainer verandering kunnen brengen in deze jonge ploeg!

##### Cercle door het oog van de naald!

In de allerlaatste minuut van de wedstrijd, sleept Cercle nog een gelijkspel uit de brand!

#### Proximus League

##### KV Mechelen wint de beker van België!

Het is van de jaren 70 geleden dat een tweedeklasser nog eens de beker van België won!

##### Wie promoveert er mee naar de Proximus League

Beerschot en KV Mechelen vechten een nek aan nek race uit. Benieuwd wie volgend jaar in de hoogste klasse zal spelen!

Hier komt meestal informatie zoals telefoon, adres, locatie, copyright, ... te staan

We zullen nu de bovenstaande code valideren.

Ga hiervoor naar de site [https://validator.w3.org/#validate\\_by\\_input](https://validator.w3.org/#validate_by_input)

W3C is de site die de richtlijnen en semantiek van het web controleert. Deze site bevat de basis en richtlijnen voor iedere website.

Kopieer de html5 code van je pagina volledig en plak deze in onderstaand scherm.

W3C® Markup Validation Service  
Check the markup (HTML, XHTML, ...) of Web documents

Validate by URI   Validate by File Upload   Validate by Direct Input

Validate by direct input

Enter the Markup to validate:

```
<!DOCTYPE html>
<html lang="nl">
<head>
    <meta charset="UTF-8">
    <title>HTML 5</title>
    <meta name="description" content="Klik per klik tutorials">
    <meta name="keywords" content="HTML 5, CSS3, BOOTSTRAP">
    <meta name="author" content="Tom Vanhoutte, Full Stack Web Developer">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
```

More Options

Check

Druk nu op de check button om een resultaat te verkrijgen.

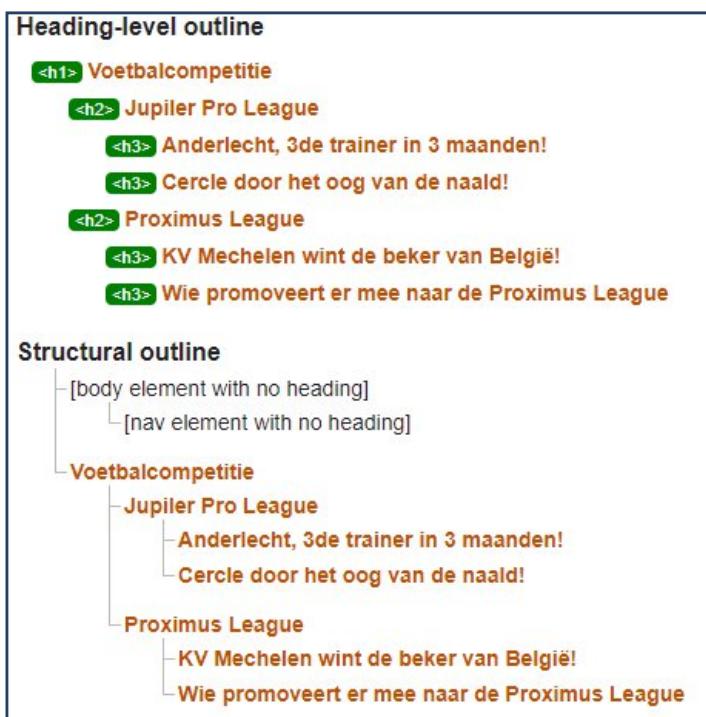
Wanneer je geen fouten in je html code hebt staan dan dien je onderstaand scherm te krijgen:

The screenshot shows the W3C Markup Validation Service interface. At the top, there are checkboxes for 'source' (checked), 'outline' (checked), 'image report' (unchecked), and 'Options...'. Below that, a dropdown menu 'Check by' is set to 'text input' (unchecked) and 'css'. The main area contains the following HTML code:

```
<!DOCTYPE html>
<html lang="nl">
<head>
    <meta charset="UTF-8">
    <title>HTML 5</title>
    <meta name="description" content="Klik per klik tutorials">
    <meta name="keywords" content="HTML 5, CSS3, BOOTSTRAP">
    <meta name="author" content="Tom Vanhoutte, Full Stack Web Developer">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
    <nav>
        <ul>
```

Below the code is a 'Check' button. A message below it says: 'Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.' There is also a 'Message Filtering' button. At the bottom, a green bar indicates: 'Document checking completed. No errors or warnings to show.'

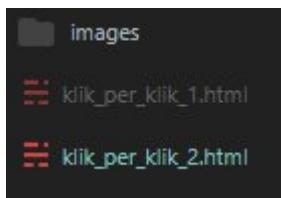
Vink nu bovenaan dit scherm **outline** aan druk opnieuw op check. Scroll nadien helemaal tot onderaan de pagina. Alles dient in het groen te staan. Je pagina is perfect opgebouwd. Merk op dat ook body en nav een header tag kunnen bevatten indien nodig. Dit geeft echter geen fout terug. Een header tag is niet verplicht. LET OP: we spreken hier over de header tag en niet over de headings zoals h1 t.e.m. h6



### 4.5.1.3. AFBEELDINGEN

We weten nu reeds hoe we text en container tags binnen de body tag kunnen opbouwen. Maar een pagina stelt natuurlijk niets voor zonder het nodige beeld materiaal. Hier zullen we in detail bespreken hoe we afbeeldingen op het scherm toveren!

Maak een mapje met de naam images aan naast je html documenten die je tot nu toe hebt aangemaakt.



Zoek nu online achter het logo van de Jupiler Pro League en deze van de Proximus League.

#### 4.5.1.3.1. IMG TAG

Image tags die we gebruiken zijn o.a. png, gif of jpg/jpeg en svg. De verschillen zullen later duidelijker worden in dit boek.

Voeg de afbeeldingen nu toe in je pagina als volgt:

```
<header>
    <h2>Jupiler Pro League</h2>
    
</header>
```

```
<header>
    <h2>Proximus League</h2>
    
</header>
```

Het src attribuut bevat de bestandsnaam en/of locatie.

Het alt attribuut bevat informatie over de afbeelding

Height en Width zorgen voor de grootte van de afbeelding op het scherm.

## RESULTAAT



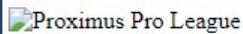
### Anderlecht, 3de trainer in 3 maanden!

Het gaat van kwaad naar erger bij Anderlecht. Zal de 3de trainer verandering kunnen brengen in deze jonge ploeg!

### Cercle door het oog van de naald!

In de allerlaatste minuut van de wedstrijd, sleept Cercle nog een gelijkspel uit de brand!

### Proximus League



Zoals je ziet wordt de link wel weergegeven maar wordt de afbeelding niet weergegeven. Dit komt omdat we in het src attribuut naar de ROOT (=de basis) verwijzen waar ook ons document staat en NIET naar het mapje afbeeldingen waarin de afbeeldingen staan. Sinds HTML5 kunnen we gebruiken maken van de <base> tag die in de head van de pagina dient te worden toegevoegd.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<base href="d:/tom/fork/cursussen/HTML5%20en%20CSS3/resources/images/" target="_blank">
```

LET OP: afhankelijk waar jullie je pagina hebben bewaard op je pc zal de href tag natuurlijk verschillen.

In realiteit wanneer deze pagina op een echt domein online komt te staan zal je onderstaande zien staan.

```
<base href="http://www.mijndomein.be/images" target="_blank">
```

## RESULTAAT



### Anderlecht, 3de trainer in 3 maanden!

Het gaat van kwaad naar erger bij Anderlecht. Zal de 3de trainer verandering kunnen brengen in deze jonge ploeg!

### Cercle door het oog van de naald!

In de allerlaatste minuut van de wedstrijd, sleept Cercle nog een gelijkspel uit de brand!

### Proximus League



Sinds de komst van HTML kunnen we ook hier meer semantiek meegeven met de tags figcaption en figure.

```
<header>
  <h2>Jupiler Pro League</h2>
  <figure>
    
    <figcaption>Afbeelding 1. Logo van de Jupiler Pro League</figcaption>
  </figure>
</header>
```

## RESULTAAT



Afbeelding 1. Logo van de Jupiler Pro League

### 4.5.1.3.2. IMAGE MAPS

Image maps zijn hotspots op een afbeelding. Met deze tags kunnen we op één afbeeldingen meerdere afzonderlijke klikbare elementen plaatsen.

Maak hiervoor een afzonderlijk bestand aan: **klik\_per\_klik\_3.html**

We bouwen vanaf nu volgens de standaarden iedere pagina op. Hier gaan we naast de afbeelding terug een voorbeeld weergeven van het gebruik van een article en section. In tegenstelling tot het vorige voorbeeld heeft deze pagina één article met verschillende sections. Daarnaast maken we gebruik van de aside tag die als randinformatie dient aangaande het article.

