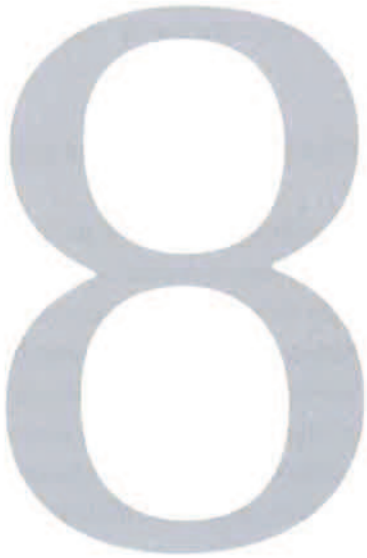
Wat er nog meer kan

8.1 Inleiding

Websites spelen op het moment van schrijven een heel grote rol in de maatschappij. Er is dan ook veel meer over websites te vertellen dan we in dit boek kunnen behandelen. Voor iedereen die zijn kennis nog verder wil uitbreiden hebben we in dit hoofdstuk een flink aantal onderwerpen op een rij gezet die in eerdere hoofdstukken niet aan de orde zijn gekomen. Het hoofdstuk is in drie hoofdparagrafen verdeeld. In de eerste paragraaf behandelen we de meer geavanceerde mogelijkheden van HTML. In de tweede paragraaf verkennen we wat er nog meer met CSS kan en in de derde paragraaf laten we enkele vaardigheden zien waarmee je een nog betere webdesigner wordt.



8.2 Wat er nog meer kan met HTML 5

Bij het schrijven van dit boek was het niet mogelijk om álle mogelijkheden van HTML en CSS te behandelen. Sommige van die mogelijkheden zijn ingewikkeld, anderen worden niet zoveel gebruikt. Hieronder staan enkele voorbeelden van nuttige, maar nog niet genoemde mogelijkheden. Voordat je daarover gaat lezen is het goed om te weten dat er een website bestaat waarop je live online een website kunt programmeren. Dit is de website www.codepen.io. Als je de website opent, kun je rechtsboven op '+New Pen' klikken en er verschijnt een venster waar je HTML, CSS en JavaScript kunt programmeren. Als je een account aanmaakt kun je ook projecten opslaan. Een handig hulpmiddel voor webdesigners!

Veel van de informatie in dit hoofdstuk is gebaseerd op wat er te vinden is op www.w3schools.com. Die website bevat zeer veel informatie over onder andere HTML en CSS. Je kunt er ook uitgebreide voorbeelden vinden over hoe bepaalde code in de praktijk werkt. Het is de moeite waard om deze website als naslagmateriaal te gebruiken om de werking van bepaalde code beter te begrijpen.

Sommige code die in dit hoofdstuk staat, wordt niet in elke webbrowser goed weer gegeven. Op het moment van schrijven ondersteunen de nieuwste versies van Chrome en Firefox alle genoemde voorbeelden. Meer informatie over ondersteuning kun je vinden in paragraaf 8.3.2.

8.2.1 Een menubalk met submenu

Op websites zie je vaak dat een deel van het menu uitklapt als je boven één van de menuknoppen zweeft met de muis. Als je op Google zoekt naar 'css drop down menu' kun je veel codevoorbeelden van dergelijke menu's vinden. Hieronder staat ook een voorbeeld. De meeste stijlregels zullen je bekend voorkomen, maar de onderste regel misschien niet. Daar staat 'nav ul li:hover ul'. Hiermee wordt in het kort aangegeven dat 'nav ul li ul' van 'display: none' (onzichtbaar) moet veranderen in 'display: block' als er sprake is van 'nav ul li:hover' (dus als de muis boven het li-element zweeft).

Dit is een belangrijke eigenschap: je kunt dus een element aanpassen als de muis boven een ander element zweeft!

```
<nav>
  <ul>
    <li class="focus"><a href="#">Home</a>
    </li>
    <li>
    <a href="#">Pagina 2</a>
      <ul>
        <li><a href="#">Submenu 1</a></li>
        <li><a href="#">Lang Submenu 2 dat
        verder gaat op deze regel</a></li>
        <li><a href="#">Submenu 3</a></li>
      </ul>
    </li>
    <li>
      <a href="#">Pagina 3</a>
      <ul>
        <li><a href="#">Submenu 1</a></li>
        <li><a href="#">Submenu 2</a></li>
        <li><a href="#">Submenu 3</a></li>
      </ul>
    </li>
    <li><a href="#">Contact</a></li>
  </ul>
</nav>
```

```
* {
  margin: 0;
  padding: 0;
}
nav ul {
  list-style: none;
}
nav ul a {
  display: block;
  color: #333333;
  text-decoration: none;
  padding: 15px 10px;
}
nav ul li {
  display: inline-block;
  margin-left: -4px;
  position: relative;
}
nav ul li.focus, nav ul
li:hover {
  background: #dddddd;
}
nav ul ul {
  display: none;
  position: absolute;
  top: 100%;
  left: 4px;
  background: #ffffff;
}
nav ul ul a {
  width: 200px;
  padding: 10px 15px;
}
nav ul li:hover ul {
  display: block;
}
```

8.2.2 Geaccepteerde opmaak in HTML

De meeste opmaak van tekst stel je in met CSS-code. Zoals je weet is het soms toch makkelijker om dat in HTML te doen, zoals bij het <b>-, <u>- en <i>-element. Er zijn nog meer van dat soort opmaakelementen. Hieronder staat een tabel met enkele daarvan:

<sup>	Superscript, bijvoorbeeld voor machten van een getal of verwijzingen: normaal^{super}
<sub>	Subscript, onder andere voor sommige wiskundige notaties: normaal$_{sub}$
<code>	Als je een andere opmaak wilt voor programmeercodes of wiskundige formules kun je dit met het code-element eenvoudig realiseren
<q>	Het q-element is een vervanging van dubbele aanhalingstekens, zodat een citaat door de webbrowser niet wordt geïnterpreteerd als een attribuut van een element

8.2.3 Bestaande scripts gebruiken (API's)

Het script-element wordt gebruikt voor het implementeren van JavaScript. Er zijn heel veel bestaande scripts te vinden op het internet, zodat je zelf maar beperkt kennis van JavaScript hoeft te hebben om een script aan je website toe te voegen. Denk aan bijvoorbeeld een slideshow. Deze scripts noemen we API's (Application Programming Interface). Ook internetbedrijven en -instellingen zoals w3schools, Google en Amazon leveren kant-en-klare scripts aan die webprogrammeurs mogen gebruiken, zoals scripts die een GPS-locatie van je telefoon doorgeven, Twitter- en Facebookfeeds, etc.

Je kunt een script op twee manieren aan je website toevoegen: door het te 'linken' of te 'embedden'. Een gelinkt script is extern beschikbaar als .js bestand. Je kunt een .js bestand linken op de volgende manier:

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

Zoals je kunt zien staan gelinkte scripts in het head-element en heeft het script-element een eind-tag, hoewel er niets tussen de begin- en eind-tag hoeft te staan. Omdat het script-element op twee manieren kan worden gebruikt is de eind-tag wel verplicht, ook al staat er niets tussen.

Je kunt ook (volledige) scripts in een HTML-bestand plakken. Als webprogrammeur is dat natuurlijk handig, omdat je in één bestand zowel de HTML- als de JavaScript-code kunt bewerken. Bij grotere websites kan de code echter onoverzichtelijk worden en heeft het de voorkeur om de scripts te linken. Een script embedden kan bijvoorbeeld op de onderstaande manier. De doctype- en html-tag zijn voor het gemak even weggelaten.

```
<head>
  <script type="text/javascript">
    document.getElementById("script").innerHTML="Tekst in p#script";
  </script>
</head>
<body>
  <p id="script"></p>
</body>
```

Het script in het voorbeeld plaatst tekst in het p-element. Je mag kleine scripts op deze manier aan het head-element van het -code HTML-document toevoegen. Het is wel belangrijk dat het HTML-document overzichtelijk blijft. Plaats daarom alle JavaScript in één script-element en gebruik één of meerdere gelinkte scripts als er zo veel JavaScript-code is dat het HTML-document er onoverzichtelijk van wordt. Wil je meer weten over wat JavaScript is en hoe je het gebruikt, dan kun je dat lezen in paragraaf 8.3.6.

