



eBook

Einstieg in den Umstieg auf HTML5

Autor

Till Sanders

Homepage

<http://www.compufreak.info/>

Datum

28.08.10

Version

0.1 (unvollständig, Bearbeitung nicht abgeschlossen)

Lizenz



Veröffentlicht unter der Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported Lizenz

Ausgabe

Komplett-Anleitung

YouTube-Channel

<http://www.youtube.com/compufreak29>



Dieses eBook befindet sich noch in der Entwicklung und kann teilweise falsche Angaben oder Tippfehler enthalten. Updates über <http://www.compufreak.info/>

Inhalt

Einleitung	Vorwort	3
Teil 1	Grundlegende Änderungen	3
Teil 2	Ziele	5
Teil 3	Neue Elemente	6-10
Teil 4	Verschwundene Elemente	11
Teil 5	Geänderte Elemente	12
Teil 6	Neue Attribute	13-15
Teil 7	Verschwundene Attribute	16-17
Teil 8	Geänderte Attribute	18
Teil 9	Neue Steuerelemente	19
Teil 10	Web-Storage	20
Teil 11	Browser	21
Anhang	Links und Empfehlung	22

Einleitung

HTML5 ist die fünfte große HTML-Version. In Anbetracht der Tatsache, dass die letzte Version (4.01) 1999 erschien kann man schon einige Neuerungen erwarten. Diese Neuerungen sollen auf den folgenden Seiten kurz und anschaulich dargestellt werden. Da aber noch nicht alle Arbeiten an HTML5 abgeschlossen sind, können sich einige Details ändern. Zudem kann dieses Dokument genauso wenig wie jedes andere beanspruchen fehlerfrei zu sein.

Dieses Dokument soll lediglich einen Einstieg darstellen. Es zeigt nicht alle neuen Funktionen und erklärt auch nicht alles. Es soll nur ein Anhaltspunkt sein, für diejenigen, die später hoffentlich auch noch umfangreichere Werke lesen möchten und nur mal wissen wollen, was auf sie zu kommt.

Attribute und Elemente wurden für die bessere Übersicht hervorgehoben. Sie werden jeweils breiter dargestellt. Attribute sind blau markiert, Elemente orange.

Teil 1 – Grundlegende Änderungen

HTML5 besitzt einige wirklich wichtige und interessante Neuerungen, die nun kurz erläutert werden:

HTML5 basiert nicht mehr auf SGML

SGML hat mit HTML5 nun endlich ausgedient. Ersetzt wurde es durch HTML. Der Unterschied ist nicht groß. Unter Anderem bietet es aber den Vorteil, dass keine DTD-Definitionen mehr nötig sind. HTML5 wird abwärtskompatibler für seine Benutzer. Dadurch ergibt sich eine wichtige Änderung: Der Doctype wird endlich deutlich einfacher:

Quellcode (1.1)

```
<!doctype html>
```

Diese Doctype-Definition reicht völlig aus.

HTML5 kann in HTML und XML verwendet werden

Was viele Leute nicht verstanden haben ist eigentlich ganz einfach: HTML5 kann in HTML (was ja SGML ersetzt) und in XML verwendet werden. Das sorgt für weitere Abwärtskompatibilität. So müssen Seiten, die noch auf HTML 4.01 basieren genauso wenig umstellen wie die, die auf XHTML, sprich XML basieren. So gibt es also immer noch 2 mögliche Syntaxen. Programmierer können die

für sie günstigere Variante auswählen. Dateien, die auf HTML basieren, sind fast immer mit dem 'text/html'-Media-Type bedient. Auf XML basierte Dateien sind mit 'application/xhtml+xml ' oder 'application/xml' auszustatten. Die Grundgerüste sehen wie folgt aus:
HTML-Basiert:

Quellcode (1.2)

```
<!doctype html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Beispiel-Dokument</title>
  </head>
  <body>
    <p>Beispiel</p>
  </body>
</html>
```

XML-Basiert:

Quellcode (1.3)

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Beispiel-Dokument</title>
  </head>
  <body>
    <p>Beispiel</p>
  </body>
</html>
```

Zeichensätze

Für die HTML-basierte Variante von HTML gibt es nach wie vor drei Möglichkeiten den Zeichensatz zu bestimmen:

- Beim Transport, indem der HTTP 'Content-Type' entsprechend gesetzt wird.
- Mithilfe eines 'Unicode-Byte-Order-Mark' (BOM) Zeichen am Anfang der Datei.
- Mit einem meta-Element, welchem ein entsprechendes Attribut übergeben wurde:
<meta charset="UTF-8"> könnte verwendet werden, folgende Syntax ist möglich aber unnötig kompliziert: <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">.

Teil 2 – Ziele und wichtige Neuerungen

Haupt-Ziele für HTML5

- Neue Funktionen/Features, basierend auf HTML, CSS und Javascript
- Reduzieren der Notwendigkeit externer Plugins (wie Flash)
- Bessere Fehler-Behandlung
- Geräte-Unabhängigkeit
- Mehr Markup-Elemente sollen Skripte ersetzen
- Der Entwicklungsprozess sollte für die Öffentlichkeit einsehbar sein.

Wichtigste Neuerungen in HTML5

- Das Video- und Audio-Element welche endlich einen standardkonformen, einfachen Weg zur multimedialen Website bieten sollen.
- Bessere Methoden Daten offline zu speichern
- Das Canvas-Element
- Die Content-spezifischen Elemente (section, article, nav, etc.)
- Die Formular-Elemente, welche unnötige Programmierarbeit ersparen und bessere Benutzerfreundlichkeit herstellen sollen.
- Geolocation soll es ermöglichen den Standpunkt des Benutzers auf Anfrage zu ermitteln.

Teil 3 – Neue Elemente

Folgende Elemente sind neu in HTML5:

- **section** stellt einen Bereich in einem Dokument oder einer Applikation dar. Es kann gut mit **h1**, **h2**, **h3**, **h4**, **h5**, und **h6** verwendet werden um die Dokument-Struktur zu verdeutlichen.
- **article** stellt einen unabhängigen Teil eines Dokuments dar, wie zum Beispiel ein Blog-Eintrag oder ein Zeitungsartikel.
- **aside** stellt ein Teil des Dokuments dar, der nur ein wenig mit dem eigentlichen Inhalt zu tun hat.
- **hgroup** stellt den Kopfteil eines Abschnitts (**section**) dar. Es enthält demzufolge meistens Überschriften (**h1**, **h2**, ...).
- **header** repräsentiert den Kopfteil (header) eines Dokuments dar.
- **footer** wird als Fußteil eines Abschnitts (**section**) verwendet und kann Informationen über Autor, Erscheinungsdatum, Copyright oder Ähnliches enthalten.
- **nav** 'vereint' die Navigationselemente des Dokuments.
- **figure** kann verwendet werden um ein Element (wie eine Grafik, ein Video, etc.) mit einem Untertitel zu versehen:

```
<figure>
  <video src="beispiel.ogg"></video>
  <figcaption>Beispiel</figcaption>
</figure>
```

- **figcaption** wird für den Untertitel verwendet (siehe oben), es ist optional.
- **embed** wird für Plugins verwendet.
- **mark** repräsentiert eine Serie von markierten Inhalten.
<mark>Es</mark> ist nicht leicht, d<mark>es</mark>en bin ich mir sicher, doch
<mark>es</mark> muss versucht werden.
- **progress** stellt einen Status dar. Beispielsweise den eines Downloads oder einer längeren Operation oder Berechnung, etc.
- **meter** stellt eine Bemessung wie zum Beispiel den verwendeten Festplattenspeicherplatz dar.
- **time** stellt ein Datum und/oder eine Uhrzeit dar: <time>2010-01-22</time>.
- **ruby**, **rt** und **rp** ermöglichen eine Auszeichnung von Ruby-Annotationen.
- **wbr** stellt eine Mögliche Textumbruchstelle dar.

- **video** stellt Videos zur Verfügung, sodass extra Plugins wie der Flashplayer unnötig sind. Das Interface lässt sich über eine API regeln. Beispiel zur Anwendung:

```
<video width="320" height="240" controls="controls">
  <source src="film.ogg" type="video/ogg" />
  <source src="film.mp4" type="video/mp4" />
  Dein Browser unterstützt den video-Tag nicht!
</video>
```

Über den Source-Tag können mehrere Versionen einer Video-Datei hinterlegt werden, welche dem Benutzer die Möglichkeit geben zu wählen. Folgende Attribute sind für das video-Element möglich:

Attribut	Wert	Erklärung
autoplay	autoplay	Bewirkt, dass das Video startet, sobald es möglich ist.
controls	controls	Bestimmt, ob Steuerelemente angezeigt werden (inkl. 'Play')
height	<i>pixel</i>	Definiert die Höhe des Elements.
loop	loop	Bewirkt, dass das Video wiederholt wird, wenn es fertig ist.
preload	preload	Bewirkt, dass der Ladevorgang des Videos beginnt, sobald die Seite geladen wurde.
src	<i>url</i>	Gibt die Quelle an, sofern es eine einzige ist.
width	<i>pixel</i>	Bestimmt die Breite des Elements.

- **audio** stellt Audio-Dateien zu Verfügung, sodass externe Plugins nicht benötigt werden. Das Interface lässt sich ebenfalls über eine API verändern.

```
<audio controls="controls">  
  <source src="song.ogg" type="audio/ogg" />  
  <source src="song.mp3" type="audio/mpeg" />  
  Dein Browser unterstützt das Audio-Element nicht.  
</audio>
```

Attribut	Wert	Erklärung
autoplay	autoplay	Bewirkt, dass die Datei startet, sobald es möglich ist.
controls	controls	Bestimmt, ob Steuerelemente angezeigt werden (inkl. 'Play')
loop	loop	Bewirkt, dass die Datei wiederholt wird, wenn sie abgespielt wurde.
preload	preload	Bewirkt, dass der Ladevorgang der Datei beginnt, sobald die Seite geladen wurde.
src	<i>url</i>	Gibt die Quelle an, sofern es eine einzige ist.

- **canvas** wird verwendet, um Grafiken wie Diagramme oder Spiele dynamisch 'on-the-fly' zu berechnen. Folgender Code würde ein canvas-Element einbinden:

```
<canvas id="einCanvas" width="200" height="100"></canvas>
```

Das canvas-Element hat jedoch selbst keine 'Malwerkzeuge', weshalb das Zeichnen von JavaScript übernommen wird:

```
<script type="text/javascript">
    var canvas=document.getElementById("einCanvas");
    var canvasWithContext=canvas.getContext("2d");
    canvasWithContext.fillStyle="#FF0000";
    canvasWithContext.fillRect(0,0,100,50);
</script>
```

Mit dem Befehl `var canvas=document.getElementById("einCanvas");` wählt JavaScript das obige Canvas-Element aus und 'speichert' es in eine Variable um schneller darauf zugreifen zu können.

In der nächsten Zeile wird über `var canvasWithContext=canvas.getContext("2d");` der Kontext festgelegt. Im '2D-Kontext' sind Rechtecke, Kreise, Dreiecke und Ähnliches möglich. In den darauf folgenden Zeilen wird ein rotes Rechteck gezeichnet.

In dieser Zeile `canvasWithContext.fillRect(0,0,100,50);` wurden Koordinaten übergeben. Die ersten beiden Zahlen stehen für die Startposition. Sie befindet sich in diesem Fall also ganz oben in der linken Ecke. Man beachte, dass das Koordinatensystem in Computersystemen anders aufgebaut ist, als in der Mathematik: Der obere 'Rand' stellt die x-Achse dar und diese verläuft von links nach rechts. Der linke 'Rand' stellt die y-Achse dar und die verläuft von oben nach unten.

Die beiden zweiten Zahlen geben die Größe des Rechtecks an: 100px auf der x- und 50 auf der y-Achse.

Innerhalb des canvas-Elements können noch viele weitere Figuren gezeichnet werden. Auch Farbverläufe und Linien sind möglich. Es können auch normale Grafiken eingebunden werden.

- **command** stellt verschiedene Zustandselemente dar (wie Radio-Button, Checkbox, etc.). In diesem schönen Beispiel vom W3C wurde eine Toolbar implementiert, welche den Benutzer eine Textausrichtung wählen lässt. So etwas wäre in einem Text-Editor denkbar:

```
<menu type="toolbar">

  <command type="radio" radiogroup="alignment" checked="checked" label="Left"
  icon="icons/alL.png" onclick="setAlign('left')">

  <command type="radio" radiogroup="alignment" label="Center" icon="icons/alC.png"
  onclick="setAlign('center')">

  <command type="radio" radiogroup="alignment" label="Right" icon="icons/alR.png"
  onclick="setAlign('right')">

  <hr>

  <command type="command" disabled label="Publish" icon="icons/pub.png"
  onclick="publish()">
</menu>
```

- **details** stellt zusätzliche Informationen oder Steuerelemente zur Verfügung, welche vom Benutzer bei Bedarf abgerufen werden können.
- **datalist** kann zusammen mit dem neuen 'list'-Attribut des input-Elements für ComboBoxen verwendet werden:

```
<input list="browsers">
<datalist id="browsers">
  <option value="Safari">
  <option value="Internet Explorer">
  <option value="Opera">
  <option value="Firefox">
</datalist>
```

- **keygen** ist ein Steuerelemente welches zum Generieren von Schlüsselpaaren verwendet werden kann.
- **output** stellt eine Ausgabe dar, die zum Beispiel durch Skripte erreicht wurde.

Teil 4 – Verschwundene Elemente

In HTML5 wurden auch einige Elemente entfernt. Für HTML5 wurden aber gesonderte Spezifikationen für Autoren (die Programmierer) und User Agents (die Browser) entworfen. So dürfen die Autoren die alten Elemente und Attribute zwar nicht mehr verwenden, die Browser müssen sie aber immer in einer Art und Weise unterstützen, die Kompatibel zu den