9 Hoofdstuk

Boxmodel, weergavemodel en positionering

Er is nog heel wat voor nodig om van alle losse HTML-code een toonbare pagina te maken, meer dan u misschien op het eerste gezicht denkt. Als de HTML-code is geladen, maakt de browser van elk element een blok. Er zijn verschillende typen blokken die zich ook verschillend gedragen. De eigenschappen en ordening van de blokken bepalen de lay-out van de pagina. Het stijlblad van de browser bevat opmaakinstructies voor alle blokken, zodat zonder uw tussenkomst een pagina wordt getoond met – min of meer – de standaardwaarden van CSS. En dan begint het pas, want úw instellingen bepalen hoe de webpagina uiteindelijk wordt vormgegeven.

U leert in dit hoofdstuk:

De kenmerken van het boxmodel: marges, padding en randen.

Hoe de grootte van een blok wordt bepaald door box-sizing.

De kenmerken van het weergavemodel en de eigenschap display.

Positionering van blokken: relatief, absoluut en float.

Het (niet) verbergen van inhoud: overflow, clip, visibility.

De opbouw van pagina's

De weergave van een niet-opgemaakte HTML-pagina komt vaak nog het meest overeen met een standaardpagina in een tekstverwerker. Alles staat netjes onder elkaar, met wat extra ruimte tussen koppen en alinea's, ingesprongen tekst bij opsommingen en vette en cursieve woorden of zinsdelen. Hoe vanzelfsprekend dat ook lijkt, er zitten mechanismen achter die al die HTML-elementen de eigenschappen geven waardoor de browser met zijn ingebouwde stijlblad deze weergave maakt. Wordt de pagina door de ontwerper met CSS opgemaakt, dan kan die volledig naar wens worden ingedeeld. Daarbij worden nog steeds dezelfde mechanismen gebruikt, alleen bepaalt de ontwerper nu de instellingen ervan.

In dit hoofdstuk komen de onderdelen aan bod waarmee de lay-out van een HTML-pagina wordt bepaald:

- het *boxmodel* maakt van elk element een blok;
- het *weergavemodel* bepaalt de ordening van die blokken;
- het *positioneringsschema* maakt uitzonderingen op die weergave mogelijk.

Browserstijlen zijn er niet voor niets

Hoewel we als ontwikkelaars de opmaakstijlen van de browser op alle mogelijke manieren aanpassen of zelfs uitschakelen, is die standaardopmaak er natuurlijk niet voor niets. Als het om wat voor reden dan ook niet lukt om de CSS-stylesheets te laden, krijgt de gebruiker toch een leesbare, bruikbare HTML-pagina voorgeschoteld. Zo kan hij toch dat artikel lezen of uw product bestellen. Dat wil zeggen: als u een bruikbare HTML-pagina hebt gecodeerd, met betekenisvolle elementen op de juiste wijze toegepast. Dat blijft de kern van uw ontwerpen, CSS of geen CSS.

Het boxmodel

Een webpagina is opgebouwd uit blokken: blokken met koptekst, alineatekst, opsommingen, tabellen enzovoort. De opbouw van zo'n blok wordt beschreven in het boxmodel. Daarover gaat deze paragraaf.

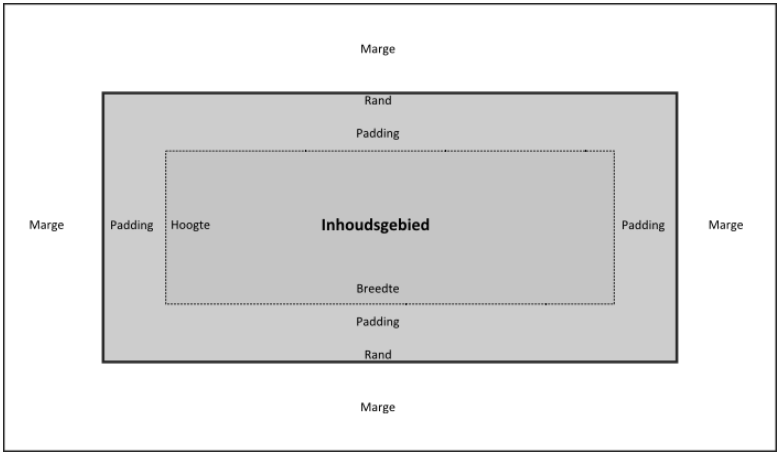
Hoe al die blokken op een pagina zich tot elkaar verhouden, met andere woorden: hoe met al die blokken de lay-out wordt opgebouwd, wordt beschreven in het weergavemodel (*visual formatting model*). Dat komt aan bod in het tweede deel van dit hoofdstuk. Hierbij speelt de eigenschap `display` een hoofdrol, want die bepaalt globaal gezien de lay-out van de pagina: komen blokken naast

of onder elkaar, is het een flexboxlay-out of een gridlay-out, enzovoort. Dan is er nog het *positioneringsschema*, het laatste deel van dit hoofdstuk. Dit gaat over afwijken van de normale plaatsing met floats of absolute positionering.

