

Inhoud

1	Introductie	1
	Inleiding	2
	Wat flexbox is	4
	Kenmerken	5
	Wat flexbox kan	8
	Wat flexbox oplost	9
	Hello flexbox	10
	Verticaal centreren	14
	Benodigde voorkennis	17
	Tips voor meer leesvoer	17
	Editor en browser	18
	Voorbeeldcode downloaden	19
	Samenvatting	20
	Praktijkoefeningen	20
2	De basis van flexbox	21
	Flexcontainers en flexitems	22
	Een flexcontainer maken	23
	Flexitems	24
	De inhoud van flexitems	25
	Ouders en kinderen	26
	Rijen en kolommen: flex-direction	27
	Twee dimensies	29
	Flex-direction toegepast	30
	Regelomloop: flex-wrap	34
	De verzameleigenschap flex-flow	36
	De volgorde veranderen: order	37
	Use case voor order	39
	Samenvatting	42
	Praktijkoefeningen	42

3	Flexitems verdelen	45
	Automatische marges	46
	Use case met automarge	49
	Automarge op kolommen	51
	Flexeigenschappen voor uitlijnen	53
	Uitlijnen in de hoofdrichting	54
	Justify-content (flexcontainer)	55
	Toelichting op de code	57
	Uitlijnen in de kruisrichting	58
	Align-items (container) en align-self (item)	58
	Align-content (flexcontainer)	60
	De producten uitlijnen	61
	Use cases uitlijnen	63
	Navigatiebalk	64
	Samenvatting	66
	Opdrachten	66
4	Een flexibel formaat	69
	Groeien en krimpen met flex	70
	Als flex niet wordt ingesteld	70
	Aanbevolen instelling: flex	71
	Het basisformaat: flex-basis	75
	Absolute en relatieve flex	77
	Flex-grow en flex-shrink	78
	Flex-shrink	79
	Conclusie	81
	Samenvatting	81
	Praktijkoefeningen	82
5	Voorbeelden van flexbox	85
	Website nos.nl	86
	Smalle weergave	87
	Breakpoint 37rem	90
	Breakpoint op 47rem	91
	Breakpoint 69rem en breder	93
	Venster voor instellingen	95
	De hoofdstructuur	97
	De header	98
	De lijst met opties	99
	De brede weergave	101

Kop en onderkop naast of onder elkaar	104
Items in een regel splitsen	105
Split screen	107
Split screen met footer	110
Hovereffect met flex	115
Show case: dobbelstenen	117
De basis	117
De één	118
De twee	118
De drie	119
De vier	119
De vijf	120
De zes	121
Samenvatting	122
6 Een opmaakraster maken met Grid Layout	123
Introductie	124
De status van gridlay-out	125
Kenmerken van gridlay-out	126
CSS-eigenschappen voor gridlay-out	128
Definitie van het raster	128
Plaatsing van de items	129
Uitlijning en tussenruimte	129
Workflow in vogelvlucht	130
Een raster maken	132
Auto-flow	135
Tussenruimte (gap)	136
Waarden voor tracks	137
De functie repeat()	138
Expliciet en impliciet raster	141
Kort: grid-template	142
Nog korter: grid	144
Samenvatting	146
Praktijkopdrachten	147
7 Items in het raster plaatsen	149
Plaatsingsmethoden	150
Plaatsen op index	152
Plaatsen op benoemde lijn	157
Index en naam combineren	159
Verkorte notatie	159
Kort met grid-area	160

Plaatsen in benoemd vlak	161
Automatisch plaatsen	164
Basisafmetingen	165
Span	167
Auto-fit en dense	168
Uitlijnen	171
Items uitlijnen in de rij	171
Items uitlijnen in de kolom	172
Items centreren met margin	173
Tracks uitlijnen	174
Samenvatting	177
Praktijkopdrachten	178
8 Voorbeelden van grid-layout	179
Website nos.nl	180
Smalle weergave	181
Breakpoint 37rem	183
Breakpoint op 47rem	184
Breakpoint 69rem en breder	186
Instellingenvenster	187
Homepage	189
Conclusie	194
Index	197

Introductie

Webdevelopment vereist steeds meer flexibiliteit. Misschien van de developer, maar zeker van het eindproduct: de pagina, app of site. Zelden is vooraf bekend op welk schermformaat de HTML-pagina zal worden bekeken. De inhoud moet echter wel altijd leesbaar en bruikbaar zijn en er ook nog aantrekkelijk uitzien. Dat zijn nogal wat eisen. CSS, de opmaaktaal voor HTML, is uitgebreid met twee technieken die flexibel ontwerp stukken eenvoudiger maken. Beide zijn rijp voor de productieomgeving. Maak kennis met flexbox en gridlay-out.

U leert in dit hoofdstuk:

Welke mogelijkheden Flexible Box Layout en Grid Layout bieden.

Wat flexbox is, wat het kan en welke problemen het oplost.

Hoe u met één CSS-declaratie een flexibele tweekolomslay-out maakt.

Inleiding

Dit boek gaat over het maken van lay-outs voor webpagina's met het nieuwste dat CSS te bieden heeft: Flexible Box Layout en Grid Layout. Deze technieken maken het leven van webdevelopers stukken eenvoudiger.

De technieken zijn bedoeld om op een eenvoudiger manier complete webpagina's, apps en onderdelen van de gebruikersinterface te lay-outen dan tot nog toe mogelijk was. Naast eenvoud is een kenmerk dat het resultaat flexibel is: kolommen en inhoudsgebieden passen zich automatisch aan de beschikbare ruimte aan. Het is fantastisch om te zien hoe een complete lay-out zich aanpast van mobiel tot desktopscherm en alles wat daartussen zit (en daarboven, denk aan grote presentatieschermen en tv's), hoe een menu naadloos van beknopt naar full size gaat, hoe een Windows-achtige tegelinterface altijd klopt of hoe het aantal tekstkolommen toe- of afneemt bij verschillende schermformaten. Dit klinkt als floats met media queries en oppervlakkig gezien is er zeker gelijkenis, maar de mogelijkheden zijn groter, het is eenvoudiger toe te passen en het resultaat is minder foutgevoelig.

De naam Flexible Box Layout – we noemen het vanaf nu kortweg flexbox – zegt precies wat het doet: blokken worden flexibel in plaats en omvang. Ze kunnen groeien en krimpen, naast of onder elkaar komen te staan en zelfs in een andere volgorde dan in het HTML-bronbestand worden weergegeven. Flexbox kan vandaag worden gebruikt. Alle moderne browsers werken er probleemloos mee.

Naast flexbox is Grid Layout (kortweg grid of gridlay-out) in ontwikkeling. Het beste voorbeeld van grid is het opmaakraster van een krant. Kolommen en witruimte bepalen de basis en inhoud

wordt geplaatst over een of meer kolommen. Ook dit systeem is flexibel en kan zich voegen naar de beschikbare schermruimte.



Browserondersteuning

Op **caniuse.com** kunt u bekijken welke browsers flexbox en gridlay-out (of elke andere CSS-eigenschap) ondersteunen. U zult zien dat Internet Explorer 11 buiten de boot valt. Of dat een belemmering is moet u zelf bepalen, waarbij u kunt overwegen dat Microsoft Internet Explorer ook niet meer ondersteunt. De opvolger, Microsoft Edge, biedt wel goede ondersteuning voor flexbox, maar loopt achter bij gridlay-out.

Flexbox en gridlay-out kunnen uitstekend worden gecombineerd. De basislay-out kan zijn opgebouwd met grid, waarbinnen met flexbox onderdelen worden ingedeeld. Zo ver is het alleen nog niet. Het tweede deel van dit boek behandelt gridlay-out uitgebreid genoeg om ermee aan de slag te gaan, maar het is nog niet geschikt voor productie. De specificatie is wel redelijk stabiel en grote veranderingen worden niet meer verwacht. Browserondersteuning begint op het moment van schrijven (voorjaar 2017) op gang te komen met als eerste Firefox, op de voet gevolgd door Chrome (ook mobiel), Safari en iOS Safari. Helaas ondersteunt Edge (net als Internet Explorer 11) nog een verouderde versie van gridlay-out, met prefix -ms- en verouderde eigenschappen. Het goede nieuws: op de dag dat dit boek naar de drukker ging maakte Microsoft bekend dat het de implementatie gaat aanpassen aan de specificatie. Daarmee komt ondersteuning van alle belangrijke browsers in zicht. Grid-lay-out wordt besproken vanaf hoofdstuk 6.

Wat flexbox is

Flexbox is een aanvulling op de huidige CSS-opmaakmogelijkheden. We kennen al:

- bloklay-out voor de indeling van pagina's;
- inline lay-out voor de indeling van tekst;
- tabellay-out voor de het indelen van gegevens in rijen en kolommen;
- gepositioneerde lay-out voor indelingen buiten de normale flow.



Afbeelding 1.1 Het opmaakraster van een krant. Een dergelijke indeling is met Grid Layout eenvoudig te realiseren in webpagina's.

Flexbox maakt ingewikkelder lay-outs mogelijk. Het doet denken aan bloklay-out, maar met flexbox kunnen blokken op meer manieren en eenvoudiger worden geordend. In tegenstelling tot de bloklay-out zijn floats niet mogelijk, maar die zijn ook helemaal niet nodig om dezelfde en zelfs betere resultaten te bereiken.

Kenmerken

Enkele kenmerken van flexbox zijn:

- items kunnen in elke richting worden geplaatst: van links naar rechts, van boven naar beneden en andersom;
- de volgorde van items kan worden omgedraaid of zelfs individueel worden ingesteld;



Afbeelding 1.2 *Interessanter is dat de webversie van die krant met Grid Layout en flexbox net zo eenvoudig is te bouwen, volledig flexibel en zonder floats.*

- items kunnen op een regel naast elkaar staan of omlopen naar de volgende regel;
- items kunnen groeien en krimpen naar gelang de ruimte;
- items kunnen in verschillende verhoudingen groeien en krimpen;
- items kunnen worden uitgelijnd ten opzichte van hun container of elkaar.



Floats niet bedoeld voor lay-out

De eigenschap `float` is aan CSS toegevoegd om afbeeldingen *in* tekstblokken te kunnen zetten in plaats van ertussen, of om een los citaat *in* tekst te kunnen plaatsen. Daarbij kan de float buiten de container vallen en dat is prima voor een afbeelding of citaat. In een paginalay-out is dat echter meestal niet gewenst en dat geeft gehannes met clearfix-klassen om het op zichzelf correcte gedrag van floats aan te passen.



Het mogelijke einde van clearfix

Er is een nieuwe waarde voor `display` in ontwikkeling: `flow-root`. Deze heeft hetzelfde effect als de clearfix-hack, maar is een stuk eenvoudiger toe te passen. Net als de clearfix wordt `display: flow-root` ingesteld op de container van de gefloate elementen. Controleer voor gebruik de browserondersteuning.

CSS Flexible Box Layout Module Level 1 is een van de vele specificaties die worden ontwikkeld en onderhouden onder het toezicht van het World Wide Web Consortium, kortweg W3C ([w3.org](http://www.w3.org)). Het is een van de vele modules van wat in de volksmond CSS3 wordt genoemd. In feite is CSS3 niet meer dan een marke-

tingterm voor alle CSS na versie CSS2.1, waarbij onderdelen in afzonderlijke modules worden doorontwikkeld.



Er komt geen CSS4

Als u zich realiseert dat CSS3 eigenlijk niet bestaat, ziet u ook dat er geen CSS4 zal komen. De tijd van één grote specificatie waarin alle onderwerpen en eigenschappen zijn verzameld is voorbij. Door de ontwikkeling van CSS-deelonderwerpen in afzonderlijke modules zal het ene onderdeel op niveau 3 zitten, terwijl een ander onderdeel niveau 1, 2 of 4 kan hebben (en later worden de nummers misschien nog hoger). Het is daarom handiger om het gewoon over CSS te hebben en de term CSS3 te vergeten.

Naast flexbox zijn dat bijvoorbeeld de al genoemde module Grid Layout, maar ook modules voor selectors, kleur, media queries, achtergronden en randen, transformaties enzovoort. Op w3.org/Style/CSS/current-work staat een compleet overzicht.

CSS Values and Units Level 3	CR	PR	
CSS Cascading and Inheritance Level 3	CR	PR	
Testing	Current	Upcoming	Notes
CSS Image Values and Replaced Content Level 3	CR	CR	
CSS Speech	CR	CR	
CSS Flexible Box Layout	CR	PR	
CSS Text Decoration Module Level 3	CR	CR	
CSS Fonts Level 3	CR	CR	
CSS Writing Modes Level 3	CR	CR	
CSS Shapes Level 1	CR	CR	
CSS Masking Level 1	CR	CR	
CSS Counter Styles Level 3	CR	PR	
CSS Fragmentation Level 3	CR	PR	
Compositing and Blending Level 1	CR	CR	
CSS Syntax Level 3	CR	CR	
CSS Grid Layout Level 1	CR	PR	
CSS Basic User Interface Level 3	CR	PR	
Geometry Interfaces Module Level 1	CR	CR	

Afbeelding 1.3 Een deel van de specificaties voor CSS op w3.org/Style/CSS/current-work.

De flexboxspecificatie is in mei 2016 van concept (*working draft*) naar ‘candidate recommendation’ gegaan. Dat is W3C-taal voor: dit stuk is bijna klaar. In de praktijk is een dergelijke specificatie dan vaak al geschikt voor productie, maar uiteraard hangt dat volledig af van de browserondersteuning. Flexbox was zelfs al bruikbaar in de conceptfase en dat komt doordat browserfabrikanten vanwege de wens van gebruikers al geruime tijd bezig zijn met flexbox. U kunt de praktijksituatie overigens altijd bekijken op **caniuse.com**. Hier wordt bijgehouden welke webtechnieken door browsers worden ondersteund. Het is een onmisbare bron voor elke ontwikkelaar. Hoe dan ook: begin 2017 staat flexbox op de nominatie voor proposed recommendation, waarna de volgende halte recommendation zal zijn.

Uiteindelijk is het enige wat echt telt of de techniek in de praktijk bruikbaar is en het antwoord daarop is een volmondig ja.



Flexbox 3.0

De huidige specificatie is de derde revisie van flexbox. Al in 2009 publiceerde het W3C de eerste conceptversie en in 2011 kwam er een herziening. In dit boek gaat het om de laatste en actuele versie: CSS Flexible Box Layout Module Level 1. De informatie is gebaseerd op de W3C Candidate Recommendation van 26 mei 2016. Op het moment van schrijven is het niet waarschijnlijk dat de inhoud nog veranderd.

Wat flexbox kan

Eenvoudig gesteld kan met flexbox elke lay-out worden gemaakt. Het traditionele model header, navigatie, inhoud, sidebar, footer komt natuurlijk als voorbeeld voorbij. Uiteraard zijn eenpagina-

ontwerpen/apps geen probleem. Voor productcatalogi, huizensites en automarkten is flexbox een prima instrument. Zelfs zonder media queries kunnen dergelijke ontwerpen zich probleemloos voegen naar de viewport, hoewel media queries ook hierbij de gebruikerservaring kunnen verbeteren en de ontwerp mogelijkheden vergroten.

Het ontwerp hoeft uiteraard niet per se een hele pagina te zijn. Sterker nog: flexbox is juist bedacht voor situaties en onderdelen waarbij het formaat, de uitlijning of zelfs de volgorde met traditionele technieken lastig – of niet! – zijn in te stellen. Denk aan (complexe) menu's, een interface van tabbladen, formulieren, invoerelementen met pictogrammen, het omdraaien van de weergegeven volgorde van hoofdinhoud en zijbalk enzovoort.

Wat flexbox oplost

Twee dingen houden webdevelopers al heel lang bezig: verticaal centreren en kolommen van gelijke hoogte. Dat laatste staat bekend als *the holy grail*. Als nadeel van flexbox zou genoemd kunnen worden dat het de creativiteit doodslaat die de afgelopen jaren bij het zoeken naar een oplossing voor beide problemen aan de dag is gelegd. Want dat doet flexbox. Verticaal centreren en kolommen van gelijke hoogte komen bij flexbox namelijk gewoon uit de doos.

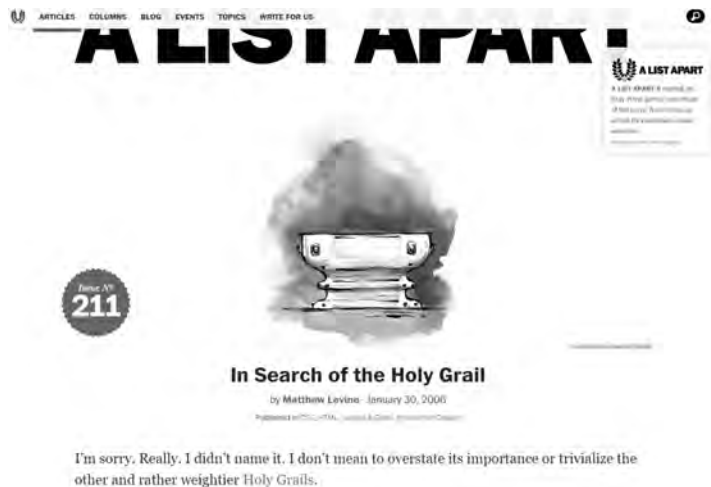
Naast verticaal centreren en kolommen van gelijke hoogte is ook een menubalk met aan de ene kant de menuopties en aan de andere kant een vrijstaand zoekvak in een handomdraai gemaakt. Zo ook een footer die altijd onder aan de pagina staat, ongeacht de inhoud van de footer of de voorgaande blokken.

In essentie zorgt flexbox ervoor dat in een omgeving waar niet alle dimensies bekend zijn of vaststaan, de proportionele verdeling van elementen gewaarborgd is.



De zoektocht naar The Holy Grail

Op 30 januari 2006 publiceerde A List Apart een uitputtend en veelgeciteerd artikel: *In search of the holy grail* (alistapart.com/article/holygrail). Het is nog steeds de moeite van het lezen waard en het geeft een mooi beeld van de creativiteit die in het zoeken naar oplossingen is gestoken.



Afbeelding 1.4 Het artikel dat verscheen bij A List Apart, nu ruim 11 jaar geleden.

Hello flexbox

Een voorbeeld kan het best illustreren waarom flexbox zo fantastisch is. De volgende code is de 'Hello World' van HTML: een pagina met een header, een inhoudsgebied met de belangrijkste inhoud en een zijkolom, en een footer ([flex0101.html](#).)

```
<body>
  <div class="page">
    <header>
      <h1>Hello flexbox</h1>
    </header>
    <div class="content-container clearfix">
      <main>
        <h2>Hoofdkolom</h2>
        <p> ... </p>
      </main>
      <div class="sidebar">
        <div class="sidebar-content">
          <h3>Zijkolom</h3>
          <p> ... </p>
        </div>
      </div>
    </div>
    <footer>
      <h2>Footer</h2>
      <p> ... </p>
    </footer>
  </div>
</body>
```

Om het resultaat uit afbeelding XX te bereiken, kan deze pagina met CSS als volgt worden ingedeeld:

```
main {
  float: left;
  width: 66%;
}
.sidebar {
  float: left;
  width: 34%;
}
```

Met deze CSS-code is gelijk duidelijk wat ook al weer die ene belangrijke tekortkoming van floats was bij het maken paginalayouts: zonder clearfix heeft de container content geen functie en volgen de kolommen hun eigen weg.

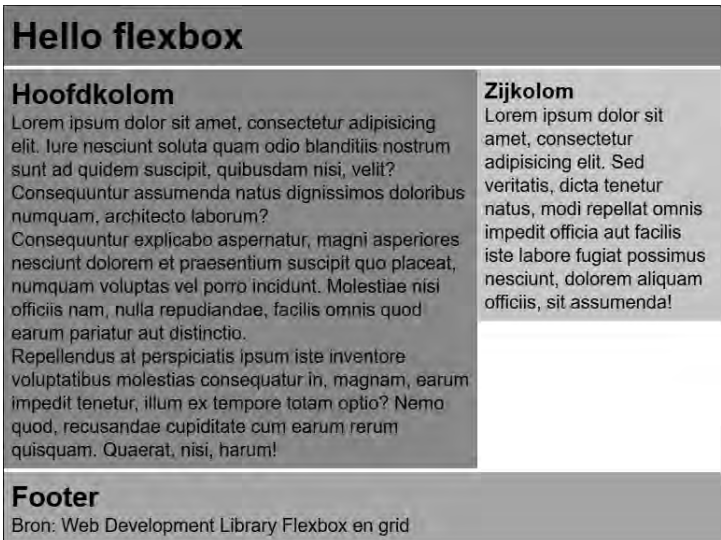


Afbeelding 1.5 Bij deze gefloate kolommen ontbreekt duidelijk een *clearfix*. De content-div is 0 pixels hoog.

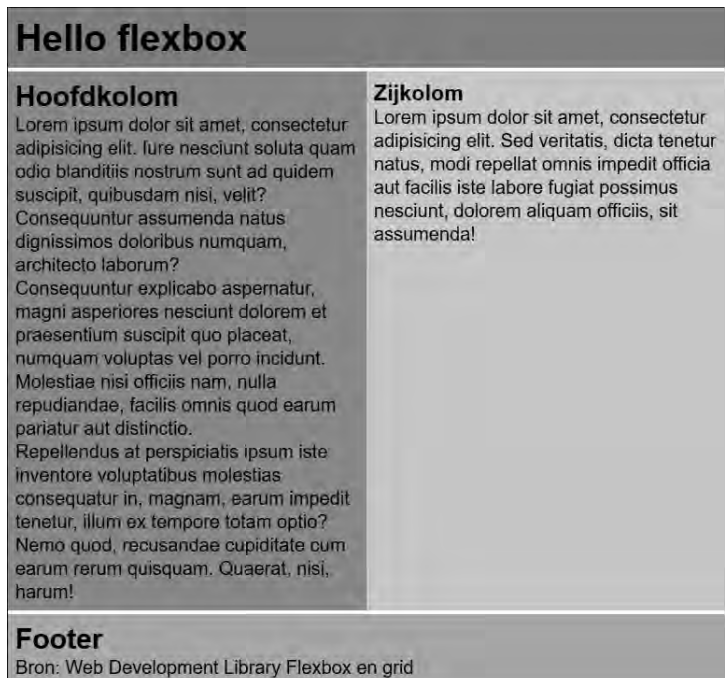
Met de toevoeging van de klasse `clearfix` aan content wordt wel het beoogde resultaat bereikt. Wel is nu die andere tekortkoming van floats te zien: de kolommen zijn niet even hoog.

```
.clearfix::after {  
  content: "";  
  display: table;  
  clear: both;  
}
```

De code zal geen verrassingen bevatten. De header en de footer openen en sluiten de pagina. Daartussenin staat de inhoud, verdeeld in twee kolommen. Deze hebben elk een deel van de beschikbare ruimte toegewezen gekregen en floats naar links. Met behulp van de `clearfix` omvat een container de floats en daardoor komt de footer eronder te staan. Met wat marge op de container is dat goed zichtbaar. Het werkt, maar dit kan handiger. Vergeet de CSS-code hiervoor (en verwijder de klasse `clearfix` uit de HTML). We vervangen alle CSS door de volgende regel:



Afbeelding 1.6 Traditioneel ingedeelde HTML-pagina op traditionele wijze vormgegeven met floats.



Afbeelding 1.7 Dezelfde HTML-code (min de klasse clearfix) en slechts één regel CSS.

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

Tada! Twee kolommen en precies even lang. Met aanmerkelijk minder CSS-code. Schone en duidelijke code ook, het lijkt wel toveren. Wat gebeurt hier?

In de HTML-bron staan de hoofdkolom en de zijkolom in de `div` met de klasse `content-container`. Deze container heeft de CSS-eigenschap `display: flex`. Hiermee wordt flexbox geactiveerd. De container wordt een *flexcontainer* en alle kindelementen zijn nu *flexitems*. Omdat flexbox net als elke CSS-eigenschap basisinstellingen heeft, gebruiken de flexcontainer en de flexitems direct die zogeheten *initiële* flexeigenschappen. Een gevolg daarvan is dat containers worden uitgerekend tot gelijke hoogte.

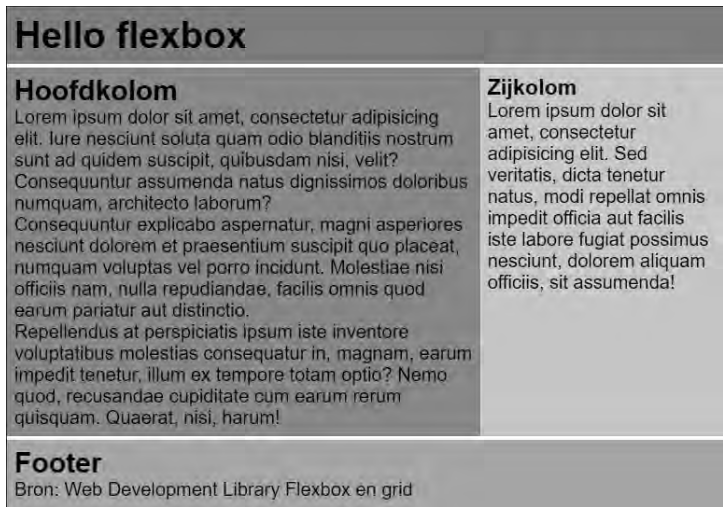
Wat nog ontbreekt is de breedteverdeling van 2 : 1. Dat vergt maar een kleine ingreep. Voeg toe:

```
main {  
  flex: 2;  
}  
.sidebar {  
  flex: 1;  
}
```

Voilà, kolommen in de juiste verhouding. Op de flexitems kan natuurlijk een (vaste, minimale of maximale) breedte worden ingesteld, maar dit voorbeeld is veelzeggend over de manier waarop flexbox werkt.

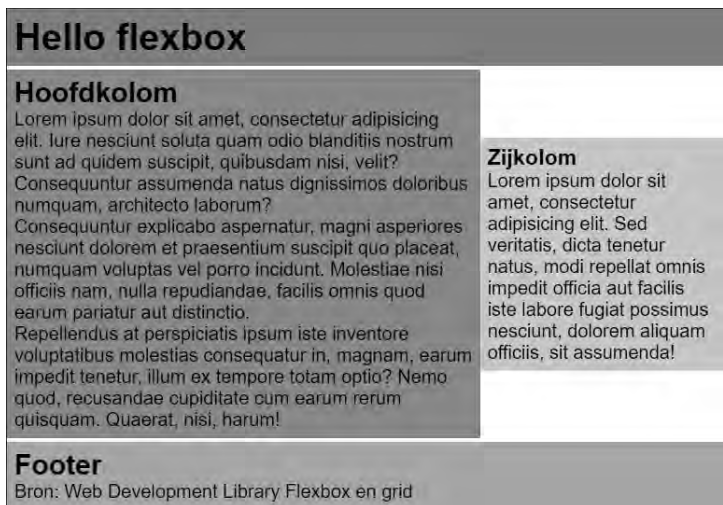
Verticaal centreren

Nu we toch bezig zijn, hier is het bewijs dat verticaal centreren geen enkele uitdaging meer is. Met de volgende CSS-declaratie wordt de zijkolom verticaal in het midden van zijn flexcontainer geplaatst:



Afbeelding 1.8 *Met de eigenschap flex wordt de gewenste verhouding ingesteld.*

```
.sidebar {
  align-self: center;
}
```



Afbeelding 1.9 *Met één regel extra is de zijkolom verticaal gecentreerd.*

Zoals u ziet zijn nu de kolommen niet meer even hoog. Ook dat is snel op te lossen. Maak in de HTML een extra flexcontainer voor de inhoud van de zijkolom en centreer vervolgens de inhoud van de zijkolom:

```
<div class="sidebar">
  <div class="sidebar-content">
    <h3>Zijkolom</h3>
    <p> ... </p>
  </div>
</div>
.sidebar {
  display: flex;
}
.sidebar-content {
  align-self: center;
}
```

Zo is ook gelijk vastgesteld dat flexcontainers kunnen worden genest.

Hello flexbox

Hoofdkolom
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Lure nesciunt soluta quam odio blanditiis nostrum sunt ad quidem suscipit, quibusdam nisi, velit? Consequuntur assumenda natus dignissimos doloribus numquam, architecto laborum? Consequuntur explicabo aspernatur, magni asperiores nesciunt dolorem et praesentium suscipit quo placeat, numquam voluptas vel porro incidunt. Molestiae nisi officiis nam, nulla repudiandae, facilis omnis quod earum pariatur aut distinctio. Repellendus at perspiciatis ipsum iste inventore voluptatibus molestias consequatur in, magnam, earum impedit tenetur, illum ex tempore totam optio? Nemo quod, recusandae cupiditate cum earum rerum quisquam. Quaerat, nisi, harum!

Zijkolom
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed veritatis, dicta tenetur natus, modi repellat omnis impedit officia aut facilis iste labore fugiat possimus nesciunt, dolorem aliquam officiis, sit assumenda!

Footer
Bron: Web Development Library Flexbox en grid

Afbeelding 1.10 *Met een geneste flexcontainer is de inhoud van .sidebar gecentreerd en is de kolom even hoog als de hoofdkolom.*

Laat het nog even allemaal tot u doordringen: met één regel CSS-code zijn twee even lange kolommen gemaakt waarvan de breedte proportioneel is. Met nog een regel CSS-code extra is een item verticaal gecentreerd. Met een extra HTML-element een nog een regel CSS erbij zijn de kolommen én even hoog én is de inhoud verticaal gecentreerd. In de komende hoofdstukken leert u alles wat u weten moet om zelf deze magie te kunnen bedrijven.

Benodigde voorkennis

Dit boek maakt deel uit van de serie *Web Development Library* (www.webdevelopmentlibrary.nl). In elk deel wordt een op zichzelf staande techniek besproken die te maken heeft met web-development. Andere, gerelateerde technieken worden bekend verondersteld. Zo betaalt u alleen voor datgene wat u echt nodig hebt.

In dit boek gaan we in op de CSS-modules Flexible Box Layout en Grid Layout. Daar kunt u op zichzelf weinig mee als u geen brede kennis hebt van HTML en CSS. Alles begint immers met goed gemarkeerde HTML-documenten. Toepassing van flexbox en grid vereist minimaal dat u weet hoe CSS-stylesheets worden gemaakt en hoe HTML-elementen worden opgemaakt. We gaan er dan ook van uit dat u voldoende ervaring hebt met HTML en CSS.

Tips voor meer leesvoer

Kan uw HTML- of CSS-voorkennis wel een opfrisbeurt gebruiken? Lees dan bijvoorbeeld eerst (van dezelfde auteur):

- *Web Development Library – HTML5*, ISBN 978 90 5940 808 1
- *Web Development Library – CSS3*, ISBN 978 90 5940 809 8
- *Handboek HTML5 en CSS3 derde editie*, ISBN 978 90 5940 916 3

Editor en browser

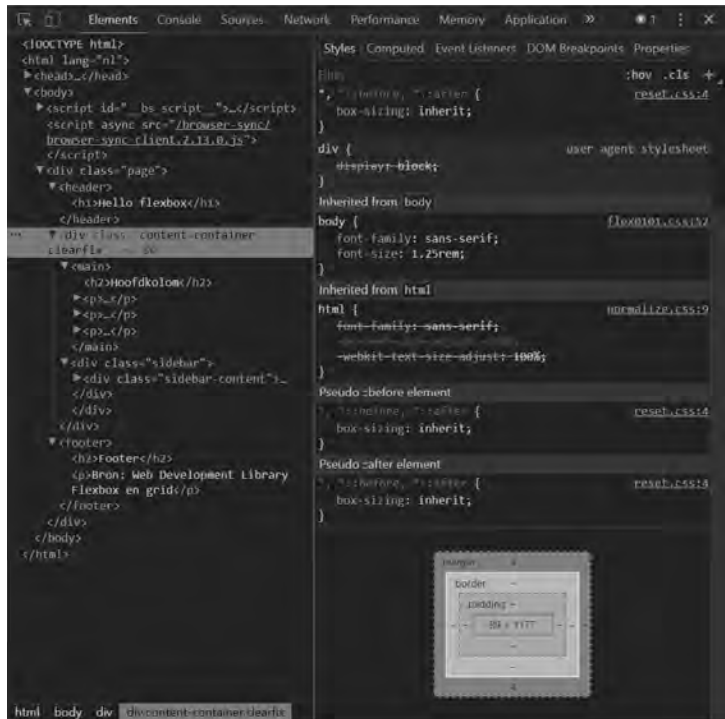
U moet kunnen werken met code. Gebruik de editor die uw voorkeur heeft. Bekende tools voor webdevelopment zijn bijvoorbeeld Brackets, Visual Studio Code, Sublime Text of Atom, of eventueel geïntegreerde ontwikkelomgevingen zoals Adobe Dreamweaver of Visual Studio. Elke teksteditor waarmee u platte tekst als .html- of .css-bestand kunt opslaan is echter geschikt. Het is wel handig als een vorm van codeaanvulling, kleurcodering en syntaxiscontrole beschikbaar is of kan worden toegevoegd met plug-ins.



Afbeelding 1.11 Nieuwe editor van Microsoft: Visual Studio Code.

We adviseren uw werk te testen in Google Chrome of Firefox. Naast de stabiele versie is er Chrome Canary, de versie voor ontwikkelaars waarin steeds de nieuwste mogelijkheden beschikbaar zijn. Firefox heeft daardoor de Developer Edition. Beide beschikken over uitstekende hulpmiddelen voor ontwikkelaars (*developer tools*) en een handige emulatiemodus. Hiermee kunt u binnen de browser de schermafmetingen en andere variabelen instellen voor tal van mobiele apparaten. Zo hoeft u niet over tientallen apparaten te beschikken om te zien hoe uw pagina er daarop uit-

ziet. U kunt het eenvoudig via een keuzemenu instellen. Overigens bieden ook andere fabrikanten dergelijke hulpmiddelen aan. Test uw werk ook in Microsoft Edge en Apple Safari.



Afbeelding 1.12 Hulprogramma's voor ontwikkelaars in Chrome Canary.

Voorbeeldcode downloaden

Alle codevoorbeelden en -fragmenten die in de tekst worden besproken zijn downloaden. Het adres hiervoor is www.webdevelopmentlibrary.nl (klik op **Downloads** en kies de titel **Flexbox en gridlay-out**). Soms gaat het maar om enkele regeltjes code, maar dat kan net genoeg zijn om u op weg te helpen. In de codevoorbeelden staat vaak aanvullend commentaar.

Er is een bestand `index.html` dat koppelingen bevat naar alle andere bestanden. De oefenbestanden zijn oplopend per hoofd-

stuk genummerd en hebben namen als `flex0201.html`, `flex0202.html` en zo verder. Bij de codeblokken in de tekst van dit boek wordt meestal genoemd wat het nummer van het bijbehorende oefenbestand is.

De CSS-code staat altijd in een CSS-bestand met dezelfde naam als het HTML-bestand, dus `flex0201.css`, `flex 0202.css` enzovoort. Deze bestanden bevatten `@import`-regels voor normalisatie en reset van browserinstellingen. Let erop dat u ook deze bestanden in uw CSS-map opslaat. Elk CSS-bestand begint met de flexbox-declaraties (of die voor gridlay-out) en daaronder staan wat algemene opmaakstijlen.

Samenvatting

In dit hoofdstuk is globaal uitgelegd wat u van dit boek kunt verwachten. Met een voorbeeld van flexbox is het topje van de ijsberg getoond, waarbij hopelijk duidelijk is geworden hoe krachtig flexbox is. In de volgende hoofdstukken worden alle hier genoemde begrippen en nog veel meer uitgebreid besproken.

Praktijkoefeningen

Net als in de andere uitgaven in de serie *Web Development Library* vindt u aan het einde van de hoofdstukken een aantal praktijkoefeningen. Zo kunt u nog eens op een andere manier herhalen wat u in het hoofdstuk hebt gelezen. De oefeningen zijn bedoeld om u verder op weg te helpen met de CSS-voorbeelden. U kunt ze zelf uitbreiden of aanpassen. De uitwerkingen van de oefeningen (indien van toepassing) zijn meestal *niet* opgenomen in de codevoorbeelden die u bij dit boek kunt downloaden, het is de bedoeling dat u zelf de uitwerkingen maakt.