

Basis statische website

Realiseren

hoofdstuk

3

Introductie CSS





Algemene informatie

Onderwerp	Introductie CSS
Leerdoel(en)	<ol style="list-style-type: none">1. De student kan benoemen wat het verschil is tussen inline-, ingesloten-, en externe CSS.2. De student kan een extern CSS-bestand koppelen aan een HTML-bestand.3. De student kan benoemen wat selectoren, eigenschappen en eigenschapwaarden zijn.4. De student is in staat om op logische wijze de volgende type selectoren toe te passen: universal, element, attribute, id, class, child, descendant, general sibling, adjacent sibling.5. De student is in staat om de tekstkleur en de achtergrondkleur van een tekst aan te passen.6. De student is in staat om kleurcodes te lezen en in te schatten wat voor een kleur het is.7. De student is in staat om het lettertype aan te passen van een tekst, ook als deze niet geïnstalleerd is.8. De student is in staat om de lettergrootte aan te passen en maakt bij het kiezen van de eenheid daarvoor een weloverwogen besluit.9. De student is in staat om teksten dikgedrukt, schuingedrukt en onderstreept weer te geven.10. De student kan benoemen wat het begrip overerving betekent.11. De student kan meerdere stylesheets koppelen aan één HTML-bestand en kan toelichten wanneer dit in de praktijk handig kan zijn.
Vereiste voorkennis	De student moet een basiskennis hebben opgebouwd over HTML waarbij de student in staat is om eenvoudige teksten te kunnen coderen in HTML. Deze kennis is opgedaan in hoofdstuk 2.
Kwalificatiedossier	<ul style="list-style-type: none"><input type="checkbox"/> B1-K1-W1: Plant werkzaamheden en bewaakt de voortgang<input type="checkbox"/> B1-K1-W2: Ontwerpt software<input checked="" type="checkbox"/> B1-K1-W3: Realiseert (onderdelen van) software<input type="checkbox"/> B1-K1-W4: Test software<input type="checkbox"/> B1-K1-W5: Doet verbetervoorstellen voor de software <input type="checkbox"/> B1-K2-W1: Voert overleg<input type="checkbox"/> B1-K2-W2: Presenteert het opgeleverde werk<input type="checkbox"/> B1-K2-W3: Reflecteert op het werk



Inhoudsopgave

Algemene informatie	2
Inhoudsopgave	3
Introductie	5
Inhoud	5
CSS koppelen	5
Inline CSS.....	5
Ingesloten CSS.....	6
Externe CSS.....	6
Selectoren.....	7
Overzicht type selectoren.....	7
Universeel.....	8
Element.....	9
Deel van een element selecteren.....	10
Id.....	10
Klasse.....	11
Afstammeling.....	12
Kind	13
Algemene buur.....	14
Naaste buur	15
Selectoren combineren	16
Overerven	17
Kleuren begrijpen	19
Kleurnamen	19
RGB-code	20
Hexadecimale-code	21
Omrekenen van hexadecimale code naar RGB-code	21
Omrekenen van RGB-code naar hexadecimale-code	22
Omrekenen met de rekenmachine	22
Eigenschappen	23
Tekstkleur.....	23
Achtergrondkleur.....	23
Tekstgrootte	24
Vetgedrukt.....	24
Schuingedrukt.....	25



Onderstrepen en doorhalen	25
Uitlijnen.....	25
Lettertype.....	26
Lettertype van internet	26



Introductie

De webpagina's die je in het vorige hoofdstuk gemaakt hebt zien er nog niet erg aantrekkelijk uit. In dit hoofdstuk ga je aan de slag met het toevoegen van opmaak aan een HTML-pagina. Zoals je eerder al geleerd hebt gebruik je hiervoor Cascading StyleSheets (CSS). Dat is weer een andere techniek dan HTML.

Inhoud

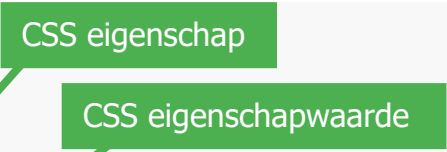
CSS koppelen

Je kunt CSS op verschillende manieren toevoegen aan je HTML-code. Deze drie manieren worden hieronder toegelicht. In de lessen is het de bedoeling dat je altijd gebruik maakt van **externe CSS**, tenzij in de opdracht vermeld staat dat het anders moet.

Inline CSS

Inline CSS wordt geschreven in het HTML-bestand. Als je bijvoorbeeld een paragraaf een andere kleur wilt geven doe je dat door het p-element het attribuut style te geven en een attribuutwaarde waarin je styling aangeeft. Binnen de attribuutwaarde geef je een CSS eigenschap op gevolgd door een dubbele punt en een CSS eigenschapwaarde. Welke CSS-eigenschappen en CSS eigenschapwaarden er allemaal zijn staat verderop in dit hoofdstuk uitgelegd.

```
<h1>
  Inline CSS
</h1>
<p style="color:red">
  Doormiddel van inline CSS wordt deze tekst nu rood.
</p>
```



In de browser ziet dit er zo uit:



Het voordeel van inline CSS is dat je snel een opmaak kunt coderen. Echter is deze manier niet erg onderhoudsvriendelijk en wordt daarom ook niet aangeraden op het moment dat je zelf CSS-code schrijft. Als je bijvoorbeeld een webpagina hebt met 10 paragrafen moet je alle p-elementen voorzien van een style-attribuut. Wil je de kleur daarna weer aanpassen, moet je dat ook weer voor alle paragrafen doen.