Het zijn stuk voor stuk belangrijke onderdelen van CSS. Zonder inzicht in de principes en kennis van de bijbehorende eigenschappen en waarden is het maken van een lay-out net CSS niet goed mogelijk.

Terug naar de blokkendoos. In de afbeelding ziet u schematisch hoe een blok is opgebouwd:

- er is inhoud (tekst of beeld);
- er is ruimte tussen de inhoud en de rand: padding;
- er is een rand: border;
- er is ruimte aan de buitenkant: margin.



Afbeelding 9.1 *Het boxmodel.*

Deze opbouw geldt voor elk blok. Dat betekent niet dat alle onderdelen ruimte hoeven in te nemen. Inhoud kan strak tegen de rand staan en dan is de padding nul. De rand kan strak tegen het volgende blok staan en dan is de marge nul. Een marge kan zelfs negatief zijn en dan verschuift de inhoud (zie de paragraaf *Negatieve marge*). En er hoeft geen rand te zijn: een rand kan nul zijn, maar ook heel dik om speciale effecten te maken.

De eigenschap box-sizing

De totale ruimte die een blok inneemt in de lay-out bestaat uit:

- `width`
- `height`
- breedte/hoogte van de padding
- dikte van de rand
- breedte/hoogte van de marge

Het is wel even opletten bij het maken van blokken, want de waarde van `width` en `height` hangt af van de eigenschap `box-sizing`. Daarmee kan de ruimte voor de inhoud aanzienlijk verschillen. De mogelijke waarden zijn:

- `content-box` (standaardwaarde)
- `border-box`

Specificatie

De eigenschap `box-sizing` maakt deel uit van de specificatie CSS Basic User Interface Module Level 3 (CSS3 UI): **www.w3.org/TR/css-ui-3/**.

Traditioneel zijn `width` en `height` de maten voor het inhoudsblok. Dit komt overeen met `box-sizing: content-box`. De totale breedte van een blok wordt dan:

> `width` + padding links/rechts + randdikte links/rechts + marge links/rechts. (Bereken de hoogte op vergelijkbare wijze.)

Bij de nieuwe waarde `border-box` bestaat de ruimte voor de inhoud uit `width` en `height` min de padding en randdikte. Voor de totale blokgrootte wordt de marge bij `width` opgeteld.

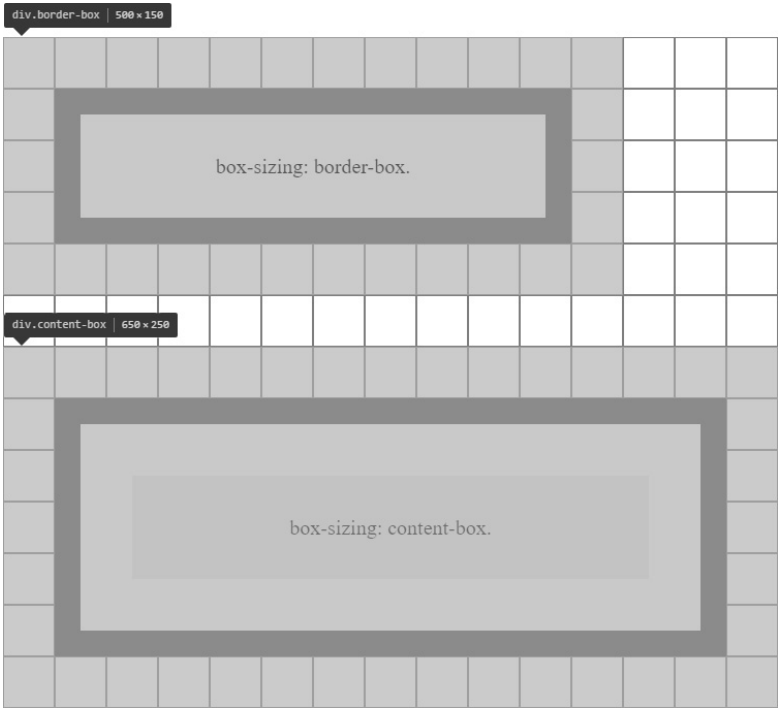
De waarde van `box-sizing` maakt dus nogal wat verschil voor de blokgrootte en de ruimte voor de inhoud. Neem bijvoorbeeld deze code:

```
.border-box {
    box-sizing: border-box; /* of content-box */
    width: 500px;
    height: 100px;
    padding: 50px;
    margin: 50px;
    border: 25px solid black;
}
```

Bij `box-sizing: content-box` is de beschikbare breedte voor de inhoud 500px, namelijk de ingestelde waarde van `width`. De totale breedte van het blok is `width:`500 + (2 x 50 padding) + (2 x 25 rand) + (2 x 50 marge) is 750px.

Bij `box-sizing: border-box` is de breedte voor de inhoud `width:`500 - (2 x 50 padding) - (2 x 25 rand) = 350px. De totale breedte van het hele blok is `width` + (2 x marge) = 600px.

