

Basis statische website

Realiseren

hoofdstuk

2

Teksten schrijven





Algemene informatie

Onderwerp	Tekst
Leerdoel(en)	<ol style="list-style-type: none">1. De student kan waardevolle commentaarregels en moduleheaders toepassen binnen HTML-code.2. De student kan minimaal 2 redenen benoemen waarom het schrijven van commentaar belangrijk is.3. De student kan minimaal 2 redenen benoemen waarom het toevoegen van een moduleheaders aan code-bestanden belangrijk is.4. De student kan in eigen woorden het verschil benoemen tussen structurele mark-up en semantische mark-up.5. De student kan in eigen woorden het verschil benoemen tussen blok-elementen en inline-elementen.6. De student kan blok-elementen en inline-elementen coderen zoals beschreven staat in de best practices.7. De student kan minimaal 2 redenen benoemen waarom het belangrijk is om code netjes te structureren.8. De student kan kopteksten h1 tot h6, paragrafen, enters, horizontale lijnen, tekst in superscript, tekst in subscript coderen.9. De student kan het em-element en het strong-element coderen en kan in eigen woorden benoemen wat het nu daar van is.10. De student kan HTML-entiteiten (escape-tekens) coderen.
Vereiste voorkennis	De student moet de kennis van het vorige hoofdstuk bezitten.
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	4
Inhoud	4
Commentaar	4
Module header	5
Structurele- en semantische mark-up.....	6
Structurele mark-up	6
Semantische mark-up.....	6
Blok- en inline-elementen.....	6
Blok-elementen	6
Inline-elementen	7
Gebruik van de element inspector.....	8
Koppen	9
Paragrafen	9
HTML-entiteiten.....	11
Superscript en subscript.....	12
Belangrijk en nadruk.....	12



Introductie

In dit hoofdstuk ga je aan de slag met het maken van je eerste webpagina's. Je gaat deze webpagina's voorzien van onder andere hoofdkoppen, subkoppen en paragrafen. Daarnaast leer je allerlei belangrijke begrippen, het schrijven van commentaar in code en hoe je HTML-code het beste kunt structureren zodat je overzicht houdt in de code.

Inhoud

Commentaar

Commentaar schrijven in code betekent dat je de code gaat voorzien van informatie voor jezelf of een andere programmeur om gemakkelijker de weg te vinden in de geschreven code. De teksten die je in commentaar zet zijn alleen in de code zichtbaar en worden niet weergegeven op de website zelf. Een bezoeker kan eventueel wel door middel van de Developer Tools (F12) de broncode bekijken en dus ook het geschreven commentaar bekijken.

Met de code `<!--` geef je aan dat je commentaar gaat schrijven. Met `-->` geef je aan dat je het commentaar wilt afsluiten.

Commentaar schrijven is zeker in het begin best lastig. Want wat zijn nu goede regels commentaar? Onderstaande 2 punten kun je in gedachten houden bij het schrijven van goed commentaar.

1. Maak gebruik van functionele uitleg zoals "Inleidende tekst". Je schrijft commentaar voor jezelf of andere codeurs / programmeurs. Dus uitleggen wat bepaalde HTML-elementen doen is niet erg waardevol, je mag ervan uitgaan dat een codeur / programmeur dat weet. Het **beschrijven van de inhoud in een paar woorden** is daarentegen wel waardevol.

Voorbeelden:

Goed commentaar (functioneel)	Slecht commentaar
Inleidende tekst	Koptekst en paragraaf
Bedrijfslogo	Afbeelding

2. Als je naar jou idee creatieve, bijzondere of lastige code hebt geschreven, of code waarvan je ervan uitgaat dat een codeur of programmeur dat niet in één keer zou begrijpen dan kun je wel gebruik maken van technisch commentaar. Het commentaar legt dan je creatieve, bijzondere of lastige code uit.