Ingesloten CSS

Ingesloten CSS wordt ook geschreven in het HTML-bestand. Ook hier gaan we kijken naar hoe je op die manier een paragraaf een andere kleur kunt geven. Bij ingesloten CSS voeg je het element style toe binnen het head-element. In het style-element plaats je een selector (een stukje code om bepaalde elementen te selecteren), welke CSS eigenschap of CSS eigenschappen je wilt aanpassen met de bijbehorende CSS eigenschapswaarden.

```
<head>
  <title>
  </title>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <style>
    p
    {
      color:red;
    }
  </style>
</head>
<body>
  <h1>
    Inline CSS
  </h1>
  <p>
    Doormiddel van inline CSS wordt deze tekst nu rood.
  </p>
</body>
```

De selector bepaalt nu welke inhoud een andere opmaak krijgt. Bij een p-selector worden alle paragrafen geselecteerd. Welke selectoren je allemaal nog meer kunt gebruiken wordt verderop in dit hoofdstuk uitgelegd.

Externe CSS

Eigenlijk ziet dit er hetzelfde uit als wat er in een style-element staat bij ingesloten CSS. Je gaat het bij externe CSS in een eigen CSS-bestand plaatsen zodat je code overzichtelijker en beter te onderhouden blijft. Als je de CSS in een eigen bestand zet moet je HTML-document natuurlijk wel weten waar de CSS te vinden is, ofwel je moet de twee bestanden aan elkaar koppelen. Dat doe je door in de head het link-element toe te voegen met het attribuut rel en src.

```
<head>
  <title>
  </title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="styles/stylesheet.css">
</head>
```

Je ziet dat link een leeg-element is, omdat je deze niet hoeft af te sluiten. Bij het attribuut rel geeft je aan wat de relatie is tussen de twee bestanden. Bij href geef je de url op waar het bestand te vinden is. Zoals in hoofdstuk 1 aangegeven plaats je CSS-bestanden in een map genaamd styles.



Selectoren

Een selector selecteert HTML-elementen. Op die manier kun je zelf bepalen welke elementen een specifieke opmaak moeten krijgen. Er zijn heel veel verschillende soorten selectoren. Een aantal veelgebruikte worden hieronder verder toegelicht. Om de selectoren toe te lichten gaan we als basis uit van onderstaande HTML-code:

```
<h1>
  Selectoren in <em>CSS</em>
</h1>
<p id="introduction" class="red">
  <strong>Een selector selecteert <em>HTML</em>-elementen</strong>. Op die manier kun
  je <em>zelf</em> bepalen welke elementen een specifieke opmaak moeten krijgen. Er
  zijn heel veel verschillende soorten selectoren.
</p>
<p>
  In dit hoofdstuk leer je maar liefst <em class="red">acht</em> verschillende soorten
  selectoren!
</p>
```

Overzicht type selectoren

Onderstaande selectoren worden in de rest van dit hoofdstuk uitgebreider beschreven.

Type selector	Omschrijving	Voorbeeld	Uitleg
Universeel Engels: Universal	Selecteert alle elementen	*	Selecteert alle elementen
Element Engels: Element / type	Selecteert alle elementen die overeenkomen met het element uit de selector	p	Selecteert alle p-elementen
Id Engels: Id	Selecteert alle elementen waarvan in HTML de attribuutwaarde van het attribuut id overeenkomt met het id uit de selector	#introduction	Selecteert alle elementen die een id-attribuut hebben met als attribuutwaarde introduction
Klasse Engels: Class	Selecteert alle elementen waarvan in HTML de attribuutwaarde van het attribuut class overeenkomt met het id uit de selector	.red	Selecteert alle elementen die een class-attribuut hebben met als attribuutwaarde red
Afstammeling Engels: Descendant	Selecteert alle elementen die binnen een ander element geplaatst (genest) zijn	p strong	Selecteert alle strong elementen die binnen een p-element geplaatst (genest zijn)
Kind Engels: Child	Selecteert alle elementen die direct binnen een ander element geplaatst (genest) zijn, er mag dus geen element tussen zitten	p>strong	Selecteert alle strong elementen die direct binnen een p-element geplaatst (genest zijn)
Algemene buur Engels: General sibling	Selecteert elementen die na een andere element komen	h1~p	Selecteert alle p-elementen die na een h1-element komen
Naaste buur Adjacent sibling	Selecteert elementen die direct na een andere element komen, er mag dus geen element tussen zitten	h1+p	Selecteert alle p-elementen die direct na een h1-element komen



Universeel

```
*  
{  
    color:red;  
}
```

De universele selector selecteert alle elementen uit de HTML-code.

```
<h1>  
    Selectoren in CSS  
</h1>  
<p id="introduction" class="red">  
    Een selector selecteert HTML-elementen Op die manier kun  
    je zelf bepalen welke elementen een specifieke opmaak moeten krijgen. Er  
    zijn heel veel verschillende soorten selectoren.  
</p>  
<p>  
    In dit hoofdstuk leer je maar liefst acht verschillende soorten  
    selectoren!  
</p>
```

In de browser ziet dat er zo uit:





Element

```
p  
{  
    color:red;  
}
```

De element selector selecteert alle elementen die overeenkomen met het element dat als selector is opgegeven. In dit voorbeeld worden alle p-elementen opgemaakt, maar dit werkt zo voor alle elementen.


```
<h1>  
    Selectoren in <em>CSS</em>  
</h1>  
<p id="introduction" class="red">  
    <strong>Een selector selecteert <em>HTML</em>-elementen</strong>. Op die manier kun  
    je <em>zelf</em> bepalen welke elementen een specifieke opmaak moeten krijgen. Er  
    zijn heel veel verschillende soorten selectoren.  
</p>  
<p>  
    In dit hoofdstuk leer je maar liefst <em class="red">acht</em> verschillende soorten  
    selectoren!  
</p>
```

In de browser ziet dat er zo uit:



Als je meerdere elementen dezelfde opmaak wilt geven kun je gebruik maken van een gecombineerde selector. Je plaatst dan tussen de selectoren een komma.

```
h1,p  
{  
    color:red;  
}
```

 Je kunt nu **Oefening 3.1** maken. Tip: kijk wat verder in de reader om te zien wat je naast kleur nog meer kunt aanpassen!



Deel van een element selecteren

Het komt regelmatig voor dat je een deel van een element wilt selecteren. Bijvoorbeeld alleen een bepaald woord in een koptekst of paragraaf. Dit kun je oplossen door het gebruik van een span-element.

```
<h1>
  Gebruik van span
</h1>
<p>
  Daarmee kun je een deel van een element selecteren.<br>
  Bijvoorbeeld om deze tekst een andere opmaak te geven.
</p>
```

Id

```
#introduction
{
  color:red;
}
```

Je kunt HTML-elementen een id-attribuut opgeven. Daarmee kun je ze als het ware een naam geven. Let op dat je de attribuutwaarde (de naam van het id) maar één keer mag gebruiken. Als je een id gebruikt is dat dus het enige element dat je met die specifieke opmaak wilt opmaken. Met een id selector selecteer je het element dat je dat id hebt gegeven. Een id selector start altijd met een hashtag (#) gevolgd door de naam van het id.

```
<h1>
  Selectoren in <em>CSS</em>
</h1>
<p id="introduction" class="red">
  <strong>Een selector selecteert <em>HTML</em>-elementen</strong>. Op die manier kun
  je <em>zelf</em> bepalen welke elementen een specifieke opmaak moeten krijgen. Er
  zijn heel veel verschillende soorten selectoren.
</p>
<p>
  In dit hoofdstuk leer je maar liefst <em class="red">acht</em> verschillende soorten
  selectoren!
</p>
```

In de browser ziet dat er zo uit:





Klasse

```
.red
{
    color:red;
}
```

Je kunt HTML-elementen een class-attribuut opgeven. Daarmee kun je ze als het ware een naam geven. Het lijkt ontzettend op de functie van een id-attribuut. Bij een class-attribuut mag je echter de attribuutwaarde (de naam van de klasse) vaker gebruiken. Als je een class gebruikt wil je dus meerdere element met dezelfde specifieke opmaak opmaken. Met een class selector selecteer je de elementen die je die class hebt gegeven. Een class selector start altijd met een punt (.) gevolgd door de naam van de klasse.

```
<h1>
  Selectoren in <em>CSS</em>
</h1>
<p id="introduction" class="red">
  <strong>Een selector selecteert <em>HTML</em>-elementen</strong>. Op die manier kun
  je <em>zelf</em> bepalen welke elementen een specifieke opmaak moeten krijgen. Er
  zijn heel veel verschillende soorten selectoren.
</p>
<p>
  In dit hoofdstuk leer je maar liefst <em class="red">acht</em> verschillende soorten
  selectoren!
</p>
```

In de browser ziet dat er zo uit:



Je kunt dit ook combineren met een element selector.

```
p.red
{
    color:red;
}
```

Hiermee selecteer je alle p-elementen die een klasse red hebben.



Je kunt nu **Oefening 3.2** maken.



Afstammeling

```
p em
{
    color:red;
}
```

Bij een afstammeling selector selecteer je elementen die voorkomen binnen een ander element. In dit voorbeeld selecteer je alle em-elementen die in een p-element staan. Je kunt ook zeggen dat hij alle em-element selecteert die in een p-element genest zijn. Een afstammeling selector herken je aan de spatie.

```
<h1>
  Selectoren in <em>CSS</em>
</h1>
<p id="introduction" class="red">
  <strong>Een selector selecteert <em>HTML</em>-elementen</strong>. Op die manier kun
  je <em>zelf</em> bepalen welke elementen een specifieke opmaak moeten krijgen. Er
  zijn heel veel verschillende soorten selectoren.
</p>
<p>
  In dit hoofdstuk leer je maar liefst <em class="red">acht</em> verschillende soorten
  selectoren!
</p>
```

In de browser ziet dat er zo uit:



Je ziet dat het em-element die in het h1-element staat nu dus niet geselecteerd wordt!

Je kunt ook hier combineren met andere type selectoren.

```
p .red
{
    color:red;
}
```

Hiermee selecteer je alle elementen met de klasse red die in een p-element genest zijn.



Kind

```
p>em
{
    color:red;
}
```

Een kind selector lijkt erg veel op een afstammeling selector. Bij een kind selector selecteer je echter alleen de elementen die direct (zonder tussenkomst van andere elementen) voorkomen binnen een ander element. In dit voorbeeld selecteer je alle em-elementen die direct in een p-element staan, zonder dat daar andere elementen tussen mogen staan. Een kind selector herken je aan het groter dan teken (>).

```
<h1>
  Selectoren in <em>CSS</em>
</h1>
<p id="introduction" class="red">
  <strong>Een selector selecteert <em>HTML</em>-elementen</strong>. Op die manier kun
  je <em>zelf</em> bepalen welke elementen een specifieke opmaak moeten krijgen. Er
  zijn heel veel verschillende soorten selectoren.
</p>
<p>
  In dit hoofdstuk leer je maar liefst <em class="red">acht</em> verschillende soorten
  selectoren!
</p>
```

In de browser ziet dat er zo uit:



Je ziet dat het eerste em-element die in de eerste paragraaf staat nu niet geselecteerd wordt. Dit komt omdat hij niet direct in een p-element staat, maar in een strong-element. Het is dus geen kind van een p-element, maar van een strong-element.