Het content-box- blok is 150 pixels breder dan het border-box-blok en daarin heeft de inhoud ook aanmerkelijk meer ruimte. In de afbeelding is het verschil goed te zien. Bedenk dat de grootte van het verschil sterk afhangt van de dikte van padding en border.



Afbeelding 9.2 *Het verschil in de ruimte voor de inhoud en de uiteindelijke afmetingen tussen border-box en content-box is aanzienlijk. De breedte in de tooltip is zonder marge.*

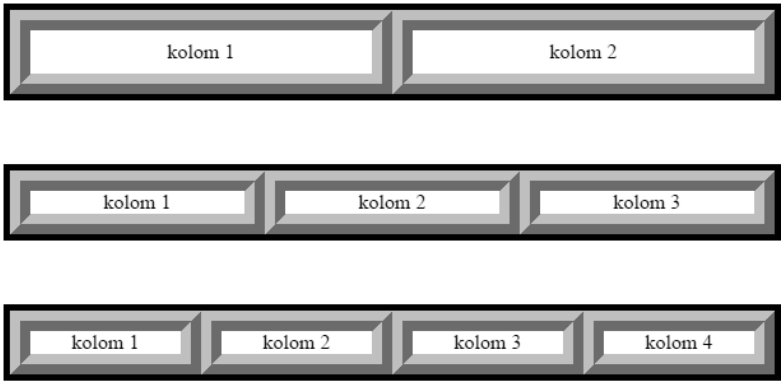
Nu hebben beide methoden zo hun nut. Als binnen een container de ruimte moet worden verdeeld tussen twee of meer blokken die padding en randdikte hebben, is `border-box` heel handig. Met een `width` als een percentage (en geen marge) krijgt elk blok precies het juiste deel van de beschikbare ruimte

zonder rekenwerk aan padding en randdikte. Als het aantal blokken verandert, hoeft alleen het percentage te worden aangepast.

In het volgende voorbeeld is er een container van 600px breed met `box-sizing: content-box`. Dat betekent dat de volle 600px beschikbaar is voor de inhoud. In de container staan twee blokken die elk een breedte van 50% hebben. Omdat dit de breedte inclusief de padding en de rand is (`box-sizing: border-box`), hoeft u met die twee waarden geen rekening te houden en nemen ze elk ook precies 300px van de breedte.

```
.container {
    box-sizing: content-box;
    width: 600px;
    border: 5px solid black;
}

.col2 {
    box-sizing: border-box;
    width: 50%;
    margin: 0;
    padding: 0.5em;
    border: 1em silver ridge;
    float: left;
}
<div class="container">
    <div class="col2">kolom 1</div>
    <div class="col2">kolom 2</div>
</div>
```



Afbeelding 9.3 *Een content-box die ook echt 600 pixels breed is huisvest twee blokken met de instelling border-box.*

Begrensde breedte en hoogte

Het is mogelijk om het formaat van inhoudsblokken te begrenzen. Zowel een minimale als een maximale grootte is daarbij mogelijk.

- `min-width` en `min-height` bepalen de minimale breedte en hoogte (lengte of een percentage);
- `max-width` en `max-height` bepalen de minimale en maximale hoogte (lengte, percentage of `none`).

De afmetingen van bijna alle elementen kunnen hiermee worden begrensd. Deze eigenschappen gelden alleen niet voor niet-vervangen inline elementen (het geldt bijvoorbeeld niet voor `<em>`, maar wel voor `<img>`), tabelrijen en tabelrijgroepen.

Waarden voor hoogte en breedte

De standaardwaarde van `width` en `height` is `auto`. Dat betekent dat de browser het zelf wel uitzoekt. De andere waarden voor hoogte- en breedtematen zijn de eenheden die in hoofdstuk 8 zijn besproken. Die werken allemaal probleemloos.

Met name in de categorie sleutelwoorden is CSS flink in beweging, zie CSS Basic box model op **dev.w3.org/csswg/css-box/**. Dit is werk in uitvoering (lees de waarschuwing), maar sommige sleutelwoorden zijn al wel in gebruik. Dat komt mede doordat die ook bij flexbox en grid worden toegepast. Wees er wel voorzichtig mee en volg **canisue.com**. Het betreft:

- `max-content` de afmeting van het grootste item in de container (bijvoorbeeld een alinea);
- `min-content` de kleinst mogelijke afmeting zonder overloop te veroorzaken, bijvoorbeeld het langste woord;
- `available` de ruimte in de container minus horizontale marge, rand en padding;
- `fit-content` de grootste van `min-content` en `max-content`;
- `border-box` en `content-box` zie de uitleg hiervoor in de paragraaf *De eigenschap box-sizing*.

Dit is een fragment van hoofdstuk 9 uit Handboek HTML5 & CSS3 vierde editie, geschreven door Peter Doolaard en uitgegeven door Van Duuren Media.

ISBN: 978-90-5940-971-2
Eerste druk: juli 2017