Voorbeelden:

Goed commentaar (technisch)	Slecht commentaar
<code>&lt;</code> betekent <code><</code>	HTML-entiteiten
Lijst in lijst geplaatst wegens positionering submenu	Lijst in een lijst



De hoeveelheid commentaar die je schrijft is sterk afhankelijk van de code die je schrijft. Je hoeft niet op iedere coderegel commentaar te schrijven, maar code waar helemaal geen commentaar in staat is sowieso niet acceptabel. Commentaar moet er wel voor zorgen dat jij, maar ook andere codeurs en programmeurs de code goed kunnen lezen. De ene keer heb je daar wat meer commentaar voor nodig dan de andere keer.

```
<!doctype html>
<html>
  <head>
    <title>
      Commentaar in HTML
    </title>
    <meta charset = "UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <!-- Artikel commentaar in HTML -->
    <h1>
      Het schrijven van commentaar in HTML
    </h1>
    <!-- &lt; is een kleiner dan teken (<) -->
    <p>
      Met de code &lt;!-- geef je aan dat je commentaar gaat schrijven. Met
      --> geef je aan dat je het commentaar wilt afsluiten.
    </p>
  </body>
</html>
```

Module header

Je hebt zojuist geleerd hoe je commentaar kunt schrijven. Direct na de doctype plaats je ook altijd een stuk commentaar met daarin informatie over de auteur, aanmaakdatum, een omschrijving van de inhoud en een versie(geschiedenis). Deze regels commentaar bovenaan een codebestand noemen we een module header.

```
<!doctype html>

<!--
Auteur:           <auteursnaam>
Aanmaakdatum:    <dd-mm-jjjj>

<omschrijving inhoud>

-->

<html>
  <head>
    <title>
    </title>
    <meta charset = "UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
  </body>
</html>
```



Het plaatsen van een module header doe je in **alle** soorten codebestanden die jij zelf maakt.

Onderdeel module header	Waarde
Auteur	Hier vul je de voornaam en achternaam in van diegene die het codebestand gemaakt heeft.
Aanmaakdatum	Hier vul je de datum in waarop het bestand is aangemaakt.
Omschrijving van de inhoud	Hier vat je in hooguit een paar zinnen de inhoud samen van het codebestand.



Je kunt nu **Oefening 2.1** maken.

Structurele- en semantische mark-up

Je kunt elementen onderverdelen in elementen die structuur beschrijven en elementen die semantiek beschrijven.

Structurele mark-up

Onder structurele mark-up worden elementen bedoeld die structuur beschrijven. Denk daarbij bijvoorbeeld aan hoofdkoppen, subkoppen, paragrafen, afbeeldingen en links.

Semantische mark-up

Naast structurele mark-up zijn er ook elementen die horen tot de semantische mark-up. Onder semantische mark-up worden elementen bedoeld die extra informatie geven, bijvoorbeeld de plaats waar nadruk in de zin wordt gelegd, dat iets wat je hebt geschreven een citaat is (en wie het gezegd heeft) en de betekenis van afkortingen. Semantische mark-up zorgt ervoor dat zoekmachines als Google de webpagina beter begrijpen en kunnen daar vervolgens de zoekresultaten op afstemmen. Daarnaast kan semantische mark-up ook gebruikt worden om voorleesprogramma's te helpen met hoe bepaalde tekst uitgesproken moeten worden. Later in dit hoofdstuk worden dit soort elementen nog verder besproken.

Blok- en inline-elementen

Elementen kun je ook onderverdelen in blok- en inline-element.

Blokelementen

Blokelementen verschijnen altijd op een nieuwe regel. Dat betekent dat je dus zelf geen enter hoeft te geven. Kijk maar eens naar onderstaand voorbeeld.

```
<body>
  <h1>
    Voorbeeld blokelementen
  </h1>
  <p>
    Een h1-element en een p-element zijn voorbeelden van blokelementen.
  </p>
  <p>
    Daarom begint deze tekst gewoon op een nieuwe regel.
  </p>
</body>
```



In de browser zie je nu dat een paragraaf altijd op een nieuwe regel start.



Om je code goed te structureren is het erg verstandig om blokelementen op de volgende manier te structureren in je code.

```
<h1>
  Voorbeeld blokelementen
</h1>
<p>
  Een h1-element en een p-element zijn voorbeelden van blokelementen.
</p>
<p>
  Daarom begint deze tekst gewoon op een nieuwe regel.
</p>
```

De openingstag, de inhoud en de sluittag staan allemaal op aparte regels. De inhoud krijgt ook één tab ten opzichte van de bijbehorende tags. Daarmee laat je heel duidelijk zien dat de inhoud bij dat element hoort.