Je kunt ook hier combineren met andere type selectoren.

```
p>.red
{
    color:red;
}
```

Hiermee selecteer je alle elementen met de klasse red die direct in een p-element genest zijn.



Algemene buur

```
h1~p
{
    color:red;
}
```

Met een algemene buur selector selecteer je elementen die komen na een ander element. In dit geval selecteer je alle p-element die na een h1-element komen.

```
<h1>
  Selectoren in <em>CSS</em>
</h1>
<p id="introduction" class="red">
  <strong>Een selector selecteert <em>HTML</em>-elementen</strong>. Op die manier kun
  je <em>zelf</em> bepalen welke elementen een specifieke opmaak moeten krijgen. Er
  zijn heel veel verschillende soorten selectoren.
</p>
<p>
  In dit hoofdstuk leer je maar liefst <em class="red">acht</em> verschillende soorten
  selectoren!
</p>
```

In de browser ziet dit er zo uit:



Je zou dit ook kunnen oplossen door beide paragrafen een klasse te geven. Door dit met een kind selector op te lossen hoef je minder code te schrijven en is het beter te onderhouden.

Je kunt ook hier combineren met andere type selectoren.

```
h1~.red
{
    color:red;
}
```

Hiermee selecteer je alle elementen met de klasse red die na een h1-element komen.



Naaste buur

```
h1+p
{
    color:red;
}
```

Deze naaste buur selector lijkt ontzettend veel op een algemene buur selector. Bij een naaste buur selector selecteer je echter alleen de eerstvolgende die er direct na komt. In dit voorbeeld selecteer je dus alleen het eerste p-element direct na een h1-element. Er mag dus geen ander element tussen staan!

```
<h1>
  Selectoren in <em>CSS</em>
</h1>
<p id="introduction" class="red">
  <strong>Een selector selecteert <em>HTML</em>-elementen</strong>. Op die manier kun
  je <em>zelf</em> bepalen welke elementen een specifieke opmaak moeten krijgen. Er
  zijn heel veel verschillende soorten selectoren.
</p>
<p>
  In dit hoofdstuk leer je maar liefst <em class="red">acht</em> verschillende soorten
  selectoren!
</p>
```

In de browser ziet dit er zo uit:



Je zou dit ook kunnen oplossen door de id-selector te gebruiken. Stel dat je meerdere webpagina's hebt waarbij je altijd de eerste paragraaf na een koptekst wilt kleuren, kun je dit beter doen met een naaste buur selector. Anders moet je namelijk in ieder HTML-bestand id-attributen gaan toevoegen.

Je kunt ook hier combineren met andere type selectoren.

```
h1+.red
{
    color:red;
}
```

Hiermee selecteer je alle elementen met de klasse red die direct na een h1-element komen.



Je kunt nu **Oefening 3.3** maken.

Selectoren combineren

Je kunt selectoren ook met elkaar combineren. Dit is handig als je bijvoorbeeld onderstaande CSS-code hebt geschreven.

```
h2
{
    color:red;
}
p
{
    color:red;
}
#introduction
{
    color:red;
    font-weight: bold;
}
```

Je ziet dat het h2-element, het p-element en het element met een id genaamd introduction, allemaal rood gemaakt moeten worden. Je kunt deze selectoren met elkaar combineren door een komma te gebruiken. Dat komt er dan zo uit te zien:

```
h2, p
{
    color:red;
}
#introduction
{
    color:red;
    font-weight: bold;
}
```

Dat scheelt al wat regels in de CSS-code. Je kunt er zelfs voor kiezen om ook de id-selector nog te combineren:

```
h2, p, #introduction
{
    color:red;
}
#introduction
{
    font-weight: bold;
}
```

Omdat de id-selector nog een font-weight eigenschap heeft die de andere selectoren niet hebben, moet je deze natuurlijk wel laten staan.



Je kunt nu **Oefening 3.4** maken.



Je ziet dat er veel verschillende manieren zijn om in CSS je opmaak te regelen. Als je je CSS-code wilt gaan combineren dien je jezelf af te vragen of dat bijdraagt aan de overzichtelijkheid en/of onderhoudbaarheid en/of compactheid van de code.

