

# HTML5 - CSS3

## 1. VOORKENNIS

Deze cursus vereist geen enkele voorkennis.

## 2. DOEL

Na het doorlopen van deze cursus bent u in staat om een volledige professionele statische responsive website te maken met het bootstrap framework. De volledige cursus bestaat uit 3 delen:

- HTML 5
- CSS 3
- BOOTSTRAP FRAMEWORK

Dit deel handelt over HTML5 en CSS3. We bespreken dit tot in het kleinste detail, waarna het BOOTSTRAP FRAMEWORK volgt om vlugger responsive websites te maken. Volgens de laatste standaarden uitgelegd met voorbeelden. Nadat u de onderdelen volledig doorlopen heeft, zullen we gezamenlijk een volledig webproject vertrekkend van een design opbouwen t.e.m. het online zetten van dit project.

### 3. INSTALLATIE TEXT EDITOR

Tijdens deze opleiding zullen we een gratis texteditor gebruiken, nl. sublime text (<https://sublimetext.com>). Iedere webontwikkelaar heeft zijn eigen specifieke keuze in de omgeving waar hij in wenst te programmeren. Er zijn ook nog andere editors zoals bijv. brackets en betalende editors zoals webstorm.

Download en installeer nu de editor.

#### 3.1. PACKAGES

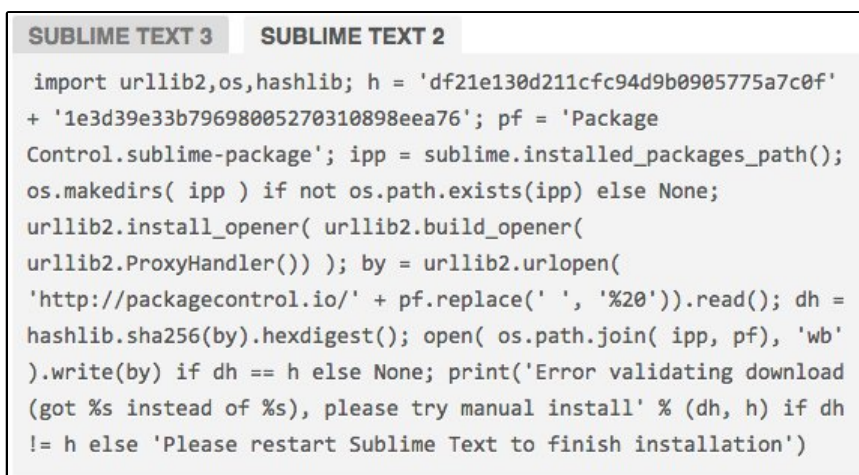
Na een eerste installatie is sublime text een vlakke editor. De enige functionaliteit die u momenteel heeft is het veranderen van de taal waar je in wenst te schrijven. Dit kan onderaan rechts het scherm aangepast worden. Klik op 'Plain Text' en scroll naar 'HTML' om dit te veranderen.

De kracht van Sublime Text ligt in het feit dat je bijkomende packages kan installeren om deze texteditor uit te bouwen tot een zeer krachtige medium om gelijk welke code te schrijven.

Hiervoor dienen we de basis package te installeren, nl. **Package Control:Install Package**. Deze package laat toe om andere packages op een eenvoudige manier te installeren binnen Sublime Text.

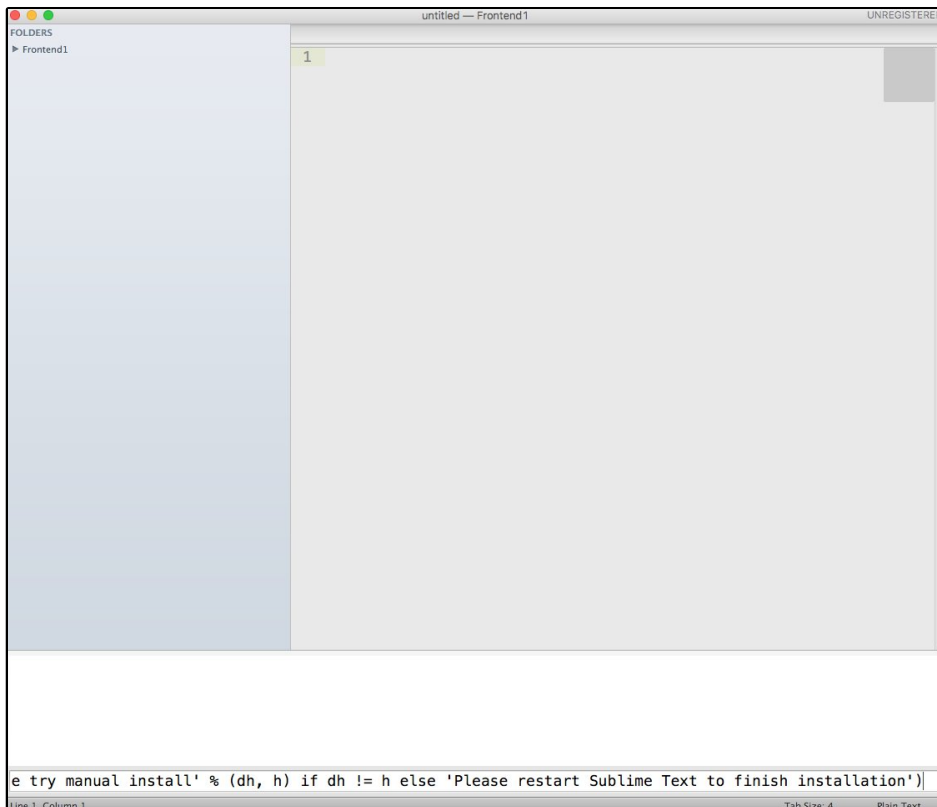
Klik hiervoor om de volgende link en voer onderstaande stappen uit

1. Link: <https://packagecontrol.io/installation>
2. Maak de keuze voor Sublime Text 3 en kopieer de code die je op het tabblad ziet. Let op: kies het tabblad Sublime Text 3