Inline-elementen

Naast blokelementen die dus altijd op een nieuwe regel starten, bestaan er ook elementen die op dezelfde regel staan. Dat worden de inline-elementen genoemd. Kijk maar eens naar onderstaand voorbeeld. De betekenis van het em-element wordt later dit hoofdstuk uitgelegd.


```
<body>
  <h1>
    Voorbeeld inline-elementen
  </h1>
  <p>
    Een <em>em-element</em> is een voorbeeld van een inline-element.
  </p>
</body>
```

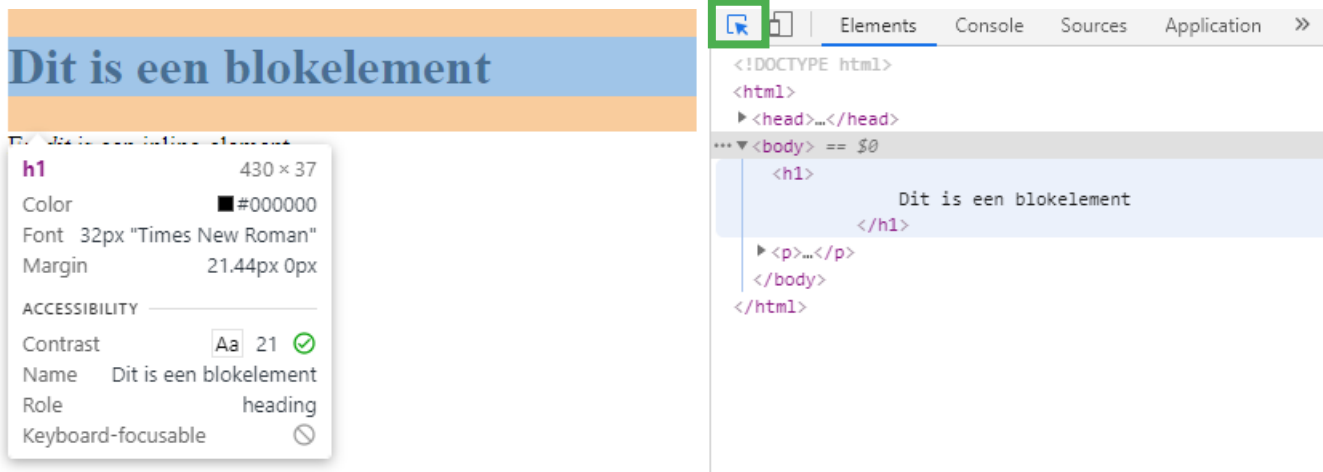


In de browser zie je nu dat een em-element altijd op dezelfde regel blijft staan.



Gebruik van de element inspector

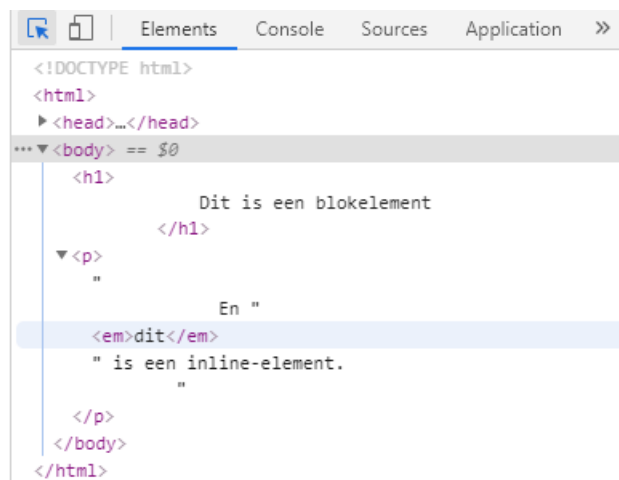
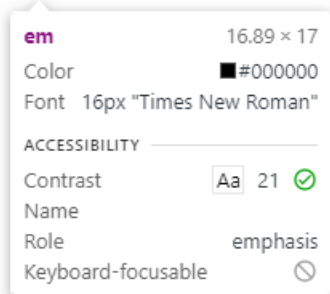
Als je gebruik maakt van de element-inspector van de developer tools (In Chrome te openen met F12) kun je ook heel mooi zien of het blok- of inline-elementen zijn. Om de element inspector te openen klik je op het  icoon. Als je daarna met je muis over een element beweegt kun je allerlei gegevens over dat element zien. Als het element helemaal de volledige breedte van de browser heeft, dan weet je dat het een blok-element is.