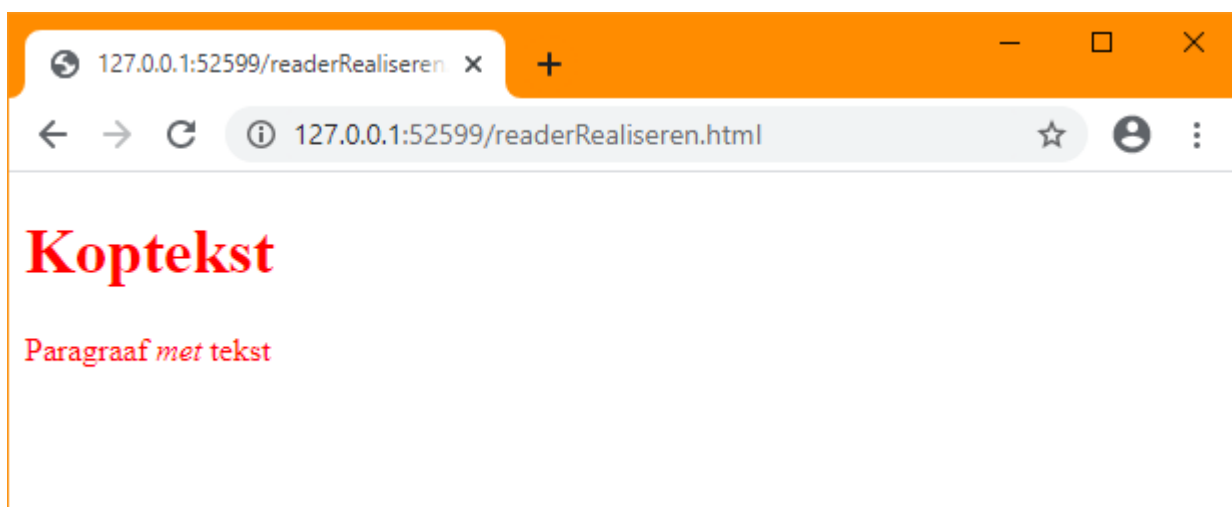
Overerven

Als je bijvoorbeeld een color eigenschap voor het body-element opgeeft, dan zullen ze op de meeste onderliggende (geneste) elementen ook van toepassing zijn. Dit komt doordat de waarde van de eigenschap door de geneste elementen geërfd wordt. Daardoor hoeft je deze eigenschappen niet voor ieder element opnieuw op te geven.

LET OP: Dit geldt niet voor alle eigenschappen! Als je bijvoorbeeld een background-color opgeeft wordt deze niet geërfd door de geneste elementen. Je kunt dat in de browser controleren. We gaan in het voorbeeld uit van onderstaande code:

```
<body>
  <h1>
    Koptekst
  </h1>
  <p>
    Paragraaf <em> met </em> tekst
  </p>
</body>
```

Als je een color instelt op het body-element is dit het resultaat:



Als je naar de Developer Tools gaat (F12) kun je de elementen met de element inspector selecteren. Doet dit wel vanuit de Verkenner en niet vanuit de Live Preview van de code editor!



readerRealiseren.html

File | C:/Users/Ralph%20Gijbsbrechts/Desktop/readerRealiseren.html

Elements

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body> == $0
    <h1>
      Koptekst
    </h1>
    <p>...</p>
  </body>
</html>
```

html body h1

Styles

Filter :hov .cls +

element.style { }

body { color: red; } stylesheet.css:2

body { display: block; margin: 8px; } user agent stylesheet

margin: 8px border: 1px solid black padding: 8px 271 x 76.44

Als je in de HTML-code die je ziet een element selecteert kun je onder styles zien welke opmaak dat element heeft. Je ziet dat op body de color:red van toepassing is. Als je nu het h1-element selecteert zie je staan dat de color:red geërf wordt van het body-element.

readerRealiseren.html

File | C:/Users/Ralph%20Gijbsbrechts/Desktop/readerRealiseren.html

Elements

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body>
    <h1>
      Koptekst
    </h1> == $0
    <p>...</p>
  </body>
</html>
```

html body h1

Styles

Filter :hov .cls +

element.style { }

h1 { display: block; font-size: 2em; margin-block-start: 0.67em; margin-block-end: 0.67em; margin-inline-start: 0px; margin-inline-end: 0px; font-weight: bold; } user agent stylesheet

Inherited from body

body { color: red; } stylesheet.css:2

margin: 21.440px border: 1px solid black padding: 8px 271 x 37

color: rgb(255, 0, 0) display: block font-size: 32px font-weight: 700

Als je de body een background-color zou geven, zou deze niet worden geërfd door de koptekst en paragraaf. Niet iedere eigenschap erft automatisch over.



Kleuren begrijpen

Je hebt al gezien dat je met behulp van de CSS eigenschap **color** de tekstkleur kunt aanpassen. Dat is nu steeds gebeurd door een kleurnaam op te geven zoals red. Nu bestaan er veel verschillende kleurnamen (op moment van schrijven 140 stuks), maar niet iedere kleur heeft dus zo'n mooie naam gekregen. Als je een hele specifieke kleur wilt gebruiken kun je ook gebruik maken van RGB(A) of hexadecimale codes. Dan zijn er maar liefst meer dan 16 miljoen kleuren mogelijk!

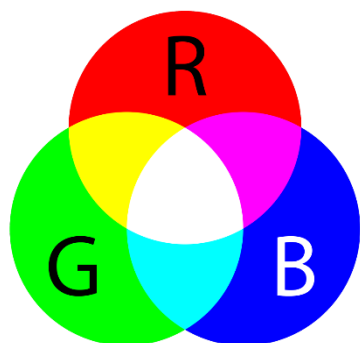
Kleurnamen

Hieronder een overzicht van de kleurnamen die je als CSS eigenschapswaarde kunt opgeven:

	black		bisque		forestgreen		slategrey
	dimgray		darkorange		limegreen		lightsteelblue
	dimgrey		burlywood		darkgreen		cornflowerblue
	gray		antiquewhite		green		royalblue
	grey		tan		lime		ghostwhite
	darkgray		navajowhite		seagreen		lavender
	darkgrey		blanchedalmond		mediumseagreen		midnightblue
	silver		papayawhip		springgreen		navy
	lightgray		moccasin		mintcream		darkblue
	lightgrey		orange		mediumspringgreen		mediumblue
	gainsboro		wheat		mediumaquamarine		blue
	whitesmoke		oldlace		aquamarine		slateblue
	white		floralwhite		turquoise		darkslateblue
	snow		darkgoldenrod		lightseagreen		mediumslateblue
	rosybrown		goldenrod		mediumturquoise		mediumpurple
	lightcoral		cornsilk		azure		rebeccapurple
	indianred		gold		lightcyan		blueviolet
	brown		lemonchiffon		paleturquoise		indigo
	firebrick		khaki		darkslategray		darkorchid
	maroon		palegoldenrod		darkslategrey		darkviolet
	darkred		darkkhaki		teal		mediumorchid
	red		ivory		darkcyan		thistle
	mistyrose		beige		aqua		plum
	salmon		lightyellow		cyan		violet
	tomato		lightgoldenrodyellow		darkturquoise		purple
	darksalmon		olive		cadetblue		darkmagenta
	coral		yellow		powderblue		fuchsia
	orangered		olivedrab		lightblue		magenta
	lightsalmon		yellowgreen		deepskyblue		orchid
	sienna		darkolivegreen		skyblue		mediumvioletred
	seashell		greenyellow		lightskyblue		deeppink
	chocolate		chartreuse		steelblue		hotpink
	saddlebrown		lawngreen		aliceblue		lavenderblush
	sandybrown		honeydew		dodgerblue		palevioletred
	peachpuff		darkseagreen		lightslategray		crimson
	peru		palegreen		lightslategrey		pink
	linen		lightgreen		slategray		lightpink



RGB-code



Als je dus nog specifiek een bepaalde kleur wilt opgeven kun je gebruik maken van een RGB-code. RGB staat voor de drie kleuren waaruit een pixel is opgebouwd. **R**ed, **G**reen en **B**lue.

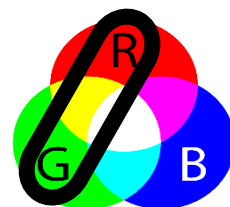
Met deze drie kleuren kun je dus iedere kleur maken die je maar wilt. Voor ieder van deze drie kleuren kun je de hoeveelheid van die kleur aangeven door een getal op te geven tussen de 0 en de 255 (dat zijn dus 256 mogelijkheden). Je kunt dus met RGB-codes in totaal $(256 \times 256 \times 256 =)$ 16.777.216 kleuren maken!

Als je van alle drie de kleuren de waarde op 255 zet, dan krijg je de kleur wit.

Als je van alle drie de kleuren de waarde op 0 zet, dan krijg je de kleur zwart.

Als je van alle drie de kleuren de waarde hetzelfde hebt staan, dan krijg je de kleur grijs. Hoe lager de getallen hoe donkerder de kleur grijs.

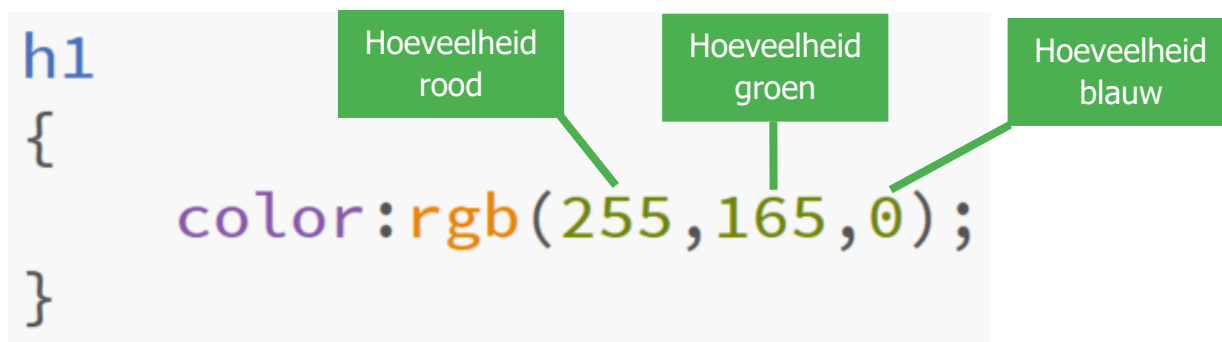
Als je de waarde van rood op 255 zet, de waarde van groen ook op 255 zet en de waarde van blauw op 0 zet, dan meng je dus eigenlijk de kleuren rood en groen. Als je dit met verf zou doen krijg je bruin, maar omdat je licht aan het mengen bent vallen de kleuren ook lichter uit. Als je een lichte versie bruin hebt, dan krijg je dus geel!



Hieronder een overzicht van een aantal kleuren opgebouwd met RGB-codes.