```
<head>
    <meta charset="UTF-8">
    <title>HTML 5 - IMAGE MAPS</title>
    <meta name="description" content="Klik per Klik tutorials">
    <meta name="keywords" content="HTML 5, CSS3, BOOTSTRAP">
    <meta name="author" content="Tom Vanhoutte, Full Stack Web Developer">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <base href="d:/tom/fork/cursussen/HTML5%20en%20CSS3/resources/images/" target="_blank">
</head>

<body>
    <!-- IMAGE MAPS -->
    <h1>IMAGE MAPS</h1>
    <main>
        <article>
            <header>
                <h2>Dagelijkse gezonde maaltijd.</h2>
            </header>
            <p>Neem een kijkje in hoe mijn perfecte maaltijd er uit ziet!</p>
            <section id="voormiddag">
                <header>
                    <h3>voormiddag</h3>
                </header>
                <article>
                    <header>
                        <h4>>>Gezond ontbijt</h4>
                    </header>
                    <p>Een goed ontbijt zal je metabolisme en vetverbranding vlugger op gang krijgen.</p>
                    <p>Op onderstaande fruitschaal of fruitsap kan je klikken. Deze 2 afbeeldingen staan in een apart html bestand.</p>
                    
                    <map name="ontbijtmap">
                        <area shape="rect" coords="571,112,425,367" href="../fruitsap.html" alt="Fruitsap">
                        <area shape="circle" coords="273,112,113" href="../fruitschaal.html" alt="Fruitschaal">
                    </map>
                </article>
            </section>
            <section id="middag">
                <header>
                    <h3>middag</h3>
                </header>
                <article>
                    <h4>Eén bord met veel groeten</h4>
                    <p>Zorg ervoor dat je je bord met de helft aan groenten vult.</p>
                    <p>Een stuk vlees tot max. 150g is meer dan voldoende voor een persoon.</p>
                </article>
            </section>
            <section id="avond">
                <header>
                    <h3>avond</h3>
                </header>
                <article>
                    <h4>Lichte maaltijd</h4>
                    <p>'s Avonds heb je in principe je lichtste maaltijd voor het slapengaan</p>
                    <p>Bruin brood met beleg is hier voldoende eventueel aangevuld met een lichte yoghurt.</p>
                    <p>Bij een hongergevoeld achteraf, drink je voldoende water.</p>
                </article>
            </section>
        </main>
    </body>
```

Zoals je in bovenstaande schermafbeelding kan zien gaat dit over één artikel, nl. "dagelijkse gezonde maaltijd" met daarin 3 sections: voormiddag, middag, en avond. Ook deze secties kunnen natuurlijk terug verschillende artikelen bevatten. Alles hangt dus in feit af van het design van de pagina waarvan we vertrekken.

Dit noemen we het nesten van sections en artikelen.

Zoals reeds gezegd plaatsen we ook hier een aside tag. De aside tag bevat randinformatie over een artikel die niet rechtstreeks betrekking heeft tot dit artikel. (aside wordt ook gebruikt om een tweede menu weer te geven).

```
<section id="avond">
  <header>
    <h3>avond</h3>
  </header>
  <article>
    <header>
      <h4>Lichte maaltijd</h4>
    </header>
    <p>'s Avonds heb je in principe je lichtste maaltijd voor het slapengaan</p>
    <p>Bruin brood met beleg is hier voldoende eventueel aangevuld met een lichte yoghurt.</p>
    <p>Bij een hongergevoeld achteraf, drink je voldoende water.</p>
  </article>
</section>
<aside>
  <p>Je kan steeds raad vragen aan een diëtist over een gezond voedingspatroon</p>
  <p>Eén van de grote tips is de volgende: kijk niet naar de producten met het
    <q>light</q> logo op de verpakking, maar bekijk het aantal <abbr title="Kilo Calorien">KCAL</abbr></p>
</aside>
</article>
```

Wanneer we ons nu focussen op de afbeelding dan zal je het volgende zien staan:

```

<map name="ontbijtmap">
  <area shape="rect" coords="571,112,425,367" href="../fruitsap.html" alt="Fruitsap">
  <area shape="circle" coords="273,112,113" href="../fruitschaal.html" alt="Fruitschaal">
</map>
```

De gebruikelijke img tag, maar ditmaal met een nieuw attribuut **usemap** voorafgegaan door een **hashtag**.

Deze usemap zorgt voor de verbinding met een nieuwe tag **<map>** en zijn attribuut **name**. Name bevat dezelfde naamgeving maar zonder de hashtag. In feite gaan we één afbeelding tonen en deze linken met 2 andere pagina's die elk hun eigen afbeelding hebben. In ons voorbeeld hieronder staan er op de foto een fruitstap en een fruitschaal afgebeeld. Deze willen we klikbaar maken. Hiervoor dienen we eerst onze fruitschaal en fruitsapje uit te knippen met een grafisch programma naar keuze. We bewaren deze afbeeldingen in ons **images** mapje. (**fruitsap.png** en **fruitschaal.png**)

Daarnaast dient elke afzonderlijke afbeelding ook in een html bestand te staan. Maak 2 nieuwe bestanden aan:

### Fruitschaal.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>HTML 5 - IMAGE MAPS</title>
    <meta name="description" content="Klik per klik tutorials">
    <meta name="keywords" content="HTML 5, CSS3, BOOTSTRAP">
    <meta name="author" content="Tom Vanhoutte, Full Stack Web Developer">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <base href="d:/tom/fork/cursussen/HTML5%20en%20CSS3/resources/images/" target="_blank">
</head>
<body>
    
</body>
</html>
```

### Fruitsap.html:

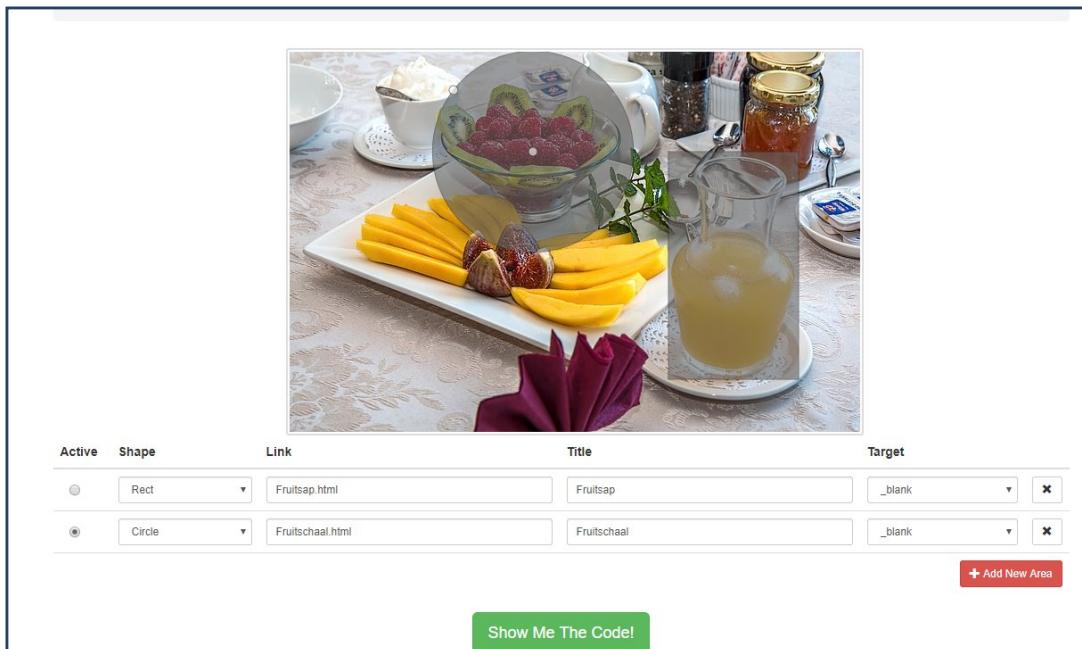
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>HTML 5 - IMAGE MAPS</title>
    <meta name="description" content="Klik per klik tutorials">
    <meta name="keywords" content="HTML 5, CSS3, BOOTSTRAP">
    <meta name="author" content="Tom Vanhoutte, Full Stack Web Developer">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <base href="d:/tom/fork/cursussen/HTML5%20en%20CSS3/resources/images/" target="_blank">
</head>
<body>
    