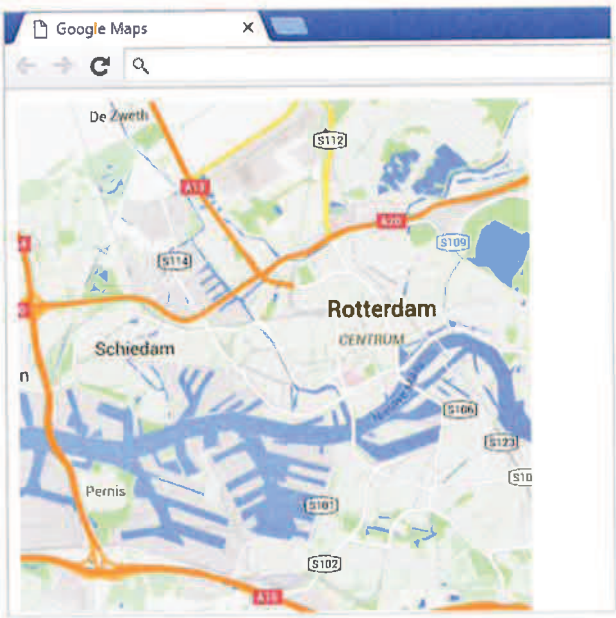
8.2.4 Een voorbeeld van een API: Google Maps

Het toevoegen van een kaart van Google Maps werkt met een API. Lees daarom, als je dit nog niet hebt gedaan, eerst de vorige paragraaf over het gebruiken van bestaande scripts. Bij de Google Maps API is het handig om twee scripts te gebruiken. De kaart zelf wordt gelinkt, maar alle persoonlijke wijzigingen aan de kaart worden geëmbed.

Een kaart van Rotterdam kun je op de volgende manier op een webpagina krijgen:

```
<!DOCTYPE html>
<html>
  <head>
    <link href="style.css" rel="stylesheet"
    type="text/css">
    <script src="https://maps.googleapis.com/
    maps/api/js"></script>
    <script type="text/javascript">
      var mapCanvas = document.
      getElementById("map");
      var mapOptions =
      center: new google.maps.
      LatLng(51.9,4.47), zoom: 11

      var map = new google.maps.
      Map(mapCanvas, mapOptions);
    </script>
  </head>

  <body>
    <div id="map"></div>
  </body>
</html>
```

```
#map {
  width: 400px;
  height: 400px;
}
```

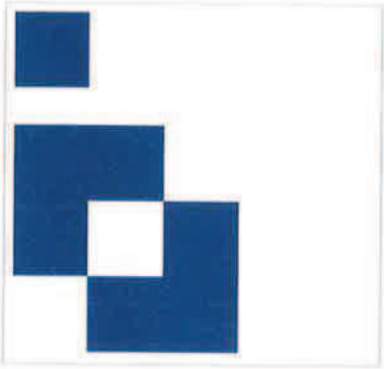


De bovenstaande code is slechts een eenvoudig voorbeeld van het gebruik van de Google Maps API. Op internet zijn voorbeelden te vinden met veel meer opties. Gecombineerd met API's voor Geolocation is het zelfs mogelijk om een routebeschrijving naar een adres te maken.

8.2.5 Tekenen met HTML 5

Het is mogelijk om met HTML 5 tekeningen te maken. Je kunt op die manier bijvoorbeeld logo's maken die niet in kwaliteit achteruit gaat als je inzoomt. Dit kan met het SVG-element of met JavaScript in combinatie met het canvas-element. Het voert wat ver om hier uitgebreid op in te gaan, maar je krijgt wel een voorbeeld zodat je zelf meer informatie kunt zoeken als je deze mogelijkheden interessant vindt. Het onderstaande voorbeeld bevat een tekening met een SVG-element van het grafische deel van het logo van Uitgeverij Instruct:

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
    <svg width="150" height="225">
      <rect x="0" y="0" width="50" height="50"
      style="fill:rgb(0,72,122)" />
      <polygon points="0,75 0,175 50,175 50,125 100,125 100,75"
      style="fill:rgb(0,72,122)" />
      <polygon points="100,125 100,175 50,175 50,225 150,225 150,125"
      style="fill:rgb(0,72,122)" />
    </svg>
  </body>
</html>
```



Met de SVG-tag wordt het SVG-element geopend. Als attributen worden de hoogte en breedte van het tekenveld meegegeven (in pixels). In het SVG-element staan in dit geval drie elementen die voor de tekening zorgen. Het eerste element is een rechthoek (rect komt van rectangle). Deze rechthoek start in het punt 0,0 (zie de eerste twee attributen). Het SVG-tekenveld gebruikt de linkerbovenhoek als 0,0; dit is dus anders dan bij wiskunde! Daarna volgen de hoogte en breedte van de rechthoek en als laatste de kleur van de vulling. Daarna volgen twee keer een 'polygon'. Dat is Engels voor veelhoek. Iedere veelhoek heeft een aantal hoekpunten. Per hoekpunt wordt de x- en y-coördinaat opgegeven en de verschillende hoekpunten worden gescheiden door een spatie.

In dit voorbeeld wordt alleen 'fill' gebruikt als attribuut van 'style'. Als je op internet zoekt naar mogelijkheden zie je dat er nog meer attributen zijn. Je kunt lijnkleur en –dikte bepalen en zelfs de transparantie instellen.

Er zijn nog meer vormen te maken met SVG. Denk aan cirkels en ovalen, maar ook tekst en ingewikkelde kwadratische curves. Met name voor die laatste is het slim om gebruik te maken van een SVG-editor. Dat is een webapplicatie waarmee je een SVG-figuur kunt maken zonder dat je zelf de code hoeft te bedenken. Google heeft een gratis SVG-editor: http://svg-edit.googlecode.com/svn-history/r1771/trunk/editor/svg-editor.html.

Wil je je verder hierin verdiepen? Zoek dan eens een manier om de twee polygonen uit het voorbeeld om te zetten naar één polygoon.

8.3 Wat er nog meer kan met CSS 3

Nog meer dan bij HTML zijn er in CSS veel functies niet aan de orde gekomen. Zeker met de introductie van CSS versie 3 zijn de mogelijkheden sterk uitgebreid. In deze paragraaf komt een aantal functies aan bod. Sommige van deze functies moest je al gebruiken in de opdrachten van de vorige hoofdstukken, maar zijn nog niet (helemaal) uitgelegd. Andere functies zijn nog niet eerder besproken.

8.3.1 Selectors

In hoofdstuk 4 t/m 6 zijn de tag-, class- en id-selector aan de orde geweest. Er zijn echter nog meer typen selectors.
Het is belangrijk om te onthouden dat er ingewikkelde selectors zijn die werken met geneste elementen. Bij nesting zijn de begrippen 'parent' en 'children' van belang. Het element waar andere elementen in genest zijn wordt 'parent' genoemd en de elementen die erin genest zijn heten 'children'. Let op! Als er een a-element in een p-element zit en dit p-element zit weer in een article-element, dan is het a-element geen child van het article-element, maar alleen van het p-element.

Hieronder staat een lijst met verschillende typen selectors.

p	Selector voor p-elementen
p.class	Selector voor p-elementen met de class 'class'
p#id	Selector voor p-elementen met het id 'id'
*	Selector voor alle elementen
p, a	Selector van alle p-elementen en alle a-elementen (een komma mag meerdere keren worden gebruikt als nog meer elementen moeten worden geselecteerd)
p a	Selector voor alle a-elementen die in een p-element zitten
p > a	Selector voor alle a-elementen die child zijn van een p-element
img + p	Selector voor alle p-elementen die na een img-element staan
img ~ p	Selector voor alle p-elementen die vóór een img-element staan
[href]	Selector voor elementen die een href-attribuut bevatten
[href=iets]	Selector voor elementen die een href-attribuut hebben waarbij de waarde 'iets' is
[href*="pag"]	Selector voor elementen die een href-attribuut hebben waarbij de waarde de tekencombinatie 'pag' bevat
[href~=pag]	Selector voor elementen die een href-attribuut hebben waarbij de waarde onder andere 'pag' is (bijvoorbeeld alt="volgende pag")
[href^="pag"]	Selector voor elementen die een href-attribuut hebben waarbij de waarde begint met de tekencombinatie 'pag'
[href$="pag"]	Selector voor elementen die een href-attribuut hebben waarbij de waarde eindigt met de tekencombinatie 'pag'

8.3.2 Pseudoclasses en pseudo-elements

In hoofdstuk 4 is de pseudoselector :hover genoemd. Er zijn naast :hover meer pseudoselectors. Er bestaan zelfs twee soorten van, namelijk pseudoclasses en pseudo-elements. Met pseudoclasses kun je elementen opmaken in een bepaalde situatie en met pseudo-elements kun je een deel van een element opmaken. Je kunt ze uit elkaar houden door een enkele en een dubbele dubbelepunt (:hover of ::first-line). Het gaat te ver om alle mogelijkheden te vermelden, maar in de tabel hieronder kun je een aantal handige pseudoselectors vinden.

Pseud
classe

Pseud
eleme

Interessante pseudoclasses

:empty	Opmaak van de elementen die geen children bevatten (uitleg over wat een child is staat in 8.2.1).
:first-of-type	Opmaak van het eerste element van dat type. Dit geldt per parent. Als er twee divisions zijn met elk drie p-elementen zal bij p:first-of-type van beide divisions de opmaak op het eerste p-element worden toegepast.
:hover	Opmaak van een element als de muis erboven zweeft.
:last-of-type	Opmaak van het laatste child-element van dat type.
:not(*p*)	Opmaak van alle elementen behalve het p-element. De p kan worden vervangen door een ander element.
:nth-of-type(*2*)	Opmaak van het tweede child-element. Het getal kan worden vervangen door een willekeurig ander getal. Er bestaat ook een :nth-last-of-type().
:only-of-type	Opmaak als het element maar één keer in een parent voorkomt.
:target	De :target-selector verdient speciale aandacht. Hiermee kan een element worden opgemaakt waarnaar wordt verwezen door middel van een a-element in combinatie met een id-selector. In paragraaf 8.2.4 staat een uitgewerkt voorbeeld.

Interessante pseudo-elementen

::after	Voeg content toe na een element.
::before	Voeg content toe vóór een element.
::first-letter	Verander de opmaak van de eerste letter van de tekst in het element.
::fist-line	Verander de opmaak van de eerste regel van de tekst in het element.
::selection	Bepaal hoe tekst eruit ziet als de gebruiker die in het element selecteert.

Bij de pseudo-elementen ::after en ::before wordt veel gebruik gemaakt van de eigenschap 'content'. Met content kan extra inhoud worden toegevoegd aan een HTML-document. Je kunt bijvoorbeeld een h1-element altijd laten eindigen met een punt met de volgende CSS-code:

```
h1::after {
  content: ".";
}
```

Een handige toepassing van de :not-pseudoclass is het opmaken van navigatie-elementen zonder een bepaalde class. Op die manier kunnen alle links met :hover dezelfde opmaak krijgen, uitgezonderd de link van de pagina die op dat moment actief is. Je ziet in dit voorbeeld overigens ook dat het mogelijk is om pseudoselectors met elkaar te combineren.

```
nav a:hover:not(.focus) {
  background-color: red;
}
```

8.3.3 Verander inhoud met een klik

Met de pseudoclass :target kun je verassende effecten maken op een website. Je kunt de eigenschappen van een element veranderen zodra er op een bepaalde

link wordt geklikt. Gebruik het volgende of een vergelijkbaar document als basis.

```
<!DOCTYPE html>
<html>
  <head>
    <link href="style.css" rel="stylesheet" type="text/css">
  </head>

  <body>
    <nav>
      <a href="#article1">Link 1</a>
      <a href="#article2">Link 2</a>
      <a href="#article3">Link 3</a>
    </nav>
    <main>
      <article id="article1">
        <p>Tekst in article 1</p>
      </article>
      <article id="article2">
        <p>Tekst in article 2</p>
      </article>
      <article id="article3">
        <p>Tekst in article 3</p>
      </article>
    </main>
  </body>
</html>
```

Zet nu de volgende stijlregel in style.css en klik op de verschillende links:

```
article:target {
  background-color: red;
}
```

Het wordt nog leuker als we de bovenstaande CSS-code vervangen door:

Index

ay

```
article {
  display: none;
}
article:target {
  display: block;
}
```

Hiermee kun je dus, afhankelijk van de geklikte link, steeds een andere division laten zien!

8.3.4 Uitgebreide box-opties

In hoofdstuk 5 is het box-model geïntroduceerd. Dit model is bedacht omdat het heel handig is om alle elementen te beschouwen als een blok (box) met hoogte, breedte, marge enz. Elk element heeft zijn unieke box-eigenschappen. Hier volgen een aantal mogelijkheden om deze unieke eigenschappen aan te passen. De enige mogelijkheid die we buiten beschouwing laten zijn de 'flex'-opties.

Display

Met display kun je een element de box-eigenschappen geven van een ander element. In paragraaf 8.1.1 kregen de li-elementen de eigenschap 'display: block;'. Block is de standaard displaywaarde voor p-elementen. Je zorgt er zo dus voor dat een li-element zich gaat gedragen als een p-element. De belangrijkste waarden voor display zijn:

inline	Gebruik de eigenschappen van een element zoals <b> of <u>
block	Gebruik de eigenschappen van een element zoals <p>
inline-block	Gebruik buiten de border inline- en binnen de border blockeigenschappen
list-item	Gebruik de eigenschappen van een li-element
table	Gebruik de eigenschappen van een table-element
table-cell	Gebruik de eigenschappen van een td-element
none	Verberg dit element

Position

Een andere box-optie die in de eerdere hoofdstukken ter sprake is geweest, is de eigenschap 'position'. Hiermee kun je een element op een andere plek neerzetten dan gewoonlijk gebeurt. In opdracht 1 van hoofdstuk 6 is deze eigenschap gebruikt om een bijschrift op een andere plek neer te zetten. Er zijn nog meer mogelijkheden.

static	Dit is de normale positioning.
relative	Een element relative maken zorgt ervoor dat je het kunt bijsturen met de eigenschappen top, bottom, right en left.
fixed	Een fixed element trekt zich nergens wat van aan en gaat met top/bottom/right/left op de plek staan die je wilt. Ook scrollen heeft geen invloed op een fixed element.
absolute	Een dergelijk element berekent zijn positie aan de hand van het element waarin het genest is. Je gebruikt absolute eigenlijk alleen als de parent (zie paragraaf 8.2.1) relative is.

Als de positioning van een element niet static is, kun je ook gebruikmaken van de CSS-eigenschap 'z-index'. Met z-index kun je elementen voor of achter elkaar plaatsen. Zie het onderstaande voorbeeld. Een z-index mag ook een negatief getal zijn. De z-index van ieder element is standaard 0.

```
position: relative;
z-index: 1;

    position: relative;
    top: -10px;
    left: 10px;
```

```
position: relative;
z-index: 2;

    position: relative;
    top: -10px;
    left: 10px;
```

Overflow

Een andere consequentie van het benaderen van elementen als blokken is dat er ook meer inhoud in kan worden geplaatst dan het element kan bevatten. Er ontstaat dan overflow. Maak maar eens een article-element van 30 pixels hoog en 100 pixels breed en plaats er zoveel tekst in dat het niet meer past. Wat er in zo'n geval moet gebeuren kun je opgeven met de CSS-eigenschap 'overflow', bijvoorbeeld:

```
overflow: auto;
```

Naast de eigenschap 'overflow' is er ook overflow-x en overflow-y, zodat je x- en y-richting apart kunt instellen. Er zijn vier mogelijke waarden:

visible	Standaard: overflow is te zien buiten de box.
hidden	Overflow wordt verborgen, alleen binnen de box is inhoud te zien.
scroll	Zowel in x- als in y-richting is een scrollbalk te zien.
auto	Er verschijnt een scrollbalk als er inhoud is die in x- of y-richting groter is dan de box.

Uiterste afmetingen

Er zijn vier eigenschappen waarmee elementen een maximale of minimale afmeting kunnen krijgen. Soms is het handig om de afmetingen van een element te laten afhangen van de inhoud, tot een bepaalde grens is bereikt. Je kunt daarvoor de regels min-width, min-height, max-width en max-height gebruiken. Deze stijlregels worden ook vaak gebruikt in media queries (zie de paragraaf over responsive design in hoofdstuk 6). Je kunt dan de opmaak van je website aanpassen als het scherm kleiner of groter is dan een bepaalde afmeting. Bijvoorbeeld: @media screen and (max-width: 600px){ }.

Borders niet meerekenen

Een vervelende klus bij het maken van een website is het berekenen van het aantal pixels dat nodig is om bepaalde elementen precies naast elkaar te krijgen. Het gebruik van borders zorgt daarbij voor een extra complexe situatie. Het is mogelijk om borders standaard uit de berekening te halen. Dit kan met de stijlregel 'box-sizing: border-box;'. Uitgeschreven zou er staan: 'Neem de borders mee als onderdeel van de box'. Als een element een breedte heeft van 200 pixels en borders van 4 pixels, dan is de breedte op het scherm zonder border-box 208 pixels en met border-box nog steeds 200 pixels.

8.3.5 Grid-view

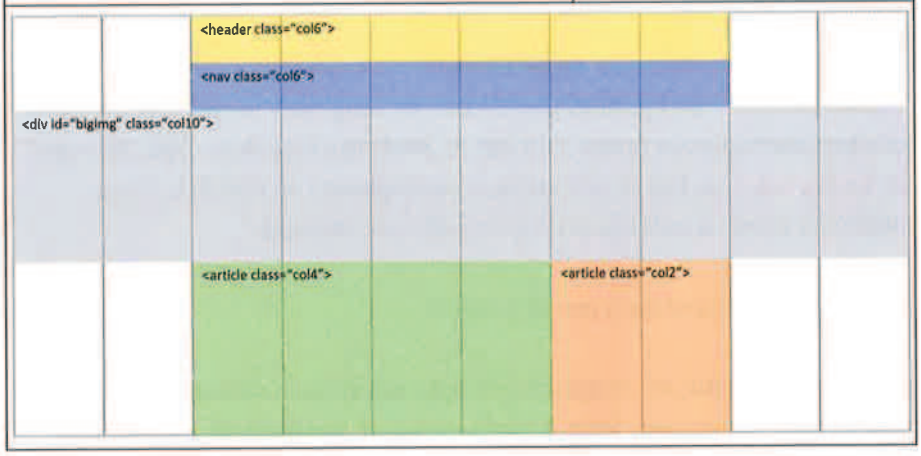
Bij modern webdesign is de zogenaamde grid-view belangrijk geworden.

Veel CMS systemen (zie paragraaf 8.3.8) maken hier gebruik van. Het is ook behulpzaam als je een website responsive maakt. Een gridview is eigenlijk een CSS-truc, waarbij de body wordt opgedeeld in kolommen en ieder element een aantal van deze kolommen mag bestrijken.

Stel je voor dat je de body denkbeeldig horizontaal opdeelt in tien stukken, dan is ieder stuk 10% van de volledige breedte. Je kunt voor elke kolombreedte een class maken (.col1 { width: 10%; } .col2 { width: 20%;}, enz.) en daarna van elk blok de juiste class invullen. Zie onderstaand voorbeeld.

```
<header class="col6"></header>
<nav class="col6"></nav>
<div id="bigimg" class="col10"></div>
<main>
  <article class="col4"></article>
  <article class="col2"></article>
</main>
```

```
.col1 {
  width: 10%;
}
.col2 {
  width: 20%;
}
.col3 {
```



Om extra gemak te hebben van de afmetingen is het handig om gebruik te maken van de stijlregel 'box-sizing: border-box;', zoals in de vorige paragraaf is uitgelegd.

8.3.6 CSS zelf laten rekenen

Als je werkt met een combinatie van pixels en procenten, is het soms lastig te bepalen waar een bepaald element moet komen te staan. In CSS 3 bestaat de mogelijkheid om CSS zelf een maat te laten uitrekenen. Dit kan met de functie calc(). Je kunt onder andere rekenen met waarden in pixels en procenten met de operatoren +, -, * en /, bijvoorbeeld: 'width: calc(100% - 80px);'. De breedte van het element waar deze opmaak van toepassing is, zal altijd 80 pixels smaller zijn dan de maximale breedte.

8.3.7 Achtergronden en kleurverlopen (gradients)

Een stijlmogelijkheid die veel leerlingen interessant vinden, is kleurverloop. Ook dit is mogelijk in CSS. Het is echter geen eigenschap, maar een waarde van de eigenschap background. Daarom wordt kleurverloop in de paragraaf over achtergronden behandeld.

In opdracht 1 van hoofdstuk 6 maken we gebruik van background-image, background-size en background-position. Er zijn nog meer mogelijkheden die je bovendien kunt opgeven in één stijlregel in plaats van meerdere. Deze stijlregel heet 'background'. Zo kan de achtergrond van bigimage uit opdracht 1 van hoofdstuk 6 samen worden gevat met de volgende stijlregel:

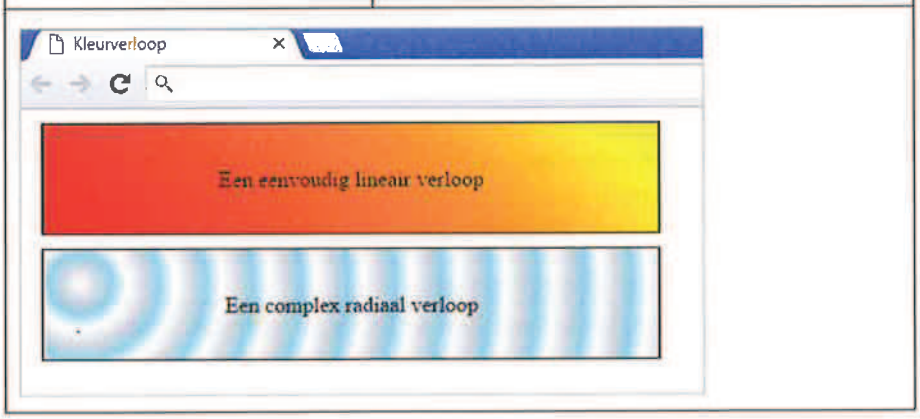
```
background: url("titan2.jpg") center / cover;
```

Als je zowel de positie als de grootte wilt opgeven, zoals in dit voorbeeld, moet je een slash ('/') gebruiken. Vóór de slash geef je de 'position' op, erna de 'size'.

In plaats van een kleur of een plaatje kan ook een kleurverloop worden toegepast. De mogelijkheden zijn bijna eindeloos. Je kunt een lineair of een radiaal verloop instellen, meer dan twee kleuren gebruiken, een lineair verloop onder een hoek laten lopen, een radiaal verloop in een hoek beginnen, transparantie gebruiken en een verloop meerdere keren achter elkaar laten herhalen. Er volgen een voorbeeld van een eenvoudig lineair en een complex radiaal verloop.

```html
<div id="verloop1">
  <p>Een eenvoudig lineair
  verloop</p>
</div>

<div id="verloop2">
  <p>Een complex radiaal
  verloop</p>
</div>
```

```css
div {
  width: 500px;
  height: 100px;
  border: 1px solid black;
  text-align: center;
  line-height: 60px;
  margin: 10px;
}

#verloop1 {
  background: linear-gradient(45deg,
  red, yellow);
}

#verloop2 {
  background: repeating-radial-
  gradient(circle closest-corner at 30px
  30px, white, lightgrey 30%, lightblue
  60%, white 90%);
}
```



Misschien heb je zelf al bedacht dat kleurverloop een gevaar kan vormen voor je ontwerp. Wees voorzichtig met het gebruiken van kleurverloop. Al snel zijn ze in strijd met de ontwerpregels uit hoofdstuk 7.

Het is ook mogelijk om naast een CSS-achtergrond één of meerdere afbeeldingen toe te voegen. Om dit uit te proberen kun je de stijlregel bij

Index

spa-
e

#verloop1 uit het voorgaande voorbeeld vervangen door onderstaande regel. Het plaatje white.png is een wit vierkantje dat in PowerPoint is gemaakt en als afbeelding is opgeslagen (zie paragraaf 8.3.3).