Kleurnaam	Hoeveelheid rood	Hoeveelheid groen	Hoeveelheid blauw
Red	255	0	0
Green	0	255	0
Blue	0	0	255
Yellow	255	255	0
Cyan	0	255	255
Magenta	255	0	255
Black	0	0	0
Gray	128	128	128
White	255	255	255
Pink	255	192	203
Orange	255	165	0
Purple	128	0	128

In CSS kun je de RGB-kleuren op de volgende manier opgeven:





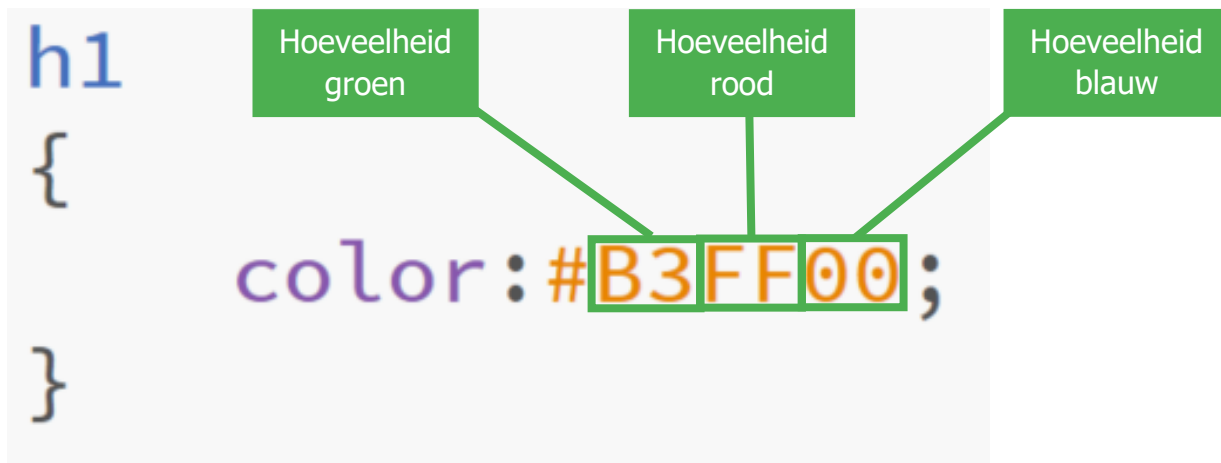
Hexadecimale-code

Naast RGB-code kun je ook gebruik maken van hexadecimale codes. Ook hiermee kun je 16.777.216 verschillende kleuren opgeven. De werking ervan is bijna hetzelfde als bij een RGB-code. Voor iedere kleur waaruit een pixel is opgebouwd (Red, Green, Blue) geef je een waarde op tussen de 0 en de 255. Echter doe je dat niet met een getal van 0 tot 255 (decimaal stelsel), maar met een waarde van 00 tot FF (hexadecimaal stelsel).

Als je gaat tellen met een het hexadecimale stelsel (zestientallig stelsel) doe je dat op de volgende manier:

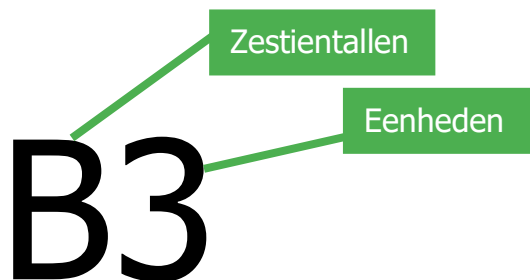
10-tallig stelsel	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16-talligstelsel	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

In CSS kun je de hexadecimale codes op de volgende manier opgeven:



Omrekenen van hexadecimale code naar RGB-code

We nemen als voorbeeld het hexadecimale getal B3.



Omdat de B hier staat voor een zestiental moet je deze waarde keer 16 doen. B staat voor 11 dus $11 \times 16 = 176$. Het hexadecimale getal heeft ook 3 eenheden. Die moet je daar nog bij optellen $176 + 3 = 179$. Het getal B3 uit het hexadecimale stelsel heeft dus de waarde **179** in het decimale stelsel.



Omrekenen van RGB-code naar hexadecimale-code

We nemen als voorbeeld het getal 179.

179

Omdat je wilt weten hoeveel zestientallen in dit getal zitten deel je dit getal door 16. $179 : 16 = 11,1875$. Dat zijn dus 11 zestientallen (je moet het altijd naar beneden afronden!). 11 is in het hexadecimale stelsel een B. $11 \times 16 = 176$. Je houdt dus nog 3 eenheden over. Zo krijg je **B3** als antwoord.

Omrekenen met de rekenmachine

Nu hoef je dit niet helemaal uit je hoofd te kunnen, maar het is wel handig om deze codes snel te kunnen lezen zodat je weet welke kleur het ongeveer is. Nu helpen de meeste editors je daar ook al bij door de kleur ook te laten zien.

Op de rekenmachine van Windows kun je het ook heel makkelijk omzetten.

The image shows two screenshots of the Windows Calculator application. The left screenshot shows the 'Rekenmachine' window with the 'Programmeren' mode selected in the sidebar. The right screenshot shows the 'Programmeren' mode with the 'HEX' tab selected, displaying the value 'FF'. A green box highlights the conversion table for the value 255:

HEX	FF
DEC	255
OCT	377
BIN	1111 1111



Je kunt nu **Oefening 3.5** maken.



Eigenschappen

Tekstkleur

Met de eigenschap **color** pas je de tekstkleur aan.

```
h1
{
    color:#B3FF00;
    /*
    of op onderstaande manieren:
    color:greenyellow;
    color:rgb(179,255,0);
    */
}
```

Je kunt hierbij de verschillende manieren gebruiken om kleuren op te geven. Waaronder kleurnamen, RGB-code of hexadecimale code, de verschillende manieren die in deze reader staan benoemd.

Achtergrondkleur

Met de eigenschap **background-color** pas je de achtergrondkleur aan.

```
h1
{
    background-color:#B3FF00;
    /*
    of op onderstaande manieren:
    background-color:greenyellow;
    background-color:rgb(179,255,0);
    */
}
```

Ook hier kun je weer de verschillende manieren gebruiken om kleuren op te geven.