</body>
</html>
```

We keren nu terug naar ons origineel bestand: `klik_per_klik_3.html` om de coördinaten te bepalen van de klikgebieden op de originele foto.

Online bestaat er een fantastische site die dit werk voor ons doet, nl.

<https://www.image-map.net/>

Enkel de foto opladen en de afbeeldingen selecteren en je krijgt de code voor de coördinaten die je nodig hebt.



Tot nu toe hebben we enkel .png of .jpg bestanden gebruikt.

Een woordje uitleg is hier op zijn plaats.

Wanneer het aankomt op het opslaan van afbeeldingen voor gebruik op het web, zijn er een aantal bestandstypen die u kunt gebruiken. De drie meest voorkomende opties zijn PNG, JPEG (of JPG) en GIF. Hoewel deze formaten populair zijn, heeft elk zijn eigen unieke voor- en nadelen:

JPEG's kunnen zeer gedetailleerde afbeeldingen met veel kleuren weergeven, waardoor ze ideaal zijn voor foto's. Tegelijkertijd zijn de bestanden vaak erg groot en houden ze niet altijd goed stand onder compressie.

PNG's zijn ideaal voor afbeeldingen zonder veel gegevens, zoals logo's of schermafbeeldingen van de interface. Ze zijn uitstekend in behoud van kwaliteit wanneer ze worden samengeperst en ondersteunen transparantie, maar werken niet goed voor foto's.

GIF's zijn uitstekend voor animaties, maar niet handig voor het opslaan van statische afbeeldingen.

WebP-afbeeldingen zijn een beeldformaat van Google waarmee u afbeeldingen op internet op een vergelijkbaar kwaliteitsniveau kunt weergeven als bestaande afbeeldingsindelingen, maar met een kleinere bestandsgrööte die dan een snellere pagina laadtijd zal garanderen..

Om dit te bereiken biedt WebP zowel 'lossy' als 'lossless' compressie-opties. De laatste bewaart meer gegevens, terwijl de eerste de resulterende bestandsgröötes nog kleiner maakt.

De toekomst is wel degelijk WebP. Er zijn sites zoals online-convert.com die je png bestand omzetten naar WebP.

Maar we dienen steeds rekening te houden met alle grote browsers. Tot op vandaag is de grote concurrent van Google, nl. Apple nog steeds niet in dit verhaal meegestapt.

Als laatste kunnen we het hebben over SVG afbeeldingen. Dit zijn VECTORIELE afbeeldingen. D.w.z. dat deze soort van afbeeldingen geschaald kunnen worden tot gelijk welke grootte zonder resolutieverlies.

We kunnen dit controleren op een site die door alle webdevelopers nauwgezet in de gaten wordt gehouden, nl. [caniuse.com](http://caniuse.com)

Hieronder zie je een schermafbeelding waar de browser van Apple, nl. Safari niet meespeelt in dit verhaal. Aangezien er een grote groep mensen hiervan gebruik maakt is WebP nog niet als standaard te beschouwen op het moment van dit schrijven.



Bovenstaand screenshot kan dus verouderd zijn omdat deze technologie in constante beweging is!

#### 4.5.1.4. FORMULIEREN

In dit onderdeel zullen we het hebben over formulieren en zijn form elementen. Het bekendste formulier wereldwijd is natuurlijk het contact formulier op een website.

##### 4.5.1.4.1. FORM TAG

Een formulier in HTML beginnen we steeds met de form tag.  
Maak een nieuw bestand aan: **klik\_per\_klik\_4.html**.

De form tag heeft 2 attributen, nl. action en method. Alles wat in een formulier wordt ingevuld, dient verstuurd te worden naar een andere locatie (pagina) die de ingevulden waarden zal ontvangen. In dit voorbeeld zal de action\_page.php deze waarden ontvangen. De methode hoe we dit versturen is simpel, nl. we POSTEN (versturen) alle ingevulden velden.

```
<!DOCTYPE html>
<html lang="nl">

<head>
    <meta charset="UTF-8">
    <title>HTML 5 - FORMULIEREN</title>
    <meta name="description" content="Klik per klik tutorials">
    <meta name="keywords" content="HTML 5, CSS3, BOOTSTRAP">
    <meta name="author" content="Tom Vanhoutte, Full Stack Web Developer">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <base href="d:/tom/fork/cursussen/HTML5%20en%20CSS3/resources/images/" target="_blank">
</head>

<body>
    <h1>Een overzicht van de meest gebruikte velden in een formulier.</h1>
    <form action="/action_page.php" method="POST">
```

## 4.5.1.4.2. FORM ELEMENTEN

### 4.5.1.4.2.1. INPUT VELDEN

Input velden zijn single line inputvelden. Er bestaan verschillende types in HTML5 die we kunnen gebruiken. De belangrijkste zal uiteindelijk de SUBMIT knop zijn die de action pagina zal oproepen. Deze button zal je meestal ook helemaal onderaan je formulier weergeven juist voor de closing form tag.

#### CODE

```
<body>
  <h1>Een overzicht van de meest gebruikte velden in een formulier.</h1>
  <form action="/action_page.php" method="POST">
    <h2>Inputvelden</h2>
    <p>Inputvelden worden gebruikt om single-line input mee te geven. De attributen zijn type, name en value.</p>
    <p>Het value attribuut bevat de inhoud van het veld</p>
    <p>Het name attribuut bevat de naam zelf. Dit wordt gebruikt om het veld als variabele aan te spreken.</p>
    <p>Het type bevat meerdere mogelijkheden: checkbox, color, date, datetime-local, email, file, hidden, image, month, number, password, radio, range, reset, search, submit, tel, text, time, url, week</p>
    <p>Hieronder volgen alle mogelijkheden</p>
    Text: <input type="text" name="VoorNaam" value="Tom"><br>
    Checkbox: <input type="checkbox"><br>
    Color: <input type="color"><br>
    Date: <input type="date"><br>
    Datetime-local: <input type="datetime-local"><br>
    Email: <input type="email"><br>
    File: <input type="file"><br>
    Image: <input type="image"><br>
    Month: <input type="month"><br>
    Number: <input type="number"><br>iNU
    Password: <input type="password"><br>
    Radio: <input type="radio"><br>
    Range: <input type="range"><br>
    Reset: <input type="reset"><br>
    Search: <input type="search"><br>
    Tel: <input type="tel"><br>
    Time: <input type="time"><br>
    Url: <input type="url"><br>
    Week: <input type="week"><br>
```

#### RESULTAAT

### Een overzicht van de meest gebruikte velden in een formulier.

#### Inputvelden

Inputvelden worden gebruikt om single-line input mee te geven. De attributen zijn type, name en value.

Het value attribuut bevat de inhoud van het veld

Het name attribuut bevat de naam zelf. Dit wordt gebruikt om het veld als variabele aan te spreken.

Het type bevat meerdere mogelijkheden: checkbox, color, date, datetime-local, email, file, hidden, image, month, number, password, radio, range, reset, search, submit, tel, text, time, url, week

Hieronder volgen alle mogelijkheden

Text:

Checkbox:

Color:

Date:

Datetime-local:

Email:

File:  Bestand kiezen Geen bestand gekozen

Image:  Verzenden

Month:

Number:

Password:

Radio:

Range:

Reset:

Search:

Tel:

Time:

Url:

Week:

#### 4.5.1.4.2.2. TEXTAREA

De text area is ook een inputveld, maar een multiline inputveld waar je kan meegeven met de attributen rows en cols hoe groot het textarea vak dient te worden weergegeven op de pagina.

##### CODE

```
<h2>TextArea</h2>
<textarea rows="4" cols="50">
    Een textarea is een multiline inputveld waar je kan kiezen hoeveel rijen en kolommen worden getoond.
</textarea><br>
```

##### RESULTAAT

**TextArea**

Een textarea is een multiline inputveld waar je kan kiezen hoeveel rijen en kolommen worden getoond.

#### 4.5.1.4.2.3. DROPODOWN LIJST

Een dropdownlijst dienen we met 2 tags te combineren, nl. select die de groep zal voorstellen en option die de waarden (value) zal bevatten. De tekst zelf in de dropdownlijst schrijven we juist voor de closing tag.

##### CODE

```
<select name="cursus" id="">
    <option value="HTML5">HTML5</option>
    <option value="CSS3">CSS3</option>
</select>
```

##### RESULTAAT

**Dropdown lijst (select)**

HTML5 ▾

## 5. CSS 3

CSS staat voor **Cascading Style Sheets**. We zitten reeds aan de 3de versie anno 2019. Tot nu toe kennen we HTML 5 als platte tekst en weten we hoe we teksten dienen te benoemen.

Start met een nieuw document: **klik\_per\_klik\_5.html**.

### 5.1. STYLES

We kennen 3 types styles:

- inline
- internal
- external

#### 5.1.1. INLINE

Inline styles is css code die tussen html tags binnen het document wordt geschreven. In onderstaand voorbeeld zullen we aan de p tag vier css text properties meegeven: color, font-size, font-style en text-align. We zullen dus een kleur, tekstgrootte, text-stijl en positionering meegeven. Merk op dat we de grootte momenteel in px. meegeven. Straks meer hierover.

#### CODE