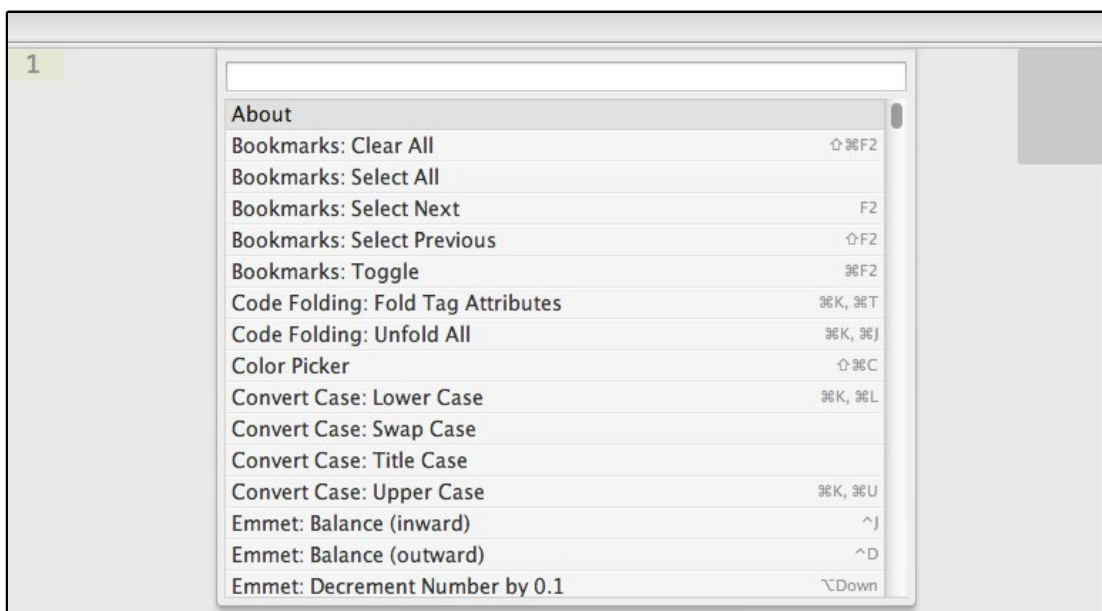
A screenshot of a code editor window showing the installation code for Sublime Text 3. The window has two tabs: 'SUBLINE TEXT 3' (selected) and 'SUBLINE TEXT 2'. The code is a Python script that uses urllib2 to download a package from packagecontrol.io and install it. The code includes comments and error handling.

```
import urllib2,os,hashlib; h = 'df21e130d211cfc94d9b0905775a7c0f'
+ '1e3d39e33b79698005270310898eea76'; pf = 'Package
Control.sublime-package'; ipp = sublime.installed_packages_path();
os.makedirs( ipp ) if not os.path.exists(ipp) else None;
urllib2.install_opener( urllib2.build_opener(
urllib2.ProxyHandler()) ); by = urllib2.urlopen(
'http://packagecontrol.io/' + pf.replace(' ', '%20')).read(); dh =
hashlib.sha256(by).hexdigest(); open( os.path.join( ipp, pf), 'wb'
).write(by) if dh == h else None; print('Error validating download
(got %s instead of %s), please try manual install' % (dh, h) if dh
!= h else 'Please restart Sublime Text to finish installation')
```

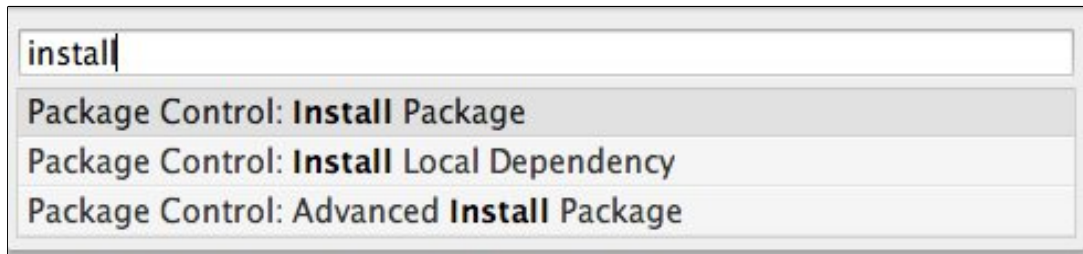
3. Open je Sublime Text en klik in de menubalk op **View - Show Console**.
4. Plak bovenstaande code in het console venster en druk op enter. De package wordt geïnstalleerd.



5. Sluit Sublime Text nu af en start opnieuw op. Deze basis package is geïnstalleerd.
6. Druk de toetscombinatie CTRL - SHIFT - P (Windows) of CMD - SHIFT - P (MAC) en u ziet onderstaand scherm.



7. Tik in het zoekscherm **install** en klik op **install control package**. Dit is de basis package die we hebben geïnstalleerd. Het package zoekt alle mogelijke bijkomende packages die je als aanvulling op je texteditor kan installeren. Zoals je zal merken zijn er heel wat die we niet allemaal nodig hebben. Wij zullen de meest gebruikte packages installeren.



8. Vervolgens ziet u het onderstaande scherm waar u ook terug een zoekbox hebt op de gewenste package(uitbreiding) te gaan installeren voor uw editor.

De volgende packages dienen we nu te installeren:

- emmet
- sublimelinter
- sidebarenhancements
- autofilename
- brackethighlighter
- HTML5
- HTML CSS JS prettify
- CSS3
- Bootstrap 4x autocomplete
- Bootstrap 4x snippets

## 4. HTML 5

Een webpagina zichtbaar maken voor het publiek vergt structuur. Deze structuur kent twee kanten in het verhaal. Het eerste gedeelte **HTML5** (Hyper Text Markup Language) geeft de volledige content (inhoud) terug van de pagina zonder enige vorm van lay-out. Het tweede gedeelte noemt men **CSS3** (Cascading Style Sheets) Dit zijn de “stijlbladen” die de html vorm geven en de webpagina’s aankleden. We zullen ook nog **JAVASCRIPT** toepassen, maar niet schrijven momenteel. Dit komt in de cursus JAVASCRIPT zelf aan bod.

### 4.1. EERSTE HTML 5 PAGINA

Maak een eerste bestand aan zoals hieronder weergegeven.

- Klik in het menu op **File**
- Klik op **New File**

Zorg ervoor dat je text editor op html staat. Klik hiervoor rechts onderaan op “plain text” en wijzig naar HTML. De extensie van een html bestand eindigt op .html of .htm

Het gebruik hiervan is te kiezen, maar .html is meestal de standaard die we ook zullen gebruiken.

## 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>
8  </body>
9  </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.

### *Characteraset*

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 geen 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.