```
background: url("white.png") calc(100% - 10px) calc(100% - 10px) /
50px no-repeat, linear-gradient(45deg, red, yellow);
```

8.3.8 Transparantie en andere filters

Veel van de in CSS beschikbare filters zijn met name geschikt voor afbeeldingen. Eén van de filters is ook voor andere elementen geschikt: transparantie.

Transparantie van elementen wordt gegeven met de eigenschap 'opacity'. De waarde van deze eigenschap wordt opgegeven als kommagetal. Een opacity van 1.0 is ondoorzichtig en een opacity van 0.5 is half doorzichtig. Als je met rgb-kleuren werkt, is het mogelijk gebruik te maken van de rgba-notering. Dan moet je bij de rgb-code ook nog het kommagetal van de opacity opgeven. Een rode kleur die voor 80% doorzichtig is, heeft dan als waarde 'rgba(255,0,0,0.2)'.

In de onderstaande tabel staan filters die voor afbeeldingen interessant zijn. Dit zijn allemaal waarden van de CSS-eigenschap 'filter'. Het is mogelijk meerdere waarden voor een filter te gebruiken, zoals:
filter: blur(3px) sepia(100%);

blur(px)	Maak een afbeelding wazig.
brightness(%)	Regel de helderheid. Een waarde van meer dan 100% is toegestaan.
contrast(%)	Regel het contrast. Een waarde van meer dan 100% is toegestaan.
drop-shadow	Deze filter lijkt op box-shadow. Kijk voor uitleg in hoofdstuk 6.
grayscale(%)	Maak een afbeelding grijs. 100% betekent volledig in grijstinten.

hue-rotate(deg)	Verander de kleuren in een afbeelding. Dit werkt met de kleurencirkel. In graden kan een rotatieverschuiving op de kleurencirkel worden aangegeven (0-360).
invert(%)	Maak een kleurennegatief van de afbeelding. 100% is volledig negatief.
opacity(%)	Dit werkt hetzelfde als de eigenschap 'opacity'.
saturate(%)	Regel de verzadiging van een afbeelding. Een waarde van meer dan 100% is toegestaan.
sepia(%)	Maak een sepia-afbeelding. 100% zorgt voor een volledig sepiaeffect.

8.3.9 Uitgebreide tekstopties

Ook voor tekst zijn er meer mogelijkheden dan we tot nu toe hebben behandeld. Misschien heb je al ontdekt dat niet alle lettertypes gebruikt kunnen worden op een webpagina. Er kan alleen gebruik worden gemaakt van zogenaamde 'web-safe fonts'. Lettertypes die geschikt zijn voor het web kun je vinden door op internet te zoeken naar 'web safe fonts'.

Het is echter ook mogelijk om zelf lettertypes toe te voegen. Zo kun je ervoor zorgen dat op jouw website een speciaal lettertype toch goed wordt weergegeven. Dat lettertype wordt dan automatisch gedownload door de gebruiker. Om dat te realiseren maak je gebruik van de regel @font-face in de CSS-code:

```
@font-face {
  font-family: Eigenlettertype;
  src: url(unsafefont.ttf);
}
```

In @font-face kun je nog meer eigenschappen toevoegen, zoals font-style, font-weight en font-stretch. Hoe dat werkt, kun je op internet vinden. Als je @font-face hebt gedeclareerd kun je naar het nieuwe lettertype verwijzen door elders in de CSS-code 'Eigenlettertype' te gebruiken als waarde bij de eigenschap font-family. Je kunt @font-face meerdere keren onder elkaar plaatsen als

je meerdere lettertypes wilt toevoegen. Houd er wel rekening mee dat alle bezoekers van je website deze bestanden moeten downloaden. Je website zal aanzienlijk langzamer laden als je lettertypes gebruikt die veel ruimte in beslag nemen en de hoeveelheid dataverkeer neemt toe. Er bestaan lettertypes die groter zijn dan 30MB.

Google heeft een database met lettertypes die al online staan, zodat je daarnaar kunt verwijzen in plaats van zelf lettertypes te moeten uploaden. Je kunt de lettertypes van Google vinden via www.google.com/fonts. Deze lettertypes moeten nog wel worden gekoppeld. Welke code daarvoor nodig is kun je op diezelfde website vinden.

Andere interessante tekstopties zijn beschikbaar via de volgende eigenschappen. De bijbehorende waarden kun je vinden op internet.

hyphens	Stel woordafbreking in bij lange woorden.
letter-spacing	Wijzig de ruimte tussen letters.
line-height	Wijzig de regelhoogte.
text-indent	Zorg ervoor dat de eerste regel van een alinea iets inspringt.
text-transform	Controleer het hoofdlettergebruik in een element (allemaal hoofdletters, geen hoofdletters, elk woord een hoofdletter).
word-spacing	Wijzig de ruimte tussen woorden.
text-decoration	Behalve het bekende underline/none bestaat er ook overline en line-through. Daarnaast kan met text-decoration-color, text-decoration-line, text-decoration-style en text-underline-position meer worden gezegd over de text-decoration.
text-shadow	Geef schaduw aan een tekst.
font-stretch	Rek de tekst meer uit dan normaal.
font-variant	Maak van de niet-hoofdletters kleine hoofdletters (ook wel klein kaptiaal genoemd). VOORBEELD

column-count	Verdeel de tekst over meerdere kolommen. Je kunt ook nog sleutelen aan de breedte van kolommen, van de ruimte tussen kolommen etc. De stijlregels die daarbij horen, laten we voor het gemak achterwege.
list-style-image	Gebruik een eigen afbeelding als punten in een ongeordende lijst.

8.3.10 Automatische hoofdstuknummering

Bij websites met veel tekst is het handig om hoofdstuknummering te gebruiken. In CSS is hier de 'counter' voor geïntroduceerd. Je kunt een counter activeren en resetten met de stijlregel 'counter-reset'. Bij de reset moet je ook een naam opgeven zodat je later naar die counter kunt verwijzen. Zo weet CSS welke counter moet worden getoond. Je kunt een counter toevoegen met het pseudo-element ::before (zie paragraaf 8.2.2). Een counter kan worden opgehoogd met de regel 'counter-increment'.

Hieronder een voorbeeld met hoofdstuk én paragraafnummering in CSS en HTML.

```
<h1>Introductie</h1>
<h2>Inleiding</h2>
<h2>De taal HTML</h2>
<h1>Basis</h1>
<h2>Inleiding</h2>
<h2>Structuur van een
HTML document</h2>
```

```
body {
  counter-reset: hoofdstuk;
}
h1 {
  counter-reset: paragraaf;
}
h1::before {
  counter-increment: hoofdstuk;
  content: counter(hoofdstuk) ". ";
}
h2::before {
  counter-increment: paragraaf;
  content: counter(hoofdstuk) "."
  counter(paragraaf) " ";
}
```

8.3.11 Overgangen en animaties

Met de introductie van CSS versie 3 is het mogelijk om elementen te vervormen en om animaties toe te voegen. Er zijn twee mogelijkheden om te animeren. Met de eigenschappen 'transition' en 'animation'. Je kunt hiermee bijvoorbeeld de achtergrondkleur van een element met een zekere snelheid naar een andere kleur laten verlopen of de breedte geleidelijk aanpassen. Niet alle eigenschappen zijn bruikbaar voor deze effecten. Eigenschappen moeten 'animatable' (animeerbaar) zijn. Op http://www.w3schools.com/cssref/css_animatable.asp is een lijst te vinden met alle eigenschappen die animeerbaar zijn.

Er zijn veel overeenkomsten tussen beide mogelijkheden, maar ook verschillen. Een transitie wordt geactiveerd door een pseudoclass (bijvoorbeeld :hover), een animatie kan al beginnen bij het laden van de website. Een animatie kan ook naar keuze worden herhaald.

Transition

Een transitie kan worden gemaakt met de eigenschap 'transition'. Als waarden van die eigenschap moeten worden opgegeven:

1. welke eigenschap er wordt geanimeerd,
2. hoe lang de animatie mag duren,
3. of er snelheidseffecten bij komen,
4. na hoeveel tijd de animatie wordt uitgevoerd.

Voer eventueel het volgende voorbeeld uit om alle vier de waarden goed te bekijken.