```
<body>
    <p style = "color:#456897;
                  font-size:100px;
                  font-style:italic;
                  text-align:center;">
        CSS3 inline styles</p>
</body>
```

#### RESULTAAT



CSS3 inline styles

## 5.1.2. INTERNAL

Net zoals bij inline css zal ook internal css BINNEN het html document worden gecodeerd. Het verschil bestaat er hierin dat we de css code in de head tag van het document zullen schrijven.

Start een nieuw document: **klik\_per\_klik\_6.html**.

### Eerste mogelijkheid: class

We schrijven nu identiek hetzelfde maar internal. Merk op dat we hier gebruiken maken van het class attribuut. In feite geef je een naam mee aan de p tag die refereert naar de styles in de header. Let op dat je class namen steeds met een PUNT laat voorafgaan. Zo weet de browser hoe hij de p-tag kan stijlen. Het voordeel hiervan is het volgende: Je kan meerdere elementen op dezelfde manier stijlen binnen je HTML pagina.

### CODE

```
<style>
    .mijnstijl {
        color: #456897;
        font-size: 100px;
        font-style: italic;
        text-align: center;
    }
</style>
</head>

<body>
    <p class="mijnstijl">
        CSS3 internal styles</p>
    <p class="mijnstijl">
        Tweede lijn, zelfde stijl</p>
</body>

</html>
```

### RESULTAAT



## Tweede mogelijkheid: id

We kunnen identiek dezelfde code gaan gebruiken als bij classes, met dit verschil. Wanneer we een id attribuut i.p.v. een class attribuut gebruiken dan mag dit slechts EEN KEER op een pagina worden weergegeven.

M.a.w. wanneer je meerdere p tags dezelfde stijl zou willen meegeven dan gebruik je class, wanneer je slechts EEN ELEMENT wil stylen op de pagina, dan gebruik je ID. Een ID wordt in css voorafgegaan door een HASHTAG.

TWEE KEER hetzelfde ID gebruiken zou een error geven in het valideren van je HTML pagina.

Onderstaande is dus correct.

## CODE

```
<style>
#mijnstijl {
    color: #456897;
    font-size: 100px;
    font-style: italic;
    text-align: center;
}
</style>
</head>

<body>
    <p id="mijnstijl">
        CSS3 internal styles</p>
</body>
```

## RESULTAAT

The result shows the text "CSS3 internal styles" displayed in a large, italicized, blue font, centered on the page. This is the output of the CSS3 internal styles provided in the code block above.

### 5.1.3. EXTERNAL

We weten nu reeds hoe we kunnen stylen binnen de HTML pagina. Dit vermijden we als developers zo veel mogelijk. Deze derde optie is de optie die we zullen gebruiken in onze projecten.

We dienen hiervoor een css bestand te creëren en te linken met onze HTML pagina.

Start een nieuw html bestand: klik\_per\_klik\_7.html

Start een nieuw css bestand: styles.css

Het linken van een stylesheet waar we onze css zullen schrijven met het html bestand doen we als volgt:

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

Wanneer we nu dezelfde oefening zouden doen, dan plakken we eigenlijk dezelfde code in ons css bestand.

```
#mijnstijl {
    color: #456897;
    font-size: 100px;
    font-style: italic;
    text-align: center;
}
```

Het html bestand heeft dat geen style tag meer binnen de head tag, maar de code blijft dezelfde.

```
<link rel="stylesheet" href="styles.css">
</head>
<body>
    <p id="mijnstijl">
        CSS3 external styles</p>
```

Probeer maar uit! Je zal zien dat dit werkt!

## 5.2. BOX MODEL

Tijdens de eerste pagina's van deze cursus hebben we het even gehad over het box model. Het box model is een voorstelling van elke individuele tag waarop je klikt.

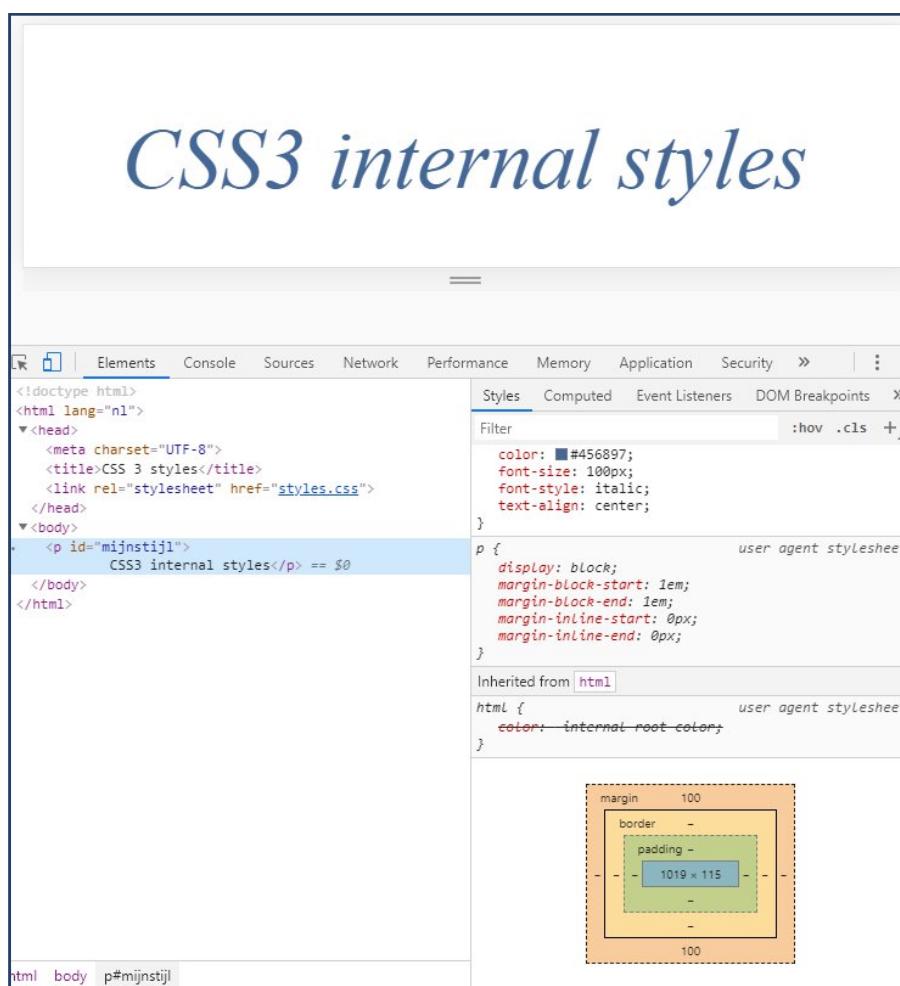
Aan de rechterkant zie je duidelijk het tabblad Styles actief staan met een grafische weergave van het boxmodel. Momenteel staan we geparkeerd op de p tag aan de linkerkant van je scherm.

Het blauwe gedeelte van het boxmodel gaat over de inhoud/content en de grootte die hiervoor wordt voorzien. In onderstaand voorbeeld is dit 1019x115. (breedte x hoogte). De breedte kan hier verschillen naargelang jullie schermweergave.

Je ziet tevens dat er bij de p tag een margin aan de bovenkant en de onderkant wordt gerekend. (margin-top en margin-bottom)

We spreken hier, van margin-top, margin-right, margin-bottom en margin-bottom.

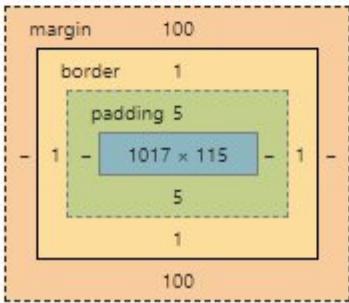
We lezen dus mee volgens de wijzers van de klok beginnend bij margin-top. Iedere tag heeft zijn eigen margin waarmee we rekening dienen te houden.



Als voorbeeld voegen we tevens een padding toe. Een padding bevat de ruimte tussen de tekst en eventueel een border (rand). Tegelijkertijd geven we meteen een border mee.

Als developer zal je steeds je styles eerst in het rechtergedeelte van het inspect element coderen vooraleer je dan de code plakt in je echte code. (element.style) Zo kan je live zien of je code correct werkt i.p.v. eerst te coderen en te gokken of je padding, margin of border goed zit.