Als het element niet de volledige breedte heeft van de browser, dan weet je dat het een inline-element is.

Dit is een blokelement

En **dit** is een inline-element.



Tip: Je kunt ook elementen selecteren door in de code van de developer tool een element aan te klikken.



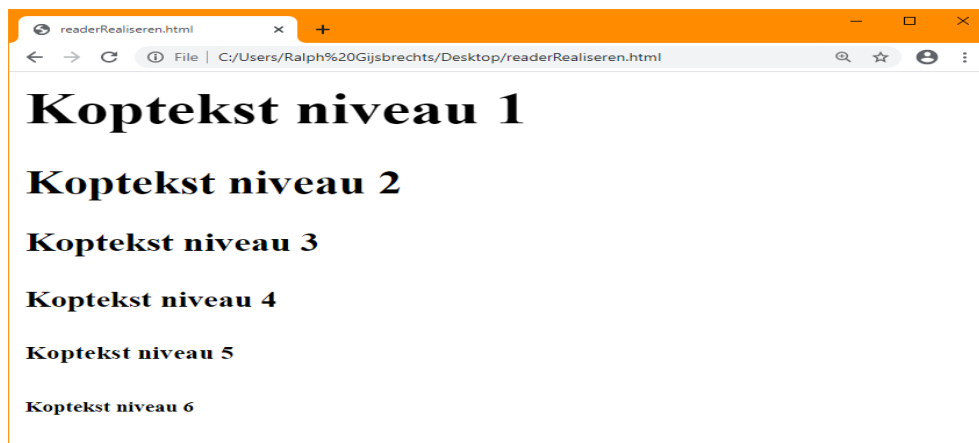
Koppen

In HTML zijn er 6 verschillende soorten elementen om een kop aan te kunnen maken. Alle koppen hebben verschillende niveaus.

Code:

```
<body>
  <h1>
    Koptekst niveau 1
  </h1>
  <h2>
    Koptekst niveau 2
  </h2>
  <h3>
    Koptekst niveau 3
  </h3>
  <h4>
    Koptekst niveau 4
  </h4>
  <h5>
    Koptekst niveau 5
  </h5>
  <h6>
    Koptekst niveau 6
  </h6>
</body>
```

Hieronder kun je zien hoe dit in de browser eruitziet. Maar let op: Je beschrijft hier een structuur, dat is wat anders dan teksten opmaken (zie hoofdstuk 1)! De browser vertaalt de HTML-code naar wat je in de browser ziet. Het kan dus best zijn dat een H1-element in Chrome er anders uit ziet dan in FireFox, Opera, Safari of een andere browser.



Zoals je kunt zien zijn alle elementen voor kopteksten ook blokelementen. Ze starten allemaal op een nieuwe regel.

Paragrafen

Door middel van het p-element maak je een paragraaf aan. Omdat er geen verschillende niveaus toe te kennen zijn aan paragrafen komt daar dus geen cijfer achter.



```
<body>
  <h1>
    Paragrafen maken in HTML
  </h1>
  <p>
    Paragraaf 1
  </p>
  <p>
    Paragraaf 2
  </p>
  <p>
    Paragraaf 3
  </p>
</body>
```

In de browser ziet dit er zo uit:



Wanneer een browser twee of meer spaties naast elkaar tegenkomt, wordt slechts één spatie getoond. Daarnaast dien je een enter altijd op te geven met een `br`-element. Dit is een leeg element dus hoef je niet te sluiten. Zonder een `br`-element staat alles nog gewoon op dezelfde regel.

```
<body>
  <h1>
    Spaties in HTML
  </h1>
  <p>
    Het maakt      niet uit hoeveel      spaties hier staan.
    De browser          toont er maar één!
  </p>
</body>
```



In de browser ziet dit er zo uit:



Je kunt nu **Oefening 2.2** maken.