```
<a href="_blank">Lege pagina</a>
```

```
a {
  color: white;
  text-decoration: none;
  padding: 5px 10px;
  background-color: darkblue;
  box-shadow: 0 0 0 grey;
  transition: box-shadow 2s ease-
  in-out 1s;
}

a:hover{
  background-color: white;
  color: black;
  box-shadow: 2px 2px 10px grey;
}
```

Op internet zijn veel voorbeelden te vinden van mooie designs door gebruik van transition.

Animation

Een animatie is iets lastiger te maken. Deze moet namelijk eerst gedeclareerd worden met een mediaquery. Hiervoor moet de eigenschap @keyframes worden gebruikt. In @keyframes kan per onderdeel van de animatie (dat je opgeeft in procenten van de verstreken tijd) worden aangeven wat de stand van zaken op dat moment moet zijn. Naast deze declaratie is er net als transition een stijlregel 'animation'. Bij animation zijn nog meer waarden op te geven dan bij transition. Als je je hierin wilt verdiepen, zul je ze zelf moeten opzoeken op internet.

Het volgende voorbeeld is niet nuttig, maar wel duidelijk en grappig:

Index

show

```
<a href="_blank">Lege pagina</a>
```

```
@keyframes voorbeeldanimatie {
  0% {top: 0px; left: 0px;}
  25% {top: 100px; left: 50px;}
  50% {top: 200px; left: 100px;}
  75% {top: 0px; left: 150px;}
  100% {top: 0px; left: 0px;}
}
a {
  color: white;
  text-decoration: none;
  padding: 5px 10px;
  background-color: darkblue;
  position: relative;
  animation: voorbeeldanimatie 1s
  infinite paused;
}
a:hover{
  animation: voorbeeldanimatie 1s
  infinite running;
}
```

8.3.12 Een mooie CSS slideshow

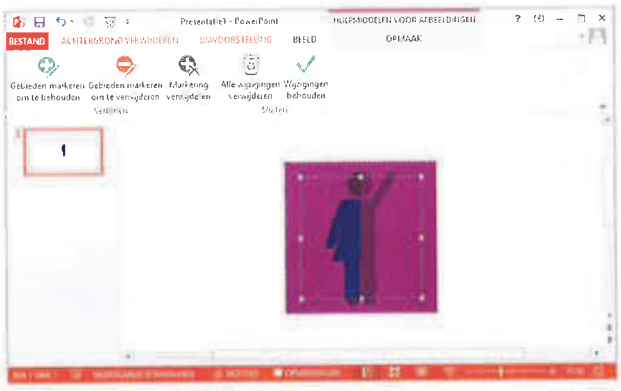
Door de eigenschappen van animation slim te gebruiken is het mogelijk om een slideshow van plaatjes te maken zonder gebruik te maken van een JavaScript. Hieronder staat een voorbeeld van een slideshow met twee foto's. Wil je het nog mooier doen? Kijk eens op http://www.alessioatzeni.com/CSS3-Cycle-Image-Slider.

Bijsnijden en transparant maken

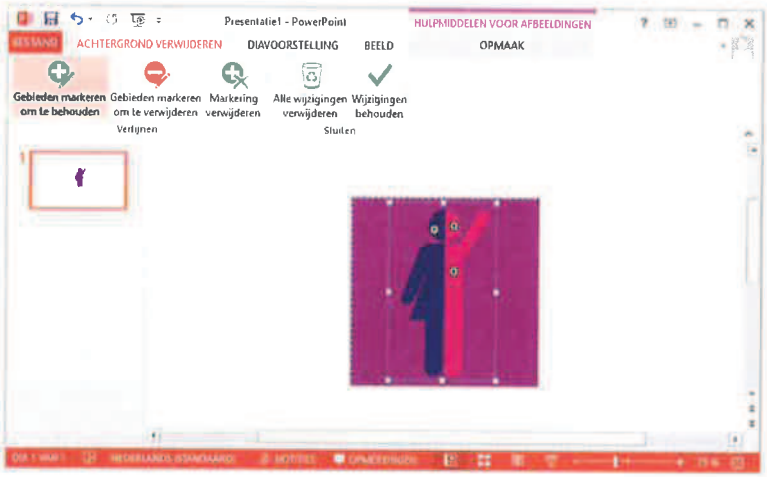
Soms wil je maar een deel van de afbeelding gebruiken. Dan moet de afbeelding worden bijgesneden. Een rechthoekige uitsnede is niet zo moeilijk. De al genoemde oplossingen kunnen ook hiervoor worden gebruikt. Als er een uitsnede moet worden gemaakt die meer grillige contouren volgt, wordt het moeilijker. Laten we voorop stellen dat het sowieso heel moeilijk is om iets uit een foto te snijden, zelfs als je een ervaren Photoshopper bent. Daarnaast ziet het er meestal niet mooi uit. Het werkt het beste als het contrast tussen het te behouden en uit te snijden gebied hoog is. Bijvoorbeeld als bij onderstaande afbeelding:

In PowerPoint bestaat een mooie functie om een dergelijke afbeelding van de achtergrond te ontdoen. Als de afbeelding in een dia is geplakt kun je gebruik maken van de functie 'Achtergrond verwijderen' in het menu 'Opmaak'.

Aanklikken levert de volgende situatie op:



Het paarse gebied zal worden weggesneden. Dit is natuurlijk meer dan we willen. Pas daarom de selectievierhoek zo aan dat de figuur er goed in past. Soms levert dat direct een goed resultaat op, maar soms moet je met de opties 'Gebieden markeren om te behouden' en 'Gebieden markeren om te verwijderen' één of meer punten neerzetten. Je helpt het programma dan te bepalen wat wel en niet moet worden weggesneden.



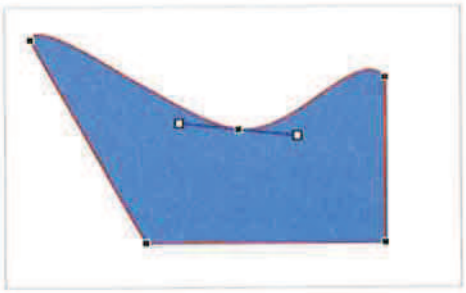
Kies als laatste voor 'Wijzigingen behouden'. De afbeelding is nu uitgesneden. Om eventuele uitsnijfouten wat minder op te laten vallen kan er bij 'Afbeeldingseffecten' nog gekozen worden voor een 'vloeiende rand' van 1 punt. Als je de afbeelding nu wilt gebruiken voor je website, kun je op de afbeelding rechtsklikken en gebruik maken van 'Opslaan als afbeelding'. Sla de afbeelding wel op als PNG, want dat formaat kan omgaan transparantie.

Je kunt aan het resultaat zien  dat de achtergrond van de afbeelding is weggesneden.

Je zult merken dat het nog niet zo makkelijk is om afbeeldingen goed bij te snijden. Bij de meest moderne websites wordt het ook nauwelijks toegepast.

In PowerPoint zitten nog veel meer effecten die op afbeeldingen kunnen worden toegepast en je kunt ook zelf figuren maken. Experimenteer eens met vrije vormen als je hierin geïnteresseerd bent. Voeg bijvoorbeeld een rechthoek toe via Invoegen > Vormen. Selecteer de rechthoek en kies in het menu voor Opmaak > Vorm bewerken > Punten bewerken. Sleep met de punten, sleep met de 'handvatten' van ieder punt, voeg punten toe door rechts te klikken op de rand van het figuur. Leef je uit!

Onderstaande figuur is een voorbeeld van een rechthoek waarvan één van de punten wordt bewerkt.



8.4.4 PHP, MySQL

PHP is de afkorting van Hypertext PreProcessor. Dit is een server-side programmeertaal. Dit betekent dat de scripts die je met deze programmeertaal maakt, op een server moeten worden uitgevoerd. Makkelijker gezegd: PHP-code wordt niet uitgevoerd op de computers van de mensen thuis, zoals HTML- en CSS-code, maar op de servers waarop de PHP-scripts staan.

PHP-code wordt direct in HTML-pagina's verwerkt. Een willekeurige bezoeker heeft niet in de gaten dat er een PHP-code wordt uitgevoerd. De PHP-scripts worden namelijk door de server verwerkt en genereren HTML-code als uitvoer. PHP-scripts kunnen met diverse databases communiceren. Een database is een verzameling tabellen waar informatie in kan worden opgeslagen. Je kunt het vergelijken met een Excel-document. Ook daarin kun je een grote hoeveelheid informatie gestructureerd opslaan. De database MySQL is veruit het meest

populair onder internetontwikkelaars. Bij de meeste hostingproviders is MySQL de enige beschikbare database. In combinatie met PHP kun je alle typen webapplicaties ontwikkelen: van een eenvoudig gastenboek tot een complete webshop.

Als je meer wilt weten over PHP en MySQL, kunnen de volgende websites voor jou handig zijn:

- www.w3schools.com/php
- www.phphulp.nl

Bij Uitgeverij Instruct is ook een boek beschikbaar over PHP en MySQL.

8.4.5 Formulieren

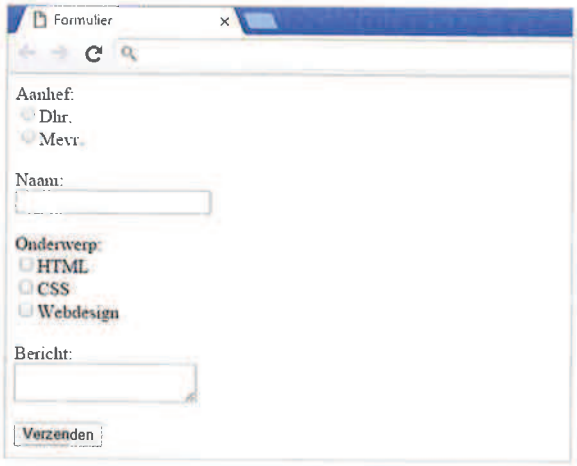
Je kunt met HTML-code een formulier maken waarin de bezoeker van de website gegevens kan invullen. De verwerking van de ingevulde formuliergegevens gebeurt over het algemeen met PHP-code, hoewel het ook met JavaScript mogelijk is. Hoe die verwerking precies gaat, komt in deze paragraaf niet aan de orde. Je kunt er alles over vinden op deze website: http://www.w3schools.com/php/php_forms.asp. Je leert in deze paragraaf wel hoe je een formulier op je website kunt plaatsen.

Een formulier kan bestaan uit tekstvakken, aanvinkvakjes en andere elementen. Je kunt zelfs bestanden van de computer selecteren en die uploaden naar de website. De meeste gebruikte HTML-formulierelementen zijn:

- Een tekstvak (textfield of textarea)
- Een aanvinkrondje (radio button)
- Een aanvinkvakje (checkbox)
- Een verzendknop (submit button)

Hieronder is een eenvoudig formulier weergegeven met invoervakken, aanvinkvakjes en -rondjes en een verzendknop.