```
element.style {  
    padding-top: 5px;  
    padding-bottom: 5px;  
    border: 1px solid #456897;  
}  
  
#mijnstijl {  
    color: #456897;  
    font-size: 100px;  
    font-style: italic;  
    text-align: center;  
}  
  
p {  
    display: block;  
    margin-block-start: 1em;  
    margin-block-end: 1em;  
    margin-inline-start: 0px;  
    margin-inline-end: 0px;  
}  
  
Inherited from html  
  
html {  
    color: internal root color;  
}
```



### 5.3. CSS TEXT PROPERTIES

Enkele voorbeelden van css text properties (attributen) die gebruikt kunnen worden. Gaandeweg zal deze lijst serieus worden uitgebreid en zullen jullie deze uit het hoofd kennen. Deze lijst is alles behalve eindig.

- color
- font-size
- font-style
- text-align
- text-decoration
- text-transform
- text-indent
- letter-spacing
- line-height
- text-direction
- word-spacing
- text-shadow
- vertical-align

### 5.4. CSS FONTS

- font-family:
  - font-family: "Times New Roman", Times, serif;
- font-style:
  - normal
  - italic
  - oblique
- font-size:
  - px
  - em
  - rem

Voor de font-size zullen we meestal gebruik maken van rem omdat het bootstrap framework vanaf versie 4 hier ook gebruik van maakt.

Voorbeeld:

Wanneer de standaardgrootte in de webbrowser voor een p tag 16 pixels is, dan staat dit gelijk aan 1em of 1rem.

Wanneer we de p tag bijvoorbeeld op 24 pixels standaard wensen te zetten voor de volledige site dan doen we dit in het css bestand als volgt:  
p{ font-size: 1.5rem; } (= 16px \* 1.5rem = 24 pixels)

De keuze van em of rem hangt af van het bepalen van de start van je pagina.

Wanneer je start vanaf de html tag = rem

Wanneer je start vanaf de body tag = em

Hieronder een voorbeeld van de mogelijkheden aangaande px, em en rem:  
de standaard font-size van een browser (google chrome) is 16px.

Wanneer we een p-tag zouden coderen dan is die **16px groot**.

Je zou in je eigen css bestand het volgende dus kunnen schrijven.

**1ste manier: hier bepalen we enkel voor de p tag de grootte.**

```
p {  
    font-size:8px;  
}
```

**2de manier: em vertrekt vanaf de body tag.** We zetten de body tag op een font-size van 100% = 16px. Kijk hieronder wat het resultaat hiervan is op een h1 tag en een p tag.

```
body{  
    font-size:100% /*16px*/
```

```
h1{  
    font-size:1em; /*16px*/  
    margin-bottom:1em; /*32px*/  
}  
p{  
    font-size:1em; /*16px*/  
    margin-bottom:1em; /*16px*/  
}
```

**3de manier: rem vertrekt vanaf de html tag.** We doen nu net hetzelfde met het volgende resultaat

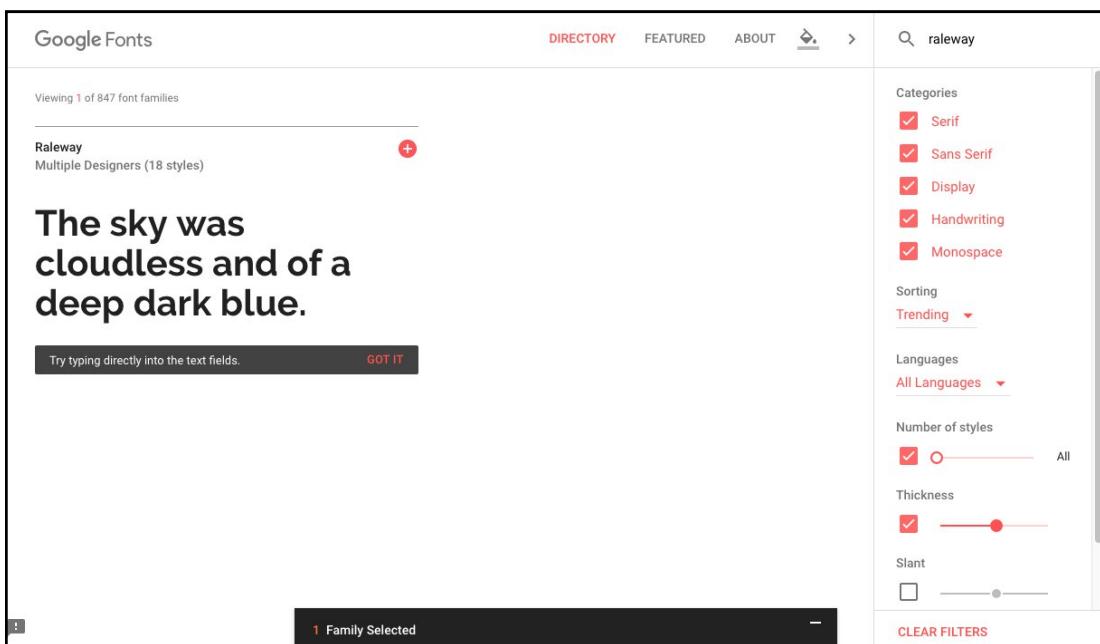
```
html{  
    font-size:100% /*16px*/  
}  
h1{  
    font-size:1rem; /*16px*/  
    margin-bottom:1rem; /*16px*/!!!!!!  
}  
p{  
    font-size:1rem; /*16px*/  
    margin-bottom:1rem; /*16px*/  
}
```

**Het bootstrap framework zal voor de 3de manier kiezen, nl. rem omdat deze éénduidig is van de root van het document, nl. de html tag.**

- font-variant:
  - normal
  - small-caps
- font-weight;
  - bold, 100, 200, ..., 900, ...

### 5.4.1. GOOGLE FONTS

Google Fonts zijn gratis fonts die je kan toepassen binnen je webpagina. Je kan natuurlijk ook betalende fonts gebruiken maar die dien je dan mee op te laden naast je site. Wanneer je fonts van adobe gebruikt (alternatief) dan kan je via de adobe typekit (betalend!), de fonts gebruiken die een designer in een webdesign heeft gebruikt. De reden hiervoor is simpel: de fonts die gebruikt worden zijn online of lokale fonts op de pc van de gebruiker. Wanneer het font niet geïnstalleerd is op de pc van de gebruiker dan zal hij niet het font volgens de opmaak weergeven, maar een standaard font gebruiken van die lokale pc. Dit kan een pagina naar lay-out en design breken.



The screenshot shows the Google Fonts interface after selecting the 'Raleway' font family. At the top, it says '1 Family Selected'. Below that, 'Your Selection' is listed with a 'Clear All' link. There are two tabs: 'EMBED' (which is selected) and 'CUSTOMIZE'. A green button at the top right says 'Load Time: Fast'. Under the 'EMBED' tab, there's a section titled 'Embed Font' with instructions to copy the provided code into the <head> of an HTML document. It offers two options: 'STANDARD' and '@IMPORT'. The standard option provides the following code:

```
<link href="https://fonts.googleapis.com/css?family=Raleway" rel="stylesheet">
```

Below this, under 'Specify in CSS', it says to use the following CSS rule:

```
font-family: 'Raleway', sans-serif;
```

At the bottom, it says 'For examples of how fonts can be added to webpages, see the [getting started guide](#)'.

## 5.5. CSS LINKS

- link : niet bezochte link
- visited: bezochte link
- hover: muis over link
- active: geselecteerde link

## 5.6. CSS LISTS

- lists-style-type
- list-style-image
- list-style-position

## 5.7. POSITIONEREN VAN HTML ELEMENTEN

### 5.7.1. FLOATING

Pas je document aan:

Je kan HTML elementen laten "floaten" naar de linkerkant of de rechterkant binnen de body van een pagina.

Wanneer je dus 3 blokken onder elkaar zou staan hebben, dan kan je de float property in css gebruiken om deze te positioneren.



### HTML

```
<body>
  <p id="mijnstijl">
    CSS3 external styles</p>

  <div id="blokA">FLOAT LEFT</div>
  <div id="blokB">FLOAT LEFT</div>
  <div id="blokC">FLOAT RIGHT</div>
</body>
</html>
```

## CSS

```
#mijnstijl {  
    color: #456897;  
    font-size: 100px;  
    font-style: italic;  
    text-align: center;  
    padding-top: 5px;  
    padding-bottom: 5px;  
    border: 1px solid #456897;  
}  
#blokA, #blokB, #blokC{  
    height: 200px;  
    width: 200px;  
}  
#blokA, #blokB{  
    float: left;  
}  
#blokA{  
    background: red;  
}  
#blokB{  
    background: blue;  
}  
#blokC{  
    background: purple;  
    float: right;  
}
```

## RESULTAAT



### ***OPMERKING:***

FLOATS werden voor de komst van HTML5 veelvuldig gebruikt om iets links of rechts uit te lijnen. Wanneer je bijvoorbeeld 2 blokken naast elkaar wil laten links floaten, dan kan je ook onderstaand alternatief gebruiken.

***display: inline-block;***

## 5.7.2. POSITIONING

Een veel gebruikte techniek in standaard css is/was het absoluut positioneren van een html element op een pagina. Het absoluut positioneren wil zeggen dat we de exacte positie op een pagina gaan duiden waar het element dient te staan, ongeacht de html structuur.

De absolute positioning van een element kan enkel en alleen werken wanneer hij zich kan relateren t.o.v. een "parent" element die ook vanaf een bepaalde positie start. Vervolgens heb je de opties: top, right, bottom en right om het aantal pixels te verzetten van een absoluut element t.o.v. de relatieve parent.

Start een nieuw document: klik\_per\_klik\_8.html

### HTML