## <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!

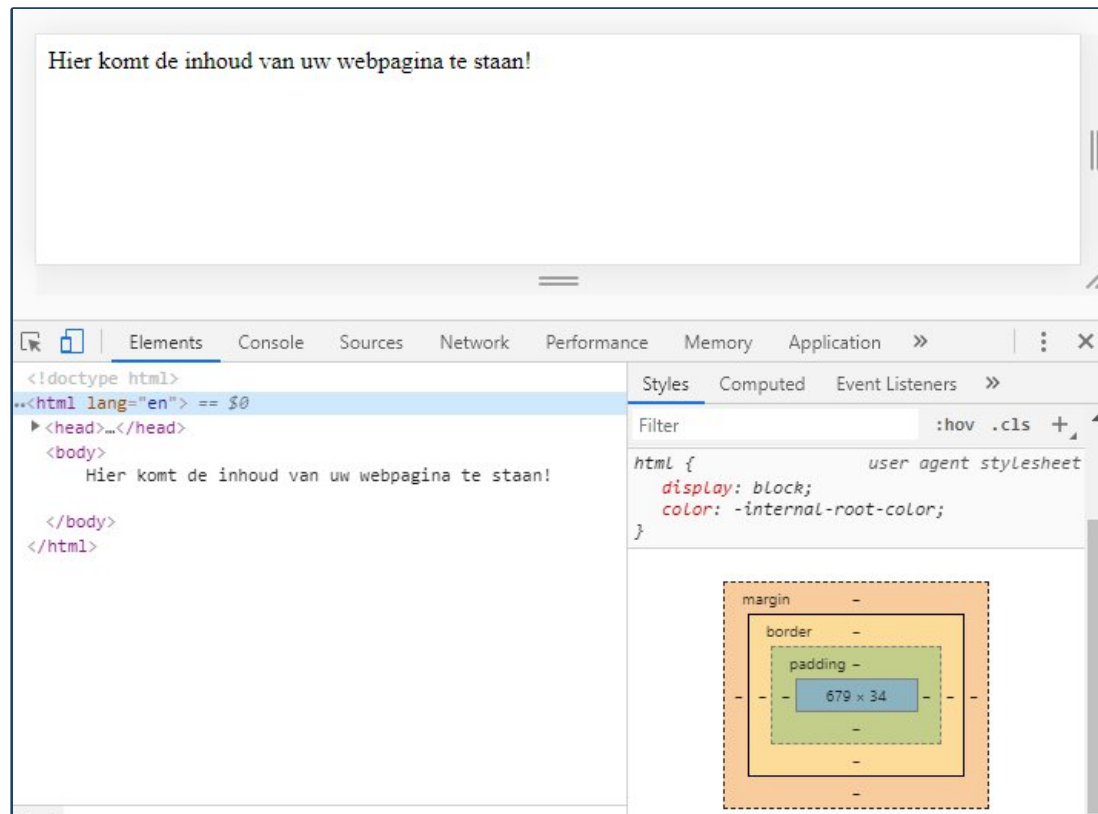
Zoals je hierboven kan zien bevat de pagina alleen platte tekst momenteel.



### 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

<b>HEADING 1</b>
<b>HEADING 2</b>
<b>HEADING 3</b>
<b>HEADING 4</b>
<b>HEADING 5</b>
<b>HEADING 6</b>

#### 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      <!-- ONGEORDEDE LIJST -->
26      <ul>
27          <li>HTML5</li>
28          <li>CSS3</li>
29          <li>BOOTSTRAP</li>
30      </ul>
31      <!-- GEORDEDE 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



#### 4.5.1.1.6. <pre>

pre tag	
Tag	<pre >
Beschrijving	De tekstopmaak 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](http://www.google.be)

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 verkleint 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 HTML5 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 HTML5 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>	
<pre> &lt;main&gt; &lt;section&gt;   &lt;header&gt;     &lt;h2&gt;subtitel van het onderwerp&lt;/h2&gt;   &lt;/header&gt;   &lt;article&gt;     &lt;header&gt;       &lt;h3&gt;artikel 1 van de subtitel&lt;/h3&gt;     &lt;/header&gt;   &lt;/article&gt;   &lt;article&gt;     &lt;header&gt;       &lt;h3&gt;artikel 2 van de subtitel&lt;/h3&gt;     &lt;/header&gt;   &lt;/article&gt; &lt;/section&gt; &lt;/main&gt; </pre>	<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>
</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.

**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 Validator interface. At the top, there's a 'Checker input' section with 'Show' options for 'source' (checked), 'outline' (checked), and 'image report' (unchecked), and an 'Options...' button. Below this, the 'Check by' dropdown is set to 'text input', and there's a checkbox for 'css'. The main area displays the HTML code being validated, which includes a DOCTYPE declaration, an HTML lang attribute, a head section with meta tags for charset, title, description, keywords, author, and viewport, and a body section with a nav element containing a list. A 'Check' button is located below the code. At the bottom, there's a 'Message Filtering' button and a green status bar that reads '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

The screenshot shows the 'Heading-level outline' and 'Structural outline' sections of the W3C Markup Validator. The 'Heading-level outline' lists the following elements: <h1> Voetbalcompetitie, <h2> Jupiler Pro League, <h3> Anderlecht, 3de trainer in 3 maanden!, <h3> Cercle door het oog van de naald!, <h2> Proximus League, <h3> KV Mechelen wint de beker van België!, and <h3> Wie promoveert er mee naar de Proximus League. The 'Structural outline' shows a tree view of the document structure, starting with '[body element with no heading]', followed by '[nav element with no heading]', and then the same heading structure as the 'Heading-level outline'.

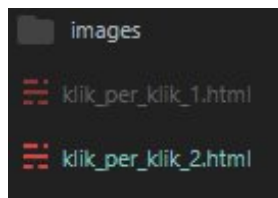




### 4.5.1.3. AFBEELDINGEN

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

 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**

 Proximus Pro 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>
          <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>
    </article>
  </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 rechtreeks 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
    <strong>light</strong> logo op de verpakking, maar bekijk het aantal <abbr title="Kilo Caloriën">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.

Active	Shape	Link	Title	Target
<input type="radio"/>	Rect	Fruitsap.html	Fruitsap	_blank
<input checked="" type="radio"/>	Circle	Fruitschaal.html	Fruitschaal	_blank

[+ Add New Area](#)

[Show Me The Code!](#)

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 bestands grootte 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 bestands groottes nog kleiner maakt.

De toekomst is wel degelijk WebP. Er zijn sites zoals [online-convert.com](http://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.

We kunnen dit controleren op een site die door alle webdevelopers nauwgezet in de gaten wordt gehouden, nl. **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.

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	BlackBerry Browser	Opera Mobile	Chrome for Android	Firefox for Android	IE Mobile	UC Browser for Android	Sams Inter
			4-8		10.1										
			9-22		11.5			2.1-3							
		2-64	23-31		12.1-18			4-4.1							
6-10	12-17	65	32-73	3.1-12	19-57	3.2-12.1		4.2-4.4.4	7	12-12.1			10		4-8
11	18	66	74	12.1	58	12.2	all	67	10	46	74	66	11	11.8	9.2
	75	67-68	75-77	TP											

#### 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>
  Password: <input type="password"><br>
  Radio: <input type="radio"><br>
  Range: <input type="range"><br>
  Reset: <input type="search"><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

Hieronder volgen alle mogelijkheden

Text:

Checkbox: ☐

Color:

Date:

Datetime-local:

Email:

File:  Geen bestand gekozen

Image:

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. DROPDOWN 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 indentiek 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

*CSS3 internal styles*

*Tweede lijn, zelfde stijl*

### 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 stijlen 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

*CSS3 internal styles*

### 5.1.3. EXTERNAL

We weten nu reeds hoe we kunnen stijlen 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.

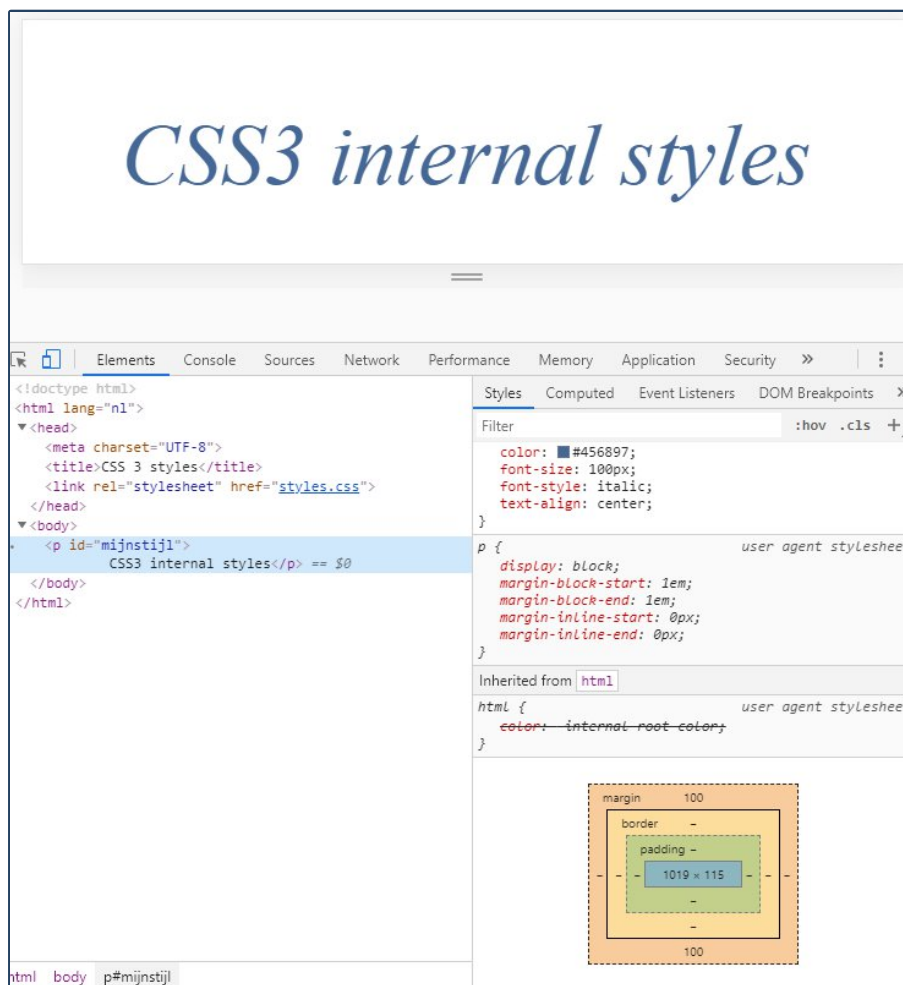
Aan de rechterkant zie je duidelijk het tabblad Styles actief staan met een grafische weergave van het boxmodel. Momenteel staan we geparkeerd op de p tag aan de linkerkant van je scherm.

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-left.

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 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;
}

```

The diagram illustrates the box model for a 1017 x 115 element. It shows three nested rectangles:

- Outermost (Margin):** A light orange rectangle with a dashed border, labeled "margin 100". It has a width of 1017 and a height of 115.
- Middle (Border):** A yellow rectangle with a solid border, labeled "border 1". It is 1px wider and 1px taller than the padding area.
- Innermost (Padding):** A green rectangle with a dashed border, labeled "padding 5". It is 5px wider and 5px taller than the content area.

The content area is a blue rectangle labeled "1017 x 115". The diagram shows the element is centered within the margin, with 100px of space on all sides.



### 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.

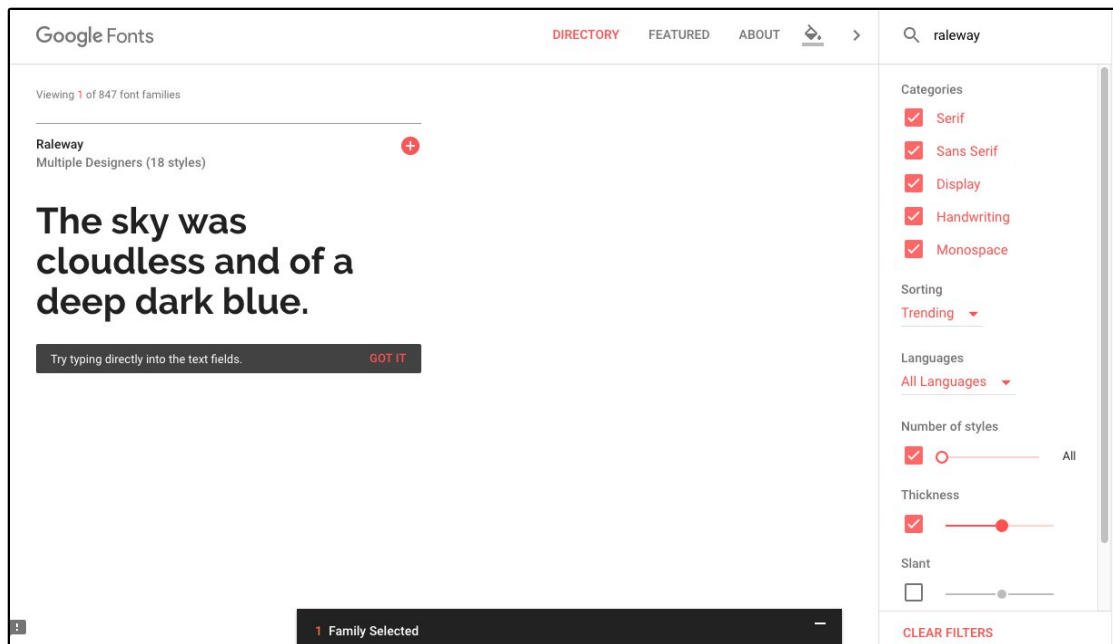
Wanneer je start vanaf de html tag = rem

Wanneer je start vanaf de body tag = em

- 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.



**1 Family Selected**

Your Selection Clear All

Raleway

EMBED

CUSTOMIZE

Load Time: Fast

### Embed Font

To embed your selected fonts into a webpage, copy this code into the <head> of your HTML document.

STANDARD @IMPORT

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

### Specify in CSS

Use the following CSS rules to specify these families:

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

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 left 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 B</div>
  </div>
</body>
</html>
```

## CSS

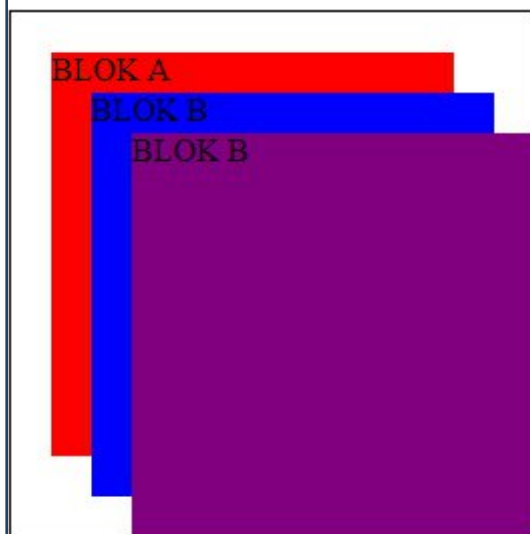
```
#mijnstijl {
  color: #456897;
  font-size: 100px;
  font-style: italic;
  text-align: center;
  padding-top: 5px;
  padding-bottom: 5px;
  border: 1px solid #456897;
}
#blokken{
  position: relative;
  border: 1px solid black;
  width: 260px;
  height: 260px;
}

#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





### 5.7.3. 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.

### 5.7.3.1. DISPLAY EN DIRECTION

Standaard worden alle child element in een rij geplaatst. Met de `direction` property kunnen we deze wijzigen naar kolommen.

#### HTML

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

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

<body>
  <h1>Flexbox</h1>
  <h2>Flexbox direction en display</h2>
  <h3>Flexbox row</h3>
  <p>row is de standaard richting van flexbox display</p>
  <div id="flexboxrow" class="flexbox-container">
    <div id="bloka">blok a</div>
    <div id="blokb">blok b</div>
    <div id="blokc">blok c</div>
  </div>
  <h4>Flexbox row reverse</h4>
  <div id="flexboxrowreverse" class="flexbox-container">
    <div id="bloka">blok a</div>
    <div id="blokb">blok b</div>
    <div id="blokc">blok c</div>
  </div>
  <h3>Flexbox column</h3>
  <div id="flexboxcolumn" class="flexbox-container">
    <div id="bloka">blok a</div>
    <div id="blokb">blok b</div>
    <div id="blokc">blok c</div>
  </div>
  <h4>Flexbox column reverse</h4>
  <div id="flexboxcolumnreverse" class="flexbox-container">
    <div id="bloka">blok a</div>
    <div id="blokb">blok b</div>
    <div id="blokc">blok c</div>
  </div>
</body>
</html>
```

## CSS

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

#flexboxcolumn{
  flex-direction: column;
}
#flexboxcolumnreverse{
  flex-direction: column-reverse;
}
#flexboxrowreverse{
  flex-direction: row-reverse;
}

#bloka, #blokb, #blokc{
  height: 200px;
  width: 200px;
}

#bloka{
  background: red;
}
#blokb{
  background: blue;
}
#blokc{
  background: purple;
}
```

## RESULTAAT

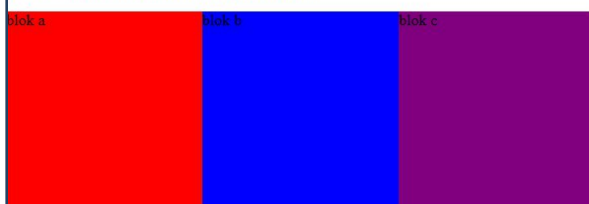
### Flex row

#### Flexbox

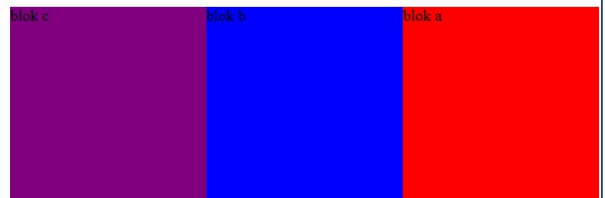
##### Flexbox direction en display

##### Flexbox row

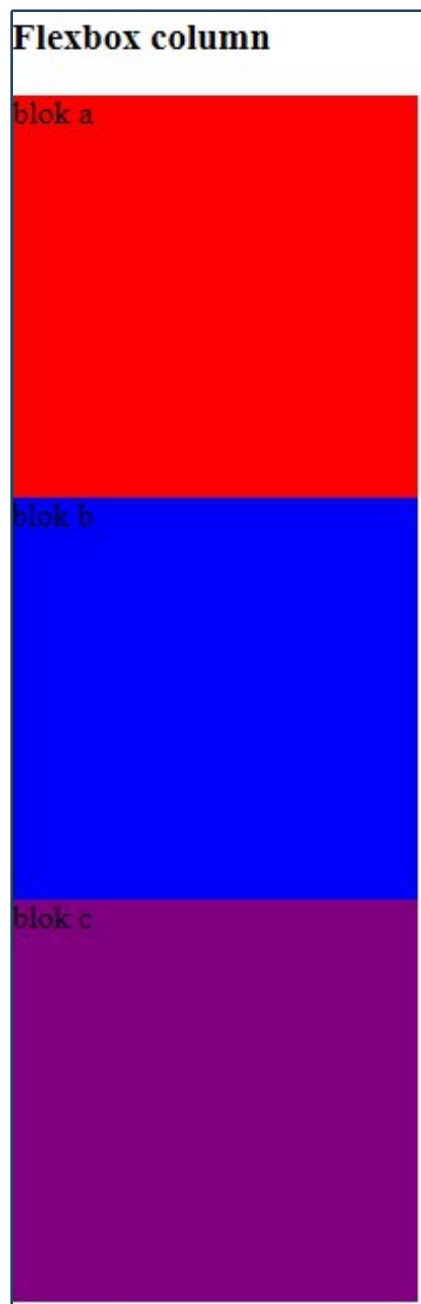
row is de standaard richting van flexbox display

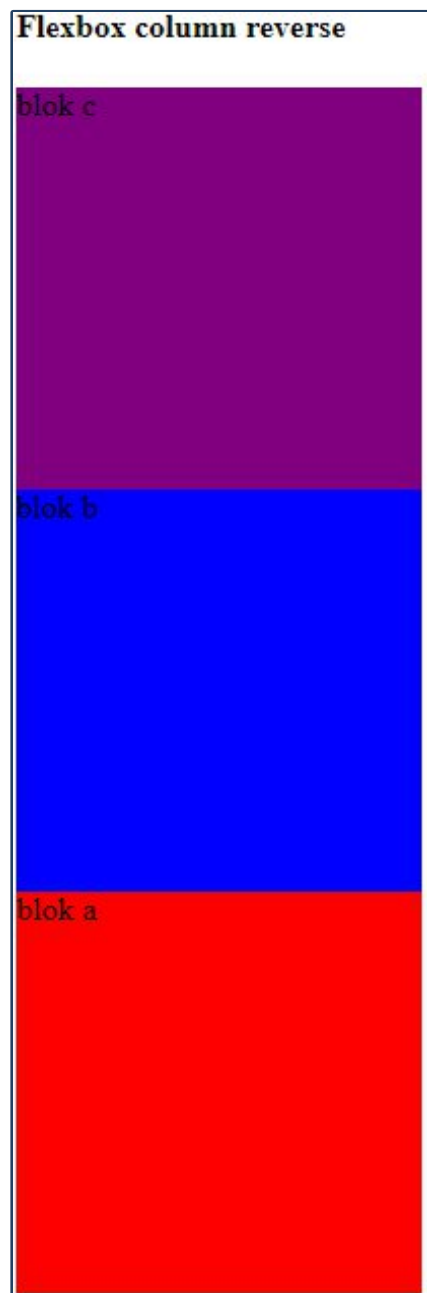


##### Flexbox row reverse



## Flex column





### 5.7.3.2. JUSTIFY CONTENT

Justify content zorgt ervoor dat de parent verantwoordelijk is voor het positioneren van zijn children. m.a.w. het html element die de kinderen omsluit zal ook de kinderen positioneren.

OPMERKING: JUSTIFY CONTENT ZORGT VOOR DE **HORIZONTALE** POSITIONERING VAN KINDEREN T.O.V. DE PARENT

#### 5.7.3.2.1. JUSTIFY CONTENT: CENTER

Hiermee centreer je de kinderen t.o.v. de parent.

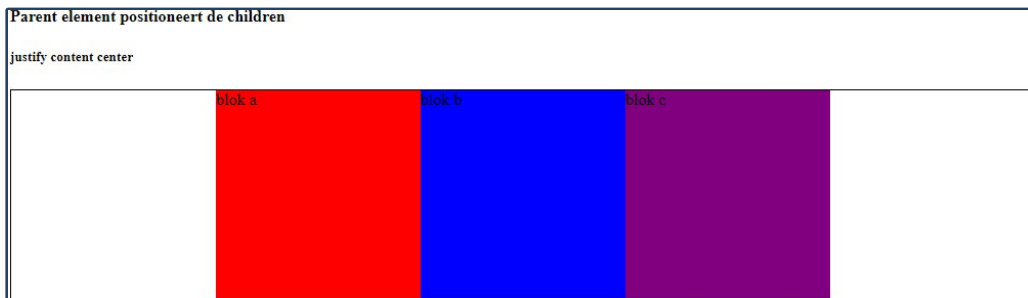
##### HTML

```
<h4>Parent element positioneert de children</h4>
<h5>justify content center</h5>
<div id="flexboxjustify" class="flexbox-container">
  <div id="bloka">blok a</div>
  <div id="blokb">blok b</div>
  <div id="blokc">blok c</div>
</div>
```

##### CSS

```
#flexboxjustify{
  width: 100%;
  height: 200px;
  border: 1px solid black;
  justify-content: center;
}
```

##### RESULTAAT



### 5.7.3.2.2. JUSTIFY CONTENT: FLEX-START

Hiermee plaats je de kinderen links t.o.v. de parent. Je kan dit vergelijken met float:left, maar dan met flexbox.

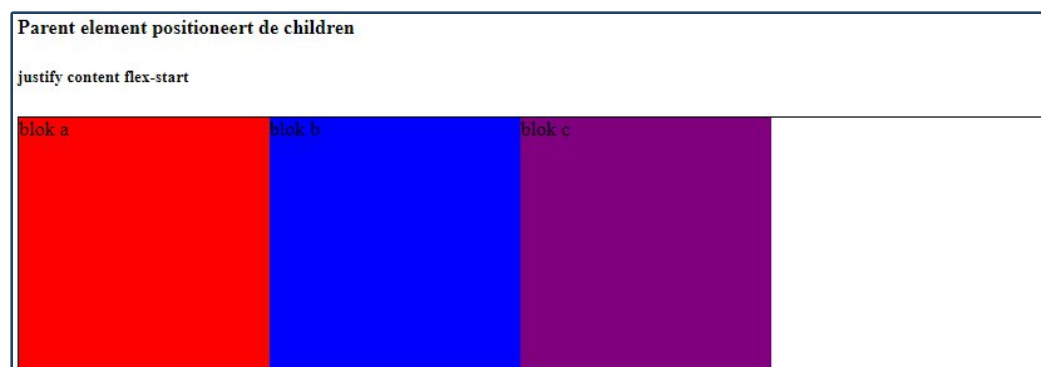
#### HTML

```
<h4>Parent element positioneert de children</h4>
<h5>justify content flex-start</h5>
<div id="flexboxjustify" class="flexbox-container">
  <div id="bloka">blok a</div>
  <div id="blokb">blok b</div>
  <div id="blokc">blok c</div>
</div>
```

#### CSS

```
#flexboxjustify{
  width: 100%;
  height: 200px;
  border: 1px solid black;
  justify-content: flex-start;
}
```

#### RESULTAAT





### 5.7.3.2.3. JUSTIFY CONTENT: FLEX-END

Hiermee plaats je de kinderen rechts t.o.v. de parent. Je kan dit vergelijken met float:right , maar dan met flexbox.

#### HTML

```
<h4>Parent element positioneert de children</h4>
<h5>justify content flex-end</h5>
<div id="flexboxjustify" class="flexbox-container">
  <div id="bloka">blok a</div>
  <div id="blokb">blok b</div>
  <div id="blokc">blok c</div>
</div>
```

#### CSS

```
#flexboxjustify{
  width: 100%;
  height: 200px;
  border: 1px solid black;
  justify-content: flex-end;
}
```

#### RESULTAAT



#### 5.7.3.2.4. JUSTIFY CONTENT: SPACE-AROUND

Hiermee krijgen de kinderen een gelijkmatige tussenafstand rond zichzelf binnen de parent.

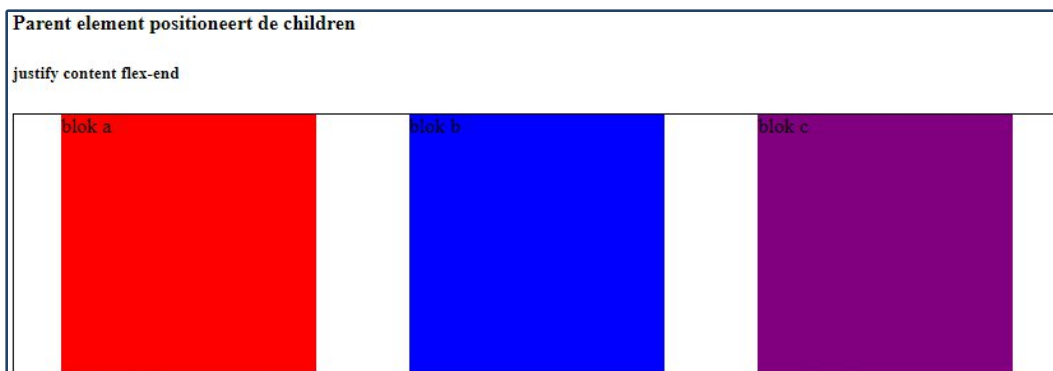
##### HTML

```
<h4>Parent element positioneert de children</h4>
<h5>justify content space-around</h5>
<div id="flexboxjustify" class="flexbox-container">
  <div id="bloka">blok a</div>
  <div id="blokb">blok b</div>
  <div id="blokc">blok c</div>
</div>
```

##### CSS

```
#flexboxjustify{
  width: 100%;
  height: 200px;
  border: 1px solid black;
  justify-content: space-around;
}
```

##### RESULTAAT



### 5.7.3.2.5. JUSTIFY CONTENT: SPACE-BETWEEN

In vergelijking met space-around zal hier enkel een tussenaafstand tussen de blokken worden weergegeven.

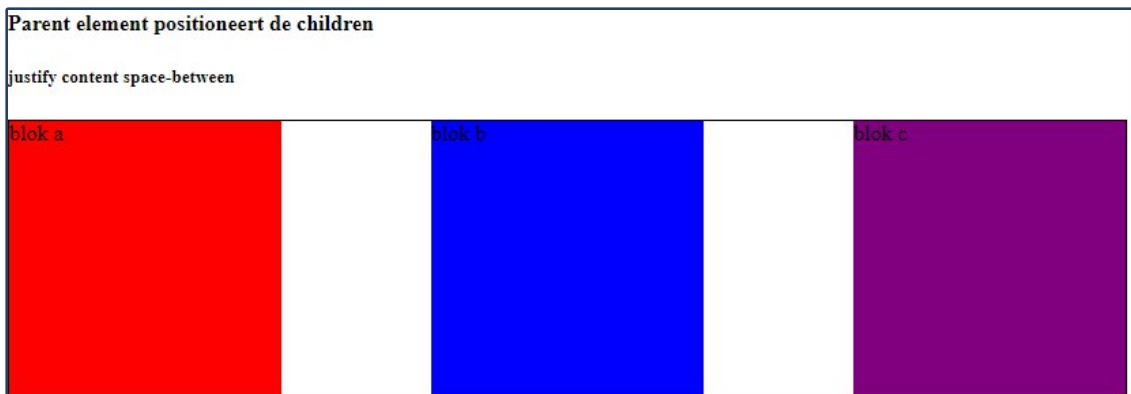
#### HTML

```
<h4>Parent element positioneert de children</h4>
<h5>justify content space-between</h5>
<div id="flexboxjustify" class="flexbox-container">
  <div id="bloka">blok a</div>
  <div id="blokb">blok b</div>
  <div id="blokc">blok c</div>
</div>
```

#### CSS

```
#flexboxjustify{
  width: 100%;
  height: 200px;
  border: 1px solid black;
  justify-content: space-between;
}
```

#### RESULTAAT



### 5.7.3.3. ALIGN-ITEMS

Align Items zorgt ervoor dat de parent verantwoordelijk is voor het positioneren van zijn children. m.a.w. het html element die de kinderen omsluit zal ook de kinderen positioneren.

OPMERKING: ALIGN-ITEMS ZORGT VOOR DE **VERTICALE** POSITIONERING VAN KINDEREN T.O.V. DE PARENT

#### 5.7.3.3.1. ALIGN-ITEMS: CENTER

Hiermee centreren we de kinderen verticaal t.o.v. de parent.

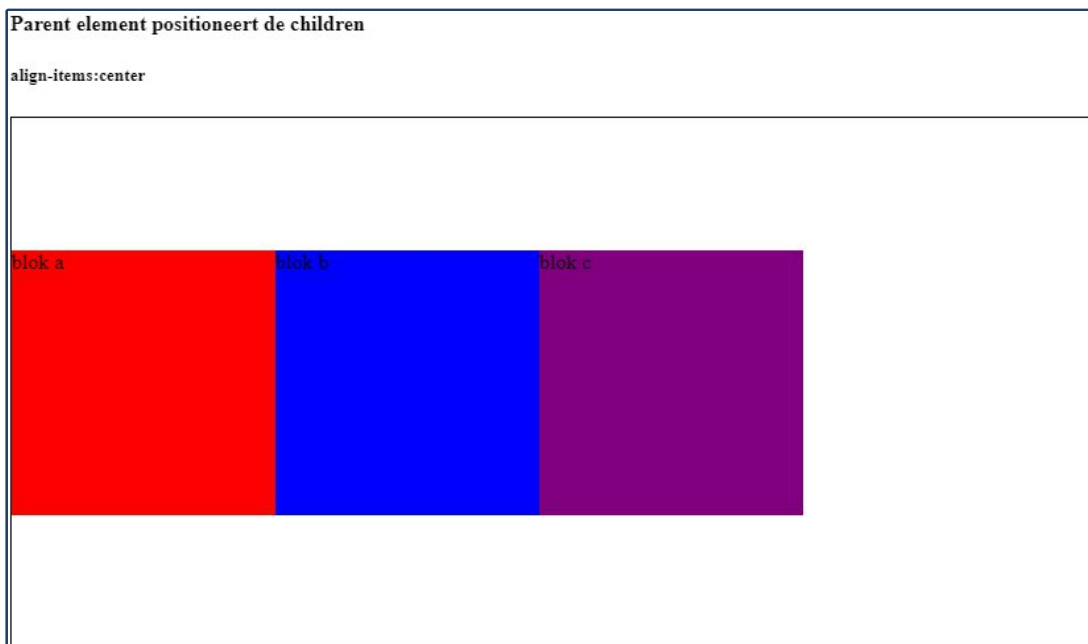
##### HTML

```
<h4>Parent element positioneert de children</h4>
<h5>align-items:center</h5>
<div id="alignitems" class="flexbox-container">
  <div id="bloka">blok a</div>
  <div id="blokb">blok b</div>
  <div id="blokc">blok c</div>
</div>
```

##### CSS

```
#alignitems{
  width: 100%;
  height: 400px;
  border: 1px solid black;
  align-items: center;
}
```

##### RESULTAAT



### 5.7.3.3.2. ALIGN-ITEMS:FLEX-START

De kinderen worden hiermee naar de linkerbovenhoek, nl. de beginpositie van de parent container gebracht.

#### HTML

```
<n4>Parent element positioneert de children</n4>
<h5>align-items:flex-start</h5>
<div id="alignitems" class="flexbox-container">
  <div id="bloka">blok a</div>
  <div id="blokb">blok b</div>
  <div id="blokc">blok c</div>
</div>
```

#### CSS

```
#alignitems{
  width: 100%;
  height: 400px;
  border: 1px solid black;
  align-items: flex-start;
}
```

#### RESULTAAT



### 5.7.3.3. ALIGN-ITEMS:FLEX-END

De kinderen worden hiermee naar de linkerhoek onderaan gebracht.

#### HTML

```
<h4>Parent element positioneert de children</h4>
<h5>align-items:flex-end</h5>
<div id="alignitems" class="flexbox-container">
  <div id="bloka">blok a</div>
  <div id="blokb">blok b</div>
  <div id="blokc">blok c</div>
</div>
```

#### CSS

```
#alignitems{
  width: 100%;
  height: 400px;
  border: 1px solid black;
  align-items: flex-end;
}
```

#### RESULTAAT



### 5.7.3.3.4. ALIGN-ITEMS:STRETCH

Hiermee wordt de height van de kinderen uitgerekt t.e.m. de height van de parent.

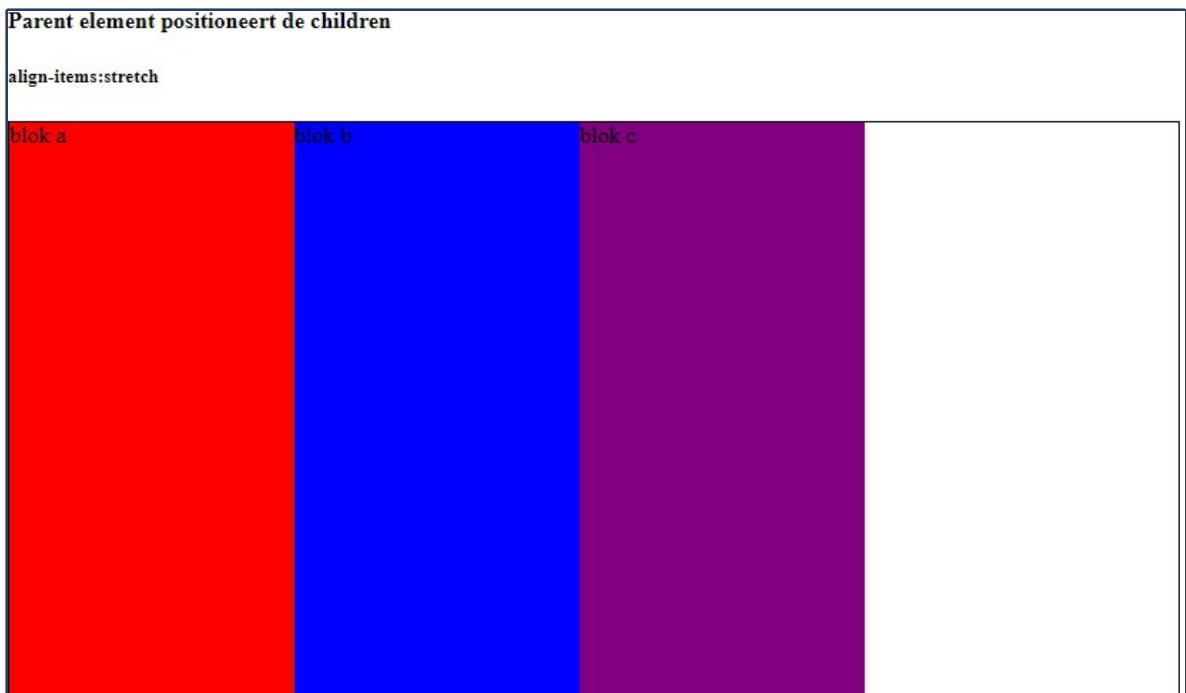
#### HTML

```
<h4>Parent element positioneert de children</h4>
<h5>align-items:stretch</h5>
<div id="alignitems" class="flexbox-container">
  <div id="bloka">blok a</div>
  <div id="blokb">blok b</div>
  <div id="blokc">blok c</div>
</div>
```

#### CSS

```
#alignitems{
  width: 100%;
  height: 400px;
  border: 1px solid black;
  align-items: stretch;
}
#bloka, #blokb, #blokc{
  /*height: 200px;*/
  width: 200px;
}
```

#### RESULTAAT



#### 5.7.3.4. ALIGN-SELF

Het verschil met alle voorgaande flexbox positioneringen is het volgende: hier zal het child element ZELF voor zijn eigen positionering zorgen. Ook hier heeft hij de mogelijkheden om zichzelf te centreren, links of rechts te plaatsen.

We passen ons voorbeeld lichtjes aan en demonstreren hier het centreren van zichzelf.

##### HTML

```
<h4>Child element positioneert zich t.o.v. de parent</h4>
<h5>align-self:center</h5>
<div id="alignitems" class="flexbox-container">
  <div id="bloka">blok a</div>
  <div id="blokb">blok b</div>
  <div id="blokc">blok c</div>
</div>
```

##### CSS

```
#alignitems{
  width: 100%;
  height: 400px;
  border: 1px solid black;
  align-items: stretch;
}
#bloka, #blokb, #blokc{
  /*height: 200px;*/
  width: 200px;
}
#bloka{
  background: red;
}
#blokb{
  background: blue;
  align-self: center;
}
#blokc{
  background: green;
}
```

##### RESULTAAT





In flexbox hebben we nog anderen mogelijkheden zoals: flex-order, flex-grow, flex-shrink die handig zijn in gebruik.

Flex-order zorgt ervoor dat je de volgorde van de elementen in de weergave kan wijzigen.

Flex-grow zorgt voor het vergroten van elementen en flex-shrink het verkleinen van elementen.

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.