```
<form method="post" action="">
  Aanhef:<br>
  <input type="radio" name="aanhef" value="dhr">Dhr.<br>
  <input type="radio" name="aanhef" value="mevr">Mevr.<br>

  <br>

  Naam:<br>
  <input type="text" name="naam"><br>

  <br>

  Onderwerp:<br>
  <input type="checkbox" name="onderwerp" value="html">HTML<br>
  <input type="checkbox" name="onderwerp" value="css">CSS<br>
   <input type="checkbox" name="onderwerp"
   value="design">Webdesign<br>

  <br>

  Bericht:<br>
  <textarea name="bericht"></textarea><br>

  <br>

  <input type="submit" name="verzenden" value="Verzenden">
</form>
```



Een formulier begint en eindigt altijd met de HTML-tag form. Deze form-tag heeft twee attributen: method en action.

Het attribuut method heeft twee mogelijke waarden: 'post' en 'get'. Het verschil hiertussen is dat bij de waarde 'get' de ingevulde gegevens in de URL van de website zichtbaar zijn, bijvoorbeeld:

mijnformulier.html?aanhef=dhr&naam=Henk&onderwerp=HTML&bericht=Hallo&verzenden=Verzenden

Het is aan te raden om altijd de waarde 'post' te gebruiken. Zeker ook als je een inlogformulier hebt, wil je niet dat het wachtwoord dat de bezoeker heeft ingevoerd na het inloggen zichtbaar is in de URL!

Als je de eigenschap action weg laat of geen waarde geeft, vindt de verwerking van het formulier plaats op dezelfde webpagina. Als je wilt dat de verwerking van het formulier op een andere webpagina plaatsvindt, geef je als waarde de bestandsnaam van deze webpagina op.

Hieronder staat een overzicht van de meest gebruikte HTML-formulierelementen:

Tekstvak	**input**	
	type="text"	type van 'input'
	name	unieke naam in formulier
	value (niet verplicht)	default waarde
	size (niet verplicht)	breedte
Tekstvak meerdere regels	**textarea**	
	name	unieke naam in formulier
	cols (niet verplicht)	aantal kolommen breed
	rows (niet verplicht)	aantal rijen hoog
Radio button	**input**	
	type="radio"	type van 'input'
	name	unieke naam in formulier
	value	unieke waarde
	selected="selected"	standaard aangevinkt
Check box	**input**	
	type="checkbox"	type van 'input'
	name	unieke naam in formulier
	value	unieke waarde
	checked="checked"	standaard aangevinkt
Verzendknop	**input**	
	type="submit"	type van 'input'
	name	unieke naam in formulier
	value	tekst op de knop

Met de komst van HTML versie 5 is er een aantal nieuwe formulierelementen beschikbaar gekomen. Enkele worden hieronder kort besproken.

Datalist

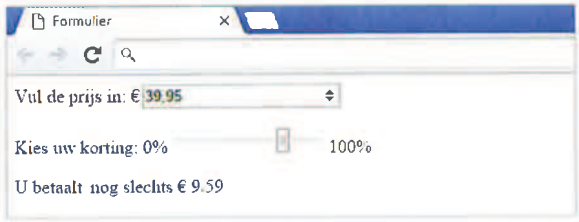
Een datalist is een uitbreiding op een tekstvak. Als webontwikkelaar kun je zelf een aantal mogelijke waarden opgeven die een bezoeker in het tekstvak kan invoeren. Het blijft voor de bezoeker daarnaast ook mogelijk om een waarde in te vullen die jij vooraf niet hebt opgegeven.

```
<form method="post" action="">
  <input list="browsers" name="browser">
  <datalist id="browsers">
    <option value="Internet Explorer">
    <option value="Firefox">
    <option value="Chrome">
    <option value="Opera">
    <option value="Safari">
  </datalist>
  <input type="submit" name="verzenden" value="Verzenden">
</form>
```

Output

Het output-element geeft het resultaat van een (wiskundige) berekening weer. In het voorbeeld hieronder komen ook twee nieuwe input-typen aan de orde, namelijk 'range' en 'number'. Ook is er een klein stukje JavaScript-code (zie paragraaf 8.3.6) in het voorbeeld verwerkt.

```
<form method="post" action="" oninput="uitkomst.value = (prijs.value
- prijs.value * korting.value / 100).toFixed(2)">
 <p>
   Vul de prijs in: &euro;
   <input type="number" id="prijs" name="prijs" value="39.95">
 </p>
 <p>
   Kies uw korting: 0%
   <input type="range" id="korting" name="korting" min="0" max="100">
   100%
  </p>
  <p>
  U betaald nog slechts &euro; <output name="uitkomst" for="prijs
  korting"></output>
 </p>
</form>
```



Een inputveld van het type 'number' is een uitbreiding op een tekstvak. Bij een tekstvak met type number is het alleen mogelijk om getallen in te voeren en staan er rechts in het tekstvak twee pijlen, waarmee je de waarde kunt verhogen of verlagen.

Een inputveld van het type 'range' geeft een schuifbalk weer. Je kunt een minimale en een maximale waarde opgeven, in het bovenstaande voorbeeld 0 en 100.

Het output-element heeft het attribuut 'for'. De waarde(n) van deze attributen zijn de namen van de formulier-elementen die worden gebruikt voor het antwoord van de berekening, gescheiden door een spatie. De berekening zelf vindt plaats in de form-tag, die is nu uitgebreid met de volgende coderegel:

```
oninput="uitkomst.value = (prijs.value - prijs.value * korting.value /
100).toFixed(2)"
```

Oninput betekent dat als een formulier-element (bijvoorbeeld een tekstvak) een andere waarde krijgt, de code tussen de aanhalingstekens (") wordt uitgevoerd. Als deze code wordt uitgevoerd, krijgt het formulier-element met de naam 'prijs' een nieuwe waarde als value, namelijk de uitkomst van de berekening:

```
(prijs.value - prijs.value * korting.value / 100).toFixed(2)
```

In deze berekening wordt het kortingspercentage met de prijs verrekend. toFixed(2) maakt vervolgens van het resultaat een kommagetal dat is afgerond op twee decimalen.

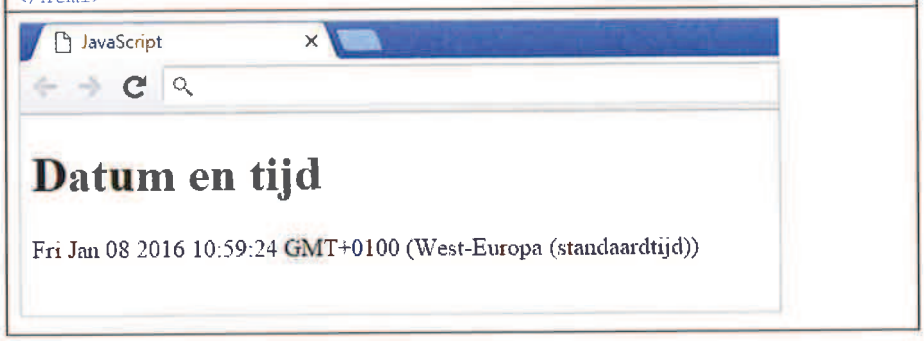
In deze paragraaf zijn slechts enkele van alle beschikbare formulier-elementen besproken. Een overzicht met nog meer elementen vind je door op internet te zoeken naar 'HTML5 forms'.

8.4.6 JavaScript

JavaScript is een programmeertaal voor het web, net als bijvoorbeeld PHP. Er is een belangrijk verschil tussen die twee. PHP wordt uitgevoerd op de server. Dat geldt niet voor JavaScript. Die code wordt, net als HTML en CSS, door de webbrowser op de computer van de gebruiker uitgevoerd. In deze paragraaf staat een korte introductie in JavaScript. Wil je hier meer over weten, kijk dan op de website www.w3schools.com/js.

In het volgende voorbeeld wordt, op het moment dat de webpagina wordt geladen, informatie over de huidige datum en tijd weergegeven in een paragraaf.