```
<!DOCTYPE html>
<html lang="nl">

<head>
    <meta charset="UTF-8">
    <title>CSS 3 styles</title>
    <link rel="stylesheet" href="styles.css">
</head>

<body>
    <p id="mijnstijl">
        CSS3 external styles</p>
    <h1>Relative en absolute positioning</h1>
    <div id="blokken">
        <div id="bloka">BLOK A</div>
        <div id="blokb">BLOK B</div>
        <div id="blokc">BLOK C</div>
    </div>
</body>

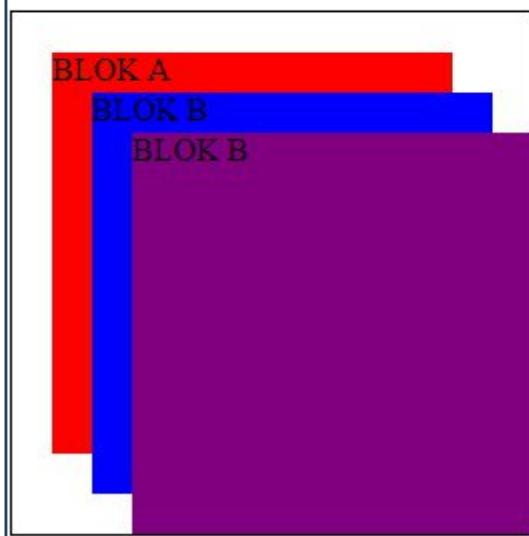
</html>
```

## CSS

```
#mijnstijl {  
    color: #456897;  
    font-size: 100px;  
    font-style: italic;  
    text-align: center;  
    padding-top: 5px;  
    padding-bottom: 5px;  
    border: 1px solid #456897;  
}  
#blokken{  
    position: relative;  
    border: 1px solid black;  
    width: 260px;  
    height: 260px;  
}  
#blokA, #blokB, #blokC{  
    height: 200px;  
    width: 200px;  
    position: absolute;  
}  
#blokA{  
    background: red;  
    top: 20px;  
    left: 20px;  
}  
#blokB{  
    background: blue;  
    top: 40px;  
    left: 40px;  
}  
#blokC{  
    background: purple;  
    top: 60px;  
    left: 60px;  
}
```

## RESULTAAT

# Relative en absolute positioning



**NAAST RELATIVE EN ABSOLUTE POSITIONING HEBBEN WE OOK NOG FIXED EN STICKY. PROBEER BEIDE EENS UIT OP BLOK B EN SCROLL VERVOLGENS NAAR BENEDEN. BEKIJK HET RESULTAAT.**

### 5.7.3. BOX-SIZING: BORDER-BOX

#### Wat doet box-sizing: border-box?

Normaal gesproken, als je een breedte en hoogte opgeeft voor een element, worden de padding en border bovenop die opgegeven waarden toegevoegd. Dit kan ertoe leiden dat een element groter wordt dan je oorspronkelijk had aangegeven.

Met **box-sizing: border-box** zorg je ervoor dat de breedte en hoogte van het element **inclusief** de padding en de border wordt berekend. Dit betekent dat de breedte en hoogte die je opgeeft, de totale ruimte van het element omvat, inclusief de inhoud (content), padding, en border.

In onderstaand voorbeeld zie je #box2 staan. Deze heeft een width van 500px. In het box model zie je dus de volgende som staan = 1 + 20 +458 +20 +1 = 500.

#box2 bevat dus box-sizing: border-box.

The screenshot shows the Chrome DevTools Elements tab. On the left, the DOM tree shows the structure of the page with nodes like #box2, #box1, and the main body. On the right, the Styles panel displays the CSS rules for these elements. The relevant rules for #box2 are:

```
<!DOCTYPE html>
<html lang="nl">
  <head>...</head>
  <body cz-shortcut-listen="true">
    <div id="pj-app">...</div>
    <div id="box1">...</div>
    <div id="box2">Hallo!</div>
  </body>
<div id="__genieContainer" style="all: initial;">...</div>
</html>
```

The Styles panel also shows the computed styles for #box1 and #box2, including their dimensions and borders. Below the styles, a detailed box model diagram illustrates the layout. It shows a large orange outer box representing the margin. Inside is a smaller yellow box representing the border. Within that is a green dashed box representing the padding. The innermost blue box represents the content area, which is labeled "458 x 158". The total width of the element is indicated as 500px.

## 5.7.4. FLEX BOX

Vóór de komst van de Flexbox-lay-out waren er vier lay-out mogelijkheden:

- blocks , voor secties op een webpagina
- inline, voor tekst
- tabel, voor tweedimensionale tabelgegevens (rijen en kolommen)
- positioning , voor expliciete positie van een element (relative, absolute)

Dankzij de komst van flex-box zijn werd het positioneren van elementen nog vereenvoudigd. Flex Box is daarnaast ook de standaard van het huidige Bootstrap Framework geworden vanaf versie 4. Het volledige Framework steunt hierop.

Later meer hierover tijdens Bootstrap zelf.

In principe kan je met flexbox alleen een volledige statisch responsive website maken, maar dan mis je al de mogelijkheden en componenten die het Bootstrap Framework later zal aanbieden.

Hieronder een overzicht van alle mogelijkheden die je dient te kennen en die we zullen gebruiken in ons projecten:

- flex-direction
  - column
  - column-reverse
  - row
  - row-reverse
- flex-wrap
  - wrap
  - nowrap
  - wrap-reverse
- flex-flow
  - row-wrap
- justify-content
  - center
  - flex-start
  - flex-end
  - space-around
  - space-between
- align-items
  - center
  - flex-start
  - flex-end
  - stretch
  - baseline
- align-content
  - space-between
  - space-around
  - stretch
  - center

- flex-start
- flex-end
- order
- flex-grow
- flex-shrink
- flex-basis
- align-self:
  - center
  - flex-start
  - flex-end

Start een nieuw document: klik\_per\_klik\_9.html

De basis van flexbox is eigenlijk het in gebruik nemen van een container als parent element en daarin ook de children te gaan positioneren, net zoals we gezien hebben bij relative en absolute positioning.

Bij flexbox hebben we echter veel meer mogelijkheden in vergelijking met de attributen top, left, bottom en right die we besproken hebben in het vorige deel van het absolute positioneren.

We overlopen even de mogelijkheden in ons document.

### 5.7.4.1. Voorbeelden

We zullen gebruiken voor deze voorbeelden maar gebruiken een inline style binnen de html tags om dit uit te leggen.

Voor onderstaande voorbeelden mag je niet vergeten je css bestand te linken in je html.

In je css bestand zet je eerst al deze classes van flexbox. Kopieer deze:

```
/* Display Flex */  
.d-flex {  
    display: flex;  
}  
  
.d-inline-flex {  
    display: inline-flex;  
}  
  
/* Flex Direction */  
.flex-row {  
    flex-direction: row;  
}  
  
.flex-row-reverse {  
    flex-direction: row-reverse;  
}  
  
.flex-column {  
    flex-direction: column;  
}  
  
.flex-column-reverse {  
    flex-direction: column-reverse;  
}  
  
/* Flex Wrap */  
.flex-wrap {  
    flex-wrap: wrap;  
}  
  
.flexnowrap {  
    flex-wrap: nowrap;  
}  
  
.flex-wrap-reverse {  
    flex-wrap: wrap-reverse;  
}  
  
/* Justify Content */  
.justify-content-start {
```

```
    justify-content: flex-start;
}

.justify-content-end {
    justify-content: flex-end;
}

.justify-content-center {
    justify-content: center;
}

.justify-content-between {
    justify-content: space-between;
}

.justify-content-around {
    justify-content: space-around;
}

.justify-content-evenly {
    justify-content: space-evenly;
}

/* Align Items */
.align-items-start {
    align-items: flex-start;
}

.align-items-end {
    align-items: flex-end;
}

.align-items-center {
    align-items: center;
}

.align-items-baseline {
    align-items: baseline;
}

.align-items-stretch {
    align-items: stretch;
}

/* Align Content */
.align-content-start {
    align-content: flex-start;
}
```

```
.align-content-end {  
    align-content: flex-end;  
}  
  
.align-content-center {  
    align-content: center;  
}  
  
.align-content-between {  
    align-content: space-between;  
}  
  
.align-content-around {  
    align-content: space-around;  
}  
  
.align-content-stretch {  
    align-content: stretch;  
}  
  
/* Align Self */  
.align-self-auto {  
    align-self: auto;  
}  
  
.align-self-start {  
    align-self: flex-start;  
}  
  
.align-self-end {  
    align-self: flex-end;  
}  
  
.align-self-center {  
    align-self: center;  
}  
  
.align-self-baseline {  
    align-self: baseline;  
}  
  
.align-self-stretch {  
    align-self: stretch;  
}  
  
/* Order */  
.order-first {  
    order: -1;
```

```
}
```

```
.order-last {  
    order: 13;  
}
```

```
.order-0 {  
    order: 0;  
}
```

```
.order-1 {  
    order: 1;  
}
```

```
.order-2 {  
    order: 2;  
}
```

```
.order-3 {  
    order: 3;  
}
```

```
.order-4 {  
    order: 4;  
}
```

```
.order-5 {  
    order: 5;  
}
```

```
.order-6 {  
    order: 6;  
}
```

```
.order-7 {  
    order: 7;  
}
```

```
.order-8 {  
    order: 8;  
}
```

```
.order-9 {  
    order: 9;  
}
```

```
.order-10 {  
    order: 10;
```

```

}

.order-11 {
  order: 11;
}

.order-12 {
  order: 12;
}

/* Grow and Shrink */
.flex-grow-0 {
  flex-grow: 0;
}

.flex-grow-1 {
  flex-grow: 1;
}

.flex-shrink-0 {
  flex-shrink: 0;
}

.flex-shrink-1 {
  flex-shrink: 1;
}

1. d-flex
<div class="d-flex">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>

```

**Uitleg:** Dit zorgt ervoor dat de vierkantjes horizontaal naast elkaar worden weergegeven.



## 2. d-inline-flex

```
<span class="d-inline-flex">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</span>
```

**Uitleg:** Hetzelfde als d-flex, maar dit maakt de flexbox inline, zodat het zich gedraagt als een inline-element.



## 3. flex-row

```
<div class="d-flex flex-row">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

**Uitleg:** De vierkantjes worden horizontaal naast elkaar weergegeven (standaard flex-richting).



## 4. flex-row-reverse

```
<div class="d-flex flex-row-reverse">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

**Uitleg:** De vierkantjes worden in omgekeerde volgorde weergegeven van rechts naar links.



### 5. flex-column

```
<div class="d-flex flex-column">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

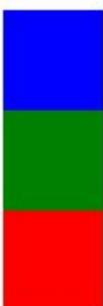
**Uitleg:** De vierkantjes worden verticaal onder elkaar weergegeven.



### 6. flex-column-reverse

```
<div class="d-flex flex-column-reverse">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

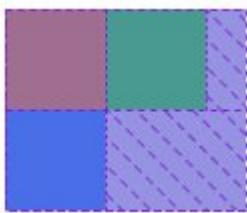
**Uitleg:** De vierkantjes worden verticaal ondersteboven weergegeven.



## 7. flex-wrap

```
<div class="d-flex flex-wrap" style="width: 120px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

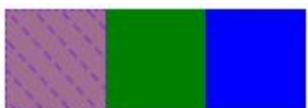
**Uitleg:** De vierkantjes worden naar de volgende regel gewrapt als er niet genoeg ruimte is. We voorzien eigenlijk een maximale breedte van 120px zoals hierboven ziet. Daardoor is er maar plaats voor de eerste twee blokjes. De wrap zorgt ervoor dat het derde blokje naar de volgende lijn wordt verschoven.



## 8. flexnowrap

```
<div class="d-flex flexnowrap" style="width: 150px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

**Uitleg:** Voorkomt dat de items naar de volgende rij verplaatsen. Zelfs als er niet genoeg ruimte is, blijven ze op één rij.



### 9. justify-content-start

```
<div class="d-flex justify-content-start" style="width: 200px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

**Uitleg:** De flex-items worden uitgelijnd aan het begin van de container (links).



### 10. justify-content-end

```
<div class="d-flex justify-content-end" style="width: 200px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

**Uitleg:** De flex-items worden uitgelijnd aan het einde van de container (rechts).



### 11. justify-content-center

```
<div class="d-flex justify-content-center">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

**Uitleg:** De vierkantjes worden horizontaal gecentreerd binnen de container.



#### 12. justify-content-between

```
<div class="d-flex justify-content-between" style="width: 200px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

**Uitleg:** De eerste en laatste flex-items worden aan de randen van de container geplaatst, en de resterende ruimte wordt gelijkmatig verdeeld tussen de items.



#### 13. justify-content-around

```
<div class="d-flex justify-content-around" style="width: 200px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```

**Uitleg:** De flex-items worden gelijkmatig verdeeld, met gelijke ruimte aan beide zijden van elk item.



#### 14. justify-content-evenly

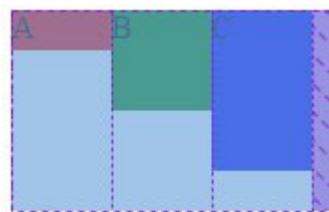
```
<div class="d-flex justify-content-evenly" style="width: 200px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```



### 15. align-items-baseline

```
<div class="d-flex align-items-baseline" style="height: 100px;">
  <div style="width: 50px; height: 20px; background-color:
red;">A</div>
  <div style="width: 50px; height: 50px; background-color:
green;">B</div>
  <div style="width: 50px; height: 80px; background-color:
blue;">C</div>
</div>
```

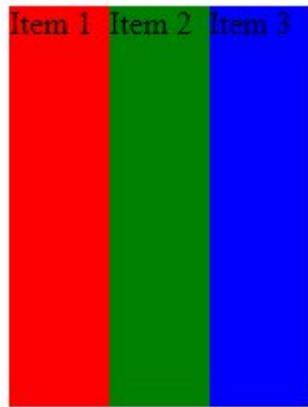
**Uitleg:** De items worden uitgelijnd op basis van hun tekst baseline. Dit zorgt ervoor dat de onderkanten van de letters gelijk uitgelijnd zijn. Dus met andere woorden de teksten blijven op dezelfde hoogte staan. De baseline is de onderkant van de letters.



#### 16. align-items-stretch

```
<div class="d-flex align-items-stretch" style="height: 200px;">
  <div style="width: 50px; background-color: red;">Item 1</div>
  <div style="width: 50px; background-color: green;">Item 2</div>
  <div style="width: 50px; background-color: blue;">Item 3</div>
</div>
```

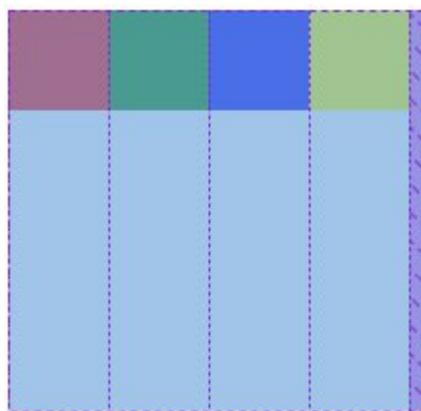
**Uitleg:** De items rekken verticaal uit om de volledige hoogte van de container te vullen.



#### 17. align-content-start

```
<div class="d-flex flex-wrap align-content-start" style="height: 200px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
  <div style="width: 50px; height: 50px; background-color: yellow;"></div>
</div>
```

**Uitleg:** De rijen worden uitgelijnd aan de bovenkant van de container.

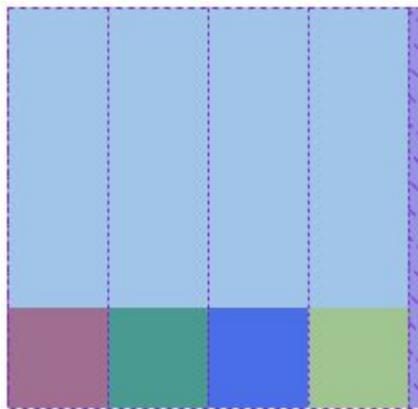


```

18. align-content-end
<div class="d-flex flex-wrap align-content-end" style="height: 200px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
  <div style="width: 50px; height: 50px; background-color: yellow;"></div>
</div>

```

**Uitleg:** De rijen worden uitgelijnd aan de onderkant van de container.

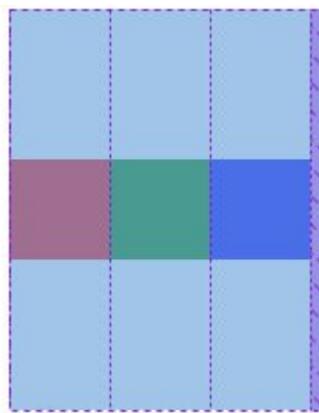


```

19. align-items-center
<div class="d-flex align-items-center" style="height: 200px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>

```

**Uitleg:** De vierkantjes worden verticaal gecentreerd binnen de container. Ik voorzie een vaste hoogte in de container van 200px. De drie divs worden hierin verticaal gecentreerd.



## 20. order-1 en order-2

```
<div class="d-flex">
  <div class="order-2" style="width: 50px; height: 50px; background-color: red;"></div>
  <div class="order-1" style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```



**Uitleg:** De volgorde van de vierkantjes is veranderd met order-klassen. Het groene vierkantje komt voor het rode, hoewel de HTML volgorde anders is.