HTML-entiteiten

Soms wil je graag bepaalde karakters plaatsen in bijvoorbeeld een koptekst of paragraaf die lastig te typen zijn zoals bijvoorbeeld een copyright-teken (©). Het kan ook zo zijn dat een teken al een andere functie heeft in HTML en je ze daarom niet zou kunnen gebruiken, zoals de `<h1>`. Als je dit intypt zal de browser dit direct vertalen naar een h1-element, terwijl je eigenlijk de letterlijke tekst op een pagina wilt laten zien. In dit soort gevallen kun je gebruik maken van zogeheten HTML-entiteiten.

Je kunt HTML-entiteiten op 2 verschillende manieren toepassen. Je kunt namen of codes van HTML-entiteiten gebruiken. Hieronder een overzicht met een aantal veelgebruikte HTML-entiteiten.

Karakter	Omschrijving	Naam HTML-entiteit	Code HTML-entiteit	Reden om HTML-entiteit te gebruiken
<	Kleiner dan	<	<	Heeft in HTML al een functie.
>	Groter dan	>	>	
©	Copyright	©	©	Is moeilijk of niet te vinden op een toetsenbord.
®	Geregistreerd handelsmerk	®	®	
™	Handelsmerk	™	ℎ	
×	Vermenigvuldigen	×	×	
÷	Delen	÷	÷	

```
<body>
  <h1>
    HTML-entiteiten
  </h1>
  <p>
    Met entiteitnamen: Je opent een paragraaf met de openingstag &lt;p>.
  </p>
  <p>
    Met entiteitcodes: Je opent een paragraaf met de openingstag &#60;p>.
  </p>
</body>
```



In de browser ziet dit er zo uit:



Superscript en subscript

Het sup-element is een inline-element dat wordt gebruikt voor tekens die in superscript gezet moeten worden, zoals bij 2^e of 2².

Het sub-element is een inline-element dat wordt gebruikt voor tekens die in subscript moeten worden gezet, zoals bij H₂O.

```
<body>
  <h1>
    Superscript en subscript
  </h1>
  <p>
    Dit is de 1<sup>e</sup> paragraaf. De e achter de 1 is superscript.
  </p>
  <p>
    H<sub>2</sub>O staat voor water. De 2 achter de H is subscript.
  </p>
</body>
```

In de browser ziet dit er zo uit:



Belangrijk en nadruk

Tot dusver hebben we alleen structurele elementen gebruikt. Nu ga je aan de slag met semantische elementen. Ze veranderen dus niet de structuur, maar ze geven extra informatie aan bepaalde inhoud. Hierdoor kan bijvoorbeeld Google jouw webpagina beter begrijpen of kan text-to-speech software bepaalde inhoud (tekst / woorden) op een andere manier uitspreken.



Het strong-element is een inline-element dat wordt gebruikt om aan te geven dat de inhoud belangrijk is. De meeste browsers tonen de inhoud van een strong-element dikgedrukt. LET OP: Dat is een vertaling van de browser, dat kan bij een andere browser anders zijn. **Een strong-element is dan ook niet bedoeld om tekst dikgedrukt weer te geven.**

Het em-element is een inline element dat wordt gebruikt om nadruk te leggen op bepaalde inhoud. Daarmee kun je op een subtiele manier de betekenis van een zin veranderen.

Bijvoorbeeld:

Ik vind dat HTML leuk is.

Ik vind dat *HTML* leuk is.

Ik vind dat HTML *leuk* is.

Spreek bovenstaande zinnen maar eens uit en leg daarbij nadruk op het groen gekleurde woord. Je merkt dat de zin op een subtiele manier van betekenis veranderd. De meeste browsers tonen de inhoud van een em-element schuingedrukt. LET OP: Dat is een vertaling van de browser, dat kan bij een andere browser anders zijn. **Een em-element is dan ook niet bedoeld om tekst schuingedrukt weer te geven.**

```
<body>
  <h1>
    Belangrijk en nadruk
  </h1>
  <p>
    <strong>Pas op</strong>: een strong-element geeft aan dat de inhoud belangrijk is,
    maar mag je nooit gebruiken om tekst dikgedrukt te maken!
  </p>
  <p>
    <em>Ik</em> vind dat iedereen vanaf nu moet weten dat een em-element aangeeft dat de
    inhoud een nadruk krijgt en je het nooit mag gebruiken om tekst schuingedrukt te
    maken!
  </p>
</body>
```

In de browser ziet dit er zo uit:



Je kunt nu **Oefening 2.3** maken.