Tekstgrootte

Met de eigenschap **font-size** pas je de tekstgrootte aan.

```
h1
{
    font-size:48px;
    /*
    of op onderstaande manieren:
    font-size:1.5em;
    font-size:150%;
    */
}
```

De lettergrootte in pixels instellen is de beste manier om ervoor te zorgen dat de tekst in de beoogde grootte verschijnt. Pixels zijn gerelateerd aan de resolutie van het scherm, dus dezelfde tekst lijkt groter wanneer een scherm een resolutie van 800x600 heeft dan bij een resolutie van 1280x800.

De standaard tekstgrootte in een webbrowser is 16 pixels. Met behulp van percentages hiervan kun je een schaal maken waarbij de standaard tekstgrootte 12 pixels is, en de grootte van koppen hieraan gerelateerd zijn. Gebruikers kunnen de standaard tekstgrootte in hun webbrowser wijzigen. Als ze dat gedaan hebben, dan worden de lettertypes in dezelfde verhouding getoond als de ontwerper bedoeld heeft, maar groter of kleiner.

Met ems kun je de tekstgrootte wijzigen ten opzichte van de tekstgrootte van het bovenliggende element. Omdat de standaard tekstgrootte in webbrowsers 16 pixels is, kun je van soortgelijke regels gebruikmaken als bij de percentages. Omdat gebruikers de standaard tekstgrootte in hun browser kunnen wijzigen, kan het gebeuren dat alle lettertypes er groter of kleiner uitzien dan de bedoeling was

Vetgedrukt

Met de eigenschap **font-weight** pas je de dikte van de tekst aan. Als eigenschapswaarde kun je onder andere een getal op geven van 100 tot en met 900. 100 is daarbij het meest dingedrukt en 900 het meest dikgedrukt.

```
#thin
{
    font-weight:100;
}
#bold
{
    font-weight:900;
}
```




Schuingedrukt

Met de eigenschap **font-style** kun je een tekst schuingedrukt weergeven. Daarbij gebruik je als eigenschapswaarde **italic**.

```
h1
{
    font-style: italic;
}
```

Onderstrepen en doorhalen

Met de eigenschap **text-decoration** kun je een tekst onderstrepen. Je maakt gebruik van de eigenschapswaarde **underline** om de tekst te onderstrepen en van **line-through** om de tekst door te halen.

```
#underline
{
    text-decoration: underline;
}
#lineThrough
{
    text-decoration: line-through;
}
```

Uitlijnen

Met de eigenschap **text-align** kun je een tekst uitlijnen. Je maakt gebruik van de eigenschapswaarden **left**, **right**, **center**, **justify** om de manier van uitlijnen op te geven.

```
h1
{
    text-align: left;
    /*
    of:
    text-align: center;
    text-align: right;
    text-align: justify;
    */
}
```

Links uitgelijnd.

In het midden uitgelijnd.

Rechts uitgelijnd.

Deze inhoud is uitgevuld. Dit betekent dat elke regel van een alinea, behalve de laatste, de volledige breedte van het kader inneemt.



Lettertype

Met de eigenschap **font-family** pas je het lettertype aan. Deze moet dan wel op het systeem van de gebruiker geïnstalleerd staan. Je kunt op een Windows besturingssysteem de geïnstalleerde lettertypen bekijken als je in de zoekbalk zoekt op lettertypen of fonts bij de Engelse versie van Windows. De eigenschapswaarde is gelijk aan de naam van het lettertype. Als de naam van het lettertype spaties bevat, moet het tussen aanhalingstekens. Als de naam dit niet heeft mag het wel, maar is het niet verplicht. We raden je daarom aan om altijd aanhalingstekens gebruiken, zodat je het consequent kunt toepassen.

```
h1
{
    font-family: "Tahoma";
}
```

Het kan zijn dat jouw gekozen lettertype niet geïnstalleerd staat op het systeem van de gebruiker. Daarom is het handig om in te stellen welke lettertype(s) er gebruikt kunnen worden als er een lettertype niet geïnstalleerd is. Dat doe je door meerdere eigenschapswaarden op te geven die je scheidt met een komma.

```
h1
{
    font-family: "Tahoma","Arial","Times New Roman";
}
```

Als Tahoma niet geïnstalleerd is, wordt er Arial gebruikt. Als Arial niet geïnstalleerd is wordt er Times New Roman gebruikt.

Lettertype van internet

Als je ervoor wilt zorgen dat de gebruiker altijd jouw gekozen lettertype te zien krijgt kun je ook gebruik maken van lettertypes die je op de server installeert. Een veel gebruikte website om lettertypes te downloaden is fonts.google.com. Download het lettertype en plaats het lettertypebestand, vaak een woff-, ttf- of otf-bestand, in de map fonts.



Plaats vervolgens onder de module header, maar boven de selectoren onderstaande code:

```
@font-face
{
    font-family: "Naam";
    src: url("../fonts/Roboto-Regular.ttf");
}

h1
{
    font-family: "Naam";
}
```

Je kunt deze naam zelf bedenken

Daarna kun je het lettertype gebruiken.

Dit is de locatie van het gedownloade lettertype.

Je ziet bij het opgeven van de locatie (url) dat er gestart wordt met twee puntjes. Dat komt omdat je vanaf het CSS-bestand eerst terug moet gaan naar de hoofdmap, voordat je naar de map fonts kunt. Twee puntjes geven aan dat je een map terug gaat.