## 21. flex-grow-1

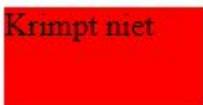
```
<div class="d-flex" style="width: 200px;">
  <div class="flex-grow-1" style="background-color: red;">Groot</div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```



**Uitleg:** Het rode vierkantje zal de resterende ruimte opvullen en groter worden dan de andere.

## 22. flex-shrink-0

```
<div class="d-flex" style="width: 100px;">
  <div class="flex-shrink-0" style="width: 100px; height: 50px; background-color: red;">Krimpt niet</div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>
```



**Uitleg:** Het rode vierkantje zal niet krimpen, ook al is de container smaller dan de inhoud. De totale breedte is hier maar 100px. Dit wil zeggen dat de twee andere vierkantjes er nog steeds zijn maar gewoon niet zichtbaar zijn.

```

23. align-content-stretch in combinatie met flex-grow
<div class="d-flex flex-wrap align-content-stretch" style="height: 400px; width: 200px;">

    <div class="flex-grow-1" style="width: 50px; background-color: red;"></div>

    <div class="flex-grow-1" style="width: 50px; background-color: green;"></div>

    <div class="flex-grow-1" style="width: 50px; background-color: blue;"></div>

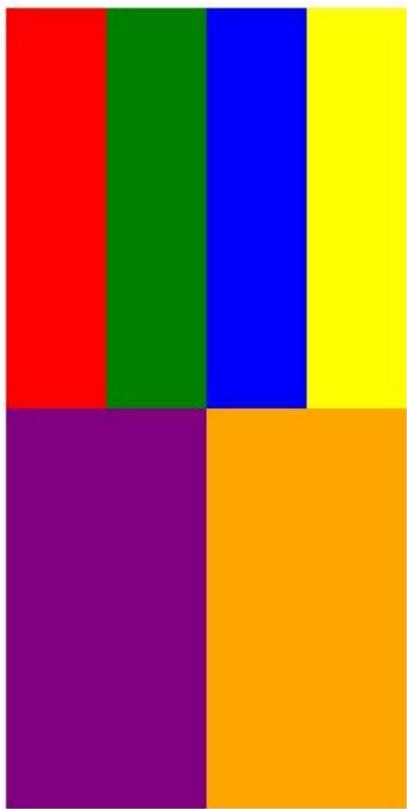
    <div class="flex-grow-1" style="width: 50px; background-color: yellow;"></div>

    <div class="flex-grow-1" style="width: 50px; background-color: purple;"></div>

    <div class="flex-grow-1" style="width: 50px; background-color: orange;"></div>

</div>

```



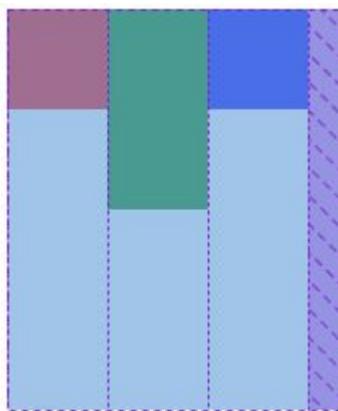
**Uitleg:** De vierkantjes worden uitgerekt om de hoogte van de container te vullen.

```

24. align-self-auto
<div class="d-flex" style="height: 200px;">
  <div style="width: 50px; height: 50px; background-color: red;"></div>
  <div class="align-self-auto" style="width: 50px; height: 100px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>

```

**Uitleg:** align-self-auto zorgt ervoor dat het element de standaard uitlijning van de flex-container behoudt (wat de standaard uitlijning is voor alle items in de flex-container).

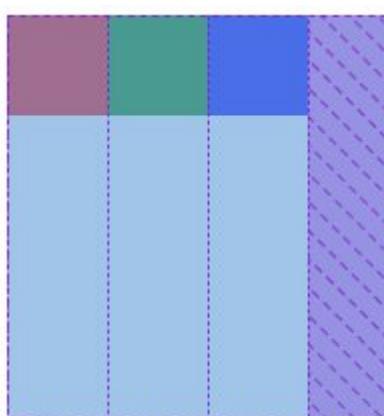


```

25. align-self-start
<div class="d-flex" style="height: 200px;">
  <div class="align-self-start" style="width: 50px; height: 50px; background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color: blue;"></div>
</div>

```

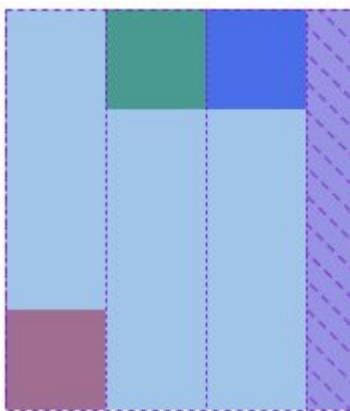
**Uitleg:** Het rode item met de klasse align-self-start wordt uitgelijnd aan de bovenkant van de container.



## 26. align-self-end

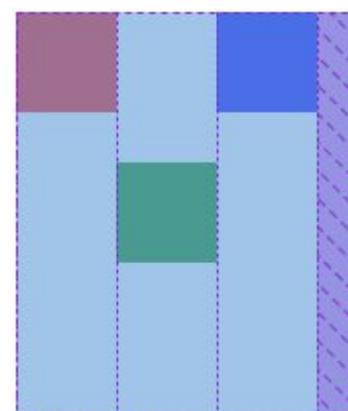
```
<div class="d-flex" style="height: 200px;">
  <div class="align-self-end" style="width: 50px; height: 50px;
background-color: red;"></div>
  <div style="width: 50px; height: 50px; background-color:
green;"></div>
  <div style="width: 50px; height: 50px; background-color:
blue;"></div>
</div>
```

**Uitleg:** Het rode item met de klasse align-self-end wordt uitgelijnd aan de onderkant van de container.



## 27. align-self-center

```
<div class="d-flex" style="height: 200px;">
  <div style="width: 50px; height: 50px; background-color:
red;"></div>
  <div class="align-self-center" style="width: 50px; height: 50px;
background-color: green;"></div>
  <div style="width: 50px; height: 50px; background-color:
blue;"></div>
</div>
```

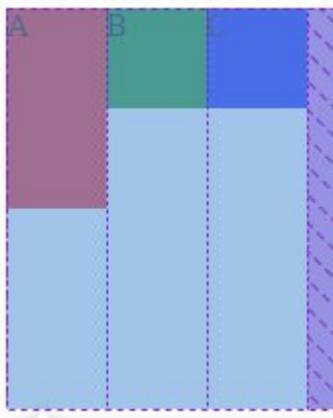


**Uitleg:** Alleen het groene vierkantje wordt verticaal gecentreerd, terwijl de andere de standaard uitlijning behouden.

## 28. align-self-baseline

```
<div class="d-flex" style="height: 200px;">
  <div class="align-self-baseline" style="width: 50px; height:
100px; background-color: red;">A</div>
  <div style="width: 50px; height: 50px; background-color:
green;">B</div>
  <div style="width: 50px; height: 50px; background-color:
blue;">C</div>
</div>
```

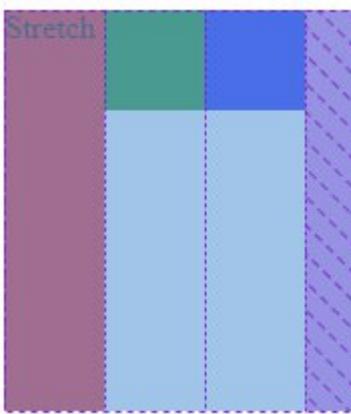
**Uitleg:** Het rode item met de klasse align-self-baseline wordt uitgelijnd volgens de baseline van de tekst in de container, zodat de onderkanten van de letters op dezelfde hoogte staan.



## 29. align-self-stretch

```
<div class="d-flex" style="height: 200px;">
  <div class="align-self-stretch" style="width: 50px; background-
color: red;">Stretch</div>
  <div style="width: 50px; height: 50px; background-color:
green;"></div>
  <div style="width: 50px; height: 50px; background-color:
blue;"></div>
</div>
```

**Uitleg:** Het rode item met de klasse align-self-stretch wordt uitgerekt om de volledige hoogte van de container te vullen.



Met al deze elementen hebben we de basis van HTML5, CSS3 en flexbox gezien en kunnen we overgaan tot een real live project in het volgende deel. Gaandeweg zullen er natuurlijk nog andere elementen bijkomen die we zullen gebruiken om een design correct om te zetten naar een eerste site.