Html5 kent veel nieuwe opmaakelementen, zoals article, section, nav, header en meer. Daarover en over nieuwe input elementen, zoals email, url, date en search en een aantal nieuwe attributen, gaat dit artikel. Ook komen de video en audio tags even langs en laten we zien hoe mindere browsers kunnen worden ondersteund. Wat we niet gaan behandelen zijn de 'sexy' zaken, daar wordt al genoeg over geschreven.

Vernieuwingen maken html5 stuk simpeler

Opmaak en meer voor luie mensen

Het gebruik van de code is in html5 sterk vereenvoudigd. Html5 maakt ons allemaal het leven wat makkelijker en tegelijkertijd zorgt het voor meer mogelijkheden. In xhtml werden we allemaal verplicht om onze p's af te sluiten, alle tags in lowercase te schrijven en alle attributen tussen quotes te zetten. Dit omdat xhtml zich aan de xml standaarden houdt. Nu worden deze regels echter weer helemaal los gelaten. Waarom? Omdat browsers er toch al niet naar keken. In html5 kan je dus losser met je code omgaan, zoals:

```
<input
        type="date"
        placeholder="datum van vandaag"
        required="required" />

<input
        TYPE=date
        placeholder="datum van vandaag"
        required>
```

(voor attributen waar je meerdere woorden in plaatst MOET je wel quotes om je waarde zetten, anders is het natuurlijk niet meer valide)
In mijn optiek zouden we echter de oude regels moeten aanhouden. Dit maakt het lezen en overdragen van code een stuk makkelijker.

Maar hoe beginnen we nu met html5? Het enige wat je moet doen is je oude (xhtml1.0 strict) doctype te vervangen door de html5 doctype. Dus in plaats van:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

begin je je document nu met:

```
<!DOCTYPE HTML>
```

Veel simpeler, toch. Zo gaat het ook met de rest van je document, er zijn minder verplichte zaken. Er is meer gekeken naar wat er door browsers gedaan werd en niet naar wat er in de standaarden stond. Het xhtml1.0 doctype zorgde er namelijk alleen maar voor dat browsers niet in 'quirksmode' gingen maar in 'standardsmode' bleven. Het leidde er niet toe dat een onafgesloten <p> of een uppercase attribute voor fouten zorgden op de pagina. Het meest voorkomende verschil is dat Internet Explorer 6 met het correcte doctype het w3c box model aanhield, zodat je verschillen in breedte en dergelijke kon vermijden.
Andere zaken die versimpeld zijn, zijn bijvoorbeeld de html en de charset declaratie. Laten we eens gaan kijken naar een simpele html opzet;
in plaats van:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.
dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
 lang="en">
<head>
        <meta http-equiv="Content-Type" content="text/
html; charset=utf-8"/>
        <title>Eenhoorns</title>
</head>
<body>
</body>
</html>
```

Doe je in html5:

```
<!DOCTYPE HTML>
<html lang="nl">
<head>
        <meta charset="UTF-8">
        <title>Eenhoorns</title>
</head>
<body>
```

Wilfred Nas
is een freelance front end developer. Je kunt meer van hem lezen op http://wnas.nl en via twitter (@wnas).

Geen div's zonder betekenis meer, maar semantische elementen.

```
</body>
</html>
```

Ook simpeler is het bijvoorbeeld om een javascript bestand te includen in je pagina. Geen:

```
<script type="text/javascript" src="script.js"></script>
```

maar simpelweg:

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

Browsers doen namelijk toch niets met 'type="text/javascript' en waarom zouden we de extra bytes oversturen (mobiel is sterk in opmars en daar is bandbreedte en snelheid heel belangrijk!).

Opmaakelementen en semantiek

Voorheen deden we alle opmaak met tabellen. Gelukkig liggen die dagen ver achter ons. Hoewel velen nog dezelfde fout maken door met div's en span's te ingewikkelde en niets betekenende constructies te maken. Nu kunnen we gelukkig gebruik maken van de verschillende nieuwe elementen. Dus ipv dit:

```
<div id="header">
        <div>
                <h1>Eenhoorns</h1>
                <h2>Een site over eenhoorns</h2>
        </div>
</div>
```

kunnen we nu:

```
<header>
        <hgroup>
                <h1>Eenhoorns</h1>
                <h2>Een site over eenhoorns</h2>
        </hgroup>
</header>
```

Wat is het voordeel hiervan? De betekenisloze div's zijn vervangen door semantische elementen. Dit maakt het makkelijker om je werk over te dragen en te onderhouden. In html5 zijn er verschillende opmaak elementen bij gekomen, waaronder: article, section, nav, sidebar, footer en header. Deze kun je gebruiken om verschillende zaken op te maken, op zo'n manier dat je code weer een stukje beter leesbaar is.
Dus voor navigatie kies je een <nav> element, voor je footer een <footer> en voor gerelateerde informatie die niet essentieel voor je artikel is een <aside>.
Het article element is voor elementen die alleenstaand ook nog betekenis hebben, terwijl een <section> voor een gedeelte van een artikel kan staan. Maar ook voor een stuk met verschillende artikelen, zoals:

```
<section>
        <article>
            <h1>Eenhoorns</h1>
            <section><p>lorem</p><p>ipsum</p></section>
            <section><p>lorem</p><p>ipsum</p></section>
        </article>

        <article>
            <h1>Regenbogen</h1>
           <section><p>lorem</p><p>ipsum</p></section>
            <section><p>lorem</p><p>ipsum</p></section>
        </article>
</section>
```

Om je een idee te geven welke mogelijkheden er qua opmaak elementen zijn is hier een korte lijst met elementen: article, aside, audio, canvas, details, figcaption, figure, footer, header, hgroup, mark, meter, nav, output, progress, section, summary, time en video.
De nieuwe elementen hebben van zichzelf nog geen styling, ook niet in browsers die ze ondersteunen, dus je zal er zelf voor moeten zorgen dat de juiste elementen een {display:block} mee krijgen...

Form elementen

Vóór html5 waren we redelijk beperkt in onze input keuze, maar nu worden deze flink opgepoetst en uitgebreid. het meest simpele input element schrijf je nu zo:

```
<input>
```

Dit zorgt automatisch voor een input type="text" in alle browsers, zelfs in IE6. Je kunt natuurlijk veel verder gaan en bijvoorbeeld kiezen voor een input type email met een placeholder en een required attribuut.

```
<input
        type="email"
        placeholder="john@foo.com"
        required>
```

Dit zorgt voor een input element dat op bijvoorbeeld de iPhone zorgt voor een ander toetsenbord, zodat je makkelijker je email kunt invoeren. Ook laat het placeholder attribuut zien wat je verwacht als input en zorgt het required attribuut voor het feit dat je dit veld in moet vullen en dat het direct wordt gevalideerd. De validatie gaat in dit geval af op de input type(email) zodat er enkel een valide email-adres kan worden ingevoerd. (In theorie althans, in de praktijk gaan browsers heel verschillend met validatie om.)
Een ander attribuut dat je kunt gebruiken om de validatie in de toekomst makkelijker te maken is het pattern attribuut. Hierin zet je een reguliere expressie die de browser dan kan valideren om dus zonder javascript je form data client site te laten valideren. De ondersteuning hiervoor is nog niet geweldig, maar je kunt het gebruiken om je validatie met javascript makkelijker te maken. Je hoeft niet in javascript je validatieregels te zetten, maar in de html. Om een url als abc.com te valideren gebruik je, in een input type=url, het volgende pattern:

```
<input
        type="url"
        placeholder="abc.com"
        pattern="^([a-zA-Z0-9]([a-zA-Z0-9\-]{0,61}
        [a-zA-Z0-9])?\.)+[a-zA-Z]{2,6}$">
```

Op deze manier kun je een andere validatieregel in je html opnemen, zonder je javascript aan te passen. Je laat je validatie script namelijk kijken naar

het pattern attribuut om zijn reguliere expressie te zoeken. Dit heeft legio voordelen, je zet je validatie regel waar hij hoort: in de content. Ook hoef je niet voor ieder wissewasje je script aan te passen (de reguliere expressies in het pattern attribuut volgen dezelfde regels als bij javascript).
Deze nieuwe input types zijn er om te gebruiken: search, tel, url, email, datetime, date, month, week, time, datetime-local, number, range, color.
Deze hebben de volgende nieuwe attributen: autocomplete, novalidate, autofocus, form, form overrides, formaction, formmethod, formnovalidate, formtarger, height and with, list, min, max and step, multiple, pattern (regexp), placeholder en required.

Video en audio

Iedereen die ooit video of audio heeft moeten toevoegen aan een site weet wat voor een crime dat is. Je moest het tot nu toe met Flash doen, of erger nog met Silverlight. Html5 maakt het allemaal wat makkelijker met de video en audio tag. Om een simpele video toe te voegen aan je site moet je het volgende doen:

```
<video
        width="320"
        height="240"
        controls>
        <source
         type='video/mp4; codecs="avc1.42E01E,
 mp4a.40.2"'                              src="movie.
mp4">
        </source>
</video>
```

En we zijn klaar, in ieder geval voor Safari. Er is namelijk een klein probleempje met video op het web, de codecs. Safari ondersteunt mp4 alsook Internet Explorer 9, maar Firefox en Opera gaan voor het open source ogg-theora en Chrome gaat voor het eigen (ook open source) webm. Verder zal het inmiddels geen verrassing zijn dat versies van Internet Explorer 8 en ouder niets van dit alles ondersteunen, ook oudere versies van Safari, Firefox, Opera en Chrome ondersteunen video niet...
Het toevoegen van meerdere sources brengt ons een stuk verder om moderne browsers te ondersteunen.

```
<video
        width="320"
        height="240"
        controls >
        <source
        , type='video/mp4; codecs="avc1.42E01E,
 mp4a.40.2"'
                src="movie.mp4">
        </source>
        <source
                type='video/ogg; codecs="theora,
 vorbis"'
                src="movie.ogv">
        </source>
        <source
                type='video/webm; codecs="vp8,
 vorbis"'
                src="movie.webm">
        </source>
</video>
```

Voor Internet Explorer en oudere browsers gebruiken we javascript om het mp4 bestand in een Flash filmpje te zetten.

Ondersteuning

Jammer genoeg is het niet allemaal rozengeur en maneschijn. De meeste moderne browsers (Safari, Chrome en Firefox) ondersteunen veel van deze nieuwe elementen, attributen en mogelijkheden. Helaas loopt Internet Explorer nog ver achter. Waar versie 9 al wat opmaak elementen (article, section enz) begrijpt, snapt hij nog niet veel van de rest. En wat oudere versies betreft, deze ondersteunen nog helemaal niets van dit alles.
Wel kun je met een simpele truc Internet Explorer aan boord brengen. Misschien heb je ooit te maken gehad met het gebrek aan ondersteuning van het <abbr> element in Internet Explorer 6. Dit is op te lossen door in de head van je document met behulp van javascript het element toe te voegen.

```
<!--[if lte IE6]>
<script>document.createElement("abbr");</script>
<![endif]>
```

En ineens kan er wel styling op worden toegepast. Deze truc kun je met behulp van Remy Sharp's html5shim ook toepassen voor meer elementen. Dus in je head zet je:

```
<!--[if lt IE 9]>
<script src="http://html5shim.googlecode.com/svn/trunk/
html5.js"></script>
<![endif]-->
```

Ook de nieuwe form elementen kun je gewoon gebruiken. Met javascript voeg je ondersteuning toe voor browsers die nog niet ver genoeg zijn. Wat je bijvoorbeeld kunt doen om te testen of iets wordt ondersteund, is het uitschrijven van een element met behulp van javascript en vervolgens te kijken of het wordt herkend:

```
// Bouw een input element om op html5 attributen te
 testen
var i = document.createElement('input');
// Kijk of het 'placeholder' attribuut ondersteund
 wordt.
if ( !('placeholder' in i) ) {
        // verzorg ondersteuning voor placeholder
}
```

Deze test doe je vervolgens voor alle attributen die je wilt ondersteunen. Wat het voordeel hiervan is, is dat we moderne browsers niet lastig vallen en alleen de oudere browsers helpen.

Conclusie

Html5 is klaar voor gebruik. Het maakt ons leven makkelijker, doordat het semantisch correcter is en dat er veel nieuwe mogelijkheden zijn. Met een beetje javascript kunnen we ook aan oudere browsers ondersteuning bieden. «

Internet Explorer kun je makkelijk aan boord brengen met javascript of html5shim.

Links

- http://diveintohtml5.org/
- http://html5doctor.com/
- http://www.wnas.nl/