```
<!DOCTYPE html>
<html>
  <head>
    <script type="text/javascript">
      function datumEnTijd() {
        document.getElementById('demo').innerHTML = Date();
      }
    </script>
  </head>

  <body onload="datumEnTijd()">
    <h1>Datum en tijd</h1>
    <p id="demo"></p>
  </body>
</html>
```



Een korte uitleg van deze code. In het head-element is een script-tag toegevoegd met JavaScript code. Deze JavaScript-code bevat een functie (een stuk code dat de webbrowser kan uitvoeren) met de naam 'datumEnTijd()'. Deze functie doet het volgende: hij zoekt het HTML-element met id 'demo' op en plaatst in dat element informatie over de huidige datum en tijd. Zoals je kunt zien in de HTML-code is er een p-element met het id 'demo'. De functie zal in dit geval dus de informatie over de huidige datum en tijd neerzetten in dat p-element.

In de body-tag staat onload="datumEnTijd()". Dit betekent dat zodra de pagina klaar is met laden, de functie met de naam 'datumEnTijd' wordt uitgevoerd. Die zal vervolgens de informatie over de huidige datum en tijd in de paragraaf met het id 'demo' weergeven.

8.4.6.1 jQuery

jQuery is een bibliotheek met standaardcodes waardoor het programmeren in JavaScript eenvoudiger wordt. In deze paragraaf staat een korte introductie in jQuery. Wil je hier meer over weten, kijk dan op de website www.w3schools.com/jquery.

In het volgende voorbeeld is het mogelijk om de afbeelding wel of niet weer te geven, afhankelijk van de tekst waarop je klikt.

```
<!DOCTYPE html>
<html>
  <head>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/
    jquery.min.js">
      </script>
      <script type="text/javascript">
        $(document).ready(function() {
          $("#aan").click(function() {
            $("#demo").show();
          });
          $("#uit").click(function() {
            $("#demo").hide();
          });
        });
    </script>
  </head>

  <body>
      <p id="aan">Toon afbeelding</p>
      <p id="uit">Verberg afbeelding</p>
      <img id="demo" src="logo.png">
  </body>
</html>
```

Om te kunnen werken met jQuery moet je eerst de bibliotheek toevoegen aan je webpagina. Dit doe je in het head-element met de script-tag. Vervolgens kun je met JavaScript programmeren, tussen de script-tags. Daarbij gebruik je de code uit de jQuery bibliotheek.

Kort gezegd doet de code het volgende. Als er op het HTML-element met het id 'aan' wordt geklikt, wordt het HTML-element met het id 'demo' weergeven. Wordt er op het HTML-element met het id 'uit' geklikt, dan wordt het HTML-element met het id 'demo' verborgen.

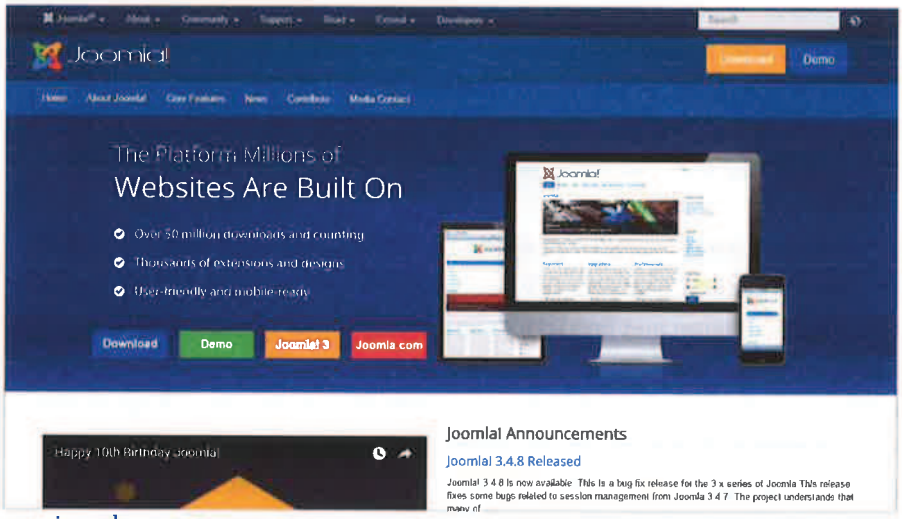
De mogelijkheden van JavaScript en jQuery zijn bijzonder uitgebreid. Daarom is er op internet heel veel over te vinden. Zoals je aan de voorbeelden kunt zien, is het programmeren met JavaScript een stuk moelijker dan met HTML. Maar soms lijkt het wel een beetje op complexe CSS functies. De organisatie die HTML en CSS ontwikkelt, probeert bij elke versie interessante functies uit JavaScript over te nemen in HTML en CSS. Op die manier worden websites sneller en hebben webdesigners minder programmeerkennis nodig bij het gebruiken van geavanceerde mogelijkheden.

8.4.7 Websites maken met een CMS

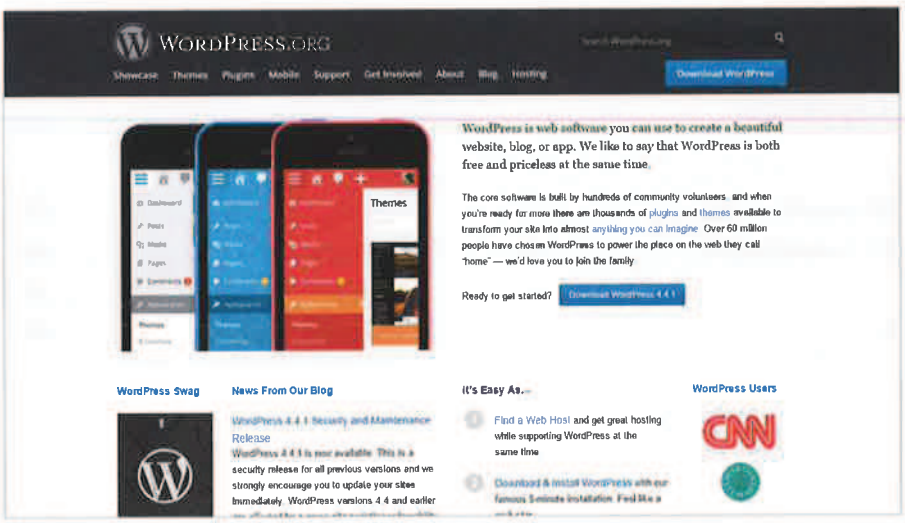
In dit boek is beschreven hoe je met HTML- en CSS-code een website kunt maken. Voor veel mensen is dat te ingewikkeld of kost het te veel tijd. Als je toch graag zelf een website wilt maken, maar geen tijd of zin hebt om HTML- en CSS-code te leren, kun je gebruik maken van een contentmanagementsysteem, vaak afgekort tot CMS.

Bij een CMS is het gebruikelijk dat je een sjabloon kiest voor het uiterlijk van je website. Daarna hoef je alleen tekst (en plaatjes) op te geven en de website is klaar. Dit klinkt heel eenvoudig en dat is het ook. Maar er zijn wel nadelen. Als je zelf een website maakt met HTML- en CSS-code kun je veel meer aanpassingen doen aan de opmaak van je website. Bij het gebruik van een CMS ben je vooral gebonden aan het sjabloon dat je gekozen hebt. Ook zijn websites die gebaseerd zijn op een CMS trager dan websites die zelf gebouwd zijn. In een CMS zitten namelijk allerlei functies die ook worden ingeladen als de gebruiker ze niet gebruikt.

Er zijn veel verschillende CMS systemen, sommigen zijn gratis. Twee bekende CMS systemen zijn Joomla! en WordPress.



www.joomla.org



nl.wordpress.org