hogeschool vives

Development 1 : Labo 8 CSS - 4

studiegebied **HWB**

bachelor in het **toegepaste Informatica**

campus **Kortrijk**

academiejaar **2024-2025**

Inhoudsopgave

1 Inleiding

Deze les behandelt voornamelijk het *visual formatting model* waarmee elementen op de pagina kunnen gezet worden.

Dit document weidt ook verder uit over enkele andere onderwerpen uit de CSS3 cursus.

1.1 Verslag

In dit deel van de cursus staan verschillende vragen die je moet beantwoorden en opdrachten om iets te maken of uit te proberen. Het is belangrijk dat je alle opdrachten zorgvuldig uitvoert!

Documenteer je werk in een verslag document (pdf of docx) "**verslag CSS deel 4**" waarin je:

- Voor elke uitprobeeropdracht een entry maakt met screenshots ter staving van wat je deed.
- Je antwoorden op de gestelde vragen neerschrijft.

Oplossingen van 'grotere' opdrachten (met veel code) bewaar je in een Webstorm project "**verslag CSS deel 4**". Per opdracht maak je in dit project een aparte folder waarin je de bestanden (en subfolders) plaatst.

2 Labo

2.1 CSS positionering

Lees Hoofdstuk 7 t.e.m. Sectie 7.3.1 (zwevende elementen)

We zagen eerder dat we inhoud en presentatie wensen te scheiden, waarbij HTML voor de content zal zorgen en CSS voor de visualisatie. Op welke manier bepaalt de HTML-code toch gedeeltelijk de layout?

2.1.1 Opdracht 1

Unzip '`CSS deel 4 voorbeelden positionering.zip`' en open '`voorbeeld1.html`' in je browser. Open de HTML en CSS bestanden in je editor.

Ga na wat het effect is van het veranderen van de `display` property voor `.area` naar achtereenvolgens:

- `block`
- `inline`
- `inline-block`

Voor welke waarden worden de opgegeven width en height genegeerd?

Voor welke waarden worden de elementen als woorden op een pagina gezet?

2.1.2 Opdracht 2

Stel `display:inline-block` voor de elementen met `class area`, en bekijk opnieuw 'voorbeeld1.html'. Vanwaar komen de lege ruimtes tussen blokken die naast elkaar staan? Resize je browservenster horizontaal en kijk hoe de blokken netjes over de regels verdeeld worden.

2.1.3 Opdracht 3

Open '`voorbeeld2.html`' in de browser.

Waarom staat de tekst in de blokken horizontaal gecentreerd?

Ga na dat een property `vertical-align:center` geen effect heeft. Waarom eigenlijk niet?

Ga na dat `vertical-align:middle` er *niet* voor zorgt dat de tekst verticaal gecentreerd wordt. Wat doet `vertical-align` dan wel?

Zoek op google of er een simpele manier is om tekst verticaal te centreren in een element.

- Wat is de 'simpele' manier om een één enkele regel tekst verticaal te centreren? (tip:`line-height`)

- Welke truc wordt vaak gebruikt om meerdere regels verticaal te centreren?
  (tip: `table cell`)

In 'styles.css' staan twee verschillende CSS regels die de height instellen van een element met `class="area tall"`. Waarom krijgt de regel voor class '`tall`' voorrang?

Verander de CSS-file om je vermoeden te bevestigen. Zet nadien alles weer terug op z'n oorspronkelijke plaats.

2.1.4 Opdracht 4

Open '`voorbeeld3.html`' in de browser. Bekijk nogmaals dat alle elementen als in een tekstverwerker gezet worden als je het browservenster vergroot en verkleint.

Geef elementen 09, 12 en 14 de '`floaty`' class.

- Ga na dat de floaty elementen naar links verplaatst werden 'in hun regel'.
- Ga na dat er geen grote ruimte achterblijft tussen elementen 08 en 10 waar element 09 stond.
- Ga na dat 12 en 14 niet altijd netjes naast elkaar staan, het hangt er maar vanaf of ze in dezelfde regel staan.
- Ga na dat er geen kleine ruimte is tussen de groene floaty elementen en het rode element rechts ernaast, maar wel tussen de rode elementen onderling.

Geef element 17 de 'break' class.

- Ga na dat dit helemaal geen invloed heeft qua positionering, er wordt niks 'gecleared'.

Het clearen van floating elementen is alleen zinvol bij display:block elementen, niet bij inline(-block) elementen!

2.1.5 Opdracht 5

Open '`voorbeeld4.html`' in de browser. Resize het browser venster (horizontaal). Ga na dat er naast element 09, 2 'regels' van blokken geplaatst worden als het venster breed genoeg is.

Geef element 13 de 'break' class. Resize je browser opnieuw. Ga na dat element 13 altijd op een nieuwe 'regel' begint en wel lager dan element 09.

Er zijn dus verschillende toepassingen voor floating elements:

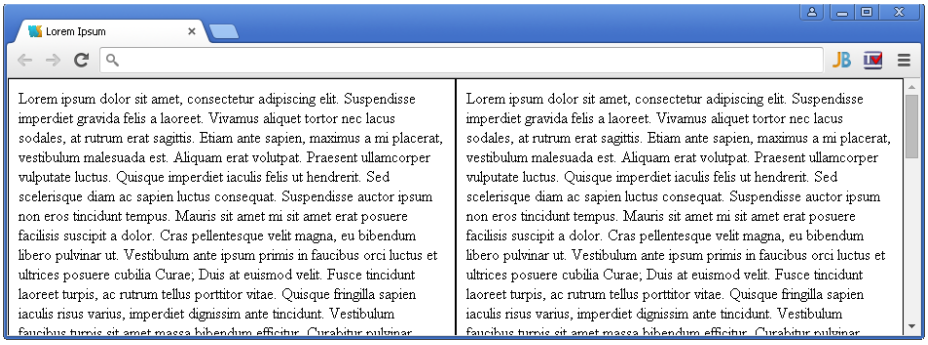
1) Een inline stuk laten zweven zodat de andere inline delen (bv. tekst) errond gezet worden. Hier heeft `clear` geen nut.

1. Een block laten zweven, met ernaast wellicht 1 of meerdere andere zwevende blokken. Hier kan `clear` nuttig zijn voor blokken die erna komen.

2.2 Opdracht Lorem ipsum

Lorem ipsum is een fake Latijnse tekst die vaak gebruikt wordt voor layout doeleinden. Op de volgende pagina kun je zoveel van deze tekst genereren als je nodig hebt alsook de achterliggende geschiedenis nalezen: http://www.lipsum.com/.

Maak een HTML5 pagina met 2 lorem ipsum sections van voldoende lengte. Maak een layout waarbij elke section een eigen kolom vormt (dus 2 kolommen naast elkaar). Elke kolom dient de helft van de breedte in beslag te nemen, met een border van 1px en een padding van 10px.



Je eerste idee is wellicht om elke section een width van 50% te geven. Probeer dit uit.

Je zult merken dat de kolommen niet naast elkaar, maar onder elkaar staan. Waarom is dit zo?

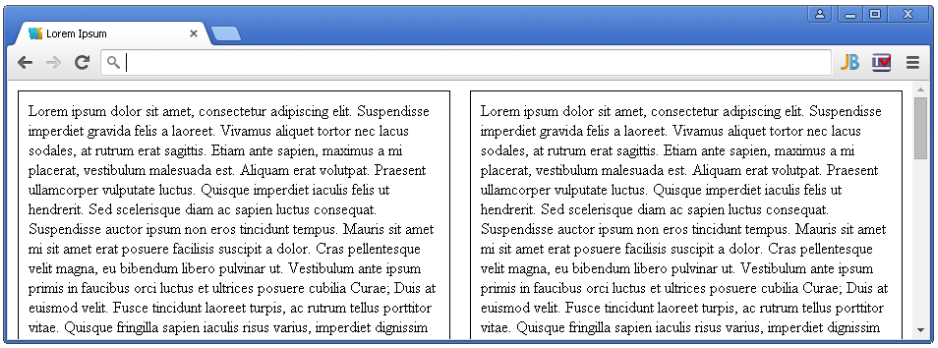
Probeer onderstaande twee pogingen uit en ga na dat ze niet het gewenste resultaat opleveren:

- De width op bv. 47% in stellen, maar dit werkt niet goed.
    - Extra lege ruimte rechts op grote schermen.
    - Kolommen onder elkaar op kleine schermen.
- De width op de helft van de beschikbare breedte in pixels in stellen, bv. 550px, maar dan is het niet exact de helft als het browservenster een ander formaat heeft.

In CSS3 is er een betere mogelijkheid: zoek op het internet naar de '`box-sizing`' property en probeer hiermee een goede oplossing te bekomen waarbij de width van elke kolom netjes op 50% is ingesteld.

2.3 Opdracht Lorem ipsum extra space

We willen nu een lege ruimte van 10px toevoegen rond de kolommen:



Merk op dat er tussen de kolommen een dubbele lege ruimte voorkomt. We zouden dit kunnen wegwerken zodat de ruimte tussen de kolommen eveneens slechts 10px is maar voor de eenvoud laten we het zo.

Hint: als je een element met expliciete width en "`box-sizing: border-box`" ook nog een `margin` geeft, zal dat element breder uitvallen dan de opgegeven width! De box-sizing is immers t.o.v. de border en de margin ligt daar nog omheen. Bedenk echter dat de lege ruimte rondom een element niet noodzakelijk van het element zelf afkomstig moet zijn, maar net zo goed bij een ancestor of descendant element kan horen!

2.4 Positioneren met CSS

Lees Sectie 7.3.2 over relatief positioneren en Sectie 7.3.3 over absoluut positioneren.

De CSS '`position`' property heeft als mogelijke waarden: `static`, `fixed`, `absolute` en `relative`. Voor de betekenis van de 'static' en 'fixed' waarden verwijzen we naar w3schools[1]. Lees beslist ook het stukje over `z-index` onderaan die pagina.

De '`relative`' waarde lijkt misschien eerder een curiositeit, maar er is 1 speciaal geval dat heel vaak gebruikt wordt! Lees dit[2] artikel op CSS-tricks.

Daar zie je dat `position: relative` voor een element X eigenlijk vooral handig is om de kinderen van X te kunnen positioneren. Immers, als de `position` van X niet '`static`' is, kunnen de kinderen van X *absoluut* gepositioneerd kunnen worden tov de box van X!

- De waarden 'fixed' of 'absolute' vallen af omdat deze ook de positie van X zelf beïnvloeden.
    - Namelijk buitenom de natuurlijk *page flow*.
- Dan blijft enkel 'relative' over en eigenlijk zou je bij hierbij ook `left` en `top` properties verwachten
    - Maar hier is het net de 'truc' om die op 0 te laten staan!

Kortom, de combinatie:

[1] http://www.w3schools.com/CSS/CSS_positioning.asp

[2] http://CSS-tricks.com/absolute-positioning-inside-relative-positioning/

```
position: relative
left: 0
top: 0
```

lijkt zinloos maar is net heel belangrijk! Merk op dat helemaal geen expliciete '`left`' en '`top`' opgeven, hetzelfde is als deze op 0 zetten, dus doorgaans zie je enkel '`position:relative`' staan. Het is de beste manier om de kinderen van een element op een exacte pixel locatie te plaatsen.

Normaliter doen we net moeite om geen expliciete posities te gebruiken, opdat de paginaopmaak zich zou kunnen aanpassen aan de beschikbare schermgrootte. Een expliciete positionering heeft echter haar toepassingen, denk bv. aan eenvoudige spelletjes waarbij *sprite-afbeeldingen* op een specifieke locatie op het speelveld moeten geplaatst worden. Een demonstratie hiervan vind je in '`demo absolute positionering.zip`', waarbij de waarde van de '`left`' property van een afbeelding meermaals per seconde wordt verhoogd om een animatie effect te verkrijgen.

Lees ook: https://CSS-tricks.com/people-share-their-CSS-ah-ha-moments/.

2.5 Pagina indeling

Lees Secties 7.4 en 7.5 waarin uitgelegd wordt hoe je de grote blokken op een pagina kunt positioneren.

2.6 Opdracht Cocktail bar

Maak de 'Cocktail bar' opdracht.