



samen sterk voor werk

HTML & CSS

projecten

Deze cursus is eigendom van de VDAB

Inhoudsopgave

1	HET PROJECTENBOEK.....	7
1.1	Bestanden	7
1.2	Software en Hardware.....	7
2	DE PROJECTEN.....	10
2.1	Een eerste pagina	10
	<i>DoeI.....</i>	10
	<i>Theorie</i>	10
	<i>Startbestanden</i>	10
	<i>Opdracht.....</i>	10
2.1.1	<i>Structuur aanbrengen</i>	12
2.1.2	<i>Kennismaking met CSS.....</i>	14
2.1.3	<i>Het Link element.....</i>	16
2.1.4	<i>de volledige pagina</i>	17
2.2	Een geneste lijst.....	18
	<i>DoeI.....</i>	18
	<i>Theorie</i>	18
	<i>Startbestanden</i>	18
2.2.1	<i>Opdracht.....</i>	18
2.2.2	<i>Nesting.....</i>	18
2.3	Een kalender als tabel.....	21
	<i>DoeI.....</i>	21
	<i>Theorie</i>	21
	<i>Startbestanden</i>	21
2.3.1	<i>Een tabel opbouwen</i>	21
2.4	Een formulier	25
	<i>DoeI.....</i>	25
	<i>Theorie</i>	25
	<i>Startbestanden</i>	25
2.4.1	<i>het formulier.....</i>	25
2.4.2	<i>Formulierelementen.....</i>	26
2.4.3	<i>Andere invulvelden</i>	27
2.4.4	<i>Een keuzelijstje</i>	28
2.4.5	<i>Nog meer types invulvelden.....</i>	28

2.4.6	<i>Radiobuttons, textarea en checkboxes</i>	29
2.4.7	<i>Knoppen</i>	30
2.5	<i>basis CSS</i>	32
	<i>Doele</i>	32
	<i>Theorie</i>	32
	<i>Startbestanden</i>	32
	<i>Opdracht</i>	32
2.5.1	<i>Hoe een stylesheet indelen?</i>	32
2.5.2	<i>Een fixed lay-out</i>	36
2.5.3	<i>Werken in relatieve eenheden</i>	37
2.5.4	<i>Normaliseren</i>	37
2.5.5	<i>Een grid systeem</i>	39
2.5.6	<i>ClearFix</i>	42
2.5.7	<i>Koppen verbergen of opmaken</i>	43
2.5.8	<i>De menu's</i>	44
2.5.9	<i>Image replacement</i>	48
2.6	<i>Werken met Floats</i>	51
	<i>Doele</i>	51
	<i>Theorie</i>	51
	<i>Startbestanden</i>	51
	<i>Opdracht</i>	51
2.6.1	<i>Images</i>	51
2.6.2	<i>Box Floats</i>	52
2.6.3	<i>Clear een box</i>	54
2.6.4	<i>clearFix lost collapsing parent op</i>	55
2.6.5	<i>Kolommen maken met floats</i>	57
2.6.6	<i>CSS classes voor clear en clearFix</i>	58
2.7	<i>Enkele veel voorkomende lay-outs</i>	61
	<i>Doele</i>	61
	<i>Theorie</i>	61
	<i>Startbestanden</i>	61
2.7.1	<i>Outline om te lay-outen</i>	62
2.7.2	<i>Fixed width met drie kolommen</i>	63
2.7.3	<i>Faux columns</i>	67
2.7.4	<i>Fluid lay-out</i>	68

2.7.5	<i>box-sizing</i>	69
2.7.6	<i>Fluid Holy Grail</i>	71
2.8	Een thumbnaillist	74
	<i>Doe!</i>	74
	<i>Theorie</i>	74
	<i>Startbestanden</i>	75
2.8.1	<i>Opdracht</i>	75
2.8.2	<i>Opmaak met CSS</i>	76
2.8.3	<i>Meer of minder tekst</i>	79
2.9	Breadcrumbs met CSS driehoekjes	84
	<i>Doe!</i>	84
	<i>Theorie</i>	84
	<i>Startbestanden</i>	84
2.9.1	<i>HTML structuur</i>	84
2.9.2	<i>Een CSS driehoek</i>	84
2.9.3	<i>Het kruimeltjesspoor</i>	85
2.10	Een dropdown menu, enkel met CSS	88
	<i>Doe!</i>	88
	<i>Theorie</i>	88
	<i>Startbestanden</i>	88
2.10.1	<i>Een geneste lijst als startpunt</i>	88
2.10.2	<i>Normalize</i>	89
2.10.3	<i>De lijst omvormen naar menu</i>	89
2.10.4	<i>Een horizontaal menu</i>	90
2.10.5	<i>De submenu's</i>	92
2.10.6	<i>Mouse-over</i>	94
2.10.7	<i>Opmaken met CSS3 gradients</i>	94
2.10.8	<i>Slagschaduw</i>	96
2.10.9	<i>CSS3 transitions</i>	97
2.11	Bootstrap: een CSS Framework.....	100
	<i>Doe!</i>	100
	<i>Theorie</i>	102
	<i>Startbestanden</i>	102
2.11.1	<i>Bootstrap downloaden</i>	102
2.11.2	<i>Structuur aanbrengen</i>	103

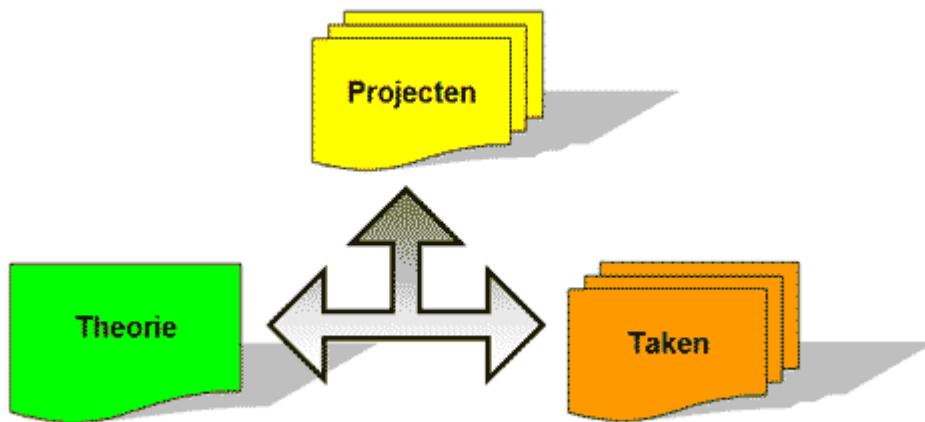
2.11.3	<i>Bootstrap koppelen en basis template</i>	104
2.11.4	<i>Een centrale container</i>	104
2.11.5	<i>een jumbotron</i>	106
2.11.6	<i>Het Bootstrap grid</i>	107
2.11.7	<i>Navigatie</i>	111
2.11.8	<i>Een openklappende navigatiebalk op Mobile</i>	113
2.11.9	<i>Het login formulier</i>	114
2.11.10	<i>Images</i>	115
2.11.11	<i>Glyphicons</i>	116
2.12	<i>Een Responsive Design, Mobile First</i>	118
	<i>Doe</i>	118
	<i>Theorie</i>	118
	<i>Startbestanden</i>	118
2.12.1	<i>Algemene css</i>	120
2.12.2	<i>Hoofdmenu</i>	122
2.12.3	<i>Helper classes</i>	123
2.12.4	<i>De thumbnailist</i>	124
2.12.5	<i>Media Queries</i>	125
2.12.6	<i>Tablet</i>	127
2.12.7	<i>Fluid images</i>	129
2.12.8	<i>De list voor een tablet</i>	130
2.12.9	<i>Grotere tablet</i>	132
2.12.10	<i>Desktops</i>	133
2.12.11	<i>Grote schermen</i>	139
2.13	<i>LESS: een CSS preprocessor</i>	142
	<i>Doe</i>	142
	<i>Theorie</i>	142
	<i>Startbestanden</i>	142
2.13.1	<i>Wat is LESS?</i>	142
2.13.2	<i>GUI compilers</i>	143
2.13.3	<i>WInless</i>	143
2.13.4	<i>Modulair werken</i>	145
2.13.5	<i>Variabelen</i>	145
2.13.6	<i>Berekeningen met eenheden</i>	147
2.13.7	<i>Geneste selectors</i>	147

2.13.8	<i>Mixins</i>	149
2.13.9	<i>Mixins met parameters</i>	150
2.13.10	<i>Navigatiebalk</i>	151
2.13.11	<i>Namespaces</i>	152
2.14	CSS omzetten naar LESS	155
	<i>DoeI</i>	155
	<i>Theorie</i>	155
	<i>Startbestanden</i>	155
2.14.1	<i>Welke CSS omzetten in LESS?</i>	155
2.14.2	<i>Ons project</i>	158
2.14.3	<i>Lusstructuur in LESS</i>	163
2.14.4	<i>Achtergrondkleuren</i>	165
2.15	Modernizr	169
	<i>DoeI</i>	169
	<i>Theorie</i>	169
	<i>Startbestanden</i>	169
2.15.1	<i>Wat is Modernizr?</i>	169
2.15.2	<i>Download Modernizr</i>	169
2.15.3	<i>CSS3 browser ondersteuning</i>	171
2.15.4	<i>Contextuele CSS</i>	171
2.15.5	<i>Multiple backgrounds</i>	172
2.15.6	<i>Textshadow</i>	173
2.15.7	<i>features combineren</i>	174
2.15.8	<i>Animations</i>	177

1 HET PROJECTENBOEK

Dit is het **projectenboek** dat hoort bij de theorie van de handleiding HTML&CSS, het praktische deel.

Er zijn nog twee andere delen: de **Theorie** en de **Taken**.



Een manier van werken is:

1. Volg een **project** uit het aparte *projectenboek*
2. Lees de **bijhorende theorie** in *diese handleiding*
3. Maak eventueel een **bijhorende taak**

Eventueel kan je 1 en 2 omwisselen. Het gaat er slechts over dat je ook de theorie in het project toegepast begrijpt.

Bij elk project staat duidelijk vermeld welk deel van de theorie je dient na te lezen.

1.1 Bestanden

Alle oefenbestanden vind je in de bijhorende *zip* file. De basisbestanden zijn in aparte projectmappen ingedeeld. Alle beeldmateriaal is voor het gemak gecentraliseerd in één map.

1.2 Software en Hardware

Welke software heb je nodig?

Editors

De oefeningen in de cursus kan je maken met eender welke **editor**.

Deze handleiding stelt zich op dat vlak neutraal op: dit is geen handleiding "websites bouwen met programma X".

Je kan een eenvoudige tekstverwerker die "platte tekst" schrijft gebruiken, zoals *Kladblok*, maar die is zeer beperkt in mogelijkheden.

Je kan ook meer gespecialiseerde editors of omgevingen gebruiken.

- voordelen: wysiwyg omgeving, syntaxherkenning, ftp-client, ..
- nadelen: niet altijd gratis, complexe omgeving

Enkele suggesties:

- Adobe **Dreamweaver** (betalend) www.adobe.com
- **Aptana Studio** (freeware) www.aptana.com
- **Crimson editor** (freeware) www.crimsoneditor.com/
- **Notepad++**(freeware) notepad-plus-plus.org/
- **Eclipse** (freeware) <http://www.eclipse.org/>

en er zijn er nog talloze andere.

Daarnaast heb je natuurlijk ook nog de volledige ontwikkelomgevingen van **.Net** of **NetBeans** voor C# en Javaontwikkelaars.

Browsers

We verwachten dat je de belangrijkste browsers installeert: *IE, FF, Chrome, Opera, Safari*. Vergeet ook de browsers op mobiele toestellen niet : Opera Mobiel, Opera Mini, ...

Controleer telkens je oefeningen in meerdere browsers! In de professionele omgeving van een bedrijf is een website die niet werkt in een belangrijke browser onaanvaardbaar.

Tools

In elke browser heb je tegenwoordig tools onder de vorm van *Add-ins* of *Plug-Ins* of zelf ingebouwd die van onschatbaar belang zijn voor het *debuggen* van fouten. Je mag de volgende tools installeren en bekijken:

Firebug voor Firefox (add-in):

Eenmaal geïnstalleerd:

- lees de documentatie op <http://getfirebug.com/wiki/index.php>
- bekijk zeker de panels *HTML, CSS, Net*

Developer tools for Chrome (geïntegreerd):

Eenmaal gevonden (menu, F12, Ctrl-Shift-I):

- lees de documentatie op <http://code.google.com/intl/nl/chrome/devtools/>
- bekijk zeker het panel *Elements*

Developer tools for IE (geïntegreerd):

Eenmaal gevonden (menu, F12):

- lees de documentatie op [http://msdn.microsoft.com/en-us/library/dd565622\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/dd565622(v=vs.85).aspx)
- bekijk zeker de panels *HTML*, *CSS*
- kijk hoe je kan switchen tussen Browser mode: IE7, 8, 9

Emulators

De doelgroepen (zie dot-brochure) die ontwikkelen voor mobiele toestellen en *Responsive Web Design* kunnen ook specifieke emulators en browsers installeren. We denken hierbij aan:

- Opera Mobile Emulator
- Fennec

Hardware

Voor dezelfde doelgroepen zijn er verschillende mobiele toestellen (*tablets* en *phones*) beschikbaar om te gebruiken: vraag je coach ernaar.

Bij sommige projecten vermelden we speciaal dat je moet testen op een mobiel toestel.

2 DE PROJECTEN

2.1 Een eerste pagina

Doel

Een eerste html pagina opbouwen. Kennismaking met de voornaamste elementen. De pagina structureren. Kennismaking met CSS

Theorie

Lees de volgende theorie topics na:

- de eerste 4 hoofdstukken - "Inleiding", "Over deze cursus", "Web standaarden", "HTML" - en lees tijdens dit project Hoofdstuk 5 "de HTML elementen" mee
- ook de topic "Structuur aanbrengen in een HTML pagina"
- CSS intro, CSS syntax, CSS en HTML

Startbestanden

geen

Opdracht

We maken een lege HTML5 webpagina in een eenvoudige tekstverwerker.

Maak een map *projecten*, maak daarin een submap *degazet*, maak daarin de submap *css*

Open een nieuw document in **kladblok** of een andere simpele tekstverwerker, gebruik voorlopig nog geen HTML editor zoals Dreamweaver.

De basis

Een webpagina bestaat in essentie uit een **declaratie** en een **root element**: het `html` element:

```
<!DOCTYPE HTML>
<html>
</html>
```

Bespreking:

- de DOCTYPE wordt conventioneel in Hoofdletters geschreven. Deze **Document Type Declaration** zegt dat dit document HTML5 **is** en dus HTML5 code zal bevatten.
Dat betekent dat we volgens de **HTML5 web standaard** werken.
- het `html` element en alle andere elementen worden conventioneel in **kleine letters** geschreven, alhoewel ze niet hoofdlettergevoelig zijn.
- het `html` element is het **root element**: alle ander elementen zullen zijn **children** zijn en er dus **in** zitten

We voegen meer toe:

```
<!DOCTYPE HTML>
<html>
  <head></head>
  <body></body>
</html>
```

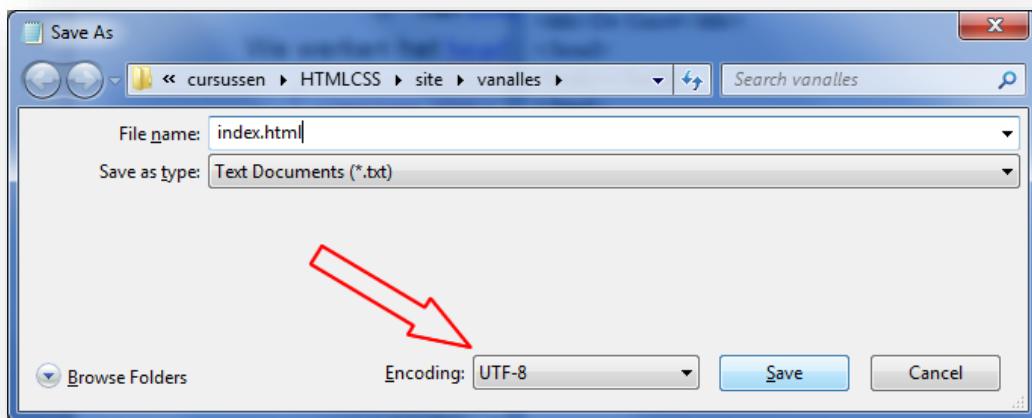
- het root element heeft twee *children*: **head** en **body**
 - het **head** element bevat *meta-informatie*: informatie die niet visueel gezien kan worden
 - het **body** element bevat de eigenlijke inhoud

We werken het **head** element verder uit:

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>De Gazer</title>
  </head>
  ...

```

- het **meta** element bevat het attribuut **charset** met de waarde "**utf-8**". dit verklaart dat dit document opgeslagen is (zal worden) met de characterset utf-8, niet ANSI of iets anders. Waarom utf-8 lees je in de theorie.
- het **title** element bevat de tekst die bovenaan het browservenster of tabblad te zien zal zijn
- Sla nu het document op als *index.html* in de map *degazer* maar let erop dat je de **encoding** op **utf-8** zet.
- Let er ook op dat de extensie *.html* is en geen *.txt*



Straks werken we verder in een html-editor, en daar zal de *encoding* waarschijnlijk standaard op utf-8 staan. We vullen de **body** tag aan:

```
...
<body>
```

```
<h1 id="logo">De Gazet</h1>
<h2>Vanaf oktober meer files op wegen</h2>
<p>Lorem ipsum dolor sit amet</p>
</body>
...
```

- de kop **h1** verschaft een zichtbare titel voor de pagina
- de kop **h2** is de titel van een nieuwsartikel
- de alinea **p** bevat een dummy tekst

Sla de pagina opnieuw op en bekijk in een browser (Firefox, Chrome, Internet Explorer, Opera,...):



Je bemerkt de **title** tekst in het tabblad, de **h1**, de **h2** en de **p** tekst.

2.1.1 Structuur aanbrengen

Je mag nu het bestand openen in een html-editor naar keuze en verder werken:
Dreamweaver, Eclipse, Visual studio etc...

Onthou van het vorige dat

- een HTML pagina een eenvoudig **tekstbestand** is met een bepaalde **encodering**
- het bevat HTML code onder de vorm van **tags**
- een **tag** stelt een HTML **element** voor

Nu brengen we **meer structuur** aan in de pagina, zowel om de **inhoud** meer structuur te geven alsook met het oog op de **lay-out** die we later willen toepassen. De volle betekenis hiervan wordt later duidelijk.

```
...
<body>
<div class="wikkel">
  <div class="inhoud">
    <header>
```

```
<h1 id="logo">De Gazet</h1>
<nav id="hoofdmenu"> </nav>

</header>
<section>
  <h2>Nieuws</h2>
  <article>
    <h2>Vanaf Oktober meer files op wegen</h2>
    <p>Lorem ipsum dolor sit amet</p>
  </article>
  <article>
    <h2>Gemeenteraad Genk wil kerncentrale Borssele dicht voor controle</h2>
    <p>Lorem ipsum dolor sit amet</p>
  </article>
</section>

<aside>
</aside>

<!--einde inhoud-->
</div>
<!--einde wikkел-->
</div>
<footer>
  <div class="wikkел">De Gazet is een oefening voor HTML & CSS</div>
</footer>
</body>
</html>
```

Bespreking:

- binnen de **body** zetten we een **div.wikkел** (een **div** met **class wikkел**). Die zal later dienen voor de lay-out
- daarbinnen volgt nog een **div.inhoud**. Deze heeft een meer semantische **class inhoud**
- daarbinnen plaatsen we een **header** element: het is duidelijk: diens inhoud zal de kop van de pagina worden
 - en die **header** krijgt een **nav** element: dat wordt straks een navigatiestructuur. Het element wordt geïdentificeerd met een **id** attribuut
 - een **header** element vormt de "kop-informatie" van een ander element: dat kan van de **body** zijn maar even goed van een **div**, **section** of **article**. Dezelfde pagina kan dus meerdere **header**'s bevatten.
- na de **header** volgt een **section**. Dit is een element dat kan meespelen als we een *outline* zouden maken. Het geeft een inhoudelijke verdeling aan: we noemen ze *sectioning elements*
- daarbinnen zitten eerst een **h2** kop,
- daarna volgen enkele **article** elementen. Hier bevatten ze exact wat ze zeggen: het zijn nieuwsartikels met elk hun tekst en kop (**h2**)

- een tweede **aside** element volgt de **section** op. De naam zegt het al, dit zal inhoud bevatten die niet tot de hoofdmoot van het nieuws behoort. Ook aside elementen zijn *sectioning elements*
- een **footer** element volgt op de **div**
 - een **footer** bevat de "voet-informatie" van een andere element: dat kan de **body** zijn maar evengoed een **div**, **section** of **article**. Dezelfde pagina kan dus meerdere **footer**'s bevatten.
 - en heeft ook een **class** *wikkel*: dit duidt er op dat een **class** niet dient om een element uniek te identificeren maar om een soort *eigenschap* mee te geven: zowel de **div** als de **footer** kunnen de class *wikkel* hebben en andere elementen kunnen die ook krijgen.
Die **class** zullen we gebruiken om een opmaak mee te geven
- Het kan nooit kwaad d.m.v. **commentaartags** aan te geven waar een gedeelte eindigt: momenteel hebben we slechts een 30-tal lijnen, straks honderden

Wat hebben we hieruit geleerd?

- een **id** attribuut dient om een element te **identificeren**: zijn waarde moet uniek zijn, m.a.w. geen enkel andere element mag hetzelfde **id** hebben.
Het wordt gebruikt in CSS, maar wordt vooral gebruikt met Javascript voor scripting
- een **class** dient vooral om **opmaak** mee toe te passen via CSS.
Een element kan meerdere classes hebben en kan dezelfde **class** delen met andere elementen
- We maken gebruik van een aantal structurele box-elementen: hier **div**, **section** en **article**.
 - hun hiërarchie is onbepaald: je kan evengoed **section > article > div** doen als **div > article > section**. Naar eigen goeddunken.
 - **section** en **article** zijn *sectioning elements*: ze spelen een rol in een eventuele outline (genummerde inhoudstafel).
 - **div** is neutraal: gebruik het als je een 'doos' nodig hebt die enkel voor lay-out gebruikt wordt

Bekijk nu de pagina: de enige zichtbare elementen zijn nog steeds de koppen en de alinea's, van de **div**, **article** en **section** elementen is niets te zien. Dat zal enkel gebeuren als we er CSS aan koppelen.

2.1.2 Kennismaking met CSS

Tijd voor een eerste kennismaking met CSS zonder echt diep te gaan, dat is voor later.



Het principe "scheiding der machten": **HTML = inhoud, CSS = opmaak**

Voeg een **link** element toe in de **head** van het document:

```
<!DOCTYPE HTML>
<html>
<head>
```

```
<meta charset="utf-8">
<link href="css/gazet.css" rel="stylesheet">
<title>De Gazet</title>
</head>
```

Maak een nieuw **stylesheet** aan - *gazet.css* - en sla dat op in de map *degazet/css*.

In dit bestand zetten we het volgende:

```
@charset "utf-8";
/* CSS Document voor deGazet */

body {
    font-family:Georgia, "Times New Roman", Times, serif;
    color:#333;
}
h1, h2, h3, h4, h5, h6 {
    color:#005689;
}
```

Bespreking:

- ook het stylesheet heeft **utf-8** als karakterset.

opmerking voor Visual studio gebruikers:

VS erkent deze de CSS **@charset** rule (onrechtmatig) niet en geeft daardoor een fout. Je mag deze rule dus achterwege laten zolang je controleert dat je stylesheet wel degelijk in utf-8 encoding opgeslagen wordt.

- commentaar kan enkel in **/* */**
- daarna volgen twee **style rules**, die elk bestaan uit een **selector** gevolgd door een **block statement { }** met CSS **properties**:
 - de eerste rule heeft **body** als selector waarvoor we het lettertype en de letterkleur instellen via de eigenschappen **font-family** en **color**
 - de tweede rule heeft als selector **h1, h2, h3, h4, h5, h6**. Dus een verzameling van elementen. De property **color** zal op al deze toegepast worden

Bekijk de pagina opnieuw dan zie je het volgende:

De Gazet

Nieuws

Vanaf Oktober meer files op wegen

Lorem ipsum dolor sit amet

Gemeenteraad Genk wil kerncentrale B

Lorem ipsum dolor sit amet

Er is opmaak toegepast op de pagina elementen: de tekst heeft een ander lettertype en is grijs, de koppen blauw.

Waarom is ook de alinea `p` grijs gekleurd? Een element **erft zijn eigenschappen van zijn parent**.

Eigenlijk zit er in het `body` element geen tekst: alle tekst zit in de *child* elementen `p` of `hx`. Dus een `p` erft zijn kleur van zijn parent `article`

die zelft erft van zijn `section`

die zelft erft van zijn `div`

die zelft erft van de `body`.

Dit noemen we **inheritance**.

Waarom hebben de `hx..` koppen dan niet de kleur #333 (grijs)? Omdat ze een andere kleur krijgen via hun eigen `selector` die het overerven onderbreekt.

Dit is voorlopig voldoende over opmaak.

2.1.3 Het `link` element

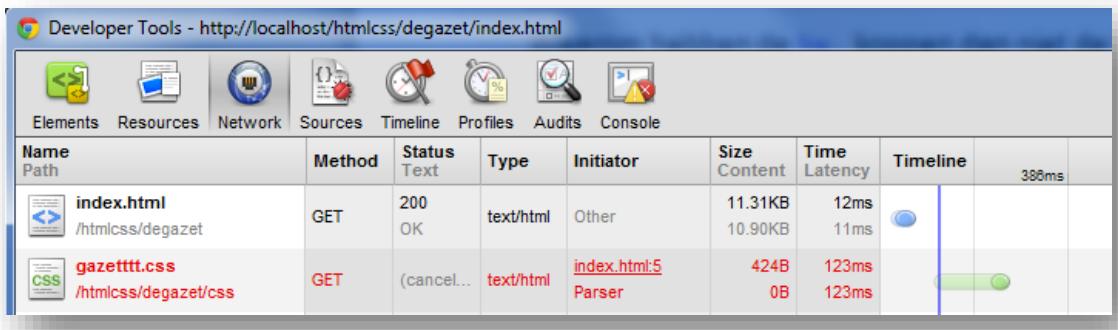
Bekijk nog even het `link` element in de HTML code. Het bevat een `href` attribuut met het pad naar het stylesheet en een `rel` attribuut die duidelijk maakt dat het hier een **stylesheet** betreft. Beide attributen zijn verplicht en noodzakelijk.

Het `link` element laadt een **externe bron**, in dit geval een *stylesheet*. Dat inladen gebeurt op het ogenblik dat de `parser` het element tegenkomt in de HTML code.

Er gaat natuurlijk ook wel eens iets fout waardoor er geen opmaak toegepast wordt. Dat kunnen we simuleren door eens een typefout in de bestandsnaam te maken, bijvoorbeeld: wijzig de bestandsnaam van het stylesheet in `gazettt.css`.

Bekijk dan de pagina in Chrome en open het `webdeveloper tool`. Ga naar het tabblad "Network" en herlaad de pagina.

Je ziet hier duidelijk welke **bronnen** allemaal geladen worden voor deze pagina. Je merkt ook dat het stylesheet niet gevonden is: het is rood aangegeven:



Een verkeerd pad of een foute bestandsnaam ingeven is een veel voorkomende foutje...

2.1.4 de volledige pagina

Omdat we je heel wat werk willen besparen, krijg je de verdere inhoud cadeau!

Open het startbestand *index.txt* en hernoem het naar *index.html* en overschrijf daarmee het vorige bestand (of geef dat een andere naam).

Bekijk de code: herken je de structuur die je daarnet maakte?

Opmerking:

de html bevat images (`img` elementen) die koppelen naar een externe site:

<http://lorempixel.com>

Die verschaft ons dummy images op elke maat en topic die je maar wil. Maar je moet er wel een internetverbinding voor hebben. Heb je dat niet dan laden ze niet in. Vervang ze eventueel door eigen dummy images.

2.2 Een geneste lijst

Doel

Ordered en unordered lists in elkaar verwerken voor een "Meest recent/gelezen/getipt" structuur - "rankings" - op de voorpagina van de krant.

Theorie

Lees de volgende theorie topics na:

- HTML lijsten

Startbestanden

degazet/index.html

2.2.1 Opdracht

Open het bestand in je editor en bekijk het terzelfdertijd in een browser. Zoek in de code de lijnen

```
<!--start rankings-->
<pre>
    rankings hier
</pre>
<!--einde rankings-->
```

Je mag de `pre` tag met zijn inhoud verwijderen en vervangen door

```
<ul id="rankings">
    <li id="meest-recent"></li>
    <li id="meest-getipt"></li>
    <li id="meest-gelezen"></li>
</ul>
```

Bespreking:

- Dit is een gewone *unordered list*: een `ul` met `li` elementen
- een `ol` of `ul` element mag nooit leeg zijn: er moet minstens één `li` element zijn
- een `ol` of `ul` element mag niets anders als *child* hebben dan `li` elementen
- de `id` attributen zullen we later nog nodig hebben.

Op het browservenster zijn enkel 3 bolletjes te zien, geen verdere inhoud.

2.2.2 Nesting

In elk van deze `li` elementen plaatsen we nu verdere inhoud: een titel en een *ordered list*.

We starten met "meest-recent":

```
<ul id="rankings">
    <li id="meest-recent">
```

```
<h3>Meest recent</h3>
<ol>

    <li>
        <time>di 10:46</time>
        <a href="/artikel/artikel.php?artid=123">Vanaf Oktober meer files op
        wegen</a></li>
    <li>
        <time>di 10:48</time>
        <a href="/artikel/artikel.php?artid=456">Gemeenteraad Genk wil
        kerncentrale Borssele dicht voor controle</a></li>
    <li>
        <time>di 10:52</time>
        <a href="/artikel/artikel.php?artid=789">Rattenplaag teistert
        Oostende</a></li>
    <li>
        <time>di 11:22</time>
        <a href="/artikel/artikel.php?artid=988">Drie gewonden bij botsing
        kusttram</a></li>
    <li>
        <time>di 11:34</time>
        <a href="/artikel/artikel.php?artid=884">Strenge winter 2012-2013
        verwacht</a></li>
    </ol>
</li>
<li id="meest-getipt"></li>
<li id="meest-gelezen"></li>
</ul>
```

Bespreking:

- om een lijst correct te nesten, zorg ervoor dat de binnenlijst **volledig in het buitenste li element zit**:

correct:

```
<ul id="rankings">
    <li> <h3>Meest recent</h3> <ol>...binnenlijst...</ol> </li>
    <li>...
</ul>
```

NIET correct:

```
<ul id="rankings">
    <li> <h3>Meest recent</h3> </li> <ol>...binnenlijst...</ol>
    <li>...
</ul>
```

- het **ol** element bevat zelf **li** elementen die elk
 - een **h3**, een **time** element en een **a** hyperlink bevatten
 - **time** is een semantisch HTML5 inline element dat we bij voorkeur gebruiken om datums en tijd in weer te geven

- de hyperlinks zijn dummy links die nergens naar leiden

Om nu de twee andere rubrieken "Meest gelezen" en "Meest getipt" te maken mag je de vorige binnenlijst kopiëren, wat aanpassen en de volgorde van de hyperlinks wat aanpassen.

Tenslotte krijg je dit:

- **Meest recent**
 1. di 10:46 [Vanaf Oktober meer files op wegen](#)
 2. di 10:48 [Gemeenteraad Genk wil kerncentrale Borssele dicht voor controle](#)
 3. di 10:52 [Rattenplaag teistert Oostende](#)
 4. di 11:22 [Drie gewonden bij botsing kusttram](#)
 5. di 11:34 [Strenge winter 2012-2013 verwacht](#)
- **Meest getipt**
 1. di 10:52 [Rattenplaag teistert Oostende](#)
 2. di 10:46 [Vanaf Oktober meer files op wegen](#)
 3. di 11:34 [Strenge winter 2012-2013 verwacht](#)
 4. di 10:48 [Gemeenteraad Genk wil kerncentrale Borssele dicht voor controle](#)
 5. di 11:22 [Drie gewonden bij botsing kusttram](#)
- **Meest gelezen**
 1. di 11:34 [Strenge winter 2012-2013 verwacht](#)
 2. di 10:52 [Rattenplaag teistert Oostende](#)
 3. di 10:46 [Vanaf Oktober meer files op wegen](#)
 4. di 11:22 [Drie gewonden bij botsing kusttram](#)
 5. di 10:48 [Gemeenteraad Genk wil kerncentrale Borssele dicht voor controle](#)

Taken:

maak nu volgende taken

- "een geneste lijst: culinair"

2.3 Een kalender als tabel

Doel

We maken een kalendertje met een HTML tabel.

Theorie

Lees de volgende theorie topics na:

- HTML tabellen

Startbestanden

werk verder in *degazet/index.html*

2.3.1 Een tabel opbouwen

Tabellen worden gebruikt voor gegevens. Ze hebben een relatief ingewikkelde structuur en zijn daardoor wat moeilijker om aan te passen. Je moet ze **nooit gebruiken voor lay-out!**

De tabel in ons voorbeeld zou kunnen dienen als basis voor een dynamische kalender-widget waar je op een datum kunt klikken en zo je agenda invullen etc..

Hier leren we enkel de HTML structuur opbouwen en straks er CSS op gebruiken, onze tabel is nergens aan gekoppeld.

Zoek in de pagina de code

```
<!--start kalender-->
<pre>
    kalender hier
</pre>
<!--einde kalender-->
```

en vervang het `pre` element door onderstaande code:

```
<table title="Oktober 2012" summary="De kalender van Oktober 2012" id="kalender">
</table>
```

- het `table` element heeft een `id` en een `title` attribuut. Dit laatste wordt de *tooltiptekst* die verschijnt als je met de muis erover gaat
- het heeft ook een `summary` attribuut dat geen zichtbare informatie geeft. Het wordt echter gebruikt door tekstlezers om duidelijk te maken aan slechtzienden waarover de tabel gaat.

We vullen verder aan:

```
<table title="Oktober 2012" summary="De kalender van Oktober 2012" id="kalender">
    <thead></thead>
    <tbody></tbody>
</table>
```

- een `table` element wordt meestal onderverdeeld in een `thead`, `tfoot` en één of meer `tbody` elementen. `thead` en `tfoot` zijn niet verplicht.

Nu maken we rijen en kolommen:

```
<table title="Oktober 2012" summary="De kalender van Oktober 2012" id="kalender">
  <thead>
    <tr>
      <th title="Maandag">Ma</th>
      <th title="Dinsdag">Di</th>
      <th title="Woensdag">Wo</th>
      <th title="Donderdag">Do</th>
      <th title="Vrijdag">Vr</th>
      <th title="Zaterdag">Za</th>
      <th title="Zondag">Zo</th>
    </tr>
  </thead>
  <tbody></tbody>
</table>
```

- een **tr** element vormt een rij
- de **th** elementen vormen kolommen. **th** wordt gebruikt voor **koppen** (zowel rijkoppen als kolomkoppen) terwijl een **td** element voor een gewone cel gebruikt wordt
- de **th** elementen hebben een **title** attribuut die een *tooltiptekst* zal tonen

Nu vullen we de **tbody** aan - type daarvoor de eerste rij, kopieer die dan 4-maal en pas ze aan:

```
...
</thead>
<tbody>
  <tr>
    <td><a href="agenda/2012/okt/1">1</a></td>
    <td><a href="agenda/2012/okt/2">2</a></td>
    <td><a href="agenda/2012/okt/3">3</a></td>
    <td><a href="agenda/2012/okt/4">4</a></td>
    <td><a href="agenda/2012/okt/5">5</a></td>
    <td><a href="agenda/2012/okt/6">6</a></td>
    <td><a href="agenda/2012/okt/7">7</a></td>
  </tr>
  <tr>
    <td class="vandaag"><a href="agenda/2012/okt/8">8</a></td>
    <td><a href="agenda/2012/okt/9">9</a></td>
    <td><a href="agenda/2012/okt/10">10</a></td>
    <td><a href="agenda/2012/okt/11">11</a></td>
    <td><a href="agenda/2012/okt/12">12</a></td>
    <td><a href="agenda/2012/okt/13">13</a></td>
    <td><a href="agenda/2012/okt/14">14</a></td>
  </tr>
  <tr>
    <td><a href="agenda/2012/okt/15">15</a></td>
    <td><a href="agenda/2012/okt/16">16</a></td>
    <td><a href="agenda/2012/okt/17">17</a></td>
```

```
<td><a href="agenda/2012/okt/18">18</a></td>
<td><a href="agenda/2012/okt/19">19</a></td>
<td><a href="agenda/2012/okt/20">20</a></td>
<td><a href="agenda/2012/okt/21">21</a></td>
</tr>
<tr>
<td><a href="agenda/2012/okt/22">22</a></td>
<td><a href="agenda/2012/okt/23">23</a></td>
<td><a href="agenda/2012/okt/24">24</a></td>
<td><a href="agenda/2012/okt/25">25</a></td>
<td><a href="agenda/2012/okt/26">26</a></td>
<td><a href="agenda/2012/okt/27">27</a></td>
<td><a href="agenda/2012/okt/28">28</a></td>
</tr>
<tr>
<td><a href="agenda/2012/okt/29">29</a></td>
<td><a href="agenda/2012/okt/30">30</a></td>
<td><a href="agenda/2012/okt/31">31</a></td>
<td class="volgendeMaand"><a href="agenda/2012/nov/1">1</a></td>
<td class="volgendeMaand"><a href="agenda/2012/nov/2">2</a></td>
<td class="volgendeMaand"><a href="agenda/2012/nov/3">3</a></td>
<td class="volgendeMaand"><a href="agenda/2012/nov/4">4</a></td>
</tr>
</tbody>
</table>
```

- normaal bevatten alle rijen (**tr**) evenveel kolommen (**td**)
- elke cel bevat dan een hyperlink - zogezegd om iemand toe te laten de agenda te raadplegen voor die dag.
- er wordt ook gebruikt gemaakt van **classes** om een aantal zaken te kunnen aanduiden:
 - de dag van *vandaag*
 - dagen die op de kalender voorkomen, maar tot de *volgende* of *vorige* maand behoren

We voegen een koprij toe aan de **thead**:

```
...
<thead>
  <tr>
    <th colspan="7"> Okt 2012
      <a title="Vorige maand" href="agenda/2012/sep" class="previous">&lt;</a>
      <a title="Volgende maand" href="agenda/2012/nov" class="next">&gt;></a>
    </th>
  </tr>
  <tr>
    <th title="Maandag">Ma</th>
    <th title="Dinsdag">Di</th>
    <th title="Woensdag">Wo</th>
    <th title="Donderdag">Do</th>
```

```
<th title="Vrijdag">Vr</th>
<th title="Zaterdag">Za</th>
<th title="Zondag">Zo</th>
</tr>
</thead>
...
...
```

- deze rij bevat slechts één **th** element dat de 7 oorspronkelijke rijen "overspant" door middel van een **colspan** attribuut. Daarmee versmelten we 7 cellen tot één.
- de inhoud bevat naast tekst ook twee hyperlinks naar de vorige en volgende maand
- de **<** *entity* toont een < teken, de **>** *entity* een > teken

Als we het resultaat bekijken zien we dit:

Okt 2012 <>						
Ma	Di	Wo	Do	Vr	Za	Zo
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

Hiermee is de HTML van onze tabel klaar. Straks passen we er opmaak op toe via CSS.

Taken:

maak nu volgende taken

- "*tabellen: het weer*"

2.4 Een formulier

Doele

Leren omgaan met formulieren. De meest courante formulier elementen leren gebruiken.

Theorie

Lees de volgende theorie topics na:

- HTML formulieren

Startbestanden

gegevens.html

2.4.1 het formulier

De GAZET biedt de mogelijkheid aan klanten om zich online te abonneren. Het abonneerproces bestaat uit een vijftal stappen:

keuze abonnement > gegevens invullen > bevestigen > betalen

In onze site worden deze stappen vertegenwoordigd door 4 aparte html pagina's. We bewerken nu de 2^{de} pagina in het proces: de gegevenspagina.

Bekijk de broncode van het startbestand. Enkel het formulier moet nog aangemaakt worden. Zoek de code:

```
<!--form hier-->
<pre>
  formulier hier
</pre>
```

en vervang het door

```
...
<!--form hier-->
<form action="bevestigen.html" method="get" id="frmgegevens" name="frmgegevens">
</form>
...
```

- een **form** element moet minstens:
 - een **action** attribuut hebben waarvan de waarde bepaalt **waar** de gegevens naartoe gaan: een andere pagina, een script,...
 - een **method** attribuut hebben dat zegt **hoe** die gegevens verstuurd worden. Meestal is dat "**get**" of "**post**", lees na in de theorie wat het verschil is en de andere mogelijkheden zijn
- kan een **id** een **name** attribuut hebben die dienen om het te kunnen identificeren (denk er aan dat er meerdere formulieren op dezelfde pagina kunnen zitten)
- Een **form** element is normaal niet zichtbaar, maar kan zichtbaar gemaakt worden door CSS en dus opgemaakt met randen, kleurtjes, etc...

2.4.2 Formulierelementen

We vullen het **form** element in met formulier elementen (*controls*):

```
...
<!--form hier-->
<form action="bevestigen.html" method="get" id="frmgegevens" name="frmgegevens">
<p>Velden met een <abbr class="verplicht" >&#9829;</abbr> zijn essentieel om uw
abonnement te kunnen registreren.</p>
<div>
    <label for="vnaam" class="verplicht" title="verplicht">Voornaam:</label>
    <input type="text" id="vnaam" name="vnaam" autofocus placeholder="uw voornaam"
title="vul hier uw voornaam in " required>
    <span class="fieldval" id="val_vnaam"></span>
</div>
</form>
...
```

- eerst een **p** element met een tekst waarin het symbool aangebracht wordt waarmee verplichte velden aangeduid worden
 - **♥** is een *Unicode entity* voor een hartje.
Je vindt een overzicht op Wikipedia "List of Unicode characters".
Je moet deze codes wel controleren voor browsercompatibiliteit!
 - we gebruiken bij voorkeur karakter *entities* voor symbooltjes om de last van in te laden figuren te verminderen
- daarna volgt een **div** die als container dienst doet voor elk formulierelement:
- een **label** element bevat de begeleidende tekst. Het heeft
 - een **for** attribuut die het koppelt aan de name van het *formcontrol*
 - een **class** attribuut zal via css het hartje tonen
 - een **title** attribuut voor tooltiptekst
- een **input** element **type=text** is een gewoon invulveld. Het heeft
 - een verplicht **type** attribuut dat aangeeft welk soort formulieveld dit is.
Lees de theorie hierop na
 - een verplicht **name** attribuut met zijn naam. Superbelangrijk, want zonder dit veld worden geen gegeven doorgegeven
 - een niet verplicht **id** attribuut dat in de meeste gevallen identiek aan name kan zijn
 - een **autofocus** attribuut dat ervoor zorgt dat de cursor eerst en automatisch in dit veld terechtkomt bij het laden van de pagina
 - een **placeholder** attribuut waarvan je de waarde als geheugensteuntje in het veld ingevuld zult zien. Als je erop klikt verdwijnt de tekst.
Placeholders kunnen labels vervangen
 - een **title** attribuut toont een tooltiptekst
 - een **required** attribuut geeft aan de browser/of aan een validatiescript aan dat dit veld verplicht is. Dit kan geen vervanger zijn voor ons "verplicht" symbooltje
- een lege **span class=fieldval** zal gebruikt worden door validatiescripts om er een foutbericht in te plaatsen indien nodig. We voorzien dit element nu reeds om straks de opmaak goed te kunnen controleren

Je ziet nu het volgende:

Uw gegevens

Velden met een ♥ zijn essentieel om uw abonnement te kunnen registreren.

Voornaam:

Voor verdere assistentie en hulp betreffende de abonnementen kunt u het gr.
U vind ook help bij de [Veel Gestelde Vragen en Voorwaarden](#) en [condities](#)

De GAZET is een oefening voor HTML & CSS

2.4.3 Andere invulvelden

Vul nu de volgende velden verder aan:

```
<div>
    <label for="fnaam" class="verplicht" >Familienaam:</label>
    <input type="text" id="fnaam" name="fnaam" placeholder="uw familienaam" title="vul hier uw familienaam" required >
    <span class="fieldval" id="val_fnaam"></span>
</div>
<div>
    <label for="straat" class="verplicht" >Straat:</label>
    <input type="text" id="straat" name="straat" title="uw straat met huisnummer" required >
    <span class="fieldval" id="val_straat"></span>
</div>
<div>
    <label for="postnr" class="verplicht" >Postnummer:</label>
    <input type="number" id="postnr" name="postnr" class="kort" title="het postnummer van uw gemeente" required>
    <span class="fieldval" id="val_postnr">hier moet u een postnummer invullen</span>
</div>
<div>
    <label for="gemeente" class="verplicht" >Gemeente:</label>
    <input type="text" id="gemeente" name="gemeente" title="de gemeente waar u woont" required >
    <span class="fieldval" id="val_gemeente"></span>
</div>
...

```

- hier is weinig nieuws te melden enkel
- een `input type=number` is een nieuw html5 `type` dat specifiek dient om enkel nummers te accepteren. Je zult merken dat sommige browsers er een tellervakje van maken waarmee je het getal kunt verhogen verlagen, andere doen dat niet
- de `class=kort` zullen we gebruiken om de breedte van het veld te beperken

- hier hebben we – om te kunnen testen – de `span class=fieldval` eens ingevuld met een vaste tekst. Die zullen we uiteindelijk verwijderen, want die moet er dynamisch ingevuld worden.

2.4.4 Een keuzelijstje

We vullen verder aan

```
...
<div>
    <label for="land" class="verplicht" >Land:</label>
    <select id='land' name='land' title='in welk land woont u?' required >
        <option value=1>België</option>
        <option value=2>Nederland</option>
        <option value=3>Swaziland</option>
    </select>
    <span class="fieldval" id="val_land"></span>
</div>
...
```

- een `select` element maakt een keuzelijst (*dropdown list*). Het heeft ook `name`, `id`, `title` en `required` attributen
- verschillende `option` elementen zijn de *children* van de `select`. Ze hebben
 - een `value` attribuut indien je wil dat de waarde die doorgegeven wordt verschilt van de tekst in de `option`. Mag de tekst zelf doorgegeven worden, dan is `value` niet nodig

Dit is een eenvoudige `select`. Ze kunnen complexer worden. Lees de theorie erop na.

2.4.5 Nog meer types invulvelden

We doen verder:

```
...
<div>
    <label for="tel">Telefoon:</label>
    <input type="tel" id="tel" name="tel" placeholder="uw telefoonnummer" title="uw
vaste of mobiele telefoon" >
    <span class="fieldval" id="val_tel"></span>
</div>
<div>
    <label for="email" class="verplicht" >Email:</label>
    <input type="email" id="email" name="email" placeholder="uw emailadres" title="mijn
emailadres" required >
    <span class="fieldval" id="val_email"></span>
</div>
<div>
    <label for="geboren">Geboortedatum:</label>
    <input type="date" id="geboren" name="geboren" title="uw geboortedatum in het formaat
1956-12-31" required >
```

```
<span class="fieldval" id="val_geboren"></span>
</div>
...
...
```

- het `input type=tel` is weer een nieuw html5 `type` dat specifiek dient om telefoonnummers te accepteren. Indien een browser deze nieuwe `type` niet kent, zullen ze behandeld worden als gewone `text` type
- het `input type=email` is opnieuw een nieuw html5 `type` voor emailadressen
- het `input type=date` is nogmaals een nieuw html5 `type` voor datums.
 - sommige browsers tonen automatisch een *kalenderwidget* als je erin klikt
 - het `placeholder` attribuut voor dit type valideert niet. Chrome toont het ook niet, daarom laten we het hier achterwege

2.4.6 Radiobuttons, textarea en checkboxes

We gaan verder:

```
...
<div>
  <label>Geslacht:</label>
  <input type="radio" id="man" value="m" name="sexe" />
  <label class="labelRadio" for="man" > Man </label>
  <input type="radio" id="vrouw" value="v" name="sexe" title="uw geslacht" />
  <label class="labelRadio" for="vrouw" >Vrouw </label>
  <span class="fieldval" id="val_sexe"></span>
</div>
<div>
  <label for="vragen" >Hebt u nog vragen?</label>
  <textarea id="vragen" name="vragen" placeholder="vragen?" title="Heeft u nog vragen?"></textarea>
  <span class="fieldval" id="val_vragen"></span>
</div>
<div>
  <label class="labelCheckbox">
    <input type="checkbox" id="actiesmail" value name="kortingen" />
    ik wens per email op de hoogte gehouden te worden van kortingen en acties van De Gazet</label>
</div>
<div>
  <label class="labelCheckbox">
    <input type="checkbox" id="actiespost" value name="actiespost" />
    ik wens per post op de hoogte gehouden te worden van kortingen en acties van De Gazet</label>
</div>
<div>
  Lees alstublieft onze <a href="">Privacy Richtlijnen</a> om te weten te komen hoe wij uw gegevens gebruiken. <br>
  U moet ook onze <a href="#">Algemene VerkoopsVoorwaarden</a> lezen en ermee akkoord gaan. U moet deze checkbox <input type="checkbox" id="akkoord" value name="akkoord" /> aanvinken om toe te stemmen met de algemene verkoopsvoorwaarden.
</div>
```

...

- het formulieveld "sexe" bestaat uit twee *radiobuttons* (keuzerondjes).
 - een radiobutton is een `input type=radio`
 - Merk op dat het `name` attribuut voor beide dezelfde waarde "sexe" heeft! dit is essentieel om ervoor te zorgen dat het hier één veld betreft. Is de `name` niet gelijk dan werken de keuzerondjes niet als één geheel samen
 - de `id` en de `value` van de velden zijn verschillend (en moeten verschillend zijn...)
 - ze worden begeleid door een eigen `label class=labelRadio`
- het volgende veld is een `textarea`: dit is een groot involveld voor tekst. je gebruikt het als er meerdere zinnen (of zelf pagina's) verwacht worden.
 - Een `textarea` heeft – in tegenstelling met een `input` – een *open-* en *closetag*. Als je wil standaardtekst voorzien schrijf je die tussen deze tags
 - de grootte van een `textarea` kunnen we bepalen ofwel met een `rows` en `cols` attribuut ofwel via CSS
- de volgende drie velden zijn *checkboxes* (aanvinkvakjes):
 - `input type=checkbox`
 - het zijn velden die hun `name=value` zullen doorgeven als ze aangevinkt zijn en helemaal niets als ze dat niet zijn

2.4.7 Knoppen

Nu nog de `submit` en `reset` knoppen:

```
<div>
    <input class="blauw" type="submit" value="Bevestig"/>
    <input class="blauw" type="reset" value="Herbegin" />
</div>
</form>
```

- een `input type=submit` is een submitknop waaraan een *eventhandler* gekoppeld is die het `submit` event van het `form` zal uitvoeren. Hiermee worden de gegevens doorgestuurd naar de `action target`. Dat gebeurt via de methode `method` die je specificeerde.
- een `input type=reset` is een resetknop waarmee alle ingevulde gegevens in het form gewist worden en eventueel weer vervangen door standaardwaarden. Dit kan een onaangename verrassing zijn voor de gebruiker –zeker voor grote formulieren, daarom laten sommige auteurs resetknoppen achterwege
- de `class=blauw` komt straks van pas bij de opmaak

We zijn klaar, nu zie je dit (in Chrome)

Uw gegevens

Velden met een ♥ zijn essentieel om uw abonnement te kunnen registreren.

Voornaam:

Familienaam:

Straat:

Postnummer: hier moet u een postnummer invullen

Gemeente:

Land:

Telefoon:

Email:

Geboortedatum:

Geslacht: Man Vrouw

Hebt u nog vragen?

ik wens per email op de hoogte gehouden te worden van kortingen en acties van De Gazet

ik wens per post op de hoogte gehouden te worden van kortingen en acties van De Gazet

Lees alstublieft onze [Privacy Richtlijnen](#) om te weten te komen hoe wij uw gegevens gebruiken.

U moet ook onze [Algemene VerkoopsVoorwaarden](#) lezen en ermee akkoord gaan. U moet deze checkbox aanvinken

Alle velden zijn aanwezig, probeer ze uit. Bemerk de placeholder teksten, het nummer veld (in Chrome met tellertje), de validatitekst, het datumveld.

De opmaak lijkt natuurlijk nergens op. We vragen je nog even geduld te hebben: de html structuur die we hier gebruikten – met container **div's**, is ideaal voor een mooie lay-out en wordt veel gebruikt.

Taken:

maak nu volgende taken

- "het vdab opleidingscentrum"

2.5 basis CSS

Doel

De basis CSS leren kennen

Theorie

Lees de volgende theorie topics eerst na:

- CSS intro
- CSS voor tekstopmaak
- CSS meer selectoren: **class** en **id**
- CSS inheritance en de cascade
- CSS het box-model

Startbestanden

degazet/index.html, degazet/gazet.css

Opdracht

We werken verder aan de opmaak van De Gazet met CSS.

Open *gazet.css* waar we reeds het volgende hadden staan:

```
@charset "utf-8";
/* CSS Document voor deGazet */

body {
    font-family:Georgia, "Times New Roman", Times, serif;
    color:#333;
}
h1, h2, h3, h4, h5, h6 {
    color:#005689;
}
```

2.5.1 Hoe een stylesheet indelen?

Een CSS file mag geen aaneengeregen lijst van selectors zijn. Schrijf altijd je CSS alsof je in een **team** werkt, zelfs al is dat niet zo.

- voorzie voldoende commentaar
- breng er orde in aan

Het is voor een collega heel wat plezieriger een CSS bestand (geschreven door een ander) te lezen dat duidelijk en geordend is zodat hij snel iets terugvindt en zich niet steeds moet afvragen waarvoor een *style rule* of property dient.

In een eerste fase kan je een bepaalde **volgorde** aanbrengen in je declaraties. Neem nu dit voorbeeld:

```
nav ul li a {  
  
    float: left;  
    display: block;  
    width: 3.8em;  
    line-height: 1.3em;  
    padding: .8em 1em .8em 1em;  
    margin: 0;  
    text-decoration: none;  
    border: none;  
    background: url(..../images/nav-bg.gif) no-repeat bottom left;  
    font-size: 1em;  
    font-weight: bold;  
    text-transform: uppercase;  
    color: #e0e0d5;  
}
```

Het is misschien volledig, maar het is ook chaotisch ...

Een eerste manier van werken:



Groepeer de **CSS properties per categorie** binnen elke selector:

```
nav ul li a {  
  
    /*typografie*/  
    font-size: 1em;  
    font-weight: bold;  
    text-decoration: none;  
    line-height: 1.3em;  
    text-transform: uppercase  
  
    /*kleur en backgrounds*/  
    color: #e0e0d5;  
    background: url(..../images/nav-bg.gif) no-repeat bottom left;  
  
    /*box*/  
    float: left;  
    display: block;  
    width: 3.8em;  
    padding: .8em 1em .8em 1em;  
    margin: 0;  
    border: none;  
}
```

Waarbij we een **consistente volgorde** aanhouden: eerst *typografie*, dan *kleuren*, dan properties van het *box-model*, etc...

We gebruiken dan steeds volgorde in alle selectoren!

Een tweede manier van werken:



Groepeer de **selectoren volgens hun doel** binnen het stylesheet:

Eerst algemene CSS die geldt voor de *veelgebruikte elementen*, daarna de algemene *structuur-elementen*, daarna de verdere *onderverdelingen*, dan de *speciale elementen*: het *hoofdmenu*, het *zij-menu*, het *logo*, een formulier, etc.

```
/* algemeen */

body { ... }
h1,h2,h3,h4 { ... }
a { ... }

/* structuur */

header { ... }
footer{ ... }
div#wrapper { ... }

/* content */

article { ... }
article h2{ ... }

/* hoofdmenu */

ul#hoofdmenu { ... }
ul#hoofdmenu li { ... }

...
```

Dit is al veel duidelijker. Sommige auteurs plaatsen zelf een inhoudstafel aan het begin! Misschien kan je een selector niet onmiddellijk onderbrengen, maar dat kan altijd bij de afwerking gebeuren. Voor de meeste ontwikkelaars is deze manier van werken voldoende.

Een **nog meer doorgedreven** benadering is de selectors volledig op te splitsen per property category: alle *typografie* properties samen, alle kleuren samen, alle lay-out properties samen. Dit heeft voor gevolg dat een selector (zoals *p*) meerdere keren voorkomt in het stylesheet, dus wordt het stylesheet (veel) langer.

Als we deze methode volgen zijn we uiteindelijk in staat de categorieën op te splitsen in aparte CSS bestanden: *kleuren.css*, *typo.css*, *grid.css*.

Waarom zouden we dit doen? om volledig **modulair** te werken zoals het gebeurt in CSS frameworks . Deze werken met een CSS preprocessor waarbij het uiteindelijke stylesheet samengesteld wordt uit verschillende modules.

Wij gaan zeker nog niet zover, maar we proberen je hier te laten wennen aan het idee.
In ons project zullen we deze laatste methode proberen toe te passen, daarom doen we wat aanpassingen:

```
@charset "utf-8";
/* CSS Document voor deGazet */
/* TYPOGRAFIE */
body {
    font-family:Georgia, "Times New Roman", Times, serif;
    font-size:100%
}
h1, h2, h3, h4, h5, h6 {
    font-weight:normal;
}
a, a:visited {
    text-decoration:none;
}
a:hover, a:active {
    text-decoration:underline;
}
/* KLEURENSCHEMAS */
body {
    background:white;
    color:#333;
}
#topbanner, footer {
    background:#EDEDED;
}
.nieuws h1,.nieuws h2,.nieuws h3,.nieuws h4,.nieuws h5,.nieuws h6{
    color:#005689;
}
.nieuws a,.nieuws a:visited ,.nieuws a:hover,.nieuws a:active {
    color:#005689;
}
```

- we maken 2 rubrieken: *typografie* en *kleurenschema's*
 - *typografie*:
 - de **font-size:100%** is in principe onnodig maar lost een probleem op dat voorkomt in IE7 bij het in-en uitzoomen op het scherm.
Niets zeggen of **100%** betekent dat de standaardlettergrootte 16px is (in bijna alle browsers).
Dit is dan het vertrekpunt voor alle andere lettergroottes die we vanaf nu in **em** zullen specificeren, dus nu: **1em = 16px**
 - we stellen de dikte van de koppen in op **normal** (standaard **bold**)
 - hyperlinks (**a** elementen) en bezochte hyperlinks worden via **text-decoration** niet onderlijnd. Actieve - en hyperlinks waar je met de muis overgaat, worden onderlijnd
- :visited, :active, :hover** zijn **pseudo-classes**: ze selecteren een element in een *bepaalde staat*

- kleurenschema's:
 - de achtergrondkleur van de `body` wordt via `background` ingesteld op "white": dit is een **named color**. Je mag dus bepaalde woorden gebruiken voor kleuren. Je vindt het overzicht van alle kleurennamen op de kleurenmodule van de CSS standaard:
<http://www.w3.org/TR/css3-color/>
 - het instellen van de achtergrondkleur op wit lijkt onnodig maar bedenk dat een gebruiker de achtergrondkleur van zijn browservenster via de instellingen een andere kleur als standaard kan ingesteld hebben
 - de basiskleur voor alle tekst wordt ingesteld via de `body` selector
 - de `div#topbanner` en het `footer` element krijgen een lichtgrijze achtergrondkleur via `background`
 - je bemerkt een `class` in de `body` tag: "nieuws". Dit designert deze pagina onder de noemer "nieuwpagina". Alle "nieuwpaginas" hebben hun dit kleurenschema. Straks maken we andere kleurenschema's, bv. de "culinaire" pagina's.
 - De selector `.nieuws h1` wordt enkel toegepast op de `h1` kop als die een child van een een `.nieuws` element is: dit noemen we een **descendant combinator**: de `h1` is een *child* maar mag ook verschillende niveaus dieper zitten.

In gewone taal kan je stellen dat de regel van toepassing is voor een `h1` binnen de *context* van `.nieuws`

- in dezelfde context bepalen we ook de kleur van de hyperlinks en hun pseudo-classes : `. nieuws a, ...`

Bekijk af en toe eens je `index.html` pagina om de veranderingen waar te nemen.

2.5.2 Een fixed lay-out

We brengen een belangrijke lay-out feature aan:

```
***** STRUCTUUR EN LAY-OUT *****

/* deze oef gaat door een aantal stadia:
1. FIXED LAY-OUT: 980px container, basis 5 kol
*/
.wikkels {
    width: 980px;
    margin: 0 auto;
}
```

- we stellen een **width** in van 980px op de containers met de **class wikkel** (controleer: er zijn er meerdere): dat zorgt ervoor dat zowat alle inhoud een breedte krijgt die beperkt is tot 980 px.
- **margin: 0 auto** (= verkorte schrijwijze) stelt de **top-** en **bottom-margin** in op nul, en de **left-** en **right-margin** op **auto**. Daarmee centreer je de content horizontaal



Door de breedte vast in te stellen in pixels maken we een **FIXED LAY-OUT**. Hoe je het scherm ook verkleint of vergroot, de centrale kolom van **980px** blijft even breed.

Bekijk zeker het resultaat. Bemerk ook dat de **#topbanner** en de **footer** wel het volledige scherm overspannen: ze bevinden zich niet in een **.wikkels**.

2.5.3 Werken in relatieve eenheden

De fixed lay-out werd uitgezet in pixels, maar lettertypes, marges, borders en padding kunnen ook uitgedrukt worden in relatieve eenheden.

We geven de voorkeur aan het werken in relatieve eenheden: **em** of **%**. We hebben net **1em** gelijkgesteld aan **16px**, dus als we een bepaald aantal pixels willen, zullen we een omrekening moeten doen, bijvoorbeeld, hoeveel is **6px** in **em** ? daarvoor doen we

$$6 / 16 = 0.375\text{em}$$

Sommige auteurs leggen daarom een kleine conversietabel aan in hun CSS:

```
/* basis 1em = 16px

    4px      = 0.25em
    6        = 0.375em
    8        = 0.5em
    10       = 0.625em
    12       = 0.75em
    14       = 0.875em
    18       = 1.125em
    20       = 1.25em
    22       = 1.375em
    24       = 1.5em
    32       = 2em

    100      = 6.25em;

*/
```

2.5.4 Normaliseren

Wat betreft ruimte in en rond containers gedragen de verschillende browsers zich nogal verschillend... hun standaardinstellingen zijn niet dezelfde. Om die reden voeren we vooraan ons stylesheet eerst een category "Normalize" in die de belangrijkste instellingen voor alle browsers gelijkstelt.

Er bestaan hiervoor speciale stylesheets, die gebruiken we verder, eerst proberen we het zelf:

```
@charset "utf-8";
/* CSS Document voor deGazet */

/* NORMALIZE */

html, body, div, section, article, aside, nav {
    margin:0;
    padding:0;
}
img {
    border:none;
    float:left;
    margin: 0 1.25em 0.5em 0;
}
table{
    border-collapse:collapse;
    border-spacing: 0;
    empty-cells:hide;
}
tr,td {
    margin:0;
    padding:0;
}
```

- de **margin** en **padding** van alle belangrijke containers wordt op **0** gezet
- voor het **img** element wordt de border afgezet, het wordt links ge-*float* (uitleg verder) en er wordt een **right**- en **bottom-margin** op geplaatst.
Je bemerkt dat de images zich links gaan plaatsen van de tekst.
- voor het **table**, **tr** en **td** element stellen we ook zaken in, maar die bespreken we als we ze nodig hebben

Dit zijn slechts *standaard-instellingen*, ze kunnen op elk moment overschreven worden door andere style-rules.

En nu passen we **margin** en **padding** van de belangrijkste structuurelementen aan:

```
header {
    padding-top: 0.375em;          /* 6px */
}
section, aside {
    margin-top: 0.25em;
```

```
margin-bottom: 0.25em;
}
article {
  margin: 0 20px 0 0;           /* gutter */
  padding: 0.375em 0 1.25em;   /* 6px 20px */
}
article p {
  margin: 0;
  padding: 0 0 0.75em 0;       /* 12px */
}
article h2 {
  margin: 0.375em 0;          /* 6px */
}
```

- Hier worden zowel **margin** als **padding** ingesteld. Je kan je afvragen waarom de ene keer margin, de andere keer padding? Lees "*margin of padding gebruiken?*" in de theorie.
- een **article** element wordt steeds een **right-margin** van 20px toegekend: dit doen we omdat we straks een **grid** (een lay-out van kolommen) gaan gebruiken en deze 20px een vaste "goot" (*gutter*) vormt rechts van elk artikel. Alle tekst zal in artikel elementen zitten
- een **p** in een **article** krijgt een **bottom-padding**
- een **h2** in een **article** krijgt een bepaalde **top-** en **bottom-margin**

2.5.5 Een grid systeem

In de HTML code bemerk je in sommige **section**, **aside** en **div** elementen classes zoals **span1**, **span2**, ..., **span5**. Dit zijn classes die we al klaargezet hebben om een eenvoudig raster (*grid*) toe te passen.

Voeg toe:

```
*****GRID*****
.span1 {
  width:196px;
  float:left;
}
.span2 {
  width:392px;
  float:left;
}
.span3 {
  width:588px;
  float:left;
}
.span4 {
  width:784px;
```

```
float:left;
}

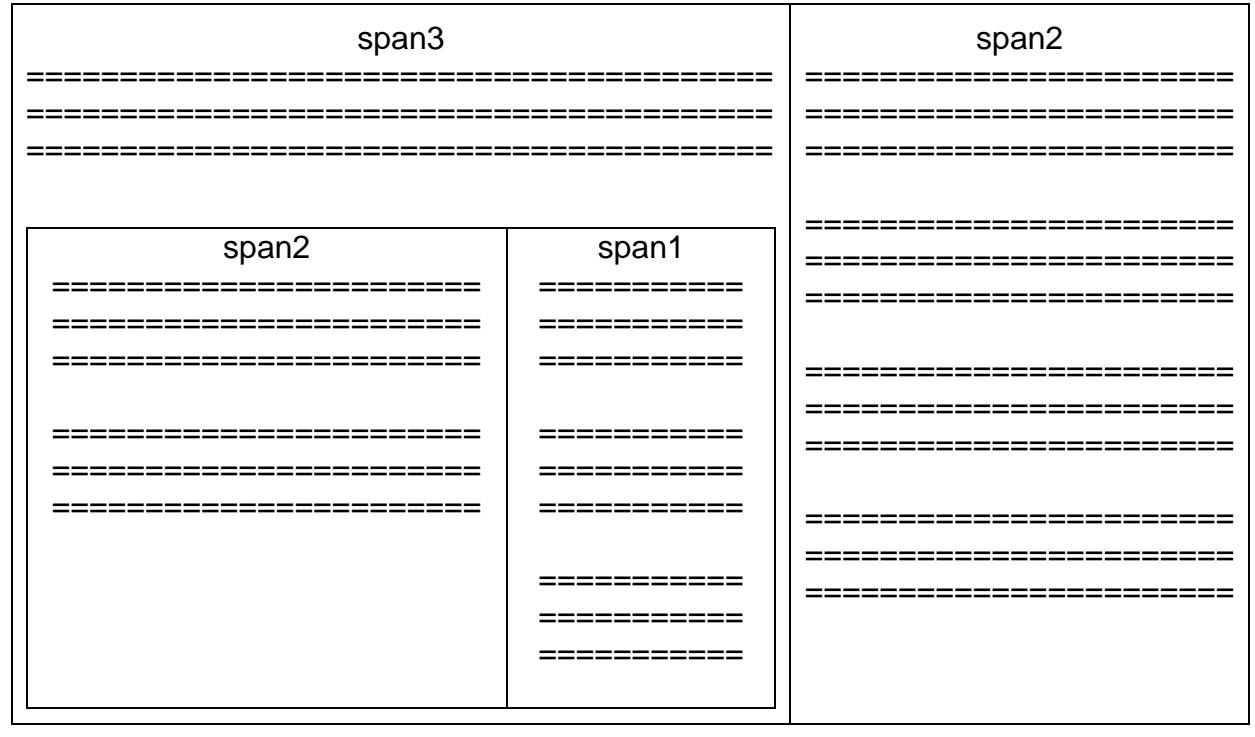
.span5 {
  width:980px;
  float:left;
}
.span5>article {
  margin-right:0; /* geen gutter in span5*/
}
.span5>h2 {
  margin-right:0; /* geen gutter in span5*/
}
```

Het werkt zo:

- Deze grid is gemaakt voor een fixed lay-out van 960px.
Als we later willen overschakelen op een andere breedte of een fluid lay-out zullen we dit moeten aanpassen
- De totale breedte van 980px wordt verdeeld in 5: een **span1** is dus 196px breed, een **span2** tweemaal zoveel,... en een **span5** is de volle breedte.
Hiermee maken we een systeem van kolommen, die exact op elkaar aansluiten en waarvan de horizontale som de totale breedte van 980px niet mag overschrijden.
- Door elke kolom een **float:left** te geven, rangschikken ze zich naast elkaar en niet onder elkaar (tenzij de som méér is dan 980px!)
- om alle mogelijke soorten inhoud toe te laten (tekst, wand-tot-wand figuren, vol gekleurde blokken, kalenders, etc..) hebben de kolommen geen **margins** en geen **padding**.
Elke **tussenruimte (gutter)** die je wil creëren moet dus komen van de inhoud van de kolom. Daarom hebben we eerder een **margin-right** van 20px voorzien voor alle **article** elementen. In onze *fixed lay-out* houden we vast aan een exacte tussenruimte
- daar maken we uitzonderingen op: als een 'inhoud' zoals een **article** element in een **span5** zit, moet het geen **right-margin** hebben, net zo min als een **h2** die rechtstreeks in een **span5** zit
- de selector **.span5>article** is een voorbeeld van een **child combinator**: het **article** element moet een **onmiddellijk child** zijn van de **.span5**, geen **kleinkind**.

Het schema van ons raster ziet er als volgt uit:

span5



Het toepassen van de grid geeft het volgende resultaat:

De Gazet

- Home
- Binnenland
- Buitenland
- Sport
- Cultuur
- Economie
- Wetenschap
- Blog
- Weer

Heet van de naald! Prins Laurent zwanger....

Nieuws

Vanaf Oktober meer files op wegen



Lollipop candy sugar
plum topping bear claw
faworki. Cotton candy
pastry candy sweet.
Apple pie cotton candy
lemon drops icing
pudding sweet jelly
beans marzipan.

[Lees verder](#)

Meer nieuws

Gemeenteraad Genk wil kerncentrale Borssele dicht voor controle



Pudding macaroon
topping topping apple
pie. Dragée sesame
snaps marshmallow
Caramels bonbon

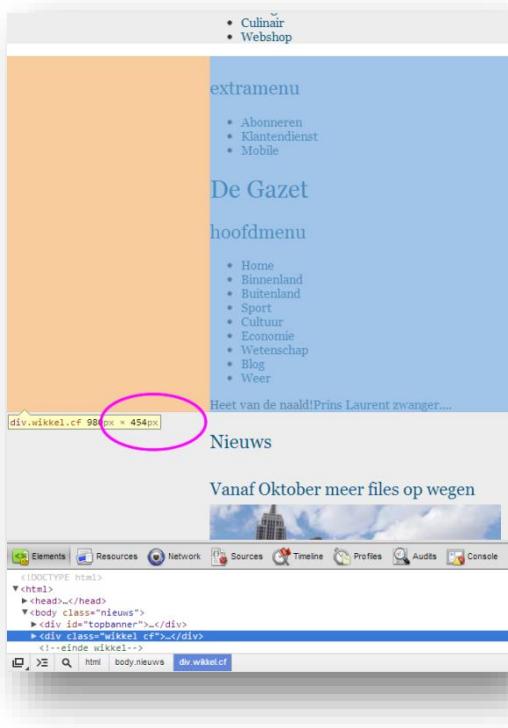
Ander nieuws

Woestijnvis zal weer programma's maken voor VRT

Je kunt hier een voorbeeld vinden van een website die gebruik maakt van de ClearFix techniek.

- Meest recent
 1. di 10:52 Rattenplaag teistert Oostende
 2. di 10:46 Vanaf Oktober meer files op wegen
 3. di 11:34 Strenge winter 2012-2013 verwacht
 4. di 10:48 Gemeenteraad Genk wil kerncentrale Borssele dicht voor controle
 5. di 11:22 Drie gewonden bij botsing kusttram
- Meest getipt
 1. di 10:52 Rattenplaag teistert Oostende
 2. di 10:46 Vanaf Oktober meer files op wegen
 3. di 11:34 Strenge winter 2012-2013 verwacht
 4. di 10:48 Gemeenteraad Genk wil kerncentrale Borssele dicht voor controle
 5. di 11:22 Drie gewonden bij botsing kusttram
- Meest gelezen
 1. di 11:34 Strenge winter 2012-2013 verwacht
 2. di 10:52 Rattenplaag teistert Oostende
 3. di 10:46 Vanaf Oktober meer files op wegen
 4. di 11:22 Drie gewonden bij botsing kusttram
 5. di 10:48 Gemeenteraad Genk wil kerncentrale Borssele dicht voor controle

2.5.6 ClearFix



The screenshot shows a browser window with developer tools open. The DOM tree on the left indicates a 'wikkel' element with a height of 454px. A specific element in the main content area is circled in pink, and its dimensions are shown as 98px x 454px. The page content includes a sidebar with a menu, a main content area with news articles, and a footer.

```

<!DOCTYPE html>
<html>
  <head></head>
  <body>
    <div id="topbanner"></div>
    <div class="wikkel cf">
      <!-- wikkel -->
      <div>
        <!-- header -->
        <div>
          <!-- sidebar -->
          <div>
            <!-- content -->
            <div>
              <!-- footer -->
            </div>
          </div>
        </div>
      </div>
    </div>
  </body>
</html>

```

Je hebt wellicht bemerkt dat de witte zone zich slechts uitstrekken tot onderaan het hoofdmenu. Waarom? We hebben de **body** toch een witte achtergrond gegeven, terwijl enkel de **footer** en de **#topbanner** een grijze achtergrond gekregen hebben? De reden is een *collapsing parent*: de **.wikkel**.

Gebruik even je *developer's tool* en selecteer het element **.wikkel**. Dan merk je dat zijn hoogte slechts een 450px is, en hij dus niet helemaal rond alle inhoud strekt, zoals wel zou moeten volgens de DOM tree!

Dit komt door het gebruik van *floats*: we leggen het haarlijn uit in het volgende project "Werken met Floats".

Nu passen we enkel een oplossing toe zodat we verder kunnen werken.

In de HTML van de pagina zie je de class **clearfix** staan in **.wikkel**.

Deze **class** voegen we nu toe helemaal onderaan je stylesheet:

```
/*
clearfix hack
http://nicolasgallagher.com/micro-clearfix-hack/
*/
.clearFix:before, .clearFix:after {
    content: " ";
    display: table;
}
.clearFix:after {
    clear: both;
}
/* enkel voor IE6/7 */
.clearFix {
    *zoom: 1;
}
```

Let op dat je geen foutjes maakt!

Bekijk nu opnieuw de pagina: probleem opgelost, het element strekt nu helemaal tot aan de **footer**.

2.5.7 Koppen verbergen of opmaken

Ondanks het feit dat onze pagina er uitziet als een lappendeken bemerkt je in de HTML toch een duidelijke structuur. We hebben verschillende **section** elementen die elk

- een **id** hebben: "nieuws", "meernieuws", "andernieuws".
Dit is voornamelijk voor *content/scripting* doeleinden: je kan je voorstellen dat een artikel thuis hoort in een bepaalde rubriek.
- een kop **h2** hebben:
die kan dienen in de *outline* van het document

Sommige van die **koppen** willen we duidelijker opmaken, andere willen we verbergen. Wat we niet doen is ze wissen: ze zijn nodig voor de outline.

Daarvoor gebruiken we twee **classes** in een **section** of **article** element:

- **.metTitel**: duidelijk met grijze achtergrond en rode bovenborder
- **.zonderTitel**: verbergt de titel

Je bemerkt deze **classes** in de HTML (en niet enkel in **section**: ook **nav** en **article** elementen).

We schrijven de CSS ervoor:

```
***** ARTICLE en SECTION OPMAAK *****

/* duidelijke grijze titel, rode bovenrand */
.metTitel { padding-top:0; }
.metTitel>h2 {
    background: #EDEDED;
    border-top: 3px solid #D61D00;
    margin: 0 20px 0.8em 0;
    padding: 0.8em 0 0.8em 0.4em;
}

/* verborgen titel, enkel voor outline */
.zonderTitel>h2 { display:none; }
```

Bespreking:

- opnieuw gebruiken we de *child combinator* `.metTitel>h2` zodat andere koppen in deze `section` niet beïnvloed worden
- `display:none` verbergt een element, zo dat het ook geen ruimte inneemt

Het resultaat is zichtbaar: de titels van de `nav` elementen zijn verdwenen en "Ander nieuws" is duidelijk zichtbaar.

We willen nog meer differentiëren:

- de kop van het hoofdartikel mag wat groter
- de `article` koppen van minder belangrijke sections mogen wat kleiner

Voeg verder toe

```
article.toparticle>h2 {
    font-size: 1.5em; /*24px*/
}
#andernieuws article>h2 {
    font-size: 1.125em; /*18px*/
    margin: 0.125em 0; /*4px*/
}
```

2.5.8 De menu's

We voorzien 3 verschillende menu's op deze website: het "*topmenu*", "*extramenu*", en het "*hoofdmenu*". Veel? helemaal niet, sommige sites hebben er meer dan tien.

Het topmenu

Dit menu geeft toegang tot delen van de website die niet onmiddellijk te maken hebben met nieuws, zoals het "*Dating*" en "*Culinair*" gedeelte.

De html structuur:

```
<div id="topbanner">
```

```
<div class="wikkel">
  <nav id="topmenu" class="zonderTitel">
    <h2>topmenu</h2>
    <ul>
      <li><a href="dating.html">Dating</a></li>
      <li><a href="culinair.html">Culinair</a></li>
      <li><a href="webshop.html">Webshop</a></li>
    </ul>
  </nav>
</div>
</div>
```

Je bemerkt dat het **nav** element de **id** "topmenu" heeft. Het bevat naast een **h2** een **ul** die de links bevat. Noteer dat er ook nog andere dingen zouden in kunnen aanwezig zijn, zoals bv. een loginformulier.

Het is belangrijk op te merken dat quasi alle menustructuren gebaseerd zijn op een *unordered list*: een **ul** element. De reden hiervoor is dat *lists* gemakkelijk aan te vullen en te wijzigen zijn. Met CSS kan je van een **ul** **eender welke opmaakstructuur** maken.

We beginnen met het belangrijkste: het omvormen van de bolletjeslijst naar een horizontaal georiënteerd menu:

```
***** TOPMENU ****/
#topmenu {text-align:right;}
#topmenu ul {
  list-style-type:none;
  margin:0;
  padding:0;
}
#topmenu ul li {display: inline;}
#topmenu ul li a {display: inline-block;}
```

Bespreking:

- we lijnen de inhoud van de **nav#topmenu** uit met **text-align:right**. Daarmee komen de hyperlinks rechts te staan, binnen de grenzen van **wikkel**
- met **list-style-type:none** verwijderen we elke vorm van *bolletje*: geen icoontjes meer
- we zetten alle **padding** en **margin** op nul zodat de **ul** perfect aansluit op de **nav**
- met **display:inline** op de **li** maak je het menu horizontaal. De **li** elementen zijn geen *block* elementen meer: ze gaan naast elkaar staan
- met **display:inline-block** op de **a** elementen maak je er *inline* blokjes van: het worden *block* elementen maar ze gaan ook naast elkaar staan. Daar is een voordeel aan verbonden: je hoeft nu niet exact op de tekst van de hyperlink te wijzen, een beetje ernaast is ook goed.

Dit is het belangrijkste. Nu merk je reeds een horizontaal menu. De rest is verdere opmaak.

We voegen toe:

```
...
#topmenu ul li a {
    display: inline-block;
    width: 7em;
    padding: 0.2em;
    font-size: 0.8em;
    font-style: italic;
    color: #499BAB;
    text-decoration:none;
}
#topmenu ul li a:hover, #topmenu ul li a:active {
    color: #005689;
    text-decoration:none;
}
```

Bespreking van het belangrijkste:

- de `width` zorgt ervoor dat alle *inline* blokjes even breed zijn
- de `text-decoration:none` verwijderd het onderlijnen van de hyperlinks
- de `:hover` en `:active` *pseudo-classes* maken de links op als je erover glijdt met de muis of als je klikt.

Waarom is het nodig de `text-decoration:none` hier te herhalen?

Omdat de *specificiteit* van `a`, `a:link`, `a:hover` en `a:active` precies gelijk is. Geen enkele heeft een grotere specificiteit, daarom is volgorde en herhalen belangrijk.

Het extramenu

Is erg gelijkend op het vorige met uitzondering van de uitlijning en een rand:

```
***** EXTRAMENU *****

#extramenu {
    border-bottom: 1px solid #bebebe;
    height: 1.375em;
}
#extramenu ul {
    list-style-type:none;
    margin: 0;
    padding: 0;
}
#extramenu ul li {
    display: inline;
}
#extramenu ul li a {
    display: inline-block;
    font-size: 0.9em;
    padding: 0.2em;
```

```
color:         #cccccc;
text-decoration:none;
}
#extramenu ul li a:hover, #topmenu ul li a:active {
color:         #505050;
text-decoration:none;
}
```

Het hoofdmenu

bevindt zich net onder de **h1** en bevat natuurlijk de belangrijke onderdelen van de nieuwssite. We maken van de gelenheid gebruik om eens op een andere manier te werk te gaan, met zeer gelijkende resultaten.

```
#hoofdmenu {
background:      #eddeded;
}
#hoofdmenu ul {
list-style-type: none;
margin:          0;
padding:         0;
float:           left;
}
#hoofdmenu ul li {
float:           left;
}
#hoofdmenu ul li a {
display:         block;
padding:         0.375em 0.5em; /*6px 8px*/
border-right:    1px solid #b1b1b1;
text-decoration: none;
letter-spacing:   0.1em;
color:           #005689;
}
```

Bespreking:

- het **ul** element wordt deze keer links *ge-float* met **float:left**. Dat betekent dat het in de linkerhoek van zijn container (de **nav**) gaat staan
- ook hier worden de blokjes afgezet met **list-style-type:none**
- de **li** elementen blijven blokjes maar worden ook links ge-float binnen hun **ul**
- de **a** elementen worden volwaardige *blocks* met **display: block**
- ze krijgen ook een rechterraand en wat tekstopmaak
- letter-spacing** is eens een andere manier om de aandacht te trekken: de letters worden wat wijder uiteen gezet

Als je dit uitprobeert kan er zich een probleem voordoen: de volgende lijn "*Heet van de naald...*" komt ernaast te staan. Dit wordt opnieuw veroorzaakt door het *floaten* van de

ul. We kunnen dit ook oplossen door er de `class .clearfix` toe te passen op het `nav` element. Pas de HTML code aan:

```
<nav id="hoofdmenu" class="zonderTitel clearfix">
```

We voegen nog meer opmaak toe aan het menu:

```
#hoofdmenu ul li a:hover,  
#hoofdmenu ul li a:active,  
#hoofdmenu ul li a.actief {  
    text-decoration:none;  
    background:#D61D00;  
    color:white;  
}
```

Bespreking:

- ook dit is een *hover* effect die we plaatsen, maar deze opmaak geldt ook voor een vast element: het `a` element met de `class .actief`
We gebruiken die om de actieve pagina aan te duiden in het menu: bekijk de HTML code

Nu maken we nog snel de *teaserline* "Heet van de naald" op: dit element is bedoeld voor een script die er het allernieuwste nieuwstje in plaatst (Breaking News!)

Hier bevat het een vaste inhoud.

```
/*Heet van de naald*/  
#heet { padding: 6px 0; }  
#heetvandenaald {  
    font-weight: bold;  
    color: #D61D00;  
    padding-right: 1em;  
}
```

Daar valt niet veel over te zeggen.

2.5.9 Image replacement

CSS *Image replacement* is een nuttige techniek die ons toelaat een mooi beeld te gebruiken als kop, zonder die kop leeg te laten. Een `h1` zonder tekst is semantisch gezien niet goed, de kop wordt gebruikt voor SEO doeleinden, komt in de outline, etc...

Zo vervangen we de tekst van de `h1#logo` door een mooier beeld:

```
/* Logo image replacement  
#logo = h1  
*/  
  
#logo a {  
    display: block;
```

```
width:      346px;
height:     77px;
margin:     14px 0 20px 0;
padding:    0;
background: transparent url(..../img/degazet.png) no-repeat;
text-indent: -99999px;
}
```

Bespreking:

- we vormen de **a** in de **h1** om naar een block element met dezelfde dimensies als ons beeldje
- we zetten het beeld al *backgroundimage*
- en we verplaatsen de tekst **99999px** naar links: geen kans dat je het nog ziet, maar de tekst is niet verwijderd

Deze eenvoudige methode wordt regelmatig gebruikt maar heeft één nadeel: als de *images* door de gebruiker afgezet worden, ziet hij helemaal niets...

Omdat ons logo uit tekst bestaat hebben we ook een alternatief: *Webfonts*.

Een veel beter alternatief is aan de horizon verschenen: CSS3 *content-replacement* is de enige echte image-replacement, maar dat kunnen we helaas nog niet gebruiken omdat nog te weinig browsers het ondersteunen (Chrome, Opera).

Het resultaat:



Borders

Nu voegen we enkele pure opmaak classes toe die volledig generiek zijn, d.w.z. dat je ze eerder waar kunt gebruiken.

Het zijn rode randen die we boven- of onderaan een blok kunnen toepassen

```
***** BORDERS ****/
/* default kleur rood */
.bt8   {border-top:8px solid #D61D00;}
.bt3   {border-top:3px solid #D61D00;}
.bb8   {border-bottom:8px solid #D61D00;}
.bb3   {border-bottom:3px solid #D61D00;}
```

We gebruiken hier pixels omdat we nooit zeker zijn in welke context we ze zullen gebruiken.

2.6 Werken met Floats

Doel

De **float** property is één van de belangrijkste CSS properties voor lay-out. We leren ermee werken en bespreken veel voorkomende de problemen en oplossingen, n.b. de *clearfix*.

Theorie

Lees de volgende theorie topics na:

- CSS box-model
- Het visual formatting model

Startbestanden

floats.html, *floats.css*

Opdracht

Het basisbestand heeft deze keer niets te maken met *De GAZET*. het project dient enkel om je inzicht te geven in de **float** property.

Bekijk het HTML bestand en het bijhorende stylesheet. Pas eventueel het pad van de beelden aan zodat ze zichtbaar zijn.

Het is een XHTML bestand waarvan de structuur bepaald wordt door een aantal genestte **div** elementen. Een buitenste **div#wrapper** heeft een relatieve breedte van 60%. Dat betekent dat de breedte van de inhoud aanpast aan een kleiner/groter venster. Test dit uit.

Merk ook op dat met de eenvoudige opmaak die er nu is de lay-out gevormd wordt door de volgorde van de structuurelement in de HTML: ideaal voor een mobiel toestel...

2.6.1 Images

Je bemerkt dat de grote foto van Elise op de *baseline* van de eerste lijn van de alinea geplaatst is. We willen dat de tekst rond het beeld vloeit. Dat kunnen we eenvoudigweg doen door het **img** element (en dus alle **img** elementen) de property **float:left** mee te geven. Pas het stylesheet aan:

```
...
img {
    margin-right:1em;
    float:left;
}
...
```

verklein-vergroot het venster en zie hoe de tekst reageert. Bij een breed venster bemerk je misschien al een probleemje.

De **float** property heeft slechts 2 mogelijke waarden: **left** of **right**.

Verander de waarde eens naar **right**. Hoe reageert de foto? Infeite zou je dan ook de **margin-right** moeten wijzigen in **margin-left**. Zet dan terug naar **left**.

Elise Crombez



Elise Crombez (24 juli 1982) Belgisch topmodel.

Crombez, opgegroeid in het Koksijde, werd in 1999 ontdekt als Miss Mannequin in Roeselare en een vriendin deed. Ze zat tegenover middelbare schoolbanken in Veurne. Ze haalde meer dan Vogue, zowel de Engelse als het gezicht van merken als Lauren en Helmut Lang. In 2001 was ze voor het eerst te zien in een H&M. Verder liep ze voor Armani, Jean-Paul Gaultier, (ontwerper: John Galliano). Crombez is een van de meest gevraagde modellen ter wereld.

Momenteel woont Crombez in New York City, maar ze zit steeds in Koksijde liggen.

Je kan ook eens experimenteren met de plaats van het `img` element binnen de `p` tag.

`img` elementen die tussen tekst moeten zitten, plaatsen we bij voorkeur als eerste element in de *parent container* en *floaten* ze links of rechts.

Deze toestand is de meest voorkomende: de *Float* (zo noemen we een container die een `float` ingesteld heeft) wordt links of rechts uitgelijnd in de container waarin hij zit (het `p` element) en de rest van de inhoud vloeit eromheen.

Een *Float* behoort niet meer tot de *Normal Flow*: de *parent container* beschouwt de *Float* als een obstakel waarrond hij de rest moet schikken.

2.6.2 Box Floats

Een tweede toepassing die veel voorkomt is het maken van een "image gallery" – een opeenvolging - stapeling van containers.

In ons voorbeeld hebben we een aantal foto's verpakt in `div` elementen met de `class` "thumb" en een `span` als onderschrift erbij:

```
...
<div class="gallery">
  <div class="thumb">
    
    <span class="caption">Milla</span>
  </div>
...
...
```



`div` elementen zijn **block elementen** en stapelen dus van nature **op** elkaar.

Pas nu het stylesheet aan:

```
...
.thumb {
  background:#A5CB4E;
```

```
width:80px;  
height:140px;  
padding:0.625em;  
margin:0.625em;  
float:left;  
}  
...
```

Het resultaat ziet er zo uit bij een wat smaller venster:



Speel nu zeker eens met de breedte van je browservenster.
Wat observeren we hier:

1. de "thumbs" zijn *gefloat*: ze volgen elkaar op van links naar rechts en op een volgende rij volgens de volgorde waarin ze in de HTML code staan
2. bemerk dat in dit voorbeeld er 5 op de bovenste rij passen, er is net geen plaats genoeg voor een zesde: dat hangt af van de beschikbare breedte
3. probleem met de *.footer*: die komt naast de *floats* te staan.

Dit is **normaal** want hetzelfde gedrag als daarnet met *img* en tekst.
Maar... dat willen we niet... we willen dat de *.footer* onder de sectie *div.anderemodellen* blijft.

Dit is een **clear** probleem

4. probleem met de *parent* container van de "thumbs": `div.gallery`: die lijkt verdwenen

de container `div.gallery` bevat niets anders dan de `div.thumb`'s: geen andere tekst of zo. Het gevolg is dat deze container geen inhoud meer heeft en dus een hoogte heeft van 0 pixels.

Dit wordt het "**collapsing parent**" **fenomeen** genoemd.

Daardoor heeft ook de `div.andermodellen` een kleinere inhoud en vouwt zich slechts rond de overgebleven tekst.

2.6.3 Clear een box

Eerst lossen we het probleem op van de `.footer` die naast de *thumbs* komt te staan. Dit is normaal gedrag!

We zullen de `.footer` dwingen zich onder zijn *sibling* – `div.rechts` – te plaatsen met de `clear` property.

Pas het stylesheet aan:

```
...
.footer {
    background:#F4E740;
    padding:1em 2em;
    clear:left;
}
...
```

De `clear` property heeft de mogelijke waarden `none`, `left`, `right`, `both` en `inherit`

Deze eigenschap duidt aan welke zijden van de box *niet* naast een *floating box* mogen staan.

`clear:left` betekent dus: "aan mijn linkerkant mag er geen *floating box* zijn, dus moet ik mij er onder plaatsen".

We hadden hier even goed de waarde `both` kunnen geven want we willen de `.footer` alleen op een "rij" hebben.

Het resultaat:

Onze andere modellen

Hieronder vind u een selectie van onze andere modellen:



Milla



Danielle



Natalie



Omahyra



Hanna



Annelies



Hannelore



Louise

Dit is een oefensite, geen reële website. De informatie op deze pagina is niet correct en niet up-to-date.

2.6.4 clearfix lost collapsing parent op

Het *collapsing parent* probleem is echter nog niet opgelost: de "thumbs" hangen in de lege ruimte als propere was aan de waslijn.

Dit probleem kwam ook al voor bij de grote foto bovenin als je een breed scherm had:

Elise Crombez



Elise Crombez (24 juli 1

Crombez, opgegroeid in wedstrijd Miss Mannequin toen nog op de middelbare school dan tien keer de cover van het gezicht van merken als H&M, Gaultier, Calvin Klein en vijf van de meest gevraagde

Momenteel woont Crombez in Koksijde.

Onze andere modellen

ook hier houdt de *parent container* geen rekening met de dimensies van de *float*.

De oplossing wordt meestal aangeduid als een **clearFix**. Het kan op meerdere manieren opgelost worden, de ene methode biedt al meer zekerheid dan de andere.

Een oplossing is de *parent* container – in dit geval **div.anderemodellen** – ook een **float** property te geven, maar dat verplaatst het probleem misschien gewoon.

Een betere oplossing is de *parent* te dwingen rekening te houden met de *floats* die hij bevat door de *parent* een **overflow** te geven.

Pas het stylesheet aan:

```
...
.sectie {
    background:#F7F7F7;
    padding:1em;
    margin: 1em 0;
    overflow:auto;
}
...
```

overflow , een property die het tonen/verbergen van inhoud regelt voor containers die te klein zijn, dwingt hier de *parent* zich rond **alle** inhoud te "vouwen".



Elise Crombez (24 juli 1982, Moeskroen), is een Belgisch topmodel.

Crombez, opgegroeid in het West-Vlaamse Koksijde, werd in 1999 ontdekt door de wedstrijd Miss Mannequin in Roeselare, die ze samen met een vriendin deed. Ze zat toen nog op de middelbare schoolbanken in het College van Veurne. Ze haalde meer dan tien keer de cover van Vogue, zowel de Engelse als de Italiaanse. Ze werd het gezicht van merken als Prada, Jil Sander, Ralph Lauren en Helmut Lang. In 2003 was ze het gezicht van H&M. Verder liep ze modeshows voor Giorgio Armani, Jean-Paul Gaultier, Calvin Klein en Dior (ontwerper: John Galliano). In 2002 stond ze in de top vijf van de meest gevraagde modellen ter wereld.

Momenteel woont Crombez in New York City, maar ze zegt dat haar hart nog steeds in Koksijde ligt.

Onze andere modellen

Hieronder vind u een selectie van onze andere modellen:



Milla



Danielle



Natalie



Omahyra



Hanna



Annelies



Hannelore



Louise

Dit is een oefensite, geen reële website. De informatie op deze pagina is niet correct en niet up-to-date.

overflow lost het probleem in het merendeel van de browsers op maar straks voorzien we een 100% oplossing.

2.6.5 Kolommen maken met floats

De containers **.links** en **.rechts** willen we als kolommen naast elkaar krijgen.

Dat lossen we ook op met *floats*.

Pas het stylesheet aan:

```
.links {  
    background:#55C5E8;  
    padding:1em;  
    width: 30%;  
    float:left;  
}  
.rechts {  
    background:#FD6943;  
    padding:1em 2em;  
    width:60%;  
    float:left;  
}
```

Door de twee containers te *floaten* en hun breedte in te stellen komen ze naast elkaar te staan. Het is echter best mogelijk dat dat bij jou het geval niet is, afhankelijk van je scherm: de breedte van elke container moet echter een *fine-tuning* ondergaan om goed te passen. De totale breedte van beide kolommen mag immers de **width** van de *parent* niet overschrijden, anders gaan ze onder elkaar staan.

Momenteel merk je een klein verschil aan de rechthoekige. En als je het venster verkleint gebeurt een *clear*. Dit komt allemaal een beetje door het vermengen van **%** en **em**:

totale breedte = $2 \times 1\text{em padding} + 30\% \text{ width } ???$

is moeilijk uit te rekenen en verandert met een breedtewijziging.

Daarom wijzigen we de padding :

```
.links {  
    background:#55C5E8;  
    padding:1em 2.5%;  
    width: 30%;  
    float:left;  
}  
.rechts {  
    background:#FD6943;  
    padding:1em 2.5%;  
    width:60%;  
    float:left;  
}
```

Dit geeft:

.links(2 x 2.5% + 30%) + .rechts(2 x 2.5% + 60%) = 100%

The screenshot shows a website layout with a pink header containing the text "Elise Modeling Agency". On the left, there is a sidebar with a blue background containing a navigation menu with links to "home", "modellen", "about", and "contact". The main content area has an orange border and features a photo of a woman named Elise Crombez. Below the photo, her name is displayed in bold black text. To the right of the photo, there is descriptive text about her.

De kolommen sluiten perfect aan elkaar.

2.6.6 CSS classes voor clear en clearFix

Omdat de clear en clearFix hacks zeer veel voorkomen – de meeste lay-outs zijn op *floats* gebaseerd – kunnen we er ook een aparte CSS class voor maken en die dan in de HTML gaan toepassen.

Voor de *Clear* voeg je toe aan onderaan je stylesheet:

```
/* clear */  
.cf {clear:both}
```

en voor de *clearFix* gebruiken we deze hack:

```
/*clearfix hack*/  
  
.clearfix:before, .clearfix:after {  
    content: " ";  
    display: table;  
}  
.clearfix:after {  
    clear: both;  
}  
  
/* enkel voor IE6/7 */  
.clearfix {  
    *zoom: 1;  
}
```

Deze hack is gebaseerd op *CSS Generated Content* waarbij **na** een element met deze class een onzichtbare **table** cel gegenereerd wordt.

Nu mag je alle **clear** en **overflow** properties die we tot nu toe toepasten verwijderen uit je stylesheet en in de HTML code geef je

- alle **div.sectie** elementen de **class** "clearFix" mee
- de **.footer** de **class** "cf"

```
<div class="sectie clearFix" id="topmodel">  
...  
    <div class="sectie clearFix" id="anderemodellen">  
    ...  
        <div class="footer cf">
```

Met hetzelfde resultaat.

Sommige ontwikkelaars verkiezen te werken zoals we eerder deden maar deze manier van werken is meer modular en laat je toe via de HTML snel een oplossing te bieden.

En tenslotte de vraag die iedereen zal stellen: *hoe krijg ik de achtergrondkleur van de eerste kolom tot beneden, dus even lang als de rechter?*

Er zijn meerdere oplossingen maar een eenvoudige is de achtergrondkleur van **div.midden** op dezelfde kleur instellen:

```
.midden{background:#55C5E8;}
```

Dit lijkt niet te werken... Kan je de oplossing raden, rekening gehouden dat zijn inhoud (**.links .rechts**) gefloatte elementen zijn?

Juist, ja...

Elise Modeling Agency

- [home](#)
- [modellen](#)
- [about](#)
- [contact](#)

Elise Crombez



Elise Crombez (24 juli 1982, Moeskroen), is een Belgisch topmodel. Crombez, opgegroeid in het West-Vlaamse Koksijde, werd in 1999 ontdekt door de wedstrijd Miss Mannequin in Roeselare, die ze samen met een vriendin deed. Ze zat toen nog op de middelbare schoolbanken in het College van Veurne. Ze haalde meer dan tien keer de cover van verschillende tijdschriften als Elle en Vogue. Ze werd het gezicht van merken als Prada, Jill Sander, Ralph Lauren en Helmut Lang. In 2003 was ze het gezicht van H&M. Verder liep ze modeshows voor Giorgio Armani, Jean-Paul Gaultier, Calvin Klein en Dior (ontwerper: John Galliano). In 2002 stond ze in de top vijf van de meest gevraagde modellen ter wereld.

Momenteel woont Crombez in New York City, maar ze zegt dat haar hart nog steeds in Koksijde ligt.

Onze andere modellen

Hieronder vind u een selectie van onze andere modellen:

Dit is een oefensite, geen reële website. De informatie op deze pagina is niet correct en niet up-to-date.

Taken:

maak nu volgende taken

- "Elise models 2"

2.7 Enkele veel voorkomende lay-outs

DoeL

We overlopen enkele veel voorkomende lay-outs en bekijken voor- en nadelen

Theorie

Lees de volgende theorie topics na:

- CSS2 box model
- floats
- CSS3 **box-sizing**

Startbestanden

vdab_lay-out.html, vdab_lay-out.css, html5shiv.js

HTML structuur

Open het basisbestand in je editor en bekijk ook het in enkele browsers.

Het is een HTML5 bestand en er is een bijhorend stylesheet *vdab_lay-out.css* waar enkele stylerules in staan, voornamelijk om de **header** op te maken.

De structuur van de *dom tree* is eenvoudig:

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <header>
      <a href="http://www.vdab.be" id="logo">...</a>
      <nav id="hoofdmenu">...</nav>
      <form method="get" action="/search">...</form>
      <hgroup>...</hgroup>
    </header>
    <div id="container">
      <aside class="kol een">...</aside>
      <div class="kol twee">...</div>
      <!--einde kol twee-->
      <div class="kol drie">...</div>
    </div>
    <!--einde container-->
    <footer>...</footer>
  </body>
</html>
```

1. een **header**
2. een **div.container** met 3 children:
 - a. een **aside**,
 - b. twee **div.kol** elementen
3. een **footer** element.

De `aside` en de twee `div.kol` elementen hebben een lichtgrijze titel, `aside`, `.twee` en `.drie` om ze gemakkelijk te kunnen identificeren.

Als we de pagina nu bekijken in IE versie vóór 9, zullen er problemen zijn, aangezien deze versies geen HTML5 elementen kennen.

Om dit op te lossen voegen we het `html5shiv.js` toe in een `script` tag.

```
...
<script src="../js/html5shiv.js"></script>
</head>
```

Zorg zelf voor het juiste pad naar de *Shiv*.

The screenshot shows a website for 'Webontwikkelaar een VDAB opleiding T&O Oostende'. The header includes the VDAB logo, a search bar, and navigation links for 'Welkom', 'Opleiding', 'Handleidingen', and 'Contact'. The main content area features an `aside` section titled 'De cursisten:' containing a list of names. Below it is a section titled 'Interesse?' with a form field for email input and a 'verzend' button. A footer at the bottom left contains a welcome message for the website.

Nu kunnen we verder werken.

Aan de HTML zullen we niet meer raken, we zullen enkel de lay-out wijzigen door er verschillende stylesheets aan te koppelen.

2.7.1 Outline om te lay-outen

Om gemakkelijk te verschillende blokken/kolommen te kunnen onderscheiden stellen we een `outline` in. Voeg de volgende lijn toe aan aan het stylesheet:

```
div, aside, header, footer {outline:1px dashed red;}
```

Alle vermeldde blokken worden nu omrand door een rood stippellijntje zodat we ze beter kunnen zien. Je zou kunnen denken dat je hetzelfde kunt doen met een **border**, maar dat klopt niet.

Borders zijn een onderdeel van het *box model* en worden dus meegerekend in de totale dimensies: 3 geneste **div's** met een **border** van 1px nemen 6px in beslag. Outlines zijn daarin neutraal.

2.7.2 Fixed width met drie kolommen

Sla het stylesheet op als `vdab_lay-out_1.css` en pas ook de koppeling in de HTML aan.

Een smalle gecentreerde container:

Eerst maken we de container wat smaller: we kiezen voor een **vaste breedte**: 960px.

Dit wijzigt weinig visueel, maar nu willen we de **.container centreren** op het scherm. Vul de selector voor de **class .container** in:

```
.container {  
    width: 960px;  
    margin: 0 auto;  
}
```

Deze waarden zetten de breedte op 960px, de **top- en bottom margin** op 0 en laat **left** en **right** automatisch gebeuren: resultaat, de container wordt in het midden van het scherm geplaatst.

Verwar dit niet met **text-align**: als we de regel **text-align:center** zouden toevoegen aan de **class .container**, dan wordt zijn inhoud gecentreerd, niet de container zelf.



Deze lay-out noemen we nu al een FIXED WIDTH lay-out: we gebruiken een vaste breedte uitgedrukt in pixels.

De inhoud van de **header** volgt deze wijziging natuurlijk niet: hij blijft de volledige breedte behouden. Wil je dat wel, dan moet je eerst even nadenken over hoe je dat gaat aanpakken:

- pas je de **class .container** toe op de **header** dan zal die ook versmallen, maar de grijze achtergrond zal dat ook doen...
- daarom kiezen we voor een andere aanpak: voeg een extra div element toe net binnen de header:

```
<header>
```

```
<div class="container">  
...  
</div>  
</header>
```

Op die manier behouden we de brede grijze strook en toch is de inhoud beperkt in breedte.

Om het logo wat te benadrukken geven we het nu een negatieve marge zodat het uitsteekt:

```
#logo{  
    float:left;  
    margin: 0 0 0 -120px;  
    padding:0;  
}
```



drie kolommen

In het stylesheet vind je een deel over `aside` en `.kol`, voeg er deze properties aan toe:

```
*****aside, kolommen****/  
aside {float:left; width:300px;}  
.kol {float:left; width:300px;}  
.twee {}  
.drie {}
```

Door de breedte van de `div.kol` en de `aside` op `300px` te zetten, maken we 3 (te) smalle kolommen. Door ze te `floaten` komen ze naast elkaar te staan.

Zoals je weet speelt nu hun volgorde in de HTML een rol: bij een `float:left` komt de eerste box links te staan, de anderen volgen van links nr rechts.

ClearFix

Alles lijkt OK tot je besluit een achtergrondkleurtje of background-image te plaatsen op de container, dan stuiten we op een gekend probleem. De `#inhoud` (die overeenkomt met `.container`) geven we een kleurtje:

```
.container {  
    width: 960px;  
    margin: 0 auto;  
}  
#inhoud{ background-color:#F2F8F9; }
```

Resultaat: niets... we verwachten een lichtblauwe achtergrondkleur achter de drie kolommen.

Je bent dit probleem al tegengekomen in één van de vorige projecten: dit is een *collapsing parent*. We weten hoe we dit eenvoudig kunnen oplossen: pas de property **overflow** toe.

```
.container {  
    width: 960px;  
    margin: 0 auto;  
    overflow:auto;  
}
```

Het probleem lijkt opgelost: we zien de achtergrond, maar ... het logo is verdwenen!

Omdat het logo met een negatieve marge uitsteekt achter de **.container** in de **header**, wordt het verborgen. Daarom kunnen we geen gebruik maken van de eenvoudige oplossing. We zullen een echte *clearfix hack* moeten gebruiken. Je kunt die kopiëren uit het project "Werken met floats":

```
/*clearfix hack*/  
.clearfix:before, .clearfix:after {  
    content: " ";  
    display: table;  
}  
.clearfix:after {  
    clear: both;  
}  
/* enkel voor IE6/7 */  
.clearfix {  
    *zoom: 1;  
}
```

Vewijder de **overflow:auto** uit **.container** selector en geef de class **.clearfix** aan de twee **div.container**:

```
<div class="container clearfix">
```

Nu we de achtergrondkleur zien bemerken we ook dat de totale breedte van de kolommen minder is dan de voorziene 960px.

Omdat we werken met een **fixed lay-out** is dat probleem snel opgelost. We voegen zoveel marge, border en padding toe tot het totaal klopt. Voeg wat extra styles toe aan de **aside** en **.kol**:

```
aside, .kol {
    float:left;
    width:300px;
    margin:4px;
    padding:4px;
    border:2px solid #E4F0F3;
}
```

De berekening voor een fixed lay-out kunnen we exact doen kloppen:

$3 \times (\text{width } 300\text{px} + 4\text{px margin (X 2)} + 2\text{px border (X 2)} + 4\text{px padding (X 2)}) = 960\text{px}$

Zet nu even de outline in commentaar, dan zien we dit:

The screenshot shows a web page layout with three columns. The first column, 'aside', contains a list of names. The second column, '.twee', contains text and a list of interests. The third column, '.drie', contains a question and a list of PHP features.

aside	.twee	.drie
De cursisten: <ul style="list-style-type: none"> • Paul van Klee • Linda Lovelace • Rudolf Valentino • Eva Green • Daniel Craig • Gerhard Richter(voltooid) • Francis Bacon(voltooid) • Rene Magritte(voltooid) • Eva Longoria(voltooid) • Toyo Ito(voltooid) Interesse? Vul dan hier je emailadres in, we contacteren je zo snel mogelijk: Email: <input type="text"/> <input type="button" value="verzend"/>	Welkom op de website van de opleiding Webontwikkeling van de VDAB Oostende. In deze opleiding leer je zelfstandig dynamische websites ontwikkelen en onderhouden op een professioneel niveau. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Wat houdt het in? Tijdens de oriënteringsfase leer je: <ul style="list-style-type: none"> • programmatilogica • MS Access • Normaliseren van databanken 	Wat is PHP? Een PHP ontwikkelaar gebruikt PHP om server-side scripts te maken die dynamisch webpagina's genereren. Je onderzoekt de wensen van de gebruikers en ontwerpt en onderhoudt het programma, test het resultaat en voorziet het van de nodige documentatie. Met dynamische pagina's bedoelen we het interpreteren van de feedback van de gebruiker (uit formulierkeuzes, hyperlinks, datums, ...) om gegevens uit een database te halen en deze te verwerken in de teruggestuurde webpagina. Daarbij gebruik je alle functionaliteit van PHP en de interactiviteit met een relationele database systeem, zoals MySQL, en leer je XML te verwerken. PHP is Open Source en geniet zo van de ondersteuning en documentatie van de development community. Het is ook gratis software. PHP heeft een wat lagere instapdrempel dan alternatieven zoals Java of .Net, maar voldoet evenzeer aan alle vereisten voor de professionele

Merk op dat als je ook maar 1 pixel teveel hebt, de laatste kolom onder de andere verdwijnt! de som moet kloppen!

We kunnen dit nog perfectioneren: om de borders links en recht te laten samenvallen met de rand van achtergrond, kunnen we met een zogenaamde "gutter" (goot) werken: dat is een ruimte die enkel **tussen** de kolommen zit. Dat doen we door de dubbele **margin** te vervangen door enkel een **margin-right** voor alle kolommen met uitzondering van de meeste rechtse. Dat betekent dat we de laatste kolom geen **margin** mogen geven.

```
aside, .kol {  
    float:left;  
    width:300px;  
    margin-right:12px;  
    padding:4px;  
    border:2px solid #E4F0F3;  
}  
.kol:last-child{ margin-right:0; }
```

Het resultaat is *exact* passend.

2.7.3 *Faux columns*

Veronderstel dat je enkel kolomregels wil: lijnen **tussen** de kolommen, niet eromheen.

Van de vorige lay-out naar deze lijkt dit een kleine stap: je plaats enkel een **border-right** en zorgt dat je evenveel ruimte krijgt aan weerszijden van de border. Er is echter een fundamenteel probleem: de border zal enkel zolang zijn als de inhoud van de kolom. Kijk maar: nu ook is de rand rond de kolom niet evenhoog.

"Faux Columns" is een techniek voorgesteld door *Dan Cederholm* die gebaseerd is op het gebruik van achtergrondfiguren die verticale lijnen bevatten.

Sla je CSS bestand op onder een andere naam *vdab_lay-out_2.css*. Pas de koppeling in *vdab_lay-out.html* aan.

We doen het volgende:

- we plaatsen de speciale ontworpen achtergrondfiguur op de kolom-container
- we verdelen de linker en rechtermarges evenredig zodat de lijn in het midden komt, in de *gutter*
- Opnieuw heeft de eerste kolom geen linker marge, de laatste geen rechter marge

Pas aan:

```
#inhoud{ background: url(..../img/scheidingslijnen.png) 0 0 repeat-y; }  
aside .kol {  
    float:left;  
    width:300px;  
    margin:0 6px;  
    padding:0 4px;  
}  
.kol:last-child{ margin-right:0; }
```

Het resultaat ziet er zo uit:

The screenshot shows a website layout with three columns. The left column ('aside') contains a list of names under 'De cursisten:' and a contact form. The middle column ('twee') contains text about the PHP course and a list of topics. The right column ('drie') contains text about PHP and its syntax. The top navigation bar includes links for Welkom, Opleiding, Handleidingen, Contact, and a login field.

De cursisten:

- Paul van Klee
- Linda Lovelace
- Rudolf Valentino
- Eva Green
- Daniel Craig
- Gerhard Richter(voltoid)
- Francis Bacon(voltoid)
- René Matisse(voltoid)
- Eva Longoria(voltoid)
- Toyo Ito(voltoid)

Interesse?

Vul hier je emailadres in, we contacteren je zo snel mogelijk:

Email:

twee

Welkom op de website van de opleiding Webontwikkelaar een VDAB opleiding T&O Oostende

Welkom op de website van de opleiding Webontwikkeling van de VDAB Oostende. In deze opleiding leer je zelfstandig dynamische websites ontwikkelen en onderhouden op een professioneel niveau.

Lorem ipsum dolor sit amet,consectetur adipiscitur elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco labors nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Wat houdt het in?

Tijdens de oriënteringsfase leer je:

- programmatiologica
- MS Access
- Normaliseren van databanken
- OOP
- XHTML & CSS
- HTML5 & CSS3
- PHP
- MySQL
- JavaScript

En ook nog communicatie- en sollicitatie training. Hiermaa doe je een praktische stage in een bedrijf.

Voor wie?

In principe kan iedereen deze opleiding aanvragen, een diploma is niet vereist. Wat wel vereist is, is dat

.drie

Wat is PHP?

Een PHP ontwikkelaar gebruikt PHP om server-side scripts te maken die dynamisch webpagina's genereren. Je onderzoekt de wensen van de gebruikers en ontwerpt en onderhoudt het programma, test het resultaat en voorziet het van de nodige documentatie.

Met dynamische pagina's bedoelen we het interpreteren van de feedback van de gebruiker (uit formulierkeuzes, hyperlinks, datums, ...) om gegevens uit een database te halen en deze te verwerken in de teruggestuurde webpagina. Daarbij gebruik je alle functionaliteit van PHP en de interactiviteit met een relationeel database systeem, zoals MySQL, en leer je XML te verwerken.

PHP is Open Source en geniet zo van de ondersteuning en documentatie van de development community. Het is ook gratis software. PHP heeft een wat lagere instapdempel dan alternatieven zoals Java of .Net, maar voldoet evenzeer aan alle vereisten voor de professionele programmeur.

Op syntaxgebied leunt de taal aan bij C, C++, Perl en Java. PHP bevat honderden zoniet duizenden basisfuncties die goed gedocumenteerd zijn op de website en becommentarieerd kunnen worden door gebruikers.

PHP wordt zowel door grote als kleine bedrijven gebruikt, maar is specifiek gericht op webapplicaties. Enkele bekende PHP toepassingen (dikwijls samen met MySQL) zijn WordPress, Drupal, Joomla en mediaWiki. PHP leent zich tot het ontwikkelen van e-commerce applicaties, Content-management systemen, blog-sites en

Faux columns bieden een oplossing voor vertikale lijnen. Ze zijn relatief makkelijk te toe te passen, maar de achtergrondfiguur - die enkel bestaat uit een lange smalle transparante figuur met op de juiste plaats twee streeppjes - moet speciaal gemaakt worden voor de voorziene breedte.

2.7.4 Fluid lay-out

Open opnieuw *vdab_lay-out_1.css* en sla dit op als *vdab_lay-out_3.css*. Pas opnieuw de koppeling in *vdab_lay-out.html* aan.

Een *Fixed lay-out* behoudt zijn breedte ongeacht de breedte van het scherm. Dat betekent dat bv op een zeer breed scherm er veel ruimte verloren gaat. Op een kleiner scherm zie je een gedeelte van de inhoud niet meer.

Nu maken we een Fluid (Liquid) lay-out.



In een **FLUID WIDTH** lay-out is breedte uitgedrukt in percentages.

Misschien kunnen we alle breedtes gewoon omzetten naar percentages?

```
.container {
  width: 80%;
  margin: 0 auto;
```

```
}
```

```
aside, .kol {
```

```
    float:left;
```

```
    width: 29%;
```

```
    margin-right: 2%;
```

```
    padding: 1%;
```

```
    border: 1% solid #E4F0F3;
```

```
}
```

```
.kol:last-child{
```

```
    margin-right:0;
```

```
}
```

De redenering is deze:

de **.container** neem 80% van zijn parent in en alle kolommen samen nemen 100% van de container in. Zolang de optelsom van de drie kolommen de 100% niet overschrijdt, is de lay-out OK.

Je mag dit proberen, maar dan merk je dat er *geen borders* zijn, want... borders mogen niet uitgedrukt worden in %, enkel in **px**, **pt**, **em** en **rem**...

Dat betekent dat we een **mix** krijgen van waarden om de totale breedte te berekenen:

```
aside, .kol {
```

```
    float:left;
```

```
    width: 32%;
```

```
    margin-right: 6px;
```

```
    padding: 4px;
```

```
    border: 2px solid #E4F0F3;
```

```
}
```

```
.kol:last-child{margin-right:0;}
```

Hier gebruiken we enkel percentages voor de kolombreedtes en pixels voor **margin**, **padding** en **border**.

Als je dit test werkt het maar je bemerkt het volgende:

- het totaal van de 3 kolommen is niet perfect 100%, het is erg moeilijk om het exact juist te krijgen
- voor een breed venster lukt het, maar zodra je versmalt, kunnen er geen 3 kolommen naast elkaar staan, in de ene browser al sneller dan in de andere

Dit komt natuurlijk omdat wij **%** en **pixels** nooit exact kan laten totaliseren op 100%, een percentage is ook altijd relatief t.o.v. zijn container dus de situatie is erg veranderlijk.

2.7.5 box-sizing

De perfectie kunnen we bereiken door de CSS3 property **box-sizing** te gebruiken. Mogelijke waarden zijn: **content-box**, **padding-box**, **border-box**.

Deze eigenschap verandert de berekeningswijze van het box model:

CSS2.1 box model	$\text{totale breedte} = (\text{margin} + \text{border} + \text{padding}) * 2 + \text{width}$
<code>box-sizing: content-box</code>	$\text{totale breedte} = (\text{margin} + \text{border} + \text{padding}) * 2 + \text{width}$ dus identiek aan CSS2.1
<code>box-sizing: padding-box</code>	$\text{totale breedte} = (\text{margin} + \text{border}) * 2 + \text{width}$ padding wordt afgetrokken van width : padding ten koste van inhoud
<code>box-sizing: border-box</code>	$\text{totale breedte} = (\text{margin}) * 2 + \text{width}$ padding en border worden afgetrokken van width : padding en border ten koste van inhoud

Opmerkingen:

- de waarde **padding-box** wordt niet goed ondersteund
- gebruik best browser-prefixes

Het komt erop neer dat we geen rekening hoeven te houden met borders en padding: ze worden toch afgetrokken van de width.

We wijzigen onze style waarden:

```
aside, .kol {
    float:left;
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
    width: 33%;
    margin-right:0.5%;
    padding:4px;
    border:2px solid #E4F0F3;
}
```

Bespreking:

- de **box-sizing** wordt op **border-box** gezet
- de **width** van de 3 kolommen is 33%, dat betekent dat er nog 1% rest
- die 1 % wordt verdeeld over twee rechtermargins (want kolom 3 heeft er geen)
- van de **border** en de **padding** trekken we ons niets meer aan

Webontwikkelaar
een VDAB opleiding
T&O Oostende

Welkom Opleiding Handleidingen Contact

aside

De cursisten:

- [Paul van Klee](#)
- [Linda Lovelace](#)
- [Rudolf Valentino](#)
- [Eva Green](#)
- [Daniel Craig](#)
- [Gerhard Richter](#)(voltooid)
- [Francis Bacon](#)(voltooid)

.twee

Welkom op de website van de opleiding Webontwikkeling van de VDAB Oostende. In deze opleiding leer je zelfstandig dynamische websites ontwikkelen en onderhouden op een professioneel niveau.

Wat ipsum dolor sit amet,consectetur adipisciing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud.

.drie

Wat is PHP?

Een PHP ontwikkelaar gebruikt PHP om server-side scripts te maken die dynamisch webpagina's genereren. Je onderzoekt de wensen van de gebruikers en ontwerpt en onderhoudt het programma, test het resultaat en voorziet het van de nodige documentatie.

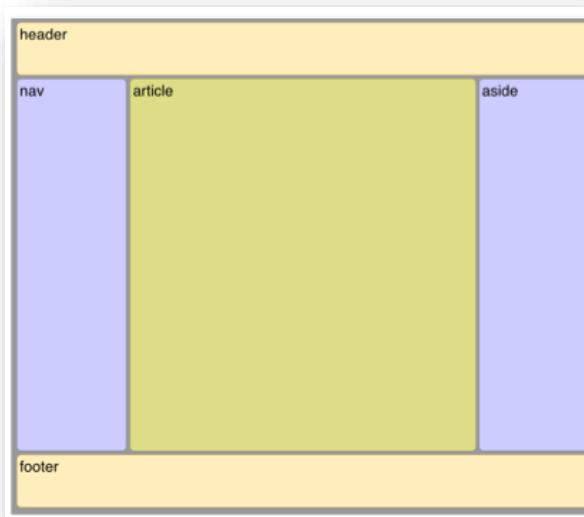
Met dynamische pagina's bedoelen we het.

Het resultaat is OK én het blijft goed als je het venster verkleint.

2.7.6 Fluid Holy Grail

The "*Holy grail*" lay-out is het lay-out type dat iedereen vroeger - en nu nog - *ideaal* vond.

Het bestaat uit een kop, een middenstuk, en een voet. Het middenstuk heeft zowel links als rechts een smalle kolom met vaste breedte en een opvullend middenstuk:



Hierboven een voorbeeldje met HTML5 elementen.

Deze lay-out werd/wordt veel gebruikt en heeft bij nader inzien een aantal voordelen die het interessant maken in responsive web design. Wij maken het hier met twee vaste kolommen links en rechts en een *fluid* middenstuk.

We moeten één wijziging aanbrengen in de HTML: de derde kolom (of eerder welk element dat als derde kolom dient), moet omhoog komen in de HTML flow:

```
<div class="container clearfix" id="inhoud">
  <aside class="kol een">
    ...
  </aside>
  <div class="kol drie">
    ...
  </div>
  <div class="kol twee">
    ...
  </div>
</div>
```

Verplaats nu eerst kolom **.drie**.

De werkwijze in de CSS is deze:

- de middenkolom(**.twee**) blijft normaal, niet *ge-float* en zonder **width**
- de linker- en rechterbalk krijgen een vaste breedte en worden respectievelijk links en rechts *ge-float*.
Floats staan best eerst in de flow, dat is de reden van de verplaatsing
- de middenkolom krijgt linker- en rechtermarges die minstens de dikte van de zijkanten hebben

Pas de CSS voor de kolommen aan:

```
aside, .kol {
  float:left;
  -webkit-box-sizing: border-box;
  -moz-box-sizing: border-box;
  box-sizing: border-box;
  width: 33%;
  margin-right:0.5%;
  padding:4px;
  border:2px solid #E4F0F3;
}

.kol:last-child{margin-right:0;}

.een {
  float:left;
  width:180px;
}
.twee {
  margin: 0 246px 0 186px;
}
.drie {
  float:right;
  width:220px;
}
```

De **box-sizing** speelt hier niet echt een rol.

Het resultaat kan je versmallen/verbreden: het is de middenkolom die *Fluid* is, de zijbalken blijven exact staan.

The screenshot shows a website layout with a header containing the VDAB logo, the title 'Webontwikkelaar een VDAB opleiding T&O Oostende', and navigation links for 'Welkom', 'Opleiding', 'Handleidingen', and 'Contact'. Below the header are three columns:

- aside**: A sidebar containing the heading 'De cursisten:' followed by a bulleted list of names: Paul van Klee, Linda Lovelace, Rudolf Valentino, Eva Green, Daniel Craig, Gerhard Richter(voltooid), Francis Bacon(voltooid), Rene Matisse(voltooid), Eva Longoria(voltooid), and Toyo Ito(voltooid).
- .twee**: The main content area containing a welcome message and a large amount of placeholder text (Lorem ipsum).
- .drie**: A sidebar containing the heading 'Wat is PHP?' followed by a text block about PHP and a note about dynamic pages.

Je ziet dat kolom **.drie** toch op zijn goede plaats staat.

We kunnen nog één aanpassing maken: als het venster te smal wordt lijkt deze lay-out nergens meer op:

```
.container {  
    width: 80%;  
    min-width:800px;  
    margin: 0 auto;  
}
```

Met **min-width** hou je een te erge versmalling tegen.

Er bestaan nog tientallen andere 'klassieke' lay-outs, maar we hopen dat je nu voldoende gewapend bent om ze te begrijpen.

Taken:

maak nu volgende taken

- "webontwikkelaar, deel 1"

2.8 Een thumbnaillist

Doel

De *unordered list* (`ul li`) leent zich erg goed voor het maken van allerhande "lijstjes": je ziet ze op forums, websites, mobile websites, apps etc.. in alle vormen en groottes, sommige aanklikbaar, andere niet.

RECENTSTE NIEWS

- 15:12 Egyptische oppositie wijst uitgestoken hand Mursi af
- 14:57 Malinese rebellen verbranden oude manuscripten
- 14:52 Van Besien counterft de N-VA-kritiek op de BHG
- 14:32 Susanne Bier en Tim Robbins in jury Berlinale
- 14:23 Zestiger redt in bevoren water gevallen baby

[alles chronologisch](#) [alles per rubriek](#)

Features

-  **Whose land is it, anyway?**
British Indians return to claim now-valuable ancestral land
-  **The solace of quantum**
Are mind-bending physics effects going on in birds, and your nose?
-  **Taking the strain**
How will Afghan security forces cope when Nato withdraws?
-  **Day in pictures**
Twenty-four hours of news photos from around the world
-  **Cold comfort**
Greeks turn to illegal logging to stay warm

PEOPLE YOU MAY KNOW

-  **Ingrid Debeuf**, sollicitatiecoach bij VDAB Competentiecentrum
[Connect](#)
-  **Anja Delaere**, beeldverwerking - grafisch ontwerper bij privé
[Connect](#)
-  **Karl Roosen**, IBO begeleider ICT-Limburg at VDAB
[Connect](#)

[See more »](#)

Het grote voordeel van lijstjes gebaseerd op een `ul` of `ol` is de supereenvoudige structuur die het zo makkelijk maakt er elementen aan toe te voegen of weg te nemen!

En weet je... ooit was er een tijd dat men hiervoor tabellen gebruikte maar die is gelukkig al lang passé!

Een van de meest voorkomende stijlvormen gemaakt uit zo'n lijst is een **thumbnaillist** (*teaser thumbnaillist*), een lijst van kleine figuurtjes of icoontjes telkens vergezeld van een korte tekst, soms ook een titeltje, hierboven zie je er twee voorbeelden van.

Wij vinden het **essentieel** dat je deze eenvoudige lay-out kunt maken uit een *unordered list*.

Theorie

Lees de volgende theorie topics na:

- HTML lijsten
- CSS lijsten opmaken
- CSS Floats

Startbestanden

thumblist.html

2.8.1 Opdracht

Open het startbestand: het bevat een lijst van *teasers* voor een overzicht van plantengroepen voor een webwinkel.

Het startbestand bevat een **style** element met een minimum aan CSS.

De HTML is een **ul** lijst met **li** elementen die elk een **a** element bevatten waarin een **img** en een **h4** element zitten. Daarna volgen één of meer **p** elementen :

```
<div class="kolom">
    <ul id="plantenmenu">
        <li><a href="plantenshop_shop.php?cat=1" title="Eacute;n- en tweejarigen" >
            
            <h4>Eacute;n- en tweejarigen</h4>
            </a>
            <p>Een- en tweejarigen kunt u zelf zaaien. Deze planten bloeien vaak meer en langer dan vaste planten en brengen kleur en variatie.</p>
            <p>Daarnaast kunnen zij goed leemtes vullen in de border. </p>
        ...
        </li>...
    </ul>
</div>
```

Het **href** attribuut van de hyperlinks is correct maar zal hier nergens naar toe leiden. De images zijn allemaal 80 x 80 pixels.

De lijst zelf zit vervat in een **div.kolom** element. Dit volgt de meest voorkomende situatie waarin de lijst vervat zit in een andere **container**. Het is deze container die zal bepalen hoe breed de lijst zal zijn: een kolom, een zijkalkje, etc... Hier is de breedte van **.kolom** ingesteld op 26em. Je mag dat straks gerust wijzigen en zien hoe de structuur reageert.

In een browser ziet het er nu zo uit:

Plantengroepen in onze winkel



Eén- en tweejarigen

Een- en tweejarigen kunt u zelf zaaien. Deze planten bloeien vaak meer en langer dan vaste planten en brengen kleur en variatie.

Daarnaast kunnen zij goed leemtes vullen in de border.

Een- en tweejarigen zijn te krijgen in vele verschillende vormen, van hele kleine tere plantjes tot klimmers die in onafzienbare tijd uw muur of schutting bedekken met een kleurige bloemenpracht. Bekijk onze selectie Een- en tweejarigen



Vaste planten en kruiden

We observeren het volgende:

- de lijst is een normale bolletjeslijst
- alle hyperlinks zijn blauw
- de figuren staan eerst
- de titels vallen niet op omdat ze in een [a](#) element zitten

Het zijn natuurlijk de *default styles* van de elementen die hier spelen. Via CSS zullen we deze dus eerst moeten veranderen.

2.8.2 Opmaak met CSS

Eerst voegen we een `class` toe aan het root element van de lijst: de `ul`. Zoals we al eens eerder vermeldden, gebruiken we liever een `class` om opmaak toe te voegen dan een `id`. Op die manier kunnen we deze opmaak ook nog ergens anders gebruiken, misschien heb je verschillende van deze lijstjes op één pagina, of in één website.

Voeg toe:

```
...
<ul id="plantenmenu" class="thumblist">
  <li>...
```

Voeg nu ook deze CSS toe aan het `style` element:

```
...
```

```
/* UL.thumblist*/
.thumblist {
    width:100%;
    margin:0;
    padding:0.5em 0;                                /* 0.5em = 6px */
    background:#EEE;
    list-style:none;
    font-size:0.75em;                               /*12px = 1em */
}
.thumblist li {
    border:1px dashed silver;
    padding: 0.5em;                                /* 6px */
    margin:0.5em;
}
...
...
```

Bespreking:

- door de **width** van de **ul** op 100% in te stellen, en de **margin** op 0, neemt de lijst zijn maximale ruimte in **.kolom** in: zijn breedte hangt nu volledig af van **.kolom**
- **list-style:none** verwijdert het opsommingsteken van de lijst
- we verkleinen het lettertype in de **class thumblist** met **font-size**: als de **body** een lettergrootte van 100% heeft, betekent dat in alle browsers 16px.
 $16px \times 0.75 = 12px$ of **0.75em**;
- de **li** elementen krijgen nu een rand en gedragen zich als blokjes in de opsommingslijst

Het resultaat ziet er toch al heel anders uit:

Eén- en tweejarigen

Een- en tweejarigen kunt u zelf zaaien. Deze planten bloeien vaak meer en langer dan vaste planten en brengen kleur en variatie.

Daarnaast kunnen zij goed leemtes vullen in de border.

Vaste planten en kruiden

De groep vaste planten omvat kruidachtige planten die geen houtige takken vormen.

Een groot deel van deze planten sterven in de winter bovengronds af, maar schieten elk jaar terug vanuit een overblijvend wortelstelsel.

alhoewel nog niet perfect.

We voegen opnieuw styles toe:

```
.thumblist img {  
    float:left;  
    margin-right: 0.5em;  
}  
.thumblist a {  
    color:inherit;  
    text-decoration:none;  
}  
.thumblist h4 {  
    font-size: 1.1666em;          /* 14px */  
    margin: 0 0 0.1em 0;  
    color:#0099FF;  
}  
.thumblist p {  
    margin: 0 0 0.1em 0;  
}
```

- de `img` wordt links `ge-float` en een rechtermarge meegegeven om wat afstand tot de tekst te creëren
- voor de hyperlinks in de `class thumblist`
 - stellen we de kleur in op de kleur van de `parent` via `inherit`, grijs dus
 - en we verwijderen de onderlijning met `text-decoration:none`
- voor de `h4`
 - stellen we het lettertype in op `1.1666em` .
 $1.1666 \times 12\text{px} (\text{parent}) = 14\text{px}$.

- stellen we de **margin** in op 0 met uitzondering van de onderkant waar hij toch erg klein gezet word om in deze beperkte ruimte te kunnen werken
 - we stellen een blauwe tekstkleur in
- ook de alinea krijgt een zeer klein ondermarge



Eén- en tweejarigen
Een- en tweejarigen kunt u zelf zaaien. Deze planten bloeien vaak meer en langer dan vaste planten en brengen kleur en variatie. Daarnaast kunnen zij goed leemtes vullen in de border.



Vaste planten en kruiden
De groep vaste planten omvat kruidachtige planten die geen houtige takken vormen. Een groot deel van deze planten sterven in de winter bovengronds af, maar schieten elk jaar terug vanuit een overblijvend wortelstelsel.



Bomen en struiken
Van oudsher is dit onze specialiteit. Bij ons vindt u dan ook een uitgelezen selectie van inheemse bomen en struiken ideaal voor alle 'echt' groen beplantingen. zoek tussen bomen en struiken

Zo, een thumbnaillist is eenvoudiger dan je denkt, niet? toch kunnen er zich problemen voordoen.

2.8.3 Meer of minder tekst

Deze lay-out steunt op een gecontroleerde hoeveelheid tekst: je moet ervoor zorgen dat er niet te veel, maar ook niet te weinig woorden zijn. Dit wordt gecontroleerd door het serverside script die dan meestal een hyperlink produceert in de aard van [lees verder...](#)

Teveel tekst

Wat gebeurt er als er teveel tekst is?

We proberen het even: voeg een alinea tekst toe aan het eerste **li** item, bv:

<p>Een- en tweejarigen zijn te krijgen in vele verschillende vormen, van hele kleine tere plantjes tot klimmers die in onafzienbare tijd uw muur of schutting bedekken met een kleurige bloemenpracht. Bekijk onze selectie Een- en tweejarigen</p>

Het resultaat:



Eén- en tweejarigen
Een- en tweejarigen kunt u zelf zaaien. Deze planten bloeien vaak meer en langer dan vaste planten en brengen kleur en variatie. Daarnaast kunnen zij goed leemtes vullen in de border.

Een- en tweejarigen zijn te krijgen in vele verschillende vormen, van hele kleine tere plantjes tot klimmers die in onafzienbare tijd uw muur of schutting bedekken met een kleurige bloemenpracht. Bekijk onze selectie Een- en tweejarigen



Vaste planten en kruiden
De groep vaste planten omvat kruidachtige planten die geen houtige takken vormen. Een groot deel van deze planten sterven in de winter bovengronds af, maar schieten elk jaar terug vanuit een overblijvend wortelstelsel.

Zodra de tekst verder dan de image uitkomt, plooit hij zich rond de ge-floatte image. De items zijn uiteraard ook verschillend in hoogte.

Wil je een soort "marge" aan de linkerzijde zodat het lijkt dat het beeld een kolom voor zich heeft? dan moet je de alinea's een linker marge geven minstens even breed als de image:

```
.thumblist p {  
    margin:0 0 0.1em 7.16666em;  
}
```

Hier geven we **7.16666em**, dus **86px** marge, dat komt perfect overeen met de **img** en zijn rechtermarge.

Eén- en tweejarigen
Een- en tweejarigen kunt u zelf zaaien. Deze planten bloeien vaak meer en langer dan vaste planten en brengen kleur en variatie. Daarnaast kunnen zij goed leemtes vullen in de border.
Een- en tweejarigen zijn te krijgen in vele verschillende vormen, van hele kleine tere plantjes tot klimmers die in onafzienbare tijd uw muur of schutting bedekken met een kleurige bloemenpracht. Bekijk onze selectie Een- en tweejarigen

Vaste planten en kruiden
De groep vaste planten omvat kruidachtige planten die geen houtige takken vormen.

Bomen en struiken
Van oudsher is dit onze specialiteit. Bij ons vindt u dan ook een uitgelezen selectie van inheemse bomen en struiken ideaal voor alle 'echt' groen beplantingen. zoek tussen bomen en struiken

Te weinig tekst

Vermits het serverside script heel waarschijnlijk een maximum stelt op de tekst, is het waarschijnlijker dat er te weinig tekst is, en dan krijgen we een gekend fenomeen.

Verwijder de extra alinea en de grote linkermarge erop.

Verwijder nu ook wat tekst in één van de blokjes. Dit zal gebeuren:

Eén- en tweejarigen Een- en tweejarigen kunt u zelf zaaien. Deze planten bloeien vaak meer en langer dan vaste planten en brengen kleur en variatie. Daarnaast kunnen zij goed leemtes vullen in de border.	Vaste planten en kruiden De groep vaste planten omvat kruidachtige planten die geen houtige takken vormen.
Bomen en struiken Van oudsher is dit onze specialiteit. Bij ons vindt u dan ook een uitgelezen selectie van inheemse bomen en struiken ideaal voor alle 'echt' groen beplantingen. zoek tussen bomen en struiken	Klim- en gevelplanten Een tuin kan niet zonder verticale accenten. Klimplanten zijn daarvoor de ideale versierders, ter verfraaiing van een muur, prieel of pergola of om een lelijk hekwerk aan het oog te onttrekken.

Inderdaad, door de *float* van de `img` elementen én het tekort aan tekst gaat het volgende `li`-item er naast staan. Vermits jullie sinds het vorige project alles weten over *floats* moet je hiervoor een oplossing kunnen vinden, lees dus niet verder, probeer het zelf op te lossen...

Geheugenprobleempje?

Er zijn inderdaad meerdere oplossingen voor deze clear-probleem:

- zet `clear:both` op de `li` elementen
- zet een `overflow` op de `li` elementen
- stel een vaste hoogte in op de `li` elementen

De `clear:both` lost het wel op maar niet goed, daarom geven we de voorkeur aan een `overflow` gecombineerd met een vaste hoogte:

```
.thumblist li {  
    border:1px dashed silver;  
    padding: 0.5em; /* 6px */  
    margin:0.5em;  
    height:7em; /* 84px */  
    overflow:hidden;  
}
```

Bespreking:

- de `height` van 7em zorgt ervoor dat de `li` elementen allemaal gelijk zijn
- de `overflow: hidden` verkiezen we boven een `overflow:auto`, want die zal scrollbars tevoorschijn toveren. Als er teveel tekst is (wat niet zou mogen), zal die gewoon niet meer te zien zijn.

2.9 Breadcrumbs met CSS driehoekjes

Doel

We vormen een bulleted list om in een *breadcrumb trail* en gebruiken daarbij CSS driehoekjes. Dit is wat we willen bereiken:



Theorie

Lees de volgende theorie topics na:

- CSS

Startbestanden

degazet/abonneren.html

2.9.1 HTML structuur

Open de pagina van "De GAZET" *abonneren.html*.

Zoek de volgende HTML structuur:

```
<nav id="abonneermenu" class="cf">
    <ul class="breadcrumbs">
        <li class="actief"><a href="#">Soort abonnement</a></li>
        <li><a href="#">Uw gegevens</a></li>
        <li><a href="#">Betalen</a></li>
        <li><a href="#">Bevestigen</a></li>
    </ul>
</nav>
```

Het is een *container* met een *bulleted list* er in.

2.9.2 Een CSS driehoek

Hoe maak je een driehoek met CSS? Je mag hiervoor een testbestandje openen. Maak de volgende HTML aan:

```
<p>Een CSS driehoek <span class="r_driehoek"></span> is gemakkelijk en een <span class="l_driehoek"></span> ook</p>
```

Het principe is het volgende: neem een leeg element met **width** en **height 0** en geef het een dikke, transparante boven- en onderrand en één gekleurde, dikke zijrand.

Maak een **style** element en plaats er volgende css in:

```
.r_driehoek{
    display: block;
```

```
width: 0;
height: 0;
border-top: 30px solid transparent;
border-bottom: 30px solid transparent;
border-left: 30px solid red;
}
```

Je hebt nu een rode driehoek die naar rechts wijst. Omdat alle border diktes gelijk zijn , is het ook een rechthoekige driehoek. Verhoog de dikte van de border-left en de pijl wordt langer.

Maak nu ook eens een driehoek aan die naar links wijst:

```
.l_driehoek{
display: block;
width: 0;
height: 0;
border-top: 30px solid transparent;
border-bottom: 30px solid transparent;
border-right: 70px solid red;
}
```

Dergelijke driehoeken zullen we gebruiken in het *breadcrumb trail*.

2.9.3 Het kruimeltjespoor

We starten van een horizontaal menu zoals we er al verschillende maakten:

```
.breadcrumbs {
list-style: none;
overflow: hidden;
margin:0;
padding:0;
}
.breadcrumbs li {
display:inline;
}
.breadcrumbs li a {
position: relative;
display: block;
float: left;
background:#E0ECF4;
text-decoration: none;
line-height: 40px;
padding-left: 30px;
}
.breadcrumbs li:first-child a{
padding-left: 10px
}
```

Om de driehoeken aan te maken gebruiken we de `:after` en `:before` selectors die **pseudo-elementen** produceren (*css generated content*). Op die manier moeten we geen extra, lege, html-elementen in onze menu-items zetten.

Eerst een gekleurd driehoekje:

```
.breadcrumbs li a:after {  
    content: " ";  
    display: block;  
    width: 0;  
    height: 0;  
    border-top: 20px solid transparent;  
    border-bottom: 20px solid transparent;  
    border-left: 20px solid #e0ecf4;  
    position: absolute;  
    top: 50%;  
    margin-top: -20px;  
    left: 100%;  
    z-index: 2;  
}
```

Bespreking:

- *achter* elk `a` element creëren we een pseudo-element met een spatie als inhoud, een block-element met een hoogte en breedte van `0`, maar
- met een 20px dikke, volle, doorzichtige boven- en onderrand
- en een gekleurde 20px dikke linkerrand
- die absoluut geplaatst is, en verticaal gecentreerd t.o.v. het `a` element (`top` en `margin-top`)
- de `left:100%` zorgt ervoor dat het element helemaal rechts van het `a` element komt te staan , of het nu via `:after` of `:before` geproduceerd wordt
- de `z-index` zorgt ervoor dat dit gekleurde driehoekje boven het witte driehoekje komt te staan

Nu een wit driehoekje erachter:

```
.breadcrumbs li a:before {  
    content: " ";  
    display: block;  
    width: 0;  
    height: 0;  
    border-top: 23px solid transparent;  
    border-bottom: 23px solid transparent;  
    border-left: 23px solid white;  
    position: absolute;  
    top: 50%;  
    margin-top: -23px;  
    margin-left: 0px;  
    left: 100%;  
    z-index: 1;  
}
```

Bespreking:

- is bijna identiek met uitzondering van de **z-index** die lager is: komt achter de vorige en
- de diktes die 3 pixels meer zijn waardoor het driehoekje wat groter is en geen gekleurde pixels doorlaat aan de hoeken: de **margin-left** moet daarom gecompenseerd worden
- de **overflow:hidden** van de container zorgt ervoor dat dit grotere driehoekje ook niet te zien zal zijn tegenover een gekleurde achtergrond

Tenslotte kleuren we de actieve delen:

```
.breadcrumbs .actief a{  
background:#005689;  
color:#fff;  
}  
.breadcrumbs .actief a:after {  
border-left-color:#005689;  
}
```

Bespreking:

- een **li** element met de **class actief** geeft aan zijn **a** element een andere achtergrond- en tekstkleur
- het driehoekje krijgt ook een andere kleur: dat komt neer op de **border-color** van de linkerrand

Je kan dus nu perfect meerdere **li** elementen de **class actief** geven, dan zie je wat we voor ogen hadden.

Het toekennen van de class **actief** zal normaal door scripting gebeuren: ofwel serverside (.Net, Java, PHP) of client-side (Javascript); dat kunnen we niet regelen via HTML of CSS, we kunnen enkel de style voorzien.

Om hier geen scripting te moeten gebruiken hebben we onze oefening opgesplitst in aparte pagina's *abonneren.html*, *gegevens.html*, *bevestig.html*, *betalen.html*.

Zo, kruimeltjesspoor klaar!

2.10 Een dropdown menu, enkel met CSS

Doel

Hoe maak je een horizontaal dropdown menu vanuit een geneste lijst, enkel met CSS en zonder Javascript?

In dit project leer je de lijst omvormen tot een horizontale structuur en gebruik maken van relatieve, absolute positionering en **visibility** om de menu's te tonen en te verbergen. Je hebt een degelijke kennis nodig van geneste CSS selectors om te begrijpen wat we gaan doen.

Als extra gebruiken we ook enkele CSS3 features.

Theorie

- het boxmodel
- relatieve en absolute positionering
- **visibility**
- CSS selectors
- CSS3 gradients, box-shadow, transitions

Startbestanden

desertLife.html, desertlife.css, normalize.css

2.10.1 Een geneste lijst als startpunt

Het startbestand *desertlife.html* bevat een geneste lijst die we zullen omvormen naar een menu:

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>DesertLife</title>
<link rel="stylesheet" href="css/desertlife.css" />
<style>
/*lokale styles */
</style>
</head>
<body>
<div class="container">
<header>
<h1>Cactusvereniging</h1>
<h2>DesertLife</h2>
<nav>
<ul>
<li><a href='desertlife.html' title='Welkom bij DesertLife'>Welkom</a>
```

```
<ul>
    <li><a href='wiewat.html' title='Wie is wie bij DesertLife en wat doen we?'>Wie zijn we?</a></li>
    <li><a href='lid.html' title='Interesse om lid te worden?'>lid worden</a></li>
    <li><a href='bib.html' title='Welke boeken zijn er in onze bibliotheek?'>bibliotheek</a></li>
    <li><a href='reizen.html' title='Welke reisjes en reizen deden we en zijn er gepland?'>reizen</a></li>
    <li><a href='media.html' title="Foto's en video's van het verenigingsleven">foto en video</a></li>
</ul>
</li>
...

```

2.10.2 Normalize

Er is ook een stylesheet *desertlife.css*: dat bevat de algemene styles voor de pagina. We hebben het niet echt nodig voor ons menu, maar interessant is wel dat het een **@import** doet van *normalize.css*.

Dit is een stylesheet dat een aantal CSS eigenschappen gelijkstelt voor alle browsers. Je kan de laatste versie downloaden op het net op <http://necolas.github.com/normalize.css/>

Het is een goed startpunt, zo krijg je geen CSS verrassingen in verschillende browsers.

2.10.3 De lijst omvormen naar menu

We starten met de niet-opgemaakte lijst. Bekijk de geneste lijst in het **nav** element.



We maken nog twee aanpassingen in de HTML:

- geef het `ul` element de `class` "hor_dropdown_menu"
- geef het eerste `li` element ("Welkom") de `class="actief"`

```
<ul class='hor_dropdown_menu'>
    <li class="actief"><a href='desertlife.html' title='Welkom bij
DesertLife'>Welkom</a>
        <ul>
            <li><a href='wiewat.html' title='Wie is wie bij DesertLife en wat doen
we?'>Wie zijn we?</a></li>
        ...
    ...

```

Dit hebben we natuurlijk nodig om zowel het menu zelf als het actieve link element te kunnen aanduiden.

2.10.4 Een horizontaal menu

Voeg nu eerst een tweede `link` element in in de `head` van het document:

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

Maak dan het nieuwe stylesheet `desertlife_menu.css` aan.

Eerst passen we wat algemene opmaak toe op het menu. Voeg toe aan het stylesheet:

```
@charset "utf-8";
/* desertlife_menu.css dropdownmenu
een horizontaal dropdown menu, enkel met CSS3
*/
/* hoofdmenu styles */
.hor_dropdown_menu {
    font-family: Verdana, Geneva, sans-serif;
    font-size: 1em; /*16px*/
    display: inline-block;
    margin: 0 auto;
    padding: 0;
    width: 100%;
    background: #e570e7;
}
```

Bespreking:

- via de `class` selector stellen we het lettertype in en zetten ook de lettergrootte op 'normaal': 16px. Dus 1em = 16px
- de `display:inline-block` zal in deze oefening niet verschillen van `block`. Enkel wanneer je op dezelfde lijn als het menu nog iets anders zou willen plaatsen is dit belangrijk
- de `margin:0 auto` zorgt er hier voor dat het menu horizontaal gecentreerd zal zijn in zijn container. Je zult er ook niet veel van merken want het `nav` element is zo breed als de volledige pagina.
- de `padding` wordt op nul gezet om alles mooi dicht op elkaar te krijgen
- de `width` wordt op 100% gezet om de volledige breedte van zijn container in te nemen. Dit betekent dat het `nav` element de breedte bepaalt
- de kleur is willekeurig: we stellen ze hier in om het `ul` element zichtbaar te maken

Momenteel is het resultaat nog zeer gelijkend op de normale lijst. Alle sgedraagd zich nog als een bolletjeslijst.

We voegen volgende CSS toe:

```
/*hoofdniveau*/
.hor_dropdown_menu li {
    list-style:none;
    float:left;
    position:relative;
    margin:0;
}
.hor_dropdown_menu a {
    display:block;
    text-decoration:none;
    padding:0.6em 1.25em;
}
```

Hier staan een paar zeer belangrijke CSS rules in:

- de selector `.hor_dropdown_menu li` geldt voor **alle** `li` elementen in dit menu:
 - ze verliezen hun status van 'bolletje' met `list-style:none`
 - ze worden links ge-`float` en stapelen dus horizontaal naast elkaar
 - ze krijgen een relatieve positionering: dit heeft geen effect op hun opmaak, maar betekent dat ze als ankerpunt dienen voor hun `children` die straks ook een position krijgen
 - hun `margin` wordt op nul gezet
- de selector `.hor_dropdown_menu a` geldt voor **alle** `a` elementen in dit menu:
 - ze worden blokjes met `display:block`
 - alle `text-decoration` wordt verwijderd
 - ze krijgen wat `padding`

Het resultaat ziet er op het eerste gezicht erg chaotisch uit, maar is toch fundamenteel anders want alle `li>a` elementen gedragen zich als blokjes:



2.10.5 De submenu's

We voegen opnieuw toe:

```
/* submenus */
.hor_dropdown_menu ul {
    position: absolute;
    padding: 0;
    width: 12em;
    background: #D566D5;
}

.hor_dropdown_menu ul li {
    float:none;
    margin:0;
}
```

Bespreking:

- de selector `.hor_dropdown_menu ul` geldt voor **alle** `ul` elementen **binnen** het hoofdmenu (dat zelf een `ul` is, maar voor wie dit niet geldt):
 - ze krijgen een absolute positionering. Aangezien ze zelf een *child* van een `li` zijn (die net een relatieve positionering gekregen heeft) is dat ten opzichte van hun onmiddellijke `li` parent
 - geen `padding`
 - een vaste `width` van 12em = 192px
 - en een achtergrondkleur
- de selector `.hor_dropdown_menu ul li` geldt voor **alle** `li` elementen **binnen** een submenu
 - ze verliezen hun `float`, maar aangezien ze ook geen 'bolletje' meer zijn (door de Cascade) zijn ze nu gewone blokjes, verticaal gestapeld
 - hun `margin` wordt expliciet op nul gezet

Het resultaat ziet er zo uit:



Je ziet dat de submenu's nu onder hun `li` element hangen. De subsubmenu's zijn nog niet goed: ze liggen gewoon bovenop de submenu's. Dat wijzigen we eerst. Voeg toe onderaan:

```
/* subsubmenu*/
.hor_dropdown_menu ul ul{
  top:0.6em;
  left:13em;
}
```

Bespreking:

- alle submenu's – ook *subsub*, *subsubsub*, etc – zijn absoluut geplaatst, maar omdat ze tot nu toe geen `top`, `bottom`, `left` of `right` gekregen hebben blijven ze op hun normale positie t.o.v. van hun *parent* staan
- hier voegen we een plaat verschuiving toe, iets naar onder en een heel stuk naar rechts voor alle submenu's van het tweede niveau en alle verdere niveau's

Nu we weten dat de positionering OK is, **verbergen** we de submenu's. Wijzig de bovenstaande selector:

```
.hor_dropdown_menu ul {
  position: absolute;
  opacity: 0;
  visibility: hidden;
  padding: 0;
  width: 12em;
  background: #D566D5;
}
```

Bespreking:

- ze worden verborgen met `visibility: hidden`
- ze krijgen een `opacity: 0`, dus volledig transparant en dus ook onzichtbaar



Welkom Kalender Cactusdokter Zoekertjes Succulenten Meer

2.10.6 Mouse-over

Nu komt het belangrijkste van allemaal: we moeten de submenu's tevoorschijn brengen als de muis over een item *hovert*, bij een **mouse-over** event dus. Voeg deze CSS toe helemaal onderaan:

```
/* drop down effect voor alle menus */

.hor_dropdown_menu li:hover > ul {
    opacity:1;
    visibility:visible;
}
```

Bespreking:

- de selector `.hor_dropdown_menu li:hover > ul` wordt toegepast op:
een `ul` element dat een onmiddellijk *child* is van een `li` element in de **hover** state (pseudo-class)
met andere woorden dit geldt voor alle submenu's van welk niveau ook, als er over zijn parent `li` element een **mouse-over** plaatsheeft.
- de **opacity** wordt op **1** gezet (niet transparant)
- de **visibility** op **visible**

Probeer maar even: de menu's worden getoond. Je bemerkt dat we hiervoor geen Javascript nodig hebben. Er zijn echter nog problemen.

2.10.7 Opmaken met CSS3 gradients

Eerst maken we het menu wat mooier met een CSS3 *gradient*.

Omdat deze eigenschap nogal complex is én browser-prefixes verlangt, zullen we die laten aanmaken door de CSS generator *Colorzilla*:

<http://www.colorzilla.com/gradient-editor/>

Je mag elke kleurovergang kiezen die je wil, maar in ons voorbeeld kozen we er eentje met slechts twee kleurstops, in een roze tint. Denk er ook aan de gewone **background-color** in een gelijkaardige kleur te zetten.

Kies je *gradient* en kopieer dan de CSS in je stylesheet op de relevante plaats.

Eerst doen we het menu zelf. Voeg toe aan de selector `.hor_dropdown_menu`

```
/* hoofdmenu styles */
.hor_dropdown_menu {
    font-family:Verdana, Geneva, sans-serif;
    font-size:1em;                                     /*16px*/
    display:inline-block;
    margin:0 auto;
    padding:0;
    width:100%;

    background: #e570e7;
    background: -moz-linear-gradient(top, #e570e7 0%, #c85ec7 47%, #a849a3 100%);
    background: -webkit-gradient(linear, left top, left bottom, color-stop(0%, #e570e7),
    color-stop(47%, #c85ec7), color-stop(100%, #a849a3));
    background: -webkit-linear-gradient(top, #e570e7 0%, #c85ec7 47%, #a849a3 100%);
    background: -o-linear-gradient(top, #e570e7 0%, #c85ec7 47%, #a849a3 100%);
    background: -ms-linear-gradient(top, #e570e7 0%, #c85ec7 47%, #a849a3 100%);
    background: linear-gradient(to bottom, #e570e7 0%, #c85ec7 47%, #a849a3 100%);
    filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#e570e7',
    endColorstr='#a849a3', GradientType=0 );

}
```

Dat ziet er al beter uit, pas nu ook de tekstkleur van de hyperlinks aan naar wit:

```
.hor_dropdown_menu a {
    display:block;
    color:white;
    text-decoration:none;
    padding:0.6em 1.25em;
}
```

Nu hebben we nog een accent nodig voor het hover-effect en voor het geselecteerde menu-item (die de class "actief" kreeg).

We maken een selector bij en gebruiken daarvoor een gelijkaardige gradient die **iets lichter getint** is:

```
/* geselecteerd menu element en hovereffect */

.hor_dropdown_menu .actief > a, .hor_dropdown_menu li:hover > a {

    background: #fb83fa;
    background: -moz-linear-gradient(top, #fb83fa 0%, #e93cec 100%);
    background: -webkit-gradient(linear, left top, left bottom, color-stop(0%, #fb83fa),
    color-stop(100%, #e93cec));
    background: -webkit-linear-gradient(top, #fb83fa 0%, #e93cec 100%);
    background: -o-linear-gradient(top, #fb83fa 0%, #e93cec 100%);
    background: -ms-linear-gradient(top, #fb83fa 0%, #e93cec 100%);
    background: linear-gradient(to bottom, #fb83fa 0%, #e93cec 100%);
```

```
filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#fb83fa',
endColorstr='#e93cec', GradientType=0 );
}
```



Deze selectors worden toegepast op de `a` elementen van het actieve `li` element en van de `li:hover`;

Je merkt dat ook de submenu's de lichtere gradient tonen bij `hover`.

2.10.8 Slagschaduw

Om het nog wat mooier te maken passen we een slagschaduw toe met de CSS3 eigenschap `box-shadow`.

Pas de `.hor_dropdown_menu` selector aan:

```
.hor_dropdown_menu {
    font-family:Verdana, Geneva, sans-serif;
    font-size:1em;                                         /*16px*/
    display:inline-block;
    margin:0 auto;
    padding:0;
    width:100%;

    background: #e570e7;
    background: -moz-linear-gradient(top, #e570e7 0%, #c85ec7 47%, #a849a3 100%);
    background: -webkit-gradient(linear, left top, left bottom, color-stop(0%, #e570e7),
color-stop(47%, #c85ec7), color-stop(100%, #a849a3));
    background: -webkit-linear-gradient(top, #e570e7 0%, #c85ec7 47%, #a849a3 100%);
    background: -o-linear-gradient(top, #e570e7 0%, #c85ec7 47%, #a849a3 100%);
    background: -ms-linear-gradient(top, #e570e7 0%, #c85ec7 47%, #a849a3 100%);
    background: linear-gradient(to bottom, #e570e7 0%, #c85ec7 47%, #a849a3 100%);
    filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#e570e7',
endColorstr='#a849a3', GradientType=0 );

    box-shadow: 0 0.25em 0.375em rgba(0,0,0, .5);
    -moz-box-shadow: 0 0.25em 0.375em rgba(0,0,0, .5);
    -webkit-box-shadow: 0 0.25em 0.375em rgba(0,0,0, .5);
}
```

Bespreking:

- de eigenschap **box-shadow** brengt hier een grijze, diffuse schaduw op het hoofd **ul** element aan.

We doen precies hetzelfde voor de submenu's:

```
.hor_dropdown_menu ul {  
    position: absolute;  
    opacity: 0;  
    visibility: hidden;  
    padding: 0;  
    width: 12em;  
    background: #D566D5;  
  
    box-shadow: 0 0.25em 0.375em rgba(0,0,0, .5);  
    -moz-box-shadow: 0 0.25em 0.375em rgba(0,0,0, .5);  
    -webkit-box-shadow: 0 0.25em 0.375em rgba(0,0,0, .5);  
  
}
```



2.10.9 CSS3 transitions

Als je de CSS theorie erop naleest dan weet je dat CSS3 **Transitions** overgangen zijn van één waarde van een eigenschap naar een andere waarde, een 'effect' dus. Dat kan een beweging veroorzaken of een gradueel tonen van iets, etc...

Maar **Transitions** zijn gebonden aan een wijziging van een CSS state of een **class** wissel, in tegenstelling tot **Animations** die onmiddellijk werken of gestart worden vanuit een script.

Wij tonen/verbergen onze menu's via de **:hover** pseudoclass. Dat kunnen we gebruiken voor een transition.

Opnieuw passen we de selector voor de submenu's aan:

```
.hor_dropdown_menu ul {  
    position: absolute;
```

```
opacity:0;
visibility:hidden;
padding:0;
width:12em;
background: #D566D5;

box-shadow:0 0.25em 0.375em rgba(0,0,0, .5);
-moz-box-shadow:0 0.25em 0.375em rgba(0,0,0, .5);
-webkit-box-shadow:0 0.25em 0.375em rgba(0,0,0, .5);

/* transition */
-moz-transition:opacity .25s linear, visibility .1s linear .1s;
-webkit-transition:opacity 0.25s linear, visibility .1s linear .1s;
-o-transition:opacity .25s linear, visibility .1s linear .1s;
transition:opacity .25s linear, visibility .1s linear .1s;
}
```

Nu bemerk je een effect bij het tonen van de menu's.

Bespreking:

- ook hier zijn browser-prefixes nodig
- de **transition** eigenschap is de verkorte schrijfwijze voor

transition-property || transition-duration || transition-timing-function || transition-delay

- we hebben twee properties in de **transition** staan: **opacity** en **visibility**
Bij de eerste is de syntax

opacity .5s linear

Dus een overgang voor opacity (van 0 nr 1) in 0.5seconden volgens een linear functie

Voor visibility is de syntax

visibility .1s linear .1s

bijna hetzelfde maar met een lichte vertraging van 0.1seconde

Alles werkt naar behoren maar de submenu's van het derde niveau zijn nog onbereikbaar want er zit een afgrondje tussen... Dat hebben we express gedaan voor nog een ander effectje. Voeg deze selector toe:

```
.hor_dropdown_menu ul li:hover > ul{
    /*enkel nodig voor extra transition effectje subsub: beweging left*/
    left:11em;
}
```

Daarmee plaatsen we het subsubmenu op de juiste plaats (en is het probleempje opgelost), maar nu voegen we nog een **transition** toe voor de subsubmenu's:

```
/* subsubmenu*/
.hor_dropdown_menu ul ul{
    top:0.6em;
    left:13em;
    /* transition */
    -moz-transition:opacity .25s linear, visibility .1s linear .1s, left .25s linear;
    -webkit-transition:opacity .25s linear, visibility .1s linear .1s, left .25s linear;
    -o-transition:opacity .25s linear, visibility .1s linear .1s, left .25s linear;
    transition:opacity .25s linear, visibility .1s linear .1s, left .25s linear;
}
```

en krijg je een mooi bewegingseffectje op de sub's van het derde niveau (en dieper).

Hier hebben we de **left** property er ook in betrokken en beweegt het menu tussen een **left** afstand van 13em tot 11em.

Taken:

maak nu volgende taken

- "Webontwikkelaar, deel 2"

2.11 Bootstrap: een CSS Framework

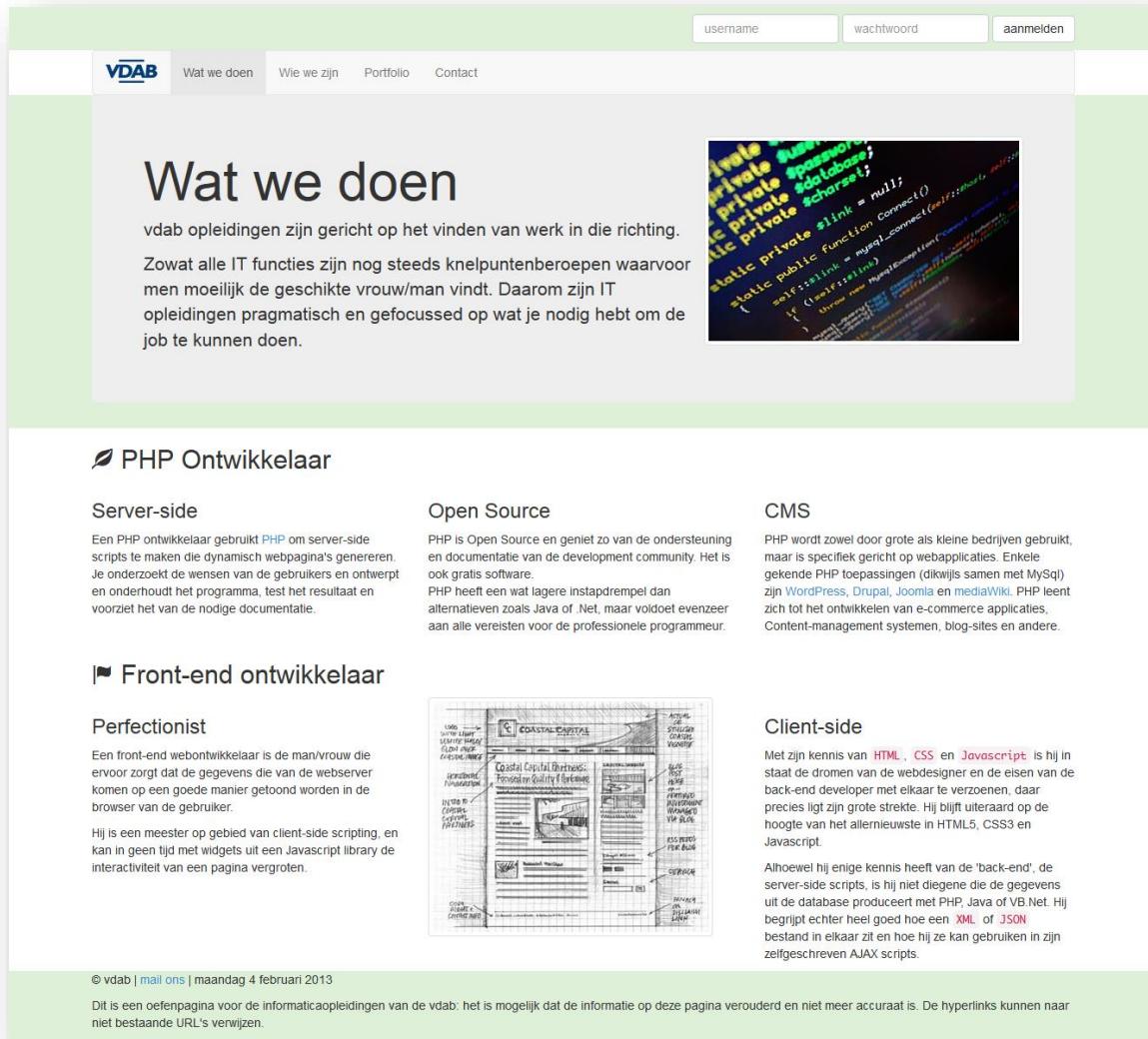
(2014 geupdate voor Bootstrap v3)

Doel

Een CSS *Framework* (hier **Bootstrap**) leren gebruiken om websites op te maken. We gebruiken zoveel mogelijk de framework classes en zo weinig mogelijk eigen CSS.

Dit stelt ons in staat om **sneller** een *mockup* van een pagina te maken die we aan de klant kunnen laten zien voor feedback.

Dit is wat we willen bereiken op een breed scherm:



The screenshot shows a responsive website layout. At the top, there's a header with the VDAB logo, a search bar with fields for 'username' and 'wachtwoord' (password), and a 'aanmelden' (login) button. Below the header, a navigation menu includes 'Wat we doen' (which is highlighted in blue), 'Wie we zijn', 'Portfolio', and 'Contact'. The main content area has a light green background. On the left, there's a large image of a person working at a computer with code visible on the screen. To the right of the image, the heading 'Wat we doen' is displayed in a large, bold, dark font. Below the heading is a paragraph of text. Further down, there are three columns: 'Server-side' (PHP Ontwikkelaar), 'Open Source' (with a sub-section for 'Perfectionist'), and 'CMS'. At the bottom of the page, there's a footer with copyright information and a note about the page being an oefenpagina.

En op een klein scherm:

The screenshot shows a web page with a header containing a logo and a search bar. Below the header, there's a login form with fields for 'username' and 'wachtwoord' (password), and a 'aanmelden' (login) button. The main content area has a green header with the 'vdab' logo. It features two main sections: 'Wat we doen' (What we do) and 'PHP Ontwikkelaar' (PHP Developer). The 'Wat we doen' section contains text about IT training being pragmatic and focused on practical skills. The 'PHP Ontwikkelaar' section is divided into 'Server-side' and 'Open Source' parts, both with descriptive text. Below this, there's a 'CMS' section and a 'Front-end ontwikkelaar' (Front-end developer) section, which includes a 'Perfectionist' subsection and a 'Client-side' subsection. At the bottom of the page, there's a footer with copyright information and a note about the page being an old version.

username
wachtwoord
aanmelden

Wat we doen

vdab opleidingen zijn gericht op het vinden van werk in die richting.

Zowat alle IT functies zijn nog steeds knelpuntenberoepen waarvoor men moeilijk de geschikte vrouw/man vindt. Daarom zijn IT opleidingen pragmatisch en gefocussed op wat je nodig hebt om de job te kunnen doen.

PHP Ontwikkelaar

Server-side

Een PHP ontwikkelaar gebruikt [PHP](#) om server-side scripts te maken die dynamisch webpagina's genereren. Je onderzoekt de wensen van de gebruikers en ontwerpt en onderhoudt het programma, test het resultaat en voorziet het van de nodige documentatie.

Open Source

PHP is Open Source en geniet zo van de ondersteuning en documentatie van de development community. Het is ook gratis software. PHP heeft een wat lagere instapdrempel dan alternatieven zoals Java of .Net, maar voldoet evenzeer aan alle vereisten voor de professionele programmeur.

CMS

PHP wordt zowel door grote als kleine bedrijven gebruikt, maar is specifiek gericht op webapplicaties. Enkele bekende PHP toepassingen (dikwijls samen met MySql) zijn [WordPress](#), [Drupal](#), [Joomla](#) en [mediaWiki](#). PHP leent zich tot het ontwikkelen van e-commerce applicaties, Content-management systemen, blog-sites en andere.

Front-end ontwikkelaar

Perfectionist

Een front-end webontwikkelaar is de man/vrouw die ervoor zorgt dat de gegevens die van de webserver komen op een goede manier getoond worden in de browser van de gebruiker.

Hij is een meester op gebied van client-side scripting, en kan in geen tijd met widgets uit een Javascript library de interactiviteit van een pagina vergroten.

Client-side

Met zijn kennis van [HTML](#), [CSS](#) en [Javascript](#) is hij in staat de dromen van de webdesigner en de eisen van de back-end developer met elkaar te verzoenen, daar precies ligt zijn grote stukje. Hij blijft uiteraard op de hoogte van het allernieuwste in HTML5, CSS3 en Javascript.

Alhoewel hij enige kennis heeft van de 'back-end', de server-side scripts, is hij niet diegene die de gegevens uit de database produceert met PHP, Java of VB.NET. Hij begrijpt echter heel goed hoe een [XML](#) of [JSON](#) bestand in elkaar zit en hoe hij ze kan gebruiken in zijn zelfgeschreven AJAX scripts.

© vdab | [mail ons](#) | maandag 4 februari 2013

Dit is een oefenpagina voor de informaticaopleidingen van de vdab: het is mogelijk dat de informatie op deze pagina verouderd en niet meer accuraat is. De hyperlinks kunnen naar niet bestaande URL's verwijzen.

Theorie

- HTML structuur aanbrengen
- CSS algemeen
- kijk ook de Bootstrap documentatie na

Startbestanden

webontwikkelaar_wat.html, images

2.11.1 Bootstrap downloaden

Bootstrap is een *CSS Framework* waarmee je zeer veel tijd kunt besparen op gebied van CSS: je hoeft enkel de Bootstrap classes te gebruiken en je hebt een mooie lay-out, die *Mobile First* en altijd *Responsive* is (v3).

Dit is exact wat we in dit project zullen doen: snel een lay-out maken met het CSS Framework zonder veel zelf CSS te schrijven (maar... voor kleine aanpassingen moet je soms zelf ingrijpen)

Wil je zo'n framework ook gebruiken als basis voor je eigen design dan kan dat zeker, maar dan zal je je moeten verdiepen in complexere CSS en ook leren de CSS preprocessor LESS gebruiken.

Om deze tekst wat in te korten gebruiken we vanaf nu de afkorting "BS" voor *Bootstrap*.

Eerst downloaden we BS: ga naar <http://getbootstrap.com>

Download de standaard versie.

Pak die uit in je map *projecten*

```
—projecten
    |—bootstrap
        |—css
        |—fonts
        |—js
            |—css (eigen)
            |—img (eigen)
            |—(andere projectbestanden)
```

De map *css* bevat een leesbare en een geminimizede versie van de gecompileerde Bootstrap stylesheet én van het BS theme zelf.

De map *fonts* bevat de ikoonlettertypes van glyphicon (nu inclusief).

De map *js* bevat Javascript plugins. Deze plugins zijn jQuery dependant.

Je kan ook een *Customized versie* samenstellen en downloaden vanaf de Customize menu optie.

2.11.2 Structuur aanbrengen

Open nu het startbestand `webontwikkelaar_wat.html` en sla het onmiddellijk op als `webontwikkelaar_wat_bs.html`.

Om deze pagina te lay-outen moeten we er meer structuur in aanbrengen. Enerzijds om de inhoud af te bakenen maar terzelfdertijd om er opmaak te kunnen op toepassen.

De lay-out die we voor ogen hebben (zie hierboven) bestaat uit een aantal brede stroken met daarin de inhoud.

We voegen een `header`, een `footer` en een aantal `section` elementen toe, de `section`'s geven we een `id` mee:

```
<body>
<header>
    <form method="get" action="/search">
        ...
    </form>
    <!--einde login-->
</header>
<section id="intro">
    <div id="logo">...</div>
    <!--einde hoofdmenu-->
</section>
<section id="wat">
    ...
    <h1>Wat we doen</h1>
    ...
</section>
<section id="opleidingen" >
    <h2>PHP Ontwikkelaar</h2>
    <article>
        <h3>Server-side</h3>
        ...
    </article>
    <article>
        <h3>Open Source</h3>
        ...
    </article>
    <article>
        <h3>CMS</h3>
        ...
    </article>
    <h2>Front-end ontwikkelaar</h2>
    <article>
        <h3>Perfectionist</h3>
        ...
    </article>
    <article>
        <h3>Client-side</h3>
        ...
    </article>
</section>
```

```
</section>
<footer>
  ...
</footer>
</body>
</html>
```

Ook zonder BS zouden we een dergelijke structuur gebruiken, het levert geen enkele opmaak op.

2.11.3 Bootstrap koppelen en basis template

Voor we verder doen koppelen we het BS stylesheet en de JS lib.

We volgen het template dat voorgesteld wordt op de GETTING STARTED pagina:

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Webontwikkelaar taak</title>
    <link rel="stylesheet" href="bootstrap/css/bootstrap.min.css">
    ...
    ...
    <script src="http://code.jquery.com/jquery.js"></script>
    <script src="js/bootstrap.min.js"></script>
  </body>
</html>
```

Bemerk

- dat we de *minified* versies gebruiken: die zijn efficiënter
- dat jQuery geladen wordt via de jQuery CDN, je kan uiteraard ook een lokale kopie van jQuery gebruiken.

Omdat de lay-out van de nieuwe Bootstrap versie standaard *Responsive* is (dus 'flexibel' op alle toestellen), moeten we nog een **meta** tag toevoegen aan de pagina.

Voeg toe na de eerste **meta** tag:

```
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Webontwikkelaar taak</title>
```

Deze zorgt voor een oorspronkelijke zoom-factor van 100% op een mobiel toestel.

2.11.4 Een centrale container

De **header**, **footer** en **section**'s zullen fungeren als horizontale stroken, om de werkelijke inhoud te beperken in breedte plaatsen we in elk van hen een extra **div** en geven die de **class** "container":

```
<body>
```

```
<header>
<div class="container">
...
</div>
</header>
<section id="intro">
<div class="container">
...
</div>
</section>
<section id="wat">
<div class="container">
...
</div>
</section>
<section id="opleidingen" >
<div class="container">
...
</div>
</section>
<footer>
<div class="container">
...
</div>
</footer>
</body>
</html>
```

De BS class `.container` zorgt onmiddellijk voor een 'kolom' die gecentreerd is en beperkt in breedte. De breedte die je nu krijgt hangt af van het scherm waarop je kijkt: het is **Responsive!**

Voor een klein scherm (onder 768px breed) is de breedte 100%. Daarboven zijn er drie vaste breedtes afhankelijk van het scherm: 750px, 970px en 1170px. Dit gebeurt met **media-queries**. Je kan dit zelf nakijken via een webdeveloper tool:

```
Styles Computed Event Listeners DOM Breakpoints Properties
element.style {
}
@media (min-width: 1200px)
.container {
    width: 1170px;
}
@media (min-width: 992px)
.container {
    width: 970px;
}
@media (min-width: 768px)
.container {
    width: 750px;
}
.container {
    padding-right: 15px;
    padding-left: 15px;
    margin-right: auto;
    margin-left: auto;
}
```

In een volgend project maken we zelf een Responsive design met media-queries.

In een volgende stap willen we visueel onderscheid maken tussen de *stroken*: het idee is alternerend een witte en een gekleurde strook te hebben.

Dit is een van de weinige styles die we zelf zullen toepassen via een eigen stylesheet:

We voegen een extra koppeling in (**onder** de BS koppeling):

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

en maken een eigen nieuw stylesheet *style.css*.

Daar voegen we deze eigen css aan toe:

```
section:nth-child(odd), header, footer {
    background-color: #dff0d8;
    border-top: #d6e9c6;
    border-bottom: #d6e9c6;
}
```

Opmerkingen:

- De **nth-child(odd)** selector selecteert alle oneven **section** elementen
- als je een eigen stylesheet koppelt, plaats deze dan steeds **na** de BS koppeling
- de **header**, de **footer** en elke alternerende **section** van dezelfde parent zullen lichtgroen gekleurd worden met een boven- en onderrand

2.11.5 een jumbotron

De eerste tekst op een website is erg belangrijk: de boodschap moet goed overkomen! We gebruiken daarvoor de BS class **.jumbotron**. Voeg een extra **div** toe binnen de **.container** en geef de **img** een class **.pull-right**:

```
<section id="wat" >
<div class="container">
```

```
<div class="jumbotron">
  <h1>Wat we doen</h1>
  <p>vdab opleidingen zijn gericht op het vinden van werk in die richting.</p>
  <p>Zowat alle IT functies zijn nog steeds knelpuntenberoepen waarvoor men moeilijk
  de geschikte vrouw/man vindt. Daarom zijn IT opleidingen pragmatisch en gefocussed op
  wat je nodig hebt om de job te kunnen doen.</p>
</div>
</div>
</section>
```

Het resultaat is een "jumbotron" waar het beeldje rechts gefloat werd. De BS classes `.pull-1` en `.pull-right` zijn shortcuts voor een linkse of rechtse `float`.

2.11.6 Het Bootstrap grid

Net zoals zoveel CSS frameworks bevat BS een **grid** (raster) systeem van classes waarmee je een container in **rijen** en **kolommen** kunt opdelen.

Het BS grid bestaat standaard uit 12 *responsive, fluid* kolommen.

Het werkt eenvoudig:

- Je plaatst alles binnen een `.container` class
- Je maakt één of meer rijen door `div`'s (of een ander element) een class `.row` te geven
- Binnenin één `.row` maak je kolommen door `div`'s (of andere elementen) een class `.col-lg-n`, `.col-md-n`, `.col-sm-n` of `.col-xs-n` te geven.

Daarbij moet je weten dat

- *n* staat voor het aantal *twaalfden* dat een kolom breed moet zijn. Bijvoorbeeld `.col-md-4` is 4/12 breed. Met 3 dergelijke kolommen vul je de volledige breedte.
- Er zijn vier types kolommen die je naar believen kan gebruiken. Ze hebben elk een maximumbreedte (voor n=1) in pixels en een *breakpoint*: als een schermbreedte onder dit *breakpoint* valt, stapelen ze verticaal, anders staan ze horizontaal naast elkaar als gewone kolommen.
 - `.col-lg-n`: voor "*Large devices*": breakpoint 1170px, max-width 95px
 - `.col-md-n`: voor "*Medium devices*": breakpoint 750px, max-width 60px
 - `.col-sm-n`: voor "*Small devices*": breakpoint 970px, max-width 78px
 - `.col-xs-n`: voor "*Extra small devices*": geen breakpoint (altijd verticaal gestapeld), `width auto`

Het is belangrijk te beseffen dat je dus op elk toestel eerder welke soort kolom kunt gebruiken, alleen, de één is er meer toepasselijk dan de ander.

Zo zal een reeks `.col-md-n` kolommen zich als normale kolommen naast elkaar gedragen op een desktop én op een tablet, maar als de schermbreedte kleiner dan 750px wordt (bv. als je de tablet verticaal houdt), stapelen ze zich boven op elkaar.

Combinaties zijn ook mogelijk, maar dat houden we voor later.

Nu willen we de twee opleidingen lay-outen als een rij met drie kolommen.

Daarvoor

- voegen we een `div class="row"` element toe rond elke opleiding en
- plaatsen een `class="col-md-4"` in elk `article` element.
- voor het `img` element dat we bij *Front-end ontwikkelaar* als tweede kolom willen, maken we een extra `div class="col-md-4"` element:

```
<section id="opleidingen">
<div class="container">
<h2>PHP Ontwikkelaar</h2>
<div class="row">
    <article class="col-md-4">
        <h3>Server-side</h3>
        ...
    </article>
    <article class="col-md-4">
        <h3>Open Source</h3>
        ...
    </article>
    <article class="col-md-4">
        <h3>CMS</h3>
        ...
    </article>
</div>
<h2>Front-end ontwikkelaar</h2>
<div class="row">
    <article class="span4">
        <h3>Perfectionist</h3>
        ...
    </article>
    <div class="col-md-4">  </div>
    <article class="col-md-4">
        <h3>Client-side</h3>
        ...
    </article>
</div>
</div>
</section>
```

Hiermee bereiken we een totaal van 12 kolommen.

Merk op dat de **h2** koppen "PHP ontwikkelaar" en "Front-end ontwikkelaar" als *children* van de **.container** staan en dus als *siblings* van de **.row** elementen zijn. Dit is perfect geldig:

```
<div class="container">
<h2>PHP Ontwikkelaar</h2>
<div class="row">
  <article class="col-md-4">
    <h3>Server-side</h3>
```

De volgende structuur is niet OK:

```
<div class="container">
  <div class="row">
    <h2>PHP Ontwikkelaar</h2>
    <article class="col-md-4">
```

Een **.row** mag enkel **.col** elementen als *immediate child* hebben.

Op een breed scherm zie je nu dit:

Wat we doen

vdab opleidingen zijn gericht op het vinden van werk in die richting. Zowat alle IT functies zijn nog steeds knelpuntenberoepen waarvoor men moeilijk de geschikte vrouw/man vindt. Daarom zijn IT opleidingen pragmatisch en gefocussed op wat je nodig hebt om de job te kunnen doen.

PHP Ontwikkelaar

Server-side

Een PHP ontwikkelaar gebruikt **PHP** om server-side scripts te maken die dynamisch webpagina's genereren. Je onderzoekt de wensen van de gebruikers en ontwerp en onderhoudt het programma, test het resultaat en voorziet het van de nodige documentatie.

Open Source

PHP is Open Source en geniet zo van de ondersteuning en documentatie van de development community. Het is ook gratis software. PHP heeft een wat lagere instapdrempel dan alternatieven zoals Java of .Net, maar voloedt evenzeer aan alle vereisten voor de professionele programmeur.

CMS

PHP wordt zowel door grote als kleine bedrijven gebruikt, maar is specifiek gericht op webapplicaties. Enkele gekende PHP toepassingen (dikwijls samen met MySQL) zijn **WordPress**, **Drupal**, **Joomla** en **mediawiki**. PHP leent zich tot het ontwikkelen van e-commerce applicaties, Content-management systemen, blog-sites en andere.

Front-end ontwikkelaar

Perfectionist

Een front-end webontwikkelaar is de man/vrouw die ervoor zorgt dat de gegevens die van de webserver komen op een goede manier getoond worden in de browser van de gebruiker. Hij is een meester op gebied van client-side scripting, en kan in geen tijd met widgets uit een Javascript library de interactiviteit van een pagina vergroten.

Client-side

Met zijn kennis van **HTML**, **CSS** en **JavaScript** is hij in staat de dromen van de webdesigner en de eisen van de back-end developer met elkaar te verzoenen, daar precies ligt zijn grote sterkte. Hij blijft uiteraard op de hoogte van het alternatieven in **HTML5**, **CSS3** en **JavaScript**.

Alhoewel hij enige kennis heeft van de 'back-end', de server-side scripts, is hij niet degene die de gegevens uit de database produceert met **PHP**, **Java** of **VB Net**. Hij begrijpt echter heel goed hoe een **XSL** of **JSON** bestand in elkaar zit en hoe hij ze kan gebruiken in zijn zelfgeschreven **AJAX** scripts.

© vdab | mail ons | maandag 4 februari 2013
Dit is een oefenpagina voor de informaticaopleidingen van de vdab: het is mogelijk dat de informatie op deze pagina verouderd en niet meer accuraat is. De hyperlinks kunnen naar niet bestaande URL's verwijzen.

Maar op een smaller scherm (<750px) zie je dit:

username wachtwoord aanmelden

VDAB

- Wat we doen
- Wie we zijn
- Portfolio
- Contact

Wat we doen

vdab opleidingen zijn gericht op het vinden van werk in die richting.



Zowat alle IT functies zijn nog steeds knelpuntenberoepen waarvoor men moeilijk de geschikte vrouw/man vindt. Daarom zijn IT opleidingen pragmatisch en gefocussed op wat je nodig hebt om de job te kunnen doen.

PHP Ontwikkelaar

Server-side

Een PHP ontwikkelaar gebruikt [PHP](#) om server-side scripts te maken die dynamisch webpagina's genereren. Je onderzoekt de wensen van de gebruikers en ontwerpt en onderhoudt het programma, test het resultaat en voorziet het van de nodige documentatie.

Open Source

PHP is Open Source en geniet zo van de ondersteuning en documentatie van de development community. Het is ook gratis software. PHP heeft een wat lagere instapdrempel dan alternatieven zoals Java of .Net, maar voldoet evenzeer aan alle vereisten voor de professionele programmeur.

CMS

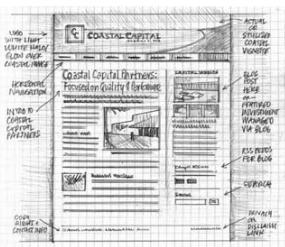
PHP wordt zowel door grote als kleine bedrijven gebruikt, maar is specifiek gericht op webapplicaties. Enkele gekende PHP toepassingen (dikwijls samen met MySql) zijn [WordPress](#), [Drupal](#), [Joomla](#) en [mediaWiki](#). PHP leent zich tot het ontwikkelen van e-commerce applicaties, Content-management systemen, blog-sites en andere.

Front-end ontwikkelaar

Perfectionist

Een front-end webontwikkelaar is de man/vrouw die ervoor zorgt dat de gegevens die van de webserver komen op een goede manier getoond worden in de browser van de gebruiker.

Hij is een meester op gebied van client-side scripting, en kan in geen tijd met widgets uit een Javascript library de interactiviteit van een pagina vergroten.



Client-side

Met zijn kennis van [HTML](#), [CSS](#) en [Javascript](#) is hij in staat de dromen van de webdesigner en de eisen van de back-end developer met elkaar te verzoenen, daar precies ligt zijn grote sterkte. Hij blijft uiteraard op de hoogte van het allernieuwste in HTML5, CSS3 en Javascript.

Afheweit hij enige kennis heeft van de 'back-end', de server-side scripts, is hij niet diegene die de gegevens uit de database produceert met PHP, Java of VB.Net. Hij begrijpt echter heel goed hoe een [XML](#) of [JSON](#) bestand in elkaar zit en hoe hij ze kan gebruiken in zijn zelfgeschreven AJAX scripts.

© vdab | [mail ons](#) | maandag 4 februari 2013
Dit is een oefenpagina voor de informaticaopleidingen van de vdab: het is mogelijk dat de informatie op deze pagina verouderd en niet meer accuraat is. De hyperlinks kunnen naar niet bestaande URL's verwijzen.

Hier zie je dus *Responsiveness* in actie: de `.col-md-n` kolommen stapelen verticaal voor kleinere schermbreedtes.

Het is nog geen ideale situatie want de figuur tussen "Perfectionist" en "Client-side" stoort nu. Straks geven we daar een redelijk goede oplossing aan, om dit volledig naar je eigen zin op te lossen moet je zelf ingrijpen in de media-queries, maar dat is voor een volgend project.

2.11.7 Navigatie

Om het hoofdmenu om te zetten naar een navigatiebalk gaan we doorheen een aantal mogelijkheden.

Pas het volgende aan:

- Geef de `ul` een `class="nav"`
- Geef het eerste `li` element een `class="active"`
- Pas de hyperlinks aan volgens de nieuwe bestandsnamen

```
<!--hoofdmenu-->
<ul class="nav">
    <li class="active"><a href="webontwikkelaar_wat_bs.html">Wat we doen</a></li>
    <li><a href="webontwikkelaar_wie_bs.html">Wie we zijn</a></li>
    <li><a href="webontwikkelaar_portfolio_bs.html">Portfolio</a></li>
    <li><a href="webontwikkelaar_contact_bs.html">Contact</a></li>
</ul>
<!--einde hoofdmenu-->
```

Het resultaat zal nogal bedroevend zijn: de hyperlinks oriënteren zich gewoon horizontaal. Dit komt omdat we naast de basisclass `.nav` ook nog een type navigatie moeten meegeven. Er zijn drie mogelijkheden: `.nav-tabs`, `.nav-pills` en `.navbar-nav`.

Probeer even deze wijziging:

```
<ul class="nav nav-tabs">
```

Dan krijg je dit:



Probeer ook dit:

```
<ul class="nav nav-pills">
```

Met dit resultaat:

Wat we doen Wie we zijn Portfolio Contact

Maar om een echte *navigatiebalk* te krijgen moeten we de *navigatie in een balk* stoppen.

Voeg een extra `nav` element toe rond de `ul` en geef er ook de class `navbar-nav` aan:

```
<nav class="navbar navbar-default" role="navigation">
  <ul class="nav navbar-nav">
    <li class="active"><a href="webontwikkelaar_wat_bs.html">Wat we doen</a></li>
    <li><a href="webontwikkelaar_wie_bs.html">Wie we zijn</a></li>
    <li><a href="opleidingscentrum_portfolio_bs.html">Portfolio</a></li>
    <li><a href="opleidingscentrum_contact_bs.html">Contact</a></li>
  </ul>
</nav>
```

Dit geeft een echte navigatiebalk:

Wat we doen Wie we zijn Portfolio Contact

Een navigatiebalk kan ook een klein logo bevatten. We vervangen het grote VDAB logo door een kleintje. Verwijder (of commenteer) het bestaande logo en voeg een `.navbar-header` toe:

```
<section id="intro">
  <div class="container">
    <!--
    <div id="logo"><a href="webontwikkelaar_home.html"></a></div>
    -->
    <!--hoofdmenu-->
    <nav class="navbar navbar-default" role="navigation">
      <div class="navbar-header">
        <a class="navbar-brand" href="#">
          </a>
        </div>
        <ul class="nav navbar-nav">
        ...
      </ul>
    </nav>
  </div>
</section>
```

Opmerkingen:

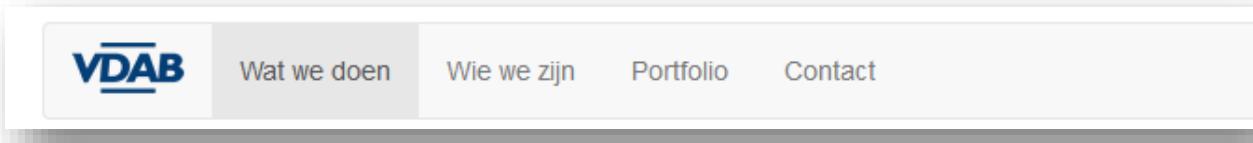
- De kleinere figuur vind je in de startbestanden.

Je bemerkt dat het figuurtje de *navbar* wat te "dik" maakt: voeg volgende style-rule toe aan *style.css*:

```
.navbar-brand {padding-top:10px; padding-bottom:10px } /* vdab_60 logo is te hoog */
```

In een bedrijf zou dit logo precies op maat gemaakt kunnen worden...

Met dit resultaat:



Een **navbar-header** element bevat alle extras die *niet* tot de eigenlijke navigatie behoren, zoals een logo en een *collapse button* die we nu toevoegen.

2.11.8 Een openklappende navigatiebalk op Mobile

Als we het scherm sterk versmallen stapelen de menuitems zich, maar we zouden liever een openklappend menu hebben.

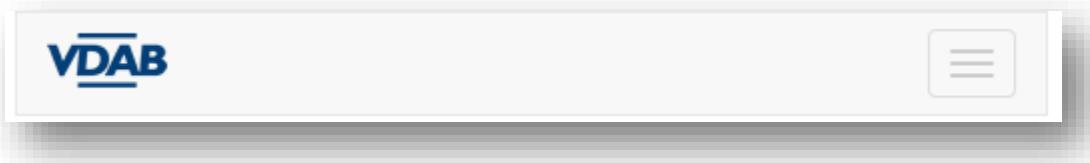
Daarvoor moeten we

- de eigenlijke navigatie groeperen in een apart `div.collapse` element
- een *collapse button* toevoegen aan de **navbar-header**
- ervoor zorgen dat de *bootstrap.js* library geladen is (reeds gebeurd)

Pas aan:

```
<!--hoofdmenu-->
<nav class="navbar navbar-default" role="navigation">
<div class="navbar-header">
<button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#hoofdmenu">
    <span class="sr-only">Toggle navigation</span>
    <span class="icon-bar"></span>
    <span class="icon-bar"></span>
    <span class="icon-bar"></span>
</button>
<a class="navbar-brand" href="#"></a>
</div>
<div class="collapse navbar-collapse" id="hoofdmenu">
    <ul class="nav navbar-nav">
        <li class="active"><a href="webontwikkelaar_wat_bootstrapv3.html">Wat we doen</a></li>
        <li><a href="webontwikkelaar_wie_bootstrapv3.html">Wie we zijn</a></li>
        <li><a href="opleidingscentrum_portfolio_bootstrapv3.html">Portfolio</a></li>
        <li><a href="opleidingscentrum_contact_bootstrapv3.html">Contact</a></li>
    </ul>
</div>
</nav>
<!--einde hoofdmenu-->
```

Als je nu op een klein scherm kijkt zie je dit:



Je kan door op de *collapse knop* te klikken het menu openen en sluiten.

Tenslotte passen we nog één iets aan dat ons een beetje stoort, nl. de **margin** onder de navbar. Voeg toe aan style.css:

```
.navbar { margin-bottom:0 }
```

2.11.9 Het login formulier

Nu bewerken we het aanmeldingsformulier.

Omdat we het op een breed scherm op één lijn bovenaan willen en toch responsive zijn geven we het de BS classes **.navbar-form** en **.navbar-right** :

```
<form method="get" action="/search" class="navbar-form navbar-right" role="search">
```

Het attribuut **role=search** dient om de *Accessibility* te verbeteren.

Nu pakken we de form-elementen aan.

Elke combinatie van **label** en **input** zetten we in een **div** element met de class **.form-group**, de **input** elementen geven we de class **.form-control**, de **submit** knop een class **class="btn btn-default"**:

```
<!--start login-->
<form method="get" action="/search" class="navbar-form navbar-right" role="search">
  <div class="form-group">
    <label for="username">username</label>
    <input type="text" class="form-control" name="username" id="username">
  </div>
  <div class="form-group">
    <label for="pw">wachtwoord</label>
    <input type="password" class="form-control" name="pw" id="pw">
  </div>
  <input type="submit" class="btn btn-default" value="aanmelden">
</form>
<!--einde login-->
```

Dit zijn standaard lay-out en classes voor een form in BS. Je kan de documentatie ervan nalezen.

Het ziet er nu zo uit:

A screenshot of a login form. It has two input fields: 'username' and 'wachtwoord', and a 'aanmelden' button.

In een volgende stap verbergen we de labels en gebruiken een **placeholder** ter vervanging

```
<form method="get" action="/search" class="navbar-form navbar-right" role="search">
  <div class="form-group">
    <label for="username" class="sr-only">username</label>
    <input type="text" class="form-control" name="username" id="username" placeholder="username">
  </div>
  <div class="form-group">
    <label for="pw" class="sr-only">wachtwoord</label>
    <input type="password" class="form-control" name="pw" id="pw" placeholder="wachtwoord">
  </div>
  <input type="submit" class="btn btn-default" value="aanmelden">
</form>
```

- De class **sr-only** verbergt de labels.
Onthou dat je altijd een **label** element moet voorzien voor *screen readers* (Accessibility)
- Het **placeholder** attribuut geeft nu een goede aanduiding van het veld

Ons form ziet er nu zo uit:

A screenshot of the login form after changes. The 'username' and 'wachtwoord' fields now have placeholder text instead of labels.

2.11.10 Images

De beelden geven we een mooi randje met de class **.img-thumbnail**:

```

...

```



We hebben daarnet opgemerkt dat de beelden op een kleiner scherm storend zijn, daarom maken we hier ook nog eens gebruik van een *Responsive helper class*. Geef beide images nog de class `.hidden-xs` mee:

```

...

```

Nu zijn ze enkel nog zichtbaar op schermen >768px.

2.11.11 Glyphicons

BS heeft heel wat ingebouwde icoontjes. We zouden graag een icoontje vóór de naam van elke opleiding zetten.

In deze versie van BS zijn de **Glyphicons** inbegrepen: kijk ze na bij *Components*.

We volgen de documentatie en voegen een `span` element in met een bepaalde glyphicon class:

```
<h2><span class="glyphicon glyphicon-leaf"></span>PHP Ontwikkelaar</h2>
...
<h2><span class="glyphicon glyphicon-flag"></span>Front-end ontwikkelaar</h2>
```

Het resultaat:



We vinden het icoontje iets te groot en er zou wat ruimte tussen ikoon en tekst moeten. Het zijn echte *true-type fonts* dus kunnen ze groter of kleiner gemaakt worden met `font-size`.

We voegen een style-rule toe aan ons eigen stylesheet `style.css`:

```
h2 .glyphicon {
```

```
    font-size: 0.8em;  
    margin-right: 0.4em;  
}
```

Het resultaat is beter:



Klaar.

Taken:

maak nu volgende taken

- "*Bootstrap gebruiken*"

2.12 Een Responsive Design, Mobile First

Doel

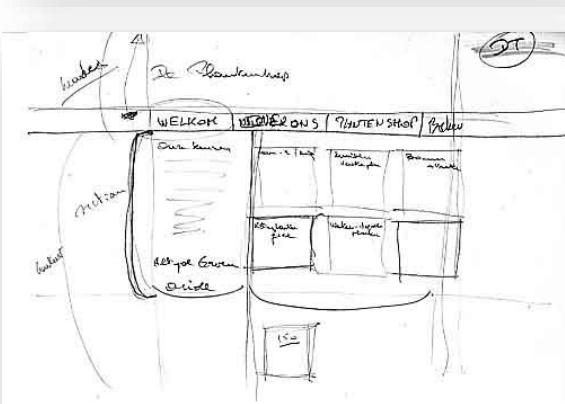
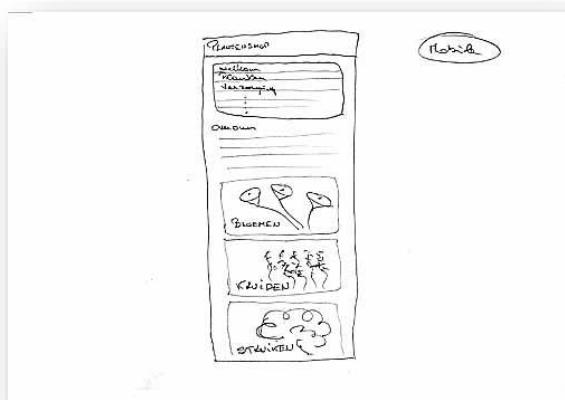
Een Responsive pagina maken volgens het *Mobile First* principe zonder Framework

Theorie

- Mobile web design

Startbestanden

plantenshop_home.html, normalize.css



In dit project werken we *Mobile First*. We doen het ook helemaal zelf: we gebruiken geen Framework.

Het eerste wat je zult doen is nadenken over de inhoud die je gaat tonen. Wat is belangrijk en **wat** ga je tonen? Daarna komt de vraag **hoe** je het gaat tonen?

Een aantal **mockups** komen daarbij van pas: maak schetsen over de schermen van phone, tablet en desktop. Hier hebben we er enkele gemaakt voor jou.

Eénmaal je een goed idee hebt over de content, in welke vorm je die gaat tonen en waar op de pagina dat zal zijn, is het erg belangrijk voldoende **structuur in die inhoud** aan te brengen. Denk na over kop-, body- en footer-zones, navigatie, titels en hoe je die in een HTML structuur gaat gieten.

Maak volop gebruik van **header** elementen, **div**, **footer**, **article**, **section**, **aside**, **navbar** etc...

Het startbestand is al volledig gestructureerd: bekijk de HTML structuur.

We bespreken eerst de **head** van het document:

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link href="css/normalize.css" rel="stylesheet">
<link href="css/plantenshop.css" rel="stylesheet">

<!-- Le HTML5 shim, for IE6-8 support of HTML5 elements -->
<!--[if lt IE 9]>
<script src="http://html5shim.googlecode.com/svn/trunk/html5.js"></script>
<![endif]-->

<title>Plantenshop</title>
</head>
```

Bespreking:

- dit is een HTML5 doc
- de **viewport** meta tag bevat de content

width=device-width, initial-scale=1.0

dit zorgt ervoor dat de initiële breedte van de pagina het beschikbare scherm vult met een zoom-factor 1. Dit is een belangrijke tag voor *Responsiveness*.

- er wordt een stylesheet *normalize.css* geladen: dit is een stylesheet dat een aantal anomalieën corrigeert die in de browserdefaults zitten.
Zo stelt het de css van alle nieuwe HTML5 elementen correct in, de **margin** van de **body** op **0**.
Wat het NIET doet is een volledige reset. Wil je **margin** en **padding** van **div** op **0** zetten, dan zal je dat zelf moeten doen.
Je komt meer te weten op <http://necolas.github.com/normalize.css/>
- er wordt een koppeling naar een stylesheet *plantenshop.css* gemaakt: die zullen wij zelf maken
- het volgende conditionele commentaar stukje dient enkel voor een Internet Explorer browser lager dan de versie 9: deze browsers herkennen de **[if lt IE9]** en laden in dat geval de *html5 shim* van Google CDN

Wat volgt is de structuur van de pagina: er is een **header**, een **div#inhoud** en een **footer**. De **#inhoud** bevat verschillende **section's**. De eerste **section** heeft een **aside** en een **div.kol**.

Bekijk nu eerst deze pagina in je browser vooraleer we starten met *plantenshop.css*:

De Plantenshop

- [Welkom](#)
- [Planten](#)
- [Verzorging](#)
- [Over ons](#)
- [Fotos](#)
- [Contact](#)

Onze keuzes

"De Plantenshop" is een kwekerij met een groot assortiment buitenplanten. Onze online shop biedt u a

Als zoon van een Vlaamse boomkweker ben ik in 1931 een kwekerij begonnen, met een oppervlakte van 10 ha. We nu het volledige areaal buitenplanten bezitten. De oppervlakte van de kwekerij is nu ruim 15.000 m². Een dergelijke sortering niet gemakkelijk, waar dan ook, in Europa te vinden zal zijn.

Groepen



[Eén- en tweejarigen](#)

Een- en tweejarigen kunt u zelf zaaien. Deze planten bloeien vaak meer en langer dan vaste plantjes tot klimmers die in onafzienbare tijd uw muur of schutting bedekken met een kleurige bloei-

De pagina bevat een eenvoudige en duidelijke structuur. Het hoofdmenu en het plantengroepenmenu bestaan uit een [ul](#) lijst.

2.12.1 Algemene css

We starten met CSS die als basis zal dienen voor alle toestellen, maar zeker en eerst voor een klein gsm scherm: dus geen media queries.

Versmal ook je browservenster helemaal en gebruik een mobile emulator met een toestel met een schermbreedte van < 480px.

```
@charset "utf-8";
/*
CSS voor Plantenshop, een Responsive, Mobile First oefening
*/
***** ALGEMEEN + MOBILE FIRST *****

body, div, article, section {
    margin:0;
    padding:0;
}
body {
    font: 16px/1.4 "Lucida Grande", Sans-Serif;      /* 1em = 16px */
    color:#333333;
}
a {
    text-decoration:none;
```

```
color:inherit;
}

h1, h2 {
    font-size: 1.125em; /* 18px */
    padding: 0 0.3333em; /* 6px */
}

header {
    position:relative;
}

section {
    clear:both;
}

p {
    margin-left:0.375em; /* 6px */
}

.container {
    overflow:auto;
}

/* kop phone: titel bovenaan met lijn eronder */

header h1 {
    margin:0.2222em; /* 4px */
}

#kopnav {
    border-top:1px solid gray;
    padding-top:0.6em;
    margin:0;
}
```



Enkele opmerkingen:

- het [a](#) element erft de kleur van zijn parent en krijgt geen onderlijning
- `section {clear: both;}` zorgt ervoor dat de `section` elementen niet naast elkaar kunnen gaan staan
 - `.container {overflow:auto;}` lost het probleem met floats op
- de twee laatste selectors hebben te maken met de lay-out op phone

Als we nu kijken zien we het volgende:

2.12.2 Hoofdmenu

In het volgende stukje maken we een verticaal georiënteerd menu:

```
...
***** hoofdmenu mobile first*****
/*
vertikaal menu
container bepaalt breedte van list */

#hoofdmenu {
    list-style: none;
    font-size: 0.875em;                      /* 14px = 1em */
    font-weight: bold;
    padding: 0;                                /* 0.5em = 6px */
    margin: 0 1em;
    width: auto;
}
#hoofdmenu li {
    display: block;                            /* onder elkaar */
    margin: 0;
    padding: 0.4286em 1.1428em;                /* 6px 16px */
    border-bottom: 1px solid silver;
    border-left: 1px solid silver;
    border-right: 1px solid silver;
}
#hoofdmenu li:first-child {
    border-top: 1px solid silver;
}
#hoofdmenu a {
    display: inline;
    cursor: pointer;
}
#hoofdmenu li:hover {
    color: white;
    background-color: darkseagreen;
}
...
...
```

Je hebt al vanuit een *bulleted list* een dergelijk menu leren maken, dus we veronderstellen dat je er de werking van begrijpt, toch enkele opmerkingen:

- **#hoofdmenu** is een **ul** element dat in een **navbar** zit. De breedte van dit menu wordt dus bepaald door de **navbar**
- de lijntjes tussen de **li** elementen bestaan enkel uit **border-bottom**. Het bovenste (eerste) element heeft dus geen lijn boven zich, daarom de **li:first-child** selector

Nu ziet het beeld er zo uit:



2.12.3 Helper classes

In een volgende deel behandelen we de thumbnaillist van de plantengroepen. Voor een GSM willen we enkel het prentje zien met de titel erbovenop, geen begeleidende tekst.

Onderdelen van je pagina tonen en verbergen kan je zeker doen via de juiste contextuele CSS selectoren , bv. `.thumbnaillist p`. Maar het kan ingewikkeld worden in situaties waarbij we het ene willen tonen op een *smartphone*, dan weer niet op een *tablet* en weer wel op een *desktop*, of in elke andere combinatie.

Om dit gemakkelijk op te vangen zullen we gebruik maken van een aantal **helper classes** die we toekennen aan die inhoud waarop ze van toepassing zijn. Je bemerkt in de HTML code dat de `p` elementen in de `.thumbnaillist` zo'n *helper classes* hebben:

```
<p class="klein-tablet"> <p class="groter-tablet"> <p class="desktop">
```

Je maakt dergelijke classes *naar eigen goeddunken*; de bedoeling hier is dat deze content getoond wordt "vanaf een tablet" en niet op een kleiner scherm. De vraag rijst natuurlijk hoe groot een "tablet" is: de keuze van de **break-points**. Maar dat bespreken we later.

We maken eerst enkele helper-classes in ons stylesheet:

```
/* helper classes */
```

```
.phone-only {  
    display:block  
}  
.klein-tablet {  
    display:none  
}  
.groter-tablet {  
    display:none  
}  
.desktop{  
    display:none  
}
```

Bespreking: bedenk dat dit voor de situatie *Mobile First* geldt

- de class **.phone-only** wordt nu getoond
- alle andere worden niet getoond
- die situaties zal veranderen eens we voor een groter scherm werken

Als je nu de pagina bekijkt, is de tekst bij de thumbnails verborgen.

2.12.4 De thumbnaillist

We voegen de CSS voor de thumbnaillist in:

```
***** thumbnaillist mobile first *****/  
/* mobile: bullets enkele titel geen foto, geen tekst */  
.thumbnaillist {  
    list-style-type: none;  
    font-size: 0.9em;           /* 12px = 1em */  
    width: 100%;  
    margin: 0;  
    padding: 0.5em 0;          /* 0.5em = 6px */  
    background: lavender;  
}  
.thumbnaillist li {  
    padding: 0 0 0.9em 3em;  
    margin: 0.5em;  
    background: transparent url(../../img/flower.png) 0 0 no-repeat;  
}  
.thumbnaillist h4 {  
    font-size: 1.125em;         /* 18px */  
    color: white;  
    margin: 0;  
    padding-top: 0.2em;  
}  
.thumbnaillist img {  
    display:none;  
}
```

Ook deze techniek heb je eerder al geleerd, we bespreken enkel wat interessante rules:

- het **ul** element zelf (**.thumbnallist**) krijgt een paarse achtergrond
- de **li** elementen krijgen
 - een achtergrondfiguur als *bullet*
 - margin en padding zijn in functie van dit figuur gekozen
- het **h4** element
 - de breedte wordt beperkt tot 60% van de container
- het **img** element
 - wordt verborgen met **display:none**

De thumbnaillist ziet er nu zo uit op een GSM:



2.12.5 Media Queries

Met *Mobile First* schreven we eerst de algemene css en specifieke css voor phone: de site werkt voor mobieljes. Maar bedenk ook



"the first mediaquery is no mediaquery"

Door *Mobile First* te werken zijn we ook perfect toegankelijk voor mobieljes die geen media queries begrijpen.

Vanaf nu werken we voor grotere schermen: we passen het design principe *progressive enhancement* toe.

Voeg onderaan de CSS de volgende mediaqueries toe:

```
/* TABLETS */  
@media screen and (min-width:361px) {  
}  
}
```

```
/* DESKTOPS */
@media screen and (min-width : 780px) {  
}  
}
```

Bespreking:

- de CSS Cascade wordt hier gebruikt als toepassing van *Progressive Enhancement*: alle CSS die eerder kwam blijft geldig, tenzij *overruled* door eventuele rules die hier volgen.
 - om die reden staat er enkel een `screen and (min-width)` directive in
 - de CSS in de MQ Tablets zal dus ook blijven gelden voor Desktops tenzij daar overschreven

Opmerking:

- we blijven nu binnen één stylesheet maar je kan dit even goed toepassen met verschillende aparte stylesheets door de MQ'q toe te passen in de [link](#) elementen, een voorbeeld:

```
<link rel="stylesheet" href="basis.css" media="screen">
<link rel="stylesheet" href="tablet.css" media="screen and (min-width:480px)">
<link rel="stylesheet" href="desktop.css" media="screen and (min-width:780px)">
```

Ook hier passen we geen mediaquery toe in het basis gedeelte, dat komt pas voor tablets en desktops. Deze manier van werken is identiek aan wat wij deden, het enige nadeel hier is dat het meerdere http requests nodig heeft.

Breakpoints

Waar zet je de break-points: de minimum breedtes waarop je pagina "respondeert"?

Deze vraag heeft geen eenvoudig antwoord. De breekpunten moet je NIET kiezen in functie van:

- je eigen, het beste, het mooiste, het nieuwste toestel: de markt evolueert zo snel dat je het toch niet kunt bijhouden.
 - voorbeelden op het web... de meeste zijn al verouderd
 - fysieke grootte van een toestel heeft tegenwoordig nog weinig te maken met resolutie: de nieuwste smartphones hebben een resolutie die even groot (of groter) is dan de middenklasse tablets...

kies WEL in functie van:

- de **meest gebruikte** toestellen van je doelpubliek:
doe een marktonderzoek, onderzoek de log files van je server. Zorg dat je geen groep wegjaagt van je site door te geavanceerd te werken.
 - **je design en lay-out:**
vanaf welk punt wordt je design lelijk (te brrrrrrrrrrrr) of gaat compleet de mist in (bv. wegens verspringende floats)

- je kan het aantal breakpoints verhogen: 3,4, 5 ipv 2: daar is niets op tegen, het betekent gewoon meer testen en aanpassingen maken

Enkele tips:

- breakpoints zijn niet belangrijk: het is de ruimte tussenin die telt. Probeer verschillende breakpoints in verschillende toestellen. Test op willekeurige toestellen van je kennissen of een willekeurig publiek
- test op zoveel mogelijk ECHTE toestellen, emulators helpen maar bieden geen zekerheid
- probeer te werken voor vandaag en voor wat komt

2.12.6 Tablet

We vullen de css voor een wat groter scherm (kleine tablet, medium smartphone) in.

Eerst doen we de helper classes:

```
/* TABLETS */
@media screen and (min-width:361px) {
    /* helper classes */
    .phone-only      { display:none; }
    .klein-tablet   { display:block; }
    .groter-tablet  { display:none; }
    .desktop         { display:none; }

}
```

Bespreking:

- let op dat je binnen de accolades van de MQ blijft!
- hier tonen we **.klein-tablet** en verbergen alle andere

Nu volgt de lay-out. Tijdens de *mock-up* kwamen we tot de volgende wijzigingen tov de phone:

- iets meer marge links en rechts
- koppen iets groter
- groene achtergrond voor de header
- titel onder het hoofdmenu dat in twee rijen van 3 items helemaal bovenaan staat
- toon de beelden in de bulleted list, verwijder de achtergrondbullets

Een eerste deel:

```
***** TABLET LAY-OUT*****
/*
    hoofdmenu bovenaan
    3 items naast elkaar: totaal = 26.8% + 3% + 1px border
    grotere header, titel onder menu
    toon figuren in de list, met titels
*/
```

```
.container {  
    width: 94%;  
    margin: 0 auto;  
}  
header {  
    height: 10em; /* 160px */  
    background: darkseagreen;  
}  
header h1 {  
    font-size: 2em; /* 32px = 1em */  
    position: absolute;  
    top: 3.5em; /* 96px */  
    left: 0;  
    margin: 0 0.5em; /* 16px */  
}  
#kopnav {  
    border: none;  
    padding: 0;  
}  
#inhoud h2 {  
    font-size: 1.25em; /* 20px */  
    padding: 0;  
}
```

Hier wijzigen we grotendeels de lay-out van de header, de **h1** wordt absoluut gepositioneerd om hem onder het menu te plaatsen.

Een volgend deel wijzigt het hoofdmenu:

```
/*hoofdmenu*/  
  
#hoofdmenu {  
    margin: 0;  
    overflow: visible;  
}  
#hoofdmenu li {  
    font-size: 1.2em; /* 20px = 1em */  
    width: 26.8%;  
    float: left;  
    border-top: none;  
    border-left: none;  
    border-bottom: 1px solid darkslategray;  
    border-right: 1px solid darkslategray;  
    padding: 0.8em 3%;  
}  
#hoofdmenu li:first-child, #hoofdmenu li:nth-child(4) {  
    border-top: none;  
    border-left: 1px solid darkslategray;  
}  
#hoofdmenu li:hover {
```

```
color: white;
background-color: darkslategray;
}
```

Alles hier is in functie van de andere lay-out van het menu:

- de `li` elementen worden gefloat en kunnen daardoor naast elkaar gaan staan
- om in twee rijen te kunnen werken wordt de breedte van elk `li` element beperkt tot 26.8%. We houden hier rekening met 3% padding en een 1px border
- de `border` situatie wordt anders dan voorheen: nu werken we met een `border` onder en rechts van elk element
- voor het eerste en vierde `li` element

```
li:first-child, li:nth-child(4)
```

wordt een linkerrand voorzien

- de `hover` kleuren worden aangepast

Dit is wat we nu te zien krijgen als we een scherm hebben breder dan 380px:



In een volgende stap moeten we ook de lay-out van de thumbnails aanpassen, maar eerst dit:

2.12.7 Fluid images

Hoe werk je met beelden in een *fluid* omgeving? Beelden hebben een vaste pixelgrootte en als hun container kleiner wordt dan ze zelf zijn, kan er vanalles gebeuren. Hoe zorg je ervoor dat een beeld mee verandert met de grootte van zijn container?

Dat kan met een eenvoudige CSS rule:

```
max-width:100%;
```

Je kan misschien eens een simpel experiment wagen. Neem een willekeurige prent, niet te klein, en plaats die drie maal in een `img` element in een nieuw HTML bestandje:

```
<div id="vb1"></div>
<div id="vb2"></div>
```

```
<div id="vb3"></div>
```

en zet de volgende css in een `style` element:

```
img.resp {max-width:100%;}  
#vb3 {width:50%}
```



In ons voorbeeld zijn de exacte dimensies van `killer-fish.jpg` 650 X 433 pixels.

Bekijk nu de pagina en verklein het venster: je bemerkt

- dat de eerste figuur niet veranderd: het normale gedrag van een `img`
- de tweede figuur verkleint volgens de grootte van het venster
- de derde figuur volgt zijn container die maar 50% van het venster inneemt

Conclusie:

`max-width:100%` toegepast op `img`, `canvas` en `video` (en in feite op alle block-) elementen maken ze "fluid" of `responsive` voor veranderingen in schermgrootte.

Dit houden we in gedachten voor de volgende stappen in onze opmaak.

2.12.8 De list voor een tablet

De bulleted list mag op een iets breder scherm – vb tablet – wat meer opgemaakt worden. We zullen het volgende doen:

- de bullets verwijderen
- de beelden tonen op beperkte grootte

- de **h4** titels komen bovenop de beeldjes te liggen
- de begeleidende tekst maken we gedeeltelijk zichtbaar

```
/* thumblist tablet
   geen bullet, image zichtbaar, titel bovenop beeld, eerste tekst zichtbaar
*/


.thumnaillist li {
  position: relative;
  background: none;
  border: none;
  padding: 1%;           /* 6px */
  margin: 1%;
  overflow: hidden;
  height: 6.3333em;      /* 80px */
}
.thumnaillist h4 {
  position: absolute;
  bottom: 0.2em;
  left: 0.2em;
  display: block;
  margin-left: 0.4em;
  color: #00FFFF;
  width: 38%;           /* wrap titels */
}
.thumnaillist img {
  display: inline-block;
  float: left;
  margin-right: 0.5em;
  max-width: 40%;
  min-width: 10em;
}
.thumnaillist p{
  margin: 0;
}
```

Bespreking:

- de **li** elementen
 - krijgen een *relatieve positionering* mee. Dat heeft geen effect op hen zelf maar is belangrijk om straks de titels te positioneren
 - alle *backgrounds* worden verwijderd, daarmee verdwijnen de bloemetjes
 - alle **margin** en **padding** worden op 1% gezet
 - de **height** en de **overflow** zorgen ervoor dat slechts een beperkte hoeveelheid tekst te zien zal zijn
- de **h4** koppen
 - worden *absoluut gepositioneerd* op de **li** elementen, **0.2 em** van de bodem en evenveel van links

- ze krijgen een blauwe kleur en een linker marge
- door hun **width** op een percentage te zetten, zullen ze 'plooien' (wrap) als de breedte van de container te klein wordt. Dit percentage valt wat aan te passen en is ook een beetje afhankelijk van de width van de figuren
- de **img** elementen
 - worden opnieuw getoond dankzij de **display:inline-block**
 - ze worden links **gefloat** in het **li** element
 - ze krijgen een **max-width** van 40% zodat er nog wat ruimte in het **li** element overblijft om de tekst te tonen
 - ze krijgen echter ook een **min-width** zodat ze niet té klein worden
 - deze breedtes moet je wat laten afhangen van de situatie: je zal bemerken – als je verkleint - dat de figuren nog comfortabel te zien zijn net voor de ondergrens van de media query
- de **p** elementen
 - worden alle marge weggenomen
 - zijn ook opgesplitst en voorzien van verschillende *helper classes* waarvan de **class** "klein-tablet" nu te zien zal zijn



2.12.9 Grottere tablet

Voor een wat groter scherm, vanaf 600px bijvoorbeeld, vertrouwen we op de CSS die we net maakten, omdat die *fluid* is en dus redelijk rekbaar.

Het enige wat we doen is een aanpassing aan de *helper classes*:

```
@media screen and (min-width : 600px) {  
    ***** grotere TABLET *****  
    /* helper classes */  
    .phone-only      { display:none; }  
    .klein-tablet   { display:block; }  
    .groter-tablet  { display:block; }  
    .desktop        { display:none; }  
  
}
```

We tonen dus ook de **class "groter-tablet"**.

2.12.10 Desktops

Vanaf ruwweg 800px veranderen we de lay-out opnieuw. We willen:

- grotere pagina marges en breedte van de pagina-kolom
- een meer-kolommen lay-out
- de paginatitel bovenaan
- een grotere header
- het paginamenu horizontaal onderaan de header
- de lijst als een meer-kolommen lay-out

Dat vraagt om een nieuwe media query waarvan we de *helper classes* eerst veranderen:

```
@media screen and (min-width : 800px) {  
    ***** DESKTOP *****  
    /*  
     Titel bovenaan  
     alle items naast elkaar: totaal = + border  
     grotere header, titel onder menu  
     */  
    .phone-only      {display:none; }  
    .klein-tablet   {display:block; }  
    .groter-tablet  {display:block; }  
    .desktop        {display:block; }  
  
}
```

Dit laten we volgen door wijzigingen aan de algemene lay-out (let op: binnen deze media-query blijven):

```
.container {  
    width:80%;  
    max-width:1000px;  
    margin: 0 auto;          /* 16px */  
    position:relative;  
    height: 100%;           /* moet header vullen */  
}  
header {
```

```
height:10em;                                /* 160px */
}
header h1 {
    font-size:2.5em;                          /* 32px = 1em */
    margin:0;
    position: static;
}
#inhoud h2 {
    font-size: 1.375em;                      /* 22px */
}
#kopnav {
    width:100%;
    position:absolute;
    bottom:0;
    left:0;
}

/* twee kol layout      25 + 72 = 99 */
aside {                                     /* 25 + 2 = 27 */
    float:left;
    width: 25%;
    padding:0 1%;
}
.kolom {                                    /*72 = 72 */
    float:left;
    width: 72%;
    padding: 0 0 1em 0;
}
p {margin:0;}
```

Bespreking:

- de class `.container` bepaalt de breedte van zowel de `header` als de `#inhoud`
 - de `width` van de `.container` wordt op 80% gezet met een `max-width` van 1000px. Dit om te voorkomen dat de lay-out té breed zou worden
 - met `margin: 0 auto` wordt de kolom gecentreerd
 - de `position: relative` gebruiken we om (eventueel) children absoluut te kunnen plaatsen
 - de `height:100%` is nodig om straks het menu correct te plaatsen
- de `header` krijgt een vaste hoogte van 10em
- de `h1` titel in de `header` wordt opnieuw normaal geplaatst: `position:static` en wordt wat groter gezet
- de `h2` titels in `#inhoud` worden wat groter
- het `nav` element `#kopnav` wordt onderaan de `header` absoluut geplaatst. Straks komt nog de CSS voor het menu zelf
- het `aside` element en het `.kolom` element vormen vanaf nu twee kolommen naast elkaar. Dat doen we door
 - ze allebei een `float:left` te geven
 - een `width` en `padding` die samengevoegd niet boven de 100% uitkomt
- de `margin` van alle `p` elementen komt op 0 te staan

Ook het hoofdmenu ondergaat nu belangrijke wijzigingen.

We recapituleren even: het menu ónder de 800px bestaat uit twee rijen van drie elementen. Dat hebben we bereikt door de breedte van de `li` elementen op 26.5% in te stellen met 3% padding. Daarenboven moesten de borders precies ingesteld worden.

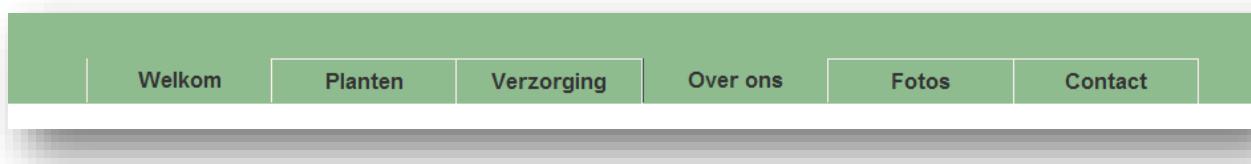
Nu verkleinen we die `width`:

```
#hoofdmenu li {  
    width: 16.4%;  
    padding: 0.3em 0;  
    border-top: 1px solid LavenderBlush;  
    border-right: 1px solid LavenderBlush;  
    border-bottom: none;  
    border-left: none;  
    text-align: center;  
}  
  
#hoofdmenu li:first-child {  
    border-left: 1px solid LavenderBlush;  
}
```

Bespreking:

- met een width van 16.4% en geen padding links of rechts komen we aan 98.4% voor 6 elementen. Dan rest nog voldoende voor 7 1px borders
- de borders komen nu boven en rechts te staan, niet meer onder of links
- tenzij voor het eerste element dat ook een border-left krijgt

Maar als we kijken bemerken we een probleem:



Het eerste en het 4^{de} `li` element hebben geen border-top... Waarom niet?

Onze developer's tool vertelt meer als we het 4^{de} `li` element aanklikken:

The screenshot shows the Developer Tools interface for a web page at http://localhost/htmlcss/projecten/plantenshop_home.html. The left panel displays the HTML structure, and the right panel shows the CSS styles. A red arrow highlights a specific `li` element in the HTML tree, which corresponds to the fourth child of the `#hoofdmenu` selector in the CSS rules.

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body class="home">
    <header>
      <div class="container">
        <h1>...</h1>
        <nav id="kopnav">
          <ul id="hoofdmenu">
            <li>...</li>
            <li>...</li>
            <li>...</li>
            <li>...</li>
            <li>...</li>
            <li>...</li>
            <li>...</li>
            <li>...</li>
          </ul>
        </nav>
      </div>
    </header>
    <div id="inhoud" class="container">
      <section id="main">
        <aside>...</aside>
        <div class="kolom">
          <h2>Groepen</h2>
          <ul id="plantenmenu" class="thumbnallist">...</ul>
        </div>
      </section>
      <section id="extra">...</section>
      <!--einde inhoud-->
    </div>
    <footer>...</footer>
  </body>
</html>
```

```
element.style {
}

Matched CSS Rules
@media screen and (min-width: 320px)
#hoofdmenu li:first-child, #hoofdmenu li:nth-child(4) {
  border-top: none;
  border-left: 1px solid darkslategray;
}

@media screen and (min-width: 800px)
#hoofdmenu li {
  width: 16.4%;
  padding: 0.3em 0;
  border-top: 1px solid lavenderblush;
  border-right: 1px solid lavenderblush;
  border-bottom: none;
  border-left: none;
  text-align: center;
}

@media screen and (min-width: 320px)
#hoofdmenu li {
  font-size: 1.2em;
  width: 26.5%;
  float: left;
  border-top: none;
  border-left: none;
  border-bottom: 1px solid darkslategray;
  border-right: 1px solid darkslategray;
  padding: 0.8em 0;
}

#hoofdmenu li {
  display: block;
  margin: 0;
  padding: 0.4286em 1.1428em;
  border-bottom: 1px solid silver;
  border-left: 1px solid silver;
  border-right: 1px solid silver;
}

li {
  user agent stylesheet
  display: list-item;
  text-align: -webkit-match-parent;
}

Inherited from ul#hoofdmenu
```

Hieruit blijkt dat de selector `#hoofdmenu li` in de 800px MQ overruled wordt door de selector `#hoofdmenu li:nth-child(4)` in de 320px MQ!

Om dit op te lossen zou je wel eens in de verleiding kunnen komen om een `!important` statement te gebruiken....

Helaas voor jou.....



dat lossen we de juiste manier op, namelijk door de specificiteit te verhogen van onze selector.

- de specificiteit van de 320px MQ `#hoofdmenu li:nth-child(4)` is 0 1 1 1
- de specificiteit van de 800px MQ `#hoofdmenu li` is slechts 0 1 0 1

Maar met een eenvoudig truckje verhogen we onze specificiteit:

```
#hoofdmenu li:nth-child(1n+0) {  
    width:16.4%;  
    padding:0.3em 0;  
    border-top:1px solid LavenderBlush;  
    border-right:1px solid LavenderBlush;  
    border-bottom:none;  
    border-left:none;  
    text-align:center;  
}
```

- de selector `#hoofdmenu li:nth-child(1n+0)` selecteert exact hetzelfde als `#hoofdmenu li` maar heeft een even hoge specificiteit van 0 1 1 1, en omdat deze regel ná de vorige komt (cascade) overschrijft hij deze

Probleem opgelost:

Welkom Planten Verzorging Over ons Fotos Contact

Ook de thumbnaillist ondergaat veranderingen.

We behandelen de `li` elementen nu eerder als blokjes die naast elkaar moeten komen.
Dat doen we zo:

```
...
.thumbnaillist {
  padding:0;
  overflow:auto;
}
.thumbnaillist li {
  float:left;
  width: 48%; /* 2 naast elkaar*/
  height:36em;
  padding:0;
}
.thumbnaillist h4 {
  position: static;
  color: darkgreen;
  margin:0;
  padding:5px 0 0 0;
  width:100%;
}
.thumbnaillist img {
  float:none;
  margin: 0;
  max-width:90%;
  border:5px solid snow;
  border-radius:0.2em;
}
} /*einde 800px MQ*/
```

Bespreking:

- vergeet niet steeds binnen de 800px MQ te blijven!
- het `ul` element `.thumbnaillist`
 - geen `padding` meer
 - om dat de `li` elementen gefloat worden (*collapsing parent*) zetten we `overflow:auto`. je kan evengoed een clearFix hack gebruiken
- de `li` elementen
 - worden links gefloat

- krijgen een relatieve width van 48%, zodat er twee naast elkaar kunnen
- krijgen een vaste hoogte van 36em
- geen padding
- de **h4** koppen
 - worden normaal in de *flow* geplaatst met **position:static**
 - krijgen een donkergroene kleur
 - een andere **margin** en **padding**
 - en expliciet een **width** van 100% om de 600px MQ te overschrijven
- de **img** elementen:
 - worden niet langer gefloat
 - worden wat verkleind en blijven *fluid* met een **max-width** van 90%
 - krijgen een **border** met afgeronde hoekjes

2.12.11 Grote schermen

Tenslotte zetten we nog eens de puntjes op de i door een kleine aanpassing te doen voor echt grote schermen. We voegen nog een MQ toe:

```
@media screen and (min-width:960px) {  
  
    .thumbnallist li {  
        width: 31%; /* 3 naast elkaar*/  
    }  
    .thumbnallist img {  
        max-width:95%;  
    }  
}
```

Bespreking:

- we passen de **width** van de **li** elementen aan zodat er 3 naast elkaar kunnen
- omdat daarom de **img** kleiner worden, geven we die een 95% breedte. Waarom geen 100%? omdat er nog wat ruimte moet zijn voor de border.

Klaar:

De Plantenshop

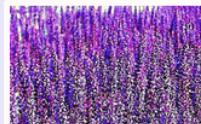
[Welkom](#) [Planten](#) [Verzorging](#) [Over ons](#) [Fotos](#) [Contact](#)

Onze keuzes

"De Plantenshop" is een kwekerij met een groot assortiment buitenplanten. Onze online shop biedt u alle mogelijkheden om op eenvoudige manier uw bestelling te plaatsen en die snel te laten leveren.

Als zoon van een Vlaamse boomkweker ben ik in 1931 een kwekerij begonnen, met een oppervlakte van ongeveer 6000 m². In de eerste jaren kweekte ik voornamelijk bomen en struiken en als hobby, vaste planten. In de loop der jaren werd ons assortiment uitgebreid zodat we nu het volledige areaal buitenplanten bezitten. De oppervlakte van de kwekerij is nu ruim 15.000 m² en alle planten, bestemd voor de verkoop, worden in de pot gekweekt. Het assortiment bestaat uit ca. 4000 soorten planten, waarop wij trots zijn en durven te beweren dat een dergelijke sortering niet gemakkelijk, waar dan ook, in Europa te vinden zal zijn.

Groepen



Eén- en tweearigen

Een- en tweearigen kunt u zelf zaaien. Deze planten bloeien vaak meer en langer dan vaste planten en brengen kleur en variatie. Daarnaast kunnen zij goed leemtes vullen in de border. Een- en tweearigen zijn te krijgen in vele verschillende vormen, van hele kleine tere plantjes tot klimmers die in onafzienbare tijd uw muur of schutting bedekken met een kleurige bloemenpracht. Bekijk onze selectie Fen- en

Vaste planten en kruiden

De groep vaste planten omvat kruidachtige planten die geen houtige takken vormen. Een groot deel van deze planten sterven in de winter bovengronds af, maar schieten elk jaar terug vanuit een overblijvend wortelstelsel. Een klein deel van de vaste planten zijn groenblijvend en zijn in de winter ook decoratief (een aantal zijn zelfs winterbloeiwers). Vaste planten worden in pot gekweekt en kunnen die

Bomen en struiken

Van oudsher is dit onze specialiteit. Bij ons vindt u dan ook een uitgelezen selectie van inheemse bomen en struiken ideaal voor alle 'echt' groen beplantingen zoek tussen bomen en struiken



Klim- en gevelplanten

Een tuin kan niet zonder verticale accenten. Klimplanten zijn daarvoor de ideale versiersels, ter verfraaiing van een muur, prieel of pergola of om een lelijk hekwerk aan het oog te onttrekken. Elke klimplant heeft wel iets aantrekkelijks, een rijke bloei, een mooi blad, opvallende bessen. Kortom in de meeste tuinen is er wel een plaatsje dat door een gepaste klimplant verfraaid kan worden. Bovendien zijn ze ook zeer geschikt voor kleine tuinen en zelfs voor geveltuinies.

Water- en vijverplanten

Waterplanten noemen we vaak vijverplanten, ze zijn een absolute noodzaak om tot een biologisch evenwicht te komen! Waterplanten zijn dus planten die groeien in of onder het water. Tot deze categorie behoren drijfplanten, onderwaterplanten, moerasplanten en waterlelies. Zuurstofplanten zijn meestal ook onderwaterplanten (er zijn uitzonderingen). Andere buitenbeentjes zijn de

Schaduwplanten

Iedereen heeft één of meer schaduwplekken in zijn tuin. Ook daar voorzien we groen, of het nu een varen is of Kweetnia*adiewa superba*, zelf een mostapijtje moet kunnen.

Powder toffee liquorice chupa chups. Marshmallow caramels oat cake. Powder tiramisu lollipop chocolate bar cupcake caramels dragée gingerbread jelly beans. Tiramisu applicake pie jelly-o sesame snaps cookie cheesecake. Tart oat cake biscuit. I love candy canes chupa chups chocolate cake I love. Gummi bears gummies apple pie macaroon.

Fruitcake marshmallow gingerbread. Gingerbread jelly beans wafer jelly liquorice I love faworki faworki oat cake. Jujubes applicake donut donut. Cotton candy sugar plum apple pie halvah I love dessert muffin. Applicake jujubes carrot cake brownie lollipop. Liquorice muffin caramels.

© vdab | responsive site by Jan

Taken:

maak nu volgende taken

- "webontwikkelaar, deel 3"

2.13 LESS: een CSS *preprocessor*

Doel

Leren werken met de CSS *Preprocessor* LESS.

Theorie

- opmerking: LESS maakt gebruik van Javascript om de CSS te compileren, maar je hoeft geen Javascript te kennen om ermee te kunnen werken
- LESS documentatie op lesscss.org

Startbestanden

opleidingscentrum_less.html

2.13.1 Wat is LESS?

LESS is een CSS *preprocessor*(*compiler*) waarbij CSS aangemaakt wordt vanuit een (.LESS) basisbestand door een javascript.

Enkele andere *processors* zijn "SASS", "Switch" en er zijn er nog meer.

De voordelen van een *processor*:

- je kan gebruik maken van **variabelen**
- je kan zogenaamde **mixins** gebruiken: dit zijn CSS classes die je kan gebruiken in een andere *class*, als een soort *inheritance*.
- hebben ingebouwde **functies** (bv. op gebied van kleuren)
- je kan classes **nesten**: een class beschrijven als onderdeel van een andere class
- je kan **dynamisch berekeningen** maken aan de hand van **geparametriseerde functies**

Dit laat je toe veel gestructureerder te werken en een CSS bestand op te bouwen vanuit een aantal aparte LESS modules.

LESS is oorspronkelijk gemaakt voor Ruby (met Python) maar er is een Javascript versie beschikbaar. De compiler kan

- *server-side* gebruikt worden
- *client-side* gebruikt worden
- je kan ook je *.less* code op voorhand compileren om er een vast stylesheet (.css) van te maken

Wij opteren voor deze laatste mogelijkheid omdat dit de snelste manier is en het client-side compileren op problemen stuit in sommige browsers.

De documentatie van alle mogelijkheden, functies,etc. van deze taal kan je vinden op <http://lesscss.org/>

2.13.2 GUI compilers

Er zijn nogal wat GUI programma's die het compileren van less bestanden vergemakkelijken, om er enkele te noemen: *Crunch*, *Codekit*, *Simpless*, *Winless*.

Op de webpagina van lesscss.org vind je een lijstje.

Wij kiezen voor Winless omdat het eenvoudig is.

- In de map *projecten* - waar je tot nu alle projecten in gemaakt hebt – maak twee submappen: *less* en *css*.
- ga naar <http://winless.org/> en download en installeer **WinLess**, een Windows GUI compiler. Deze versie bevat waarschijnlijk de laatste versie van *less.js*
- wil je *less* apart downloaden en client-side uittesten, ga dan naar <http://lesscss.org/> en download de laatste versie van *less.js*. Volg de instructies op de website
- ga naar <http://necolas.github.com/normalize.css/> en download *normalize.css*, een modern HTML5 normaliseer stylesheet.

2.13.3 Winless

Open het basisbestand *opleidingscentrum_less.html*. Het is een HTML5 bestand zonder stylesheet.

Plaats de volgende koppeling in de **head** van het bestand:

```
<!DOCTYPE HTML>
<html lang=nl>
<head>
<meta charset=utf-8>
<link rel="stylesheet" type="text/css" href="css/opl.css">
<!--[if lt IE 9]>
<script src="../js/html5shiv.js"></script>
<![endif]-->
<title>vdab opleiding Webontwikkelaar</title>
</head>
```

Een *less* stylesheet is grotendeels zuivere CSS met die extra's die we aanhaalden. Daarom openen we nu een nieuw CSS bestand en slaan het op in de map *less* als *opl.less*. Typ er volgende code in:

```
@charset "utf-8";
/* LESS Document */
// een eerste less test
@color: #4D926F;

header {
  color: @color;
}
h2 {
  color: @color;
```

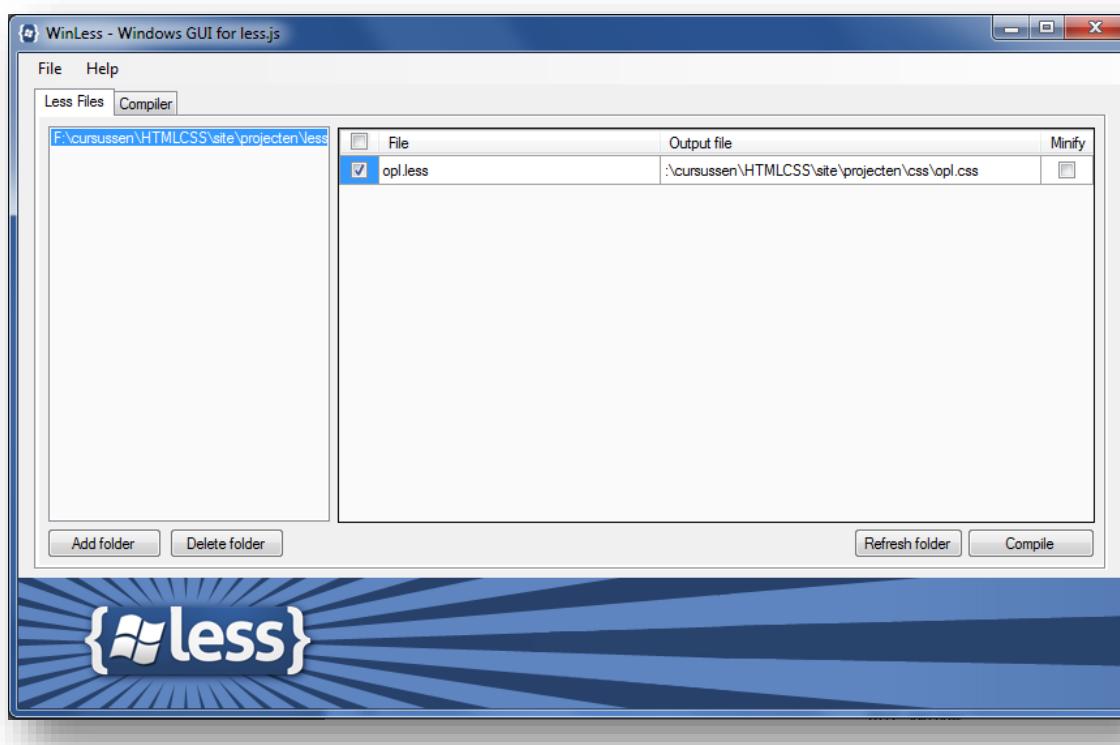
{}

Bespreking:

- de `@color` is een `less` variabele. Zo iets kan je nooit doen met zuivere css
- alle normale css wordt onveranderd doorgegeven
- omdat `less` bestanden beschouwd worden als Javascript mag je er Javascript commentaar // in plaatsen. Deze worden eruit gefilterd tijdens de compilatie

Start *WinLess* en klik op *Add folder*. Duidt de folder `../projecten/less` aan.

Je ziet nu het volgende:



Klik op *Compile*: je bemerkt dat een bestand `opl.css` automatisch aangemaakt is in de zustermap `css`. Open het en bekijk de inhoud:

```
@charset "utf-8";
/* CSS Document */
header {
    color: #4d926f;
}
h2 {
    color: #4d926f;
}
```

De variabele waarden zijn vervangen door de hex waarden.

Vanaf nu hoef je zelf niet meer op de knop *Compile* te klikken: zolang je *WinLess* open houdt zal deze elke wijziging aan het *.less* bestand automatisch compileren naar *opl.css*.

Je zal ook merken dat *WinLess* compileert fouten toont als je een schrijffout maakt.

Vermits het stylesheet voor *opleidingscentrum_less.html* nu aangemaakt is, zijn de kleuren voor **header** en **h2** automatisch toegepast.

2.13.4 Modulair werken

Eén van de redenen waarom we LESS gebruiken is om meer **modulair** te werken: we houden welomlijnde stukken CSS als aparte less bestanden (*kleuren.less*, *typo.less*, *forms.less*, ...) bij om ze tenslotte te compileren in één CSS stylesheet. Zo gebruiken grote CSS Frameworks gemakkelijk 30-40 LESS/SASS bestanden die gecompileerd worden tot 1 CSS stylesheet. Voor de ontwikkelaar is het dan gemakkelijk om aanpassingen te doen.

Het bestand *opl.less* is niets meer dan het verzamelpunt voor alle aparte modules, het integreren gebeurt met **@import** statements. Het zal gecompileerd worden naar *opl.css*.

Verwijder alle vorige code uit *opl.less* en pas aan:

```
@charset "utf-8";
/* LESS Document */

@import "normalize.css";
```

Opmerkingen:

- Het eerste statement in dit bestand is een **@import** van *normalize.css*. Omdat dit een normaal CSS bestand is, zal Winless er niets aan veranderen: het **@import** statement wordt gewoon overgenomen.
- Om die reden moet je ermee rekening houden dat het **pad** naar *normalize.css* moet kloppen vanuit *opl.css*. *normalize.css* zit dus in dezelfde map als *opl.css*.

We importeren nu ook het bestand *variabelen.less* in *opl.less*:

```
@charset "utf-8";
/* LESS Document */

@import "normalize.css";
@import "variabelen.less";
```

Open dit bestand apart en bekijk het.

2.13.5 Variabelen

Het bestand *variabelen.less* bevat kleur en lettertype definities als variabelen. Je mag het vergelijken met een declaratie- en initialisatiezone in een programma. We pikken er enkele voorbeelden uit:

```
...
@blauw:           #049cdb;
@blauwDonker:    #0064cd;
@groen:           #46a546;
@rood:            #9d261d;
...
```

- dit zijn kleurdefinities die bv. we later met de variabelenaam `@blauw` zullen gebruiken.
- alles wat je in CSS mag gebruiken kan hier ook: `hex`-waarden, `rgb` functie, `rgba`, kleurnamen

```
...
@bodyAchtergrond:  @wit;
@tekstKleur:        @grijsDonker;
...
```

- hier maken we gebruik van een variabele om een andere variabele te maken

```
...
@linkKleur:         #08c;
@linkKleurHover:   darken(@linkKleur, 15%);
...
```

- `darken` is een functie die inwerkt op de var `@linkKleur`.
Andere kleurfuncties zijn `lighten()`, `saturate()`, `desaturate()`, `fadein()`, `fadeout()`, `spin()`, `mix()`. Uitleg op de site.

```
...
@sansFontFamily:      "Helvetica Neue", Helvetica, Arial, sans-serif;
@baseFontSize:         100%; /* 16px */
@baseFontFamily:       @sansFontFamily;
@baseLineHeight:      1.125em; /* 18px */
...
```

- hier stellen we de lettertypes, de lettergrootte en de lineheight in

Het staat je uiteraard vrij nog andere variabelen in te stellen.

Nu passen we een aantal van die variabelen toe in een ander less bestand: `tekst.less`.

Importeer dit bestand ook in `opl.less`:

```
@charset "utf-8";
/* LESS Document */

@import "normalize.css";
@import "variabelen.less";
@import "tekst.less";
```

Open dit bestand ook even.

2.13.6 Berekeningen met eenheden

We zien o.a. het volgende

```
p {  
    margin: 0 0 @baseLineHeight / 2;  
    ...  
}
```

- de **margin** van het **p** element gebruikt de helft van de **@baseLineHeight**: dit illustreert dat je een berekening kunt maken met elke eenheid, zij het pixels, ems, kleurwaarden

```
h2 {  
    font-size: @baseFontSize * 1.6;  
    line-height: @baseLineHeight * 2;  
}
```

- dit illustreert dit principe opnieuw: de **@baseFontSize** (uitgedrukt in ems) wordt vermenigvuldigt met factor 1.6

2.13.7 Geneste selectors

```
h1, h2, h3, h4, h5, h6 {  
    margin: 0;  
    font-family: @headingsFontFamily;  
    font-weight: @headingsFontWeight;  
    color: @headingsColor;  
  
    small {  
        font-weight: normal;  
        line-height: 1;  
        color: @grijsLicht;  
    }  
}
```

- hier zien we een nested rule: een **small** element dat voorkomt in een **h1, h2 of h6**. Dit is het equivalent van de selector

h1 small, h2 small, h3 small, h4 small, h5 small, h6 small { }

Nog twee voorbeelden vinden we :

```
p {
```

```
...
.rood{
  color:@rood;
}
}
```

- dit is het equivalent van de selector `p .rood { }`

```
a {
  color: @linkKleur;

  &:hover, &:active {
    text-decoration:none;
    color:@linkKleurHover;
  }
}
```

- hier wordt gebruik gemaakt van de `&` operator, om wat volgt te concateneren:
`&:hover` betekent dus `a:hover`

Geneste rules zijn dikwijls korter dan pure CSS en geven ook beter de specificity weer.

Als Winless actief is zal `opl.css` opnieuw gecompileerd worden telkens je een `/less` bestand opslaat.

Open nu even `opl.css` (in de map `css`) en bekijk het resultaat van de compilatie:

```
@import "normalize.css";
@charset "utf-8";
/*      stylesheet voor Opleidingen
       gecompileerd uit LESS
*/
/* TYPOGRAFIE */
/* alineas */
p {
  margin: 0 0 0.5625em;
  font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
  font-size: 100%;
  line-height: 1.125em;
}
p .rood {
  color: #c53a2f;
}
/* headings */
h1,
h2,
h3,
h4,
h5,
h6 {
```

```
margin: 0;  
font-family: inherit;  
font-weight: bold;  
color: inherit;  
}  
h1 small,  
...
```

- Het resultaat is zuivere CSS.
- Bemerk het ongewijzigde `@import` statement bovenaan.
`normalize.css` zit dus niet mee gecompileerd in deze code, maar wordt geladen via een `@import`
- bemerk ook de aanwezigheid van de CSS commentaren `/* */` en de afwezigheid van de Javascript `//` commentaren die we gebruikten in het less bestand

Het resultaat op de pagina `opleidingscentrum_less.html` is voorlopig beperkt tot kleuren en lettertypes:

Webontwikkelaar

een vdab opleiding

Welkom op de website van de opleiding Webontwikkeling van VDAB Oostende.

- [WELKOM](#)
- [OPLEIDINGEN](#)
- [HANDLEIDINGEN](#)
- [PLANNING](#)
- [MEER](#)

2.13.8 Mixins

Voeg nu ook de pagina's `mixins.less` en `structuur.less` toe aan `opl.less`:

```
@charset "utf-8";  
/* LESS Document */  
  
@import "normalize.css";  
@import "variabelen.less";  
@import "mixins.less";  
@import "tekst.less";  
@import "structuur.less";
```

In de pagina `structuur.less` vinden we op meerdere plaatsen mixins, bijvoorbeeld:

```
header{
```

```
position: relative;
background: @headerAchtergrond;
padding: 0;
margin: 0;
.clearfix();
}
```

.clearFix() is een voorbeeld van een "Mixin": een gedefinieerde class waarvan de code ingevoegd wordt tussen de rest. De class wordt gedefinieerd in de pagina *mixins.less*:

```
.clearfix {
  *zoom: 1;
  &:before,
  &:after {
    display: table;
    content: "";
    line-height: 0;
  }
  &:after {
    clear: both;
  }
}
```

Deze class bevat CSS code voor de gekende *clearFix hack*. Als een selector deze hack nodig heeft roepen we gewoon de "mixin" op en alle nodige code wordt er automatisch aan toegevoegd.

Mixins zijn dus vooral handig voor styles die grote, ingewikkelde CSS rules vergen, zoals deze hack en ook bv. CSS3 styles met veel browser-prefixes zoals gradients.

2.13.9 Mixins met parameters

in *structuur.less* vinden ook de selector .container die een *geparametriseerde mixin* gebruikt:

```
.container{
  //fixed gecentreerd
  .container-fixed(@containerWidth);          // @width
  background: @asideAchtergrond;
  .clearfix();
}
```

- de variabele `@containerWidth` is gedefinieerd in *variabelen.less*
- de mixin `container-fixed` gebruikt deze parameter

In *mixins.less* vinden we de class:

```
.container-fixed(@width){
  margin-left: auto;
```

```
margin-right:auto;  
width:@width;  
padding:0;  
}
```

In *structuur.less* worden ook nog de *mixin* classes *paddingLinks()*, *paddingRechts()*, *paddingOpzij()*, *paddingRondom()*, en *clear()* gebruikt.

2.13.10 Navigatiebalk

Navigatiebalken en menusystemen kan je ook zien als aparte modules. Ook hier hebben we een horizontale navigatiebalk voorzien. We voegen het less bestand ook toe aan *opl.less*:

```
@import "normalize.css";  
@import "variabelen.less";  
@import "mixins.less";  
@import "tekst.less";  
@import "structuur.less";  
@import "navbar.less";
```

en we bekijken het bestand waarin je de volgende zaken moet opmerken:

- de **class navbar** bevat verschillende geneste selectoren:

```
.navbar {  
    a{  
        &:hover{  
            }  
    }  
    ul{  
        li {  
            &:first-child{  
                }  
            &:hover{  
                }  
        }  
    }  
}
```

Wat zal gecompileerd worden naar

```
.navbar {}  
.navbar a {}  
.navbar a:hover {}  
.navbar ul {}  
.navbar ul li {}  
.navbar ul li:first-child {}  
.navbar ul li:hover {}
```

- de `class navbar` bevat ook een mixin `.navbarhor(@aantal,@paddingPx: 2px)`
De parameters zijn
 - het `@aantal` menu-items
 - de `@paddingPx, padding` in pixels die links-rechts opgezet moet worden
 deze parameter heeft een standaardwaarde van 2 px. Dus als je dit argument achterwege laat, zal een standaardwaarde van 2 pixels gebruikt worden

De mixin berekent uit deze waarden de `padding` en `width` die elk `li` element moet hebben om maximaal gebruik te maken van de beschikbare ruimte in de `navbar`

Het resultaat ziet er zo uit:

The screenshot shows a website for a 'Webontwikkelaar' opleiding. At the top is a light blue header with the title 'Webontwikkelaar'. Below it is a red navigation bar with tabs: 'WELKOM', 'OPLEIDINGEN', 'HANDLEIDINGEN', 'PLANNING', and 'MEER'. The 'OPLEIDINGEN' tab is active. A dropdown menu is open under the 'Cursisten' section, listing names: Paul van Klee, Linda Lovelace, Rudolf Valentino, and Eva Green. To the right of the dropdown, text explains that the mixin calculates padding and width for each list item to make the best use of available space. The main content area has a dark background and features the heading 'De inhoud'.

2.13.11 Namespaces

We zeiden al eerder dat één van de grote voordelen van mixins dat je ingewikkelde style rules éénmaal kunt schrijven en daarna overal gebruiken. We demonstreren dat hier even.

De mixin bestaat reeds in `mixins.less`:

```
#gradient {  
    .horizontal(@startColor: #555, @endColor: #333) {  
        ...  
        code  
        ...  
    }  
    .vertical(@startColor: #555, @endColor: #333) {  
        ...  
        code  
        ...  
    }  
}
```

- de echte code hebben we aangemaakt via een zogenaamde gradient generator op het internet. Daar hebben we dan de absolute waarden vervangen door de variabelen `@startColor` en `@endColor`, die ook defaultwaarden meekregen
- we gebruiken de `id` selector `#gradient` hier als **Namespace** om `.vertical` en `.horizontal` in te verzamelen: er bestaat geen element met deze `id`
- de mixin `.gradvertGroen` (onderaan `mixins.less`) gebruikt nu de Namespace gevuld door de mixin en geeft twee argumenten mee:

```
.gradVertGroen{  
    #gradient.vertical(@extraAchtergrondHigh, @extraAchtergrondLow) ;  
}
```

- De argumenten zijn kleurvariabelen die gedefinieerd staan in `variabelen.less`

Je kan een *Namespace* gebruiken om eerder wat te groeperen: variabelen, mixins, functies , een voorbeeld:

```
#demo {  
    @achtergrondKleur: #fff;  
    @tekstKleur: #ccc;  
    .codevenster{  
        background:@achtergrondKleur;  
        color:@tekstKleur  
    }  
}
```

dit kan je dus gebruiken als

```
.voorbeelden {  
    #demo .codevenster()  
}
```

We moeten dit nu nog enkel toepassen op een element: dat doen we in `structuur.less` op de container `#extra`:

```
/* extra kader */  
#extra {  
    margin:1em;  
    border: 4px solid @wit;  
    color:@koolstof;  
    background:@honing;  
    .paddingRondom();  
    .gradVertGroen();  
}
```

het resultaat ziet er zo uit:

Extra info

- Meer info over beroep en arbeidsvoorraarden: [bekijk de beroepenfiche](#)
- Dit is een opleiding die naar een *knelpuntberoep* leidt. **Je kan in aanmerking komen voor een premie!**
- Als werkzoekende heb je bepaalde rechten en plichten
- Deze opleiding is erkend door de VDAB. Als je als werkzoekende deze opleiding volgt, biedt de VDAB een aantal wettelijke garanties en voordelen (kinderopvang, verplaatsingskosten, ...)
- Heb je [hulp nodig bij het inschrijven?](#)

Taken:

maak nu volgende taken

- "Omzetten CSS naar LESS"

2.14 CSS omzetten naar LESS

Doel

We tonen hoe je de CSS voor een gesofisticeerd CSS3 menu kan omzetten naar LESS.

Dit project gaat diep in op de vele mogelijkheden binnenin LESS: we maken gebruik van een diep geneste structuur, geparametriseerde mixins, concatenatie, variabele variabelen, en bouwen zelf een lus structuur.

Theorie

- LESS documentatie op lesscss.org

Startbestanden

Dit project is oorspronkelijk verschenen als tutorial op Tutorialzine:

<http://demo.tutorialzine.com/2010/06/css3-minimalistic-navigation-menu/demo.html>.

We gebruiken het eindresultaat van deze tutorial om te leren converteren naar een LESS vorm. Maak gebruik van deze startbestanden:

menudemo.html, menudemo_styles.css

Open beide startbestanden en bekijk zowel HTML als CSS.

Je merkt dat het menu gebruik maakt van een **sprite**: *navigation.jpg* om alle knopjes te tonen. Controleer eventueel het pad van het beeldbestand in de CSS als je geen figuurtjes ziet.

2.14.1 Welke CSS omzetten in LESS?

We bekijken het CSS bestand inhoudelijk. Waar we vooral op letten:

1. **kleuren** die herhaald worden doorheen de volledige CSS
2. **elementen** die herhaald worden,
vb. `a{}`, `a:visited{}`, `a:hover{}`, `a img{}`, ...
3. **eigenschappen** die herhaald worden bij meerdere elementen, maar telkens met een andere waarde.
vb:

```
#menu1{
    background-color: #af1e83;
    color: #460f35;
    text-shadow: 0 0 5px #d8b54b;
}
en
#menu2{
    background-color: #d0a525;
    color: #604e18;
    text-shadow: 0 0 5px #c95f73;
}
```

Alles wat herhaald wordt gaan we omzetten naar

1. *variables*
2. *nested rules*
3. *mixins*

Dit demonstreren we aan de hand van een paar voorbeelden:

1. **variables**: declareer een naam en een waarde en gebruik deze variabele doorheen je code

```
@color: #4D926F;  
  
#header { color: @color; }  
h2 { color: @color; }
```

wordt na compilatie:

```
#header { color: #4D926F; }  
h2 { color: #4D926F; }
```

2. **nested rules**: het repeterende element wordt vervangen door een & teken en we gaan de elementen nesten

```
a {  
a:visited { }  
a:hover { }  
a img { }}
```

wordt:

```
a{  
  
  &:visited { }  
  &:hover { }  
  & img {}  
}
```

of indien de eigenschappen bij **visited**, **hover** en **img** dezelfde zijn:

```
a{  
  
  &:visited, &:hover, & img{}  
}
```

3. **mixins**: eigenschappen die in meerdere elementen voorkomen, zetten we in een **class** en roepen deze aan:

```
.menuItem{  
    background-color: #af1e83;  
    color: #460f35;  
    text-shadow: 0 0 5px #d8b54b;  
}  
#menu1{  
    .menuItem;  
}  
#menu2{  
    .menuItem;  
}
```

Mixins kunnen **parameters** gebruiken, dus is een nog betere manier van werken om de kleuren telkens door te geven.

```
.menuItem (@bgkleur, @kleur, @tshadow){  
    background-color: @bgkleur;  
    color: @kleur;  
    text-shadow: 0 0 5px @tshadow;  
}  
#menu1{  
    .menuItem(#af1e83, #460f35, #d8b54b);  
}  
#menu2{  
    .menuItem(#d0a525, #604e18, #c95f73);  
}
```

Je kan de kleuren ook op voorhand declareren in een variabele:

```
@rood:      #af1e83;  
@groen:     #460f35;  
@blauw:     #d8b54b;  
@geel:      #d0a525;  
@paars:     #604e18;  
@bruin:     #c95f73;
```

en het menu op deze manier opbouwen:

```
.menuItem (@bgkleur, @kleur, @tshadow){  
    background-color: @bgkleur;  
    color: @kleur;  
    text-shadow: 0 0 5px @tshadow;  
}  
#menu1{
```

```
.menuItem(@rood, @groen,@blauw);
}
#menu2{
    .menuItem(@geel, @paars, @bruin);
}
```

Na compilatie verkrijg je deze css code:

```
#menu1 {
    background-color: #af1e83;
    color: #460f35;
    text-shadow: 0 0 5px #d8b54b;
}
#menu2 {
    background-color: #d0a525;
    color: #604e18;
    text-shadow: 0 0 5px #c95f73;
}
```

2.14.2 Ons project

We keren terug naar onze opdracht: maak nu een `.less` bestand aan en hou het originele CSS bestand ernaast open. We bekijken opnieuw de inhoud van het CSS bestand.

De universele selector `*` : daarvan kan je de code niet vereenvoudigen, we nemen dit gewoon over naar het less bestand.

```
* {
/* A universal CSS reset */
margin:0;
padding:0;
}
```

Het `body` element bevat:

```
body{
    font-size:      14px;
    color:         #666;
    background:     #111 no-repeat;

    /* CSS3 Radial Gradients */
    background-image:-moz-radial-gradient(center -100px 45deg, circle farthest-corner, #444
150px, #111 300px);
    background-image:-webkit-gradient(radial, 50% 0, 150, 50% 0, 300, from(#444),
to(#111));
    font-family:Arial, Helvetica, sans-serif;
}
```

Hier zou je eventueel een aantal variabelen kunnen declareren, zoals de kleuren, maar omdat we ons vooral op het menu zullen concentreren nemen we dit stuk code gewoon over naar het less bestand.

Het volgende element in de rij is dit:

```
#navigationMenu li{  
    list-style:none;  
    height:39px;  
    padding:2px;  
    width:40px;  
}
```

Ook hier kan je niets omzetten maar als je de volgende elementen bekijkt valt het op dat het **id #navigationMenu** regelmatig herhaald wordt doorheen het volledige css bestand. We kiezen er dan ook voor om gebruik te maken van *nested rules*, m.a.w. we gaan alle elementen die beginnen met dit **id** groeperen.

Het betreft deze elementen:

```
#navigationMenu li{...}  
  
#navigationMenu span{...}  
  
#navigationMenu a{...}  
  
#navigationMenu a:hover span{...}  
  
#navigationMenu a:hover{...}  
  
#navigationMenu .home {...}*  
  
#navigationMenu .home:hover {...}*  
  
#navigationMenu .home span{...}*  
  
#navigationMenu .about {...}*  
  
#navigationMenu .about:hover {...}*  
  
#navigationMenu .about span{...}*  
  
#navigationMenu .services {...}*  
  
#navigationMenu .services:hover {...}*  
  
#navigationMenu .services span{...}*
```

```
#navigationMenu .portfolio {...}*  
  
#navigationMenu .portfolio:hover {...}*  
  
#navigationMenu .portfolio span{...}*  
  
#navigationMenu .contact {...}*  
  
#navigationMenu .contact:hover {...}*  
  
#navigationMenu .contact span{...}*
```

Opmerking *:

- merk op dat bij deze elementen telkens een styling gebeurt van
 - de **class** zelf (vb **.portfolio**)
 - de pseudoclass **:hover** ervan en
 - het child element **span**

Ideaal zou dus zijn dat je deze 3 styling regels slechts éénmaal zou moeten oplijsten. Nog idealer zou zijn dat je dan ook nog gebruik maakt van variabelen om de properties een waarde te geven.

We hebben 5 menu item **classes**: *home*, *about*, *services*, *portfolio* en *contact*. Zou het mogelijk zijn om gebruik te maken van een lussen structuur in less? Dit zouden we willen bereiken:

```
voor teller := 1 t/m 5 {  
    #navigationMenu .m_itemteller {...}  
    #navigationMenu .m_itemteller:hover {...}  
    #navigationMenu .m_itemteller span{...}  
}
```

We houden dit nog even in gedachten en gaan eerst de andere elementen met id **#navigationMenu** omzetten.

```
#navigationMenu li{  
    ...  
}  
#navigationMenu span{  
    ...  
}
```

```
#navigationMenu a{  
...  
}  
#navigationMenu a:hover span{  
...  
}  
#navigationMenu a:hover{  
...  
}
```

wordt in less als volgt geschreven

```
#navigationMenu {  
  
    li {  
        ...  
    } /* einde li */  
    span {  
        ...  
    } /* einde span */  
    a {  
        ...  
        &:hover {  
            ...  
            & span{  
                ...  
            } /* einde & span */  
  
        } /* einde &:hover */  
  
    } /* einde a */  
  
} /* einde #navigationMenu */
```

De volledige code met eigenschappen inbegrepen ziet eruit als volgt:

```
#navigationMenu {  
    li {  
        list-style: none;  
        height: 39px;  
        padding: 2px;  
        width: 40px;  
    } /* einde li */  
    span {  
        /* Container properties */  
        width: 0;  
        left: 38px;
```

```
padding:      0;
position:     absolute;
overflow:    hidden;

/* Text properties */
font-family:  'Myriad Pro', Arial, Helvetica, sans-serif;
font-size:    18px;
font-weight:   bold;
letter-spacing:0.6px;
white-space: nowrap;
line-height:  39px;

/* CSS3 Transition: */
-webkit-transition:  @seconden;

/* Future proofing (these do not work yet): */
-moz-transition:    @seconden;
transition:         @seconden;
} /* einde span */

a {
background: url('img/navigation.jpg') no-repeat;
height:    39px;
width:    38px;
display:   block;
position:  relative;

&:hover {
text-decoration:none;
/* CSS outer glow with the box-shadow property */
-moz-box-shadow:    0 0 5px @outerglow;
-webkit-box-shadow: 0 0 5px @outerglow;
box-shadow:         0 0 5px @outerglow;

& span{
width:      auto;
padding:    0 20px;
overflow:   visible;
} /* einde & span */
} /* einde &:hover */
} /* einde a */
} /* einde #navigationMenu */
```

Opmerkingen:

- we hebben in één beweging ook een aantal variabelen toegevoegd, nl. **@seconden** en **@outerglow**.
- Declareer deze variabelen bovenaan je css bestand:

```
@seconden: 0.25s;  
@outerglow: #9ddff5;
```

2.14.3 Lusstructuur in LESS

We keren terug naar het idee om de elementen van het menu in een lussen structuur onder te brengen.



Lees het topic over 'Guards' op <http://lesscss.org/> en zie hoe je dit kan gebruiken om een voorwaardelijke lus te construeren:

```
.lusGuardedMixin (@index) when (@index < 6) {  
  
    element{  
        eigenschap ...  
    }  
  
    .lusGuardedMixin(@index + 1); /* verhoog de lus telkens met 1 */  
}  
  
.lusGuardedMixin (6) {} /* eindig de lus nadat het max. aantal lussen bereikt werd */  
  
.lusGuardedMixin (1); /* start de lus */
```

Nu moeten we enkel nog een manier vinden om het *home*, *about*, *services*, ..., menu in de lus te verwerken. Dit kunnen we eenvoudig oplossen door de naam van de **class** van de menu items in de html code te wijzigen als volgt:

```
<li><a class="home" href="#"><span>Home</span></a></li>  
<li><a class="about" href="#"><span>About</span></a></li>  
<li><a class="services" href="#"><span>Services</span></a></li>  
<li><a class="portfolio" href="#"><span>Portfolio</span></a></li>  
<li><a class="contact" href="#"><span>Contact us</span></a></li>
```

wordt:

```
<li><a class="m_item1" href="#"><span>Home</span></a></li>  
<li><a class="m_item2" href="#"><span>About</span></a></li>  
<li><a class="m_item3" href="#"><span>Services</span></a></li>  
<li><a class="m_item4" href="#"><span>Portfolio</span></a></li>  
<li><a class="m_item5" href="#"><span>Contact us</span></a></li>
```

Pas de HTML code aan.

De structuur van de lus kennen we al, we dienen enkel nog de styling regels om te zetten. We vertrekken van deze code:

```
#navigationMenu .m_item1 {  
    background-position: 0px 0;  
}  
#navigationMenu .m_item1:hover {  
    background-position: 0px -39px;  
}  
#navigationMenu .m_item1 span {  
    background-color: #7da315;  
    color: #3d4f0c;  
    text-shadow: 0 0 5px #99bf31;  
}
```

... in totaal 5 menu items.



Lees ook het topic over 'Escaping' op <http://lesscss.org/> meer bepaald over hoe je bepaalde code kan *escapen*.

We willen de **class** `.m_itemteller` schrijven binnen de lus, dit doen we in less als volgt:

(~".m_item@{index}")

- `@index` is de teller
- om een variabele `@var` binnen een **string** te schrijven en te laten interpoleren, moet je die noteren als `@{var}`
- de waarde van `@{index}` wordt dus in de lus door de waarde van de teller ingevuld en nadien aan de tekst "`.m_item`" geconcateneerd
- je bekomt uiteindelijk de waarden `.m_item1`, `.m_item2`, ..., `.m_item5`

Pas de *nested rules* toe voor `m_itemteller`, `m_itemteller:hover` en `m_itemteller span`. Dit moet je in principe nu reeds in de vingers hebben.

Nu rest ons enkel nog de eigenschappen binnenin de 'nested rules' aan te passen.

Binnenin `m_itemteller` vind je de eigenschap **background-position**. Die wordt gebruikt om de vertikale en horizontale positie van de sprite als achtergrond te bepalen.

Als je dit voor elk menu item nakijkt dan zie je dat er gestart wordt op `0px` en dat er telkens `38px` (de breedte van één figuurtje in de *sprite*) afgetrokken worden. Dit kan je eenvoudig in je lus berekenen, gebruik makend van de teller van de lus.

De berekening is als volgt `-((38 * waardeteller)-38)`, de eerste keer heeft de teller de waarde `1`, dus `- ((38 * 1) - 38) = 0`, in de tweede lus wordt dat `- ((38 * 2) - 38) = - 38`, enz..

We stoppen de berekening in een aparte variabele `@waarde`, doen we dit niet dan gaat less na de compilatie een spatie zetten tussen de berekende waarde en de eenheid `px` en zal je css code niet werken!

Nadien gaan we via ‘escaping’ nog de tekst "px" concateneren aan de waarde die we berekend hebben.

```
@waarde: -((38*@index)-38);
background-position: ~"#{@waarde}px" 0;
```

De class `.m_itemteller:hover` bevat dezelfde berekening die we reeds gedaan hebben, we kunnen de variabele `@waarde` dus hergebruiken, op voorwaarde dat we deze als eerste regel code binnen de lus zetten.

2.14.4 Achtergrondkleuren

De class `.m_itemteller span` bevat 3 styling regels, die steeds herhaald worden, we maken hiervoor een mixin als volgt:

```
.menuspan (@bgc, @col, @tshad) {
  background-color: @bgc;
  color: @col;
  text-shadow: 0 0 5px @tshad;
}
```

en we kunnen ook gebruik maken van de lus structuur om onze variabelen op te vullen, we definiëren daarom deze variabelen bovenaan:

```
//m_item1
@bgcolor_1:      #7da315;
@color_1:        #3d4f0c;
@tshadow_1:      #99bf31;
//m_item2
@bgcolor_2:      #1e8bb4;
@color_2:        #223a44;
@tshadow_2:      #44a8d0;
//m_item3
@bgcolor_3:      #c86c1f;
@color_3:        #5a3517;
@tshadow_3:      #d28344;
//m_item4
@bgcolor_4:      #d0a525;
@color_4:        #604e18;
@tshadow_4:      #d8b54b;
//m_item5
@bgcolor_5:      #af1e83;
@color_5:        #460f35;
@tshadow_5:      #d8b54b;
```

De *mixin .menuspan* die we zopas aangemaakt hebben klopt niet volledig, de structuur zit goed maar aangezien we niet rechtstreeks aan onze ‘mixin’ de *hexwaarde* van de kleur meegeven, maar een andere variabele, dienen we deze aan te passen.



Lees de topic ‘Variables’ op <http://lesscss.org/> en meer bepaald het voorbeeld ‘*it is also possible to define variables with a variable name*’.

We zetten dus een extra @ voor de variabele, het resultaat ziet eruit als volgt:

```
.menuspan (@bgc, @col, @tshad) {  
    background-color: @@bgc;  
    color: @@col;  
    text-shadow: 0 0 5px @@tshad;  
}
```

We integreren nu alle styling regels in onze lus met als eindresultaat dit:

```
.lusGuardedMixin (@index) when (@index > 0) {  
  
    @waarde: -((38*@index)-38);  
  
    (~".m_item@{index}") {  
        background-position: ~"@{waarde}px" 0;  
  
        &:hover {  
            background-position: ~"@{waarde}px" -39px;  
        }  
  
        & span {  
            .menuspan(~"bgcolor_@{index}", ~"color_@{index}", ~"tshadow_@{index}")  
        }  
    }  
    .lusGuardedMixin(@index + 1);  
}  
.lusGuardedMixin (6) {}  
.lusGuardedMixin (1);
```

We hebben het meest complexe stuk code van de css omgezet naar less. De code die nu nog overblijft is deze:

```
#main{  
    margin:80px auto;
```

```
position:relative;
width:40px;
}
h1{
  color:#fff;
  font-size:30px;
  font-weight:normal;
  padding:60px 0 20px;
  text-align:center;
}
h2{
  font-weight:normal;
  text-align:center;
}
h1,h2{
  font-family:"Myriad Pro",Arial,Helvetica,sans-serif;
}
a, a:visited,a:active {
  color:#0196e3;
  text-decoration:none;
  outline:none;
}
a:hover{
  text-decoration:underline;
}

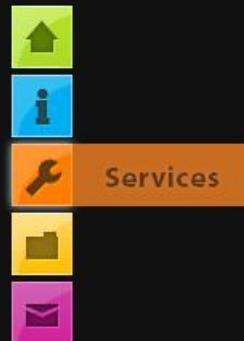
a img{
  border:none;
}

p.note{
  color:#707070;
  font-size:10px;
  text-align:center;
  margin:50px;
}
```

Je zou eventueel nog een aantal variabelen kunnen declareren maar we gaan er hier verder niet meer op in. Kopieer de resterende code naar het less bestand en compileer!

CSS3 Minimalistic Navigation Menu

[Read the tutorial on Tutorialzine »](#)



Dit is een copy van het online voorbeeld op <http://demo.tutorialzine.com/2010/08/css3-minimalistic-navigation-menu/demo.html>

2.15 Modernizr

Doel

Gebruik maken van *Modernizr* om browserproblemen met HTML5 en CSS3 features op te lossen.

Theorie

- CSS3 features

Startbestanden

modernizr_gebruiken.html, algemeen.css, normalize.css

2.15.1 Wat is Modernizr?

Niet alle browsers ondersteunen de nieuwste HTML5 en CSS3 features.

Modernizr is een Javascript library die ons kan helpen HTML5 en CSS3 feature ondersteuning te detecteren en (soms) te verhelpen. De *HTML5 shim* die we al enkele keren gebruiktten, zit bij Modernizr ingebouwd, dus die hoef je niet meer apart te koppelen.

De shim maakt echter alleen de *missing elements* aan, meer niet, het voegt geen enkele HTML5 of CSS3 functionaliteit toe! Modernizr is dus enkel een **testframework**.

Het bevat ook nog de Javascript library *yepnope.js* (zie download opties), die gebruikt kan worden om scripts "*on demand*" op te roepen.

Modernizr zelf zit soms ingesloten in andere Frameworks, bijvoorbeeld in *Bootstrap* en in *Foundation*.

Wat kan je er mee doen?

- selectief CSS selectors toepassen om toch nieuwe CSS3 toe te passen op nieuwe browsers en de problemen op te vangen in oudere browsers
- selectief javascripts opstarten die HTML5 features gebruiken (via *yepnope.js*)
- HTML5 sectioning elementen opmaken

2.15.2 Download Modernizr

Download de laatste versie van Modernizr.js vanaf <http://modernizr.com/>. Ga naar de download pagina, kies de production version en stip alle opties aan, en ook **Modernizr.load** (*yepnope.js*) en media queries:

Download Modernizr 2.6.2

Use the [Development version](#) to develop with and learn from. Then, when you're ready for production, use the build tool below to pick only the tests you need.

CSS3 TOGGLE

- @font-face
- background-size
- border-image
- border-radius
- box-shadow
- Flexible Box Model (flexbox)
- Flexbox Legacy
- hsla()
- multiple backgrounds
- opacity
- rgba()
- text-shadow
- CSS Animations
- CSS Columns
- CSS Generated Content (before/after)
- CSS Gradients
- CSS Reflections
- CSS 2D Transforms
- CSS 3D Transforms
- CSS Transitions

HTML5 TOGGLE

- applicationCache
- Canvas
- Canvas Text
- Drag 'n Drop
- hashchange
- History (pushState)
- HTML5 Audio
- HTML5 Video
- IndexedDB
- Input Attributes

Note: does not add classes
- Input Types

Note: does not add classes
- localStorage
- postMessage
- sessionStorage
- Web Sockets
- Web SQL Database
- Web Workers

Misc. TOGGLE

- Geolocation API
- Inline SVG
- SMIL
- SVG
- SVG clip paths
- Touch Events
- WebGL

Extra

- html5shiv v3.6
- html5shiv v3.6 w/ printshiv
- Modernizr.load ([yepnope.js](#))
- Media Queries
- Add CSS Classes

className prefix:

Je krijgt een *custom download*, vb *modernizr.custom.82634.js*. Plaats dat in de *js* map van je website.

Open het startbestand en plaats een koppeling naar de library in een script tag:

```
<script src="../js/modernizr.custom.82638.js"></script>
```

Voeg ook een *class* toe aan het *html* element:

```
<html class="no-js">
```

Modernizr zal deze *class* vervangen door een massa andere classes. Gebeurt dat niet, dan is er iets aan de hand, bv. Javascript is niet geactiveerd in de browser van de gebruiker.

Bekijk nu de pagina in een browser en gebruik een *developer tool* om de gegenereerde code te bekijken. Dan merk je dat het *class* attribuut in de *html* tag opgevuld is met een heleboel classes, bv.

```
<html class=" js flexbox flexboxlegacy canvas canvastext webgl no-touch geolocation postmessage websqldatabase indexeddb hashchange history draganddrop websockets rgba hsla multiplebgs backgroundsize borderimage borderradius boxshadow textshadow opacity cssanimations csscolumns cssgradients cssreflections csstransforms csstransforms3d csstransitions fontface generatedcontent video audio localstorage sessionstorage webworkers applicationcache svg inlinesvg smil svgclippaths" style="">
<head>
```

...

Wat je ziet zijn alle features die getest zijn, bijvoorbeeld:

- **multiplebg**s staat voor de CSS3 eigenschap *multiple backgrounds*, die dus ondersteund wordt door je browser/toestel
- **no-touch** betekent dat het feature **touch** niet ondersteund wordt door deze browser/toestel. Geteste features die een negatief resultaat opleveren krijgen de **no-** prefix
- vergeet niet dat deze classes verschillen afhankelijk van de browser waarin je ze bekijkt!

2.15.3 CSS3 browser ondersteuning

Sommige CSS3 features werken enkel in de allerlaatste versies van een browser.



Het is altijd noodzakelijk om een pagina met CSS3 features te testen in verschillende browsers, inclusief mobiele browsers.

Om goed te kunnen testen heb je ook minstens één oudere browser nodig: via de developer's tool van IE kan je terugschakelen naar browser modus IE7.

De volgende sites bieden een goed overzicht van de ondersteuning:

- www.quirksmode.org bij *Compatibility – CSS*.
- caniuse.com

Voor de juiste syntax van de CSS3 features die we hierna gebruiken, lees je de theorie erop na.

Er zullen dus NIET browsers zijn en WEL browsers. Daarom gebruiken we *progressive enhancement*: eerst zorgen voor de "mindere" browsers en daarna voorzien we voor de "betere browser".

Door gebruik te maken van de **class** waarden in het **html** element, kunnen we selectief tewerk gaan: *contextuele CSS*.

2.15.4 Contextuele CSS

Een van de nieuwe CSS3 features is de mogelijkheid om *multiple background images* toe te passen op dezelfde container. Gedaan zijn de tijden waar je twee/drie extra **div**'s moest voorzien om verschillende achtergronden te gebruiken.

Maar lukt dat wel altijd?

Het startbestand heeft al twee koppelingen naar *normalize.css* en *algemeen.css*, een minimale stylesheet voor het oefenbestand.

Maak nu een derde koppeling naar een nieuw stylesheet *mod_gebruiken.css* en maak dat stylesheet aan als leeg CSS bestand.

```
<link rel="stylesheet" href="css/normalize.css" >
<link rel="stylesheet" href="css/algemeen.css" >
<link rel="stylesheet" href="css/mod_gebruiken.css">
<script src="../js/modernizr.custom.82638.js"></script>
```

2.15.5 Multiple backgrounds

We starten met een eenvoudige achtergrond. We voegen aan *mod_gebruiken.css* een selector toe voor de **header**:

```
header {
    min-height: 220px;
    background: url(../../img/clouds_groot.png) 50% top repeat-x;
}
```

bespreking:

- de **header** krijgt een minimale hoogte via **min-height**
 - en een achtergrondfiguur die horizontaal, centraal geplaatst en herhaald wordt.
- Pas eventueel het pad naar het image aan.

Alle browsers kunnen dit aan. Nu maken we gebruik van de classes die Modernizr voor ons voorzien heeft in het *root* element.

We voegen toe ná de vorige selector:

```
header { ... }
.multiplebgs header {
    background: url(../../img/clouds_groot.png) 50% top repeat-x, ⚡
                url(../../img/clouds_klein.png) right bottom repeat-x;
}
```

bespreking:

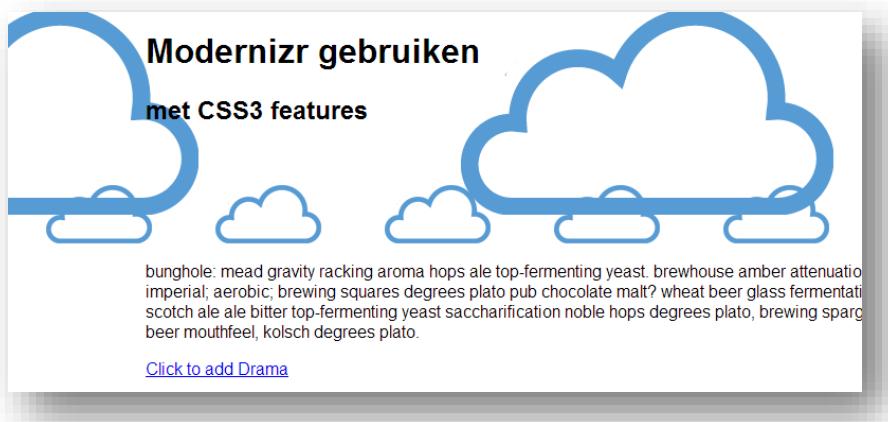
- de selector **.multiplebgs header** heeft een **grotere specificiteit** dan **header** alleen en overschrijft daarmee het vorige statement, maar natuurlijk enkel voor die browsers waar de **class .multiplebgs** in de **html** tag staat!

Controleer in verschillende browsers:

In IE7:



In Chrome:



Speel ook eens met de breedte van het venster in Chrome bemerkt je de verschillende bewegingen van de wolken: een *parallax effect*.

Inspelen op de specificiteit van deze selectoren noemen we ook **contextuele CSS**: alleen in de *context* van `.multiplebgs` wordt dit toegepast.

Wat zou er gebeuren moesten we de syntax van multiple backgrounds gebruiken zonder de `.multiplebgs` context?

In dit geval zou je gewoon een enkele (de eerste) background krijgen, maar in andere gevallen, als je meerdere style rules in de selector stopt, zou het kunnen dat de browser – die iets niet begrijpt – de volledige selector laat vallen!

Nu kunnen we alle CSS3 features veilig gaan gebruiken.

2.15.6 Textshadow

Nog zo'n nieuwheidje is *textshadow*. Die gebruiken we op de `h1` en `h2`:

```
.textshadow h1, .textshadow h2 {  
  -moz-text-shadow: 20px 20px 5px silver;  
  -webkit-text-shadow: 20px 20px 5px silver;  
  -o-text-shadow: 20px 20px 5px silver;
```

```
text-shadow: 20px 20px 5px silver;
}
```

bespreking:

- we passen opnieuw een Modernizr class toe: `.textshadow`
- deze rules gebruiken browserprefixes. Die zijn voor deze eigenschap grotendeels overbodig want de standard property wordt ondersteund in de meeste browsers. Toch gebruiken we ze voor de veiligheid

Het resultaat in *Opera Mobile Emulator* voor een *HTC Desire*:



2.15.7 features combineren

In de tekst vind je twee hyperlinks met de `class` "drama".

We willen die links tonen als mooie knoppen door gebruik te maken van de CSS3 features *kleurovergangen*, *ronde hoeken* en *schaduw*.

Omdat het letterlijk uitschrijven van een *gradient* niet evident is, maken we gebruik van een *gradient generator* zoals *Colorzilla* of de *CSS-Tricks button generator* (en er zijn er meer...).

Zorg er alleszins voor dat je code toegepast wordt op de `a.drama` en dat ze de verschillende features bevat zoals hieronder:

```
a.drama { margin-right:1em; }
a.drama:link, a.drama:visited {

    color: #CCC;
    font-size: 1.2em;
    font-family: Helvetica, Arial, Sans-Serif;
    text-decoration: none;
    vertical-align: middle;
    text-shadow: rgba(0,0,0,0.4) 0 1px 0;
```

```
padding: 0.5em 1em;
border-top: 1px solid #f7afaf;

background: #f0182a;
background: -moz-linear-gradient(top, #f0182a 1%, #e872e8 100%);
background: -webkit-gradient(linear, left top, left bottom, color-stop(1%,#f0182a), color-stop(100%,#e872e8));
background: -webkit-linear-gradient(top, #f0182a 1%,#e872e8 100%);
background: -o-linear-gradient(top, #f0182a 1%,#e872e8 100%);
background: -ms-linear-gradient(top, #f0182a 1%,#e872e8 100%)/;
background: linear-gradient(to bottom, #f0182a 1%,#e872e8 100%);
filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#f0182a', endColorstr='#e872e8',GradientType=0 );

-webkit-border-radius: 1em;
-moz-border-radius: 1em;
border-radius: 1em;

-webkit-box-shadow: rgba(0,0,0,1) 3px 3px 3px;
-moz-box-shadow: rgba(0,0,0,1) 3px 3px 3px;
box-shadow: rgba(0,0,0,1) 3px 3px 3px;
}

a.drama:hover {
border-top-color: #e609f2;
background: #e609f2;
color: #f5edf5;
}
a.drama:active {
border-top-color: #f70a0a;
background: red;
color: white;
}
```

Bespreking:

- eerst voegen we wat **margin-right** toe om de knoppen uit elkaar te houden
- je hoeft niet dezelfde kleuren te gebruiken
- zorg ervoor dat je goed het onderscheid kunt maken tussen de verschillende opmaak features:
 - we hebben voor de hyperlink in normale staat
 - teksteigenschappen met **text-shadow**
 - **padding** en een bovenrand
 - gradient kleur
 - ronde hoeken met **border-radius**
 - slagschaduw met **box-shadow**
- voor de **:hover** en **:active** pseudo-classes gebruiken we een volle kleur

Dit toont mooi in een browser die het allemaal ondersteunt, zoals Opera12:



Click to add Drama

Maar helemaal niet zo mooi in een browser die deze features gedeeltelijk of helemaal niet ondersteunt: bv. IE9,8,7:



Click to add Drama

Hier werkt *gradient* niet, en de ronde hoekjes en de schaduw zijn van slechte kwaliteit.

Dank zij Modernizr kunnen we nu een selector schrijven die enkel zal toegepast worden **als alle features** ondersteund worden, zo niet kiezen we resoluut voor een eenvoudiger oplossing.

Ook hier werken we best volgens het principe van *progressive enhancement*, m.a.w. we voorzien **eerst** de opmaak die **alle** browsers aankunnen, pas daarna zorgen we voor het betere werk. Om die reden moeten we alle "onzekere" opmaak uit de gewone selector houden en onderbrengen in een volgende selector:

```
a.drama:link, a.drama:visited {  
    color: #CCC;  
    font-size: 1.2em;  
    font-family: Helvetica, Arial, Sans-Serif;  
    text-decoration: none;  
    vertical-align: middle;  
    padding: 0.5em 1em;  
    border-top: 1px solid #f7afaf;  
    background: #f0182a; /* Old browsers */  
}  
a.drama:hover {  
    border-top-color: #e609f2;  
    background: #e609f2;  
    color: #f5edf5;  
}  
a.drama:active {  
    border-top-color: #f70a0a;  
    background: red;  
    color:white;  
}  
.cssgradients.boxshadow.borderradius a.drama:link,  
.cssgradients.boxshadow.borderradius a.drama:visited {  
    text-shadow: rgba(0,0,0,0.4) 0 1px 0;  
    background: -moz-linear-gradient(top, #f0182a 1%, #e872e8 100%);
```

```
background: -webkit-gradient(linear, left top, left bottom, color-stop(1%,#f0182a), color-stop(100%,#e872e8));
background: -webkit-linear-gradient(top, #f0182a 1%,#e872e8 100%);
background: -o-linear-gradient(top, #f0182a 1%,#e872e8 100%);
background: -ms-linear-gradient(top, #f0182a 1%,#e872e8 100%);
background: linear-gradient(to bottom, #f0182a 1%,#e872e8 100%);
filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#f0182a', endColorstr='#e872e8',GradientType=0 );

-webkit-border-radius: 1em;
-moz-border-radius: 1em;
border-radius: 1em;

-webkit-box-shadow: rgba(0,0,0,1) 3px 3px 3px;
-moz-box-shadow: rgba(0,0,0,1) 3px 3px 3px;
box-shadow: rgba(0,0,0,1) 3px 3px 3px;

}
```

bespreking:

- alle CSS2 eigenschappen blijven in de *basis* selector
- alle CSS3 eigenschappen zijn vervat in de *contextuele* selector

.cssgradients.boxshadow.borderradius a.drama

Bemerkt de combinatie (het aaneensluiten) van de Modernizr classes:
dit komt neer op een AND logica:

cssgradient AND boxshadow AND borderradius
moet ondersteund worden, anders beginnen we er niet aan.

Als je vindt dat sommige features wel kunnen, dan pas je dit aan natuurlijk: we demonstreren hier enkel het principe.

2.15.8 Animations

Tenslotte proberen we even wat beweging in de zaak te krijgen met CSS3 animations.

Eerst splitsen we de *multiple background* rules op in:

```
.multiplebgs header {
  background-image: url(../../img/clouds_groot.png), url(../../img/clouds_klein.png);
  background-position: 100% 0%, 0% 100%;
  background-repeat: repeat-x, repeat-x;
}
```

Nu voegen we **animation** properties toe:

```
.multiplebgs header {
  background-image: url(../../img/clouds_groot.png), url(../../img/clouds_klein.png);
```

```
background-position: 100% 0%,0% 100%;  
background-repeat: repeat-x, repeat-x;  
  
-webkit-animation-name: zweven;  
-webkit-animation-duration: 10s;  
-webkit-animation-iteration-count: infinite;  
-webkit-animation-direction: alternate;  
-webkit-animation-timing-function: ease-in-out;  
  
animation-name: zweven;  
animation-duration: 10s;  
animation-iteration-count: infinite;  
animation-direction: alternate;  
animation-timing-function: ease-in-out;  
}
```

Er wordt verwezen naar een benoemd **@keyframes** sleutelwoord, dat plaatsen we er nu onder:

```
@keyframes zweven {  
    0% {background-position: 0% 0%,100% 100%;}  
    50% {background-position: 50% 0%,50% 50%;}  
    100% {background-position: 100% 0%,0% 100%;}  
}  
@-webkit-keyframes zweven {  
    FROM {background-position: 0% 0%,100% 150%;}  
    50% {background-position: 50% 0%,50% 50%;}  
    TO {background-position: 100% 0%,0% 150%;}  
}
```

Je bemerkt natuurlijk de browser prefixes voor webkit browsers. Nu bewegen je achtergronden!

We kunnen ook nog iets extra's toevoegen: een opkomende zon:

Voeg de volgende HTML toe aan de **header**:

```
<header>  
    <div class="content">  
        <h1>Modernizr gebruiken</h1>  
        <h2>met CSS3 features</h2>  
    </div>  
    <div id="zon"></div>  
</header>
```

Maak de header klaar:

```
header {  
    min-height: 320px;  
    background: transparent url(../../../../img/clouds_groot.png) 50% top repeat-x ;  
    position: relative;
```

```
}
```

Maak de zon klaar:

```
#zon{
    position: absolute;
    z-index:-10;
    bottom:5em;
    left:0;
    width:4em;
    height:4em;
    background-color:orange;
    border-radius:50%;
    opacity:0;
    transform:scale(0.5);

    -webkit-animation-name: rijzen;
    -webkit-animation-duration: 10s;
    -webkit-animation-iteration-count: 1;
    -webkit-animation-direction: normal;
    -webkit-animation-timing-function: linear;

    animation-name: rijzen;
    animation-duration: 10s;
    animation-iteration-count: 1;
    animation-direction: normal;
    animation-timing-function: linear;

}
```

Maak de @keyframes:

```
@keyframes rijzen {
    0% {top:400px; left:0; opacity:0; width:4em; height:4em;
        box-shadow: #FCC 0px 0px 10px 10px ;}
    50% {top:0; left:50%; opacity:1; background-color:yellow; width:6em; height:6em;
        box-shadow: #FF6 0px 0px 200px 200px ;}
    100% {top:400px; left:100%; opacity:1; background-color:red; width:8em; height:8em;
        box-shadow: #F69 0px 0px 200px 200px ;}
}

@-webkit-keyframes 'rijzen' {
    0% {top:400px; left:0; opacity:0; width:4em; height:4em;
        -webkit-box-shadow: #FCC 0px 0px 10px 10px ;}
    50% {top:0; left:50%; opacity:1; background-color:yellow; width:6em; height:6em;
        -webkit-box-shadow: #FF6 0px 0px 200px 200px ;}
    100% {top:400px;left:100%; opacity:1; background-color:red; width:8em; height:8em;
        -webkit-box-shadow: #F69 0px 0px 200px 200px ;}
}
```

Nu komt de zon ook op!

COLOFON

Domeinexpertisemanager	Rita Van Damme
Moduleverantwoordelijke	Jean Smits
Auteurs	Adinda Mattens Ilse Palmaers Jan Vandorpe
Versie	10/1/2014
Codes	Peoplesoftcode: Wettelijk depot:

Omschrijving module-inhoud

Abstract	Doelgroep	ICT cursisten
	Aanpak	zelfstudiecursus
	Doelstelling	Ontwerpen van webpagina's met html en css, theorie
Trefwoorden		Html css html5 css3 webpagina website
Bronnen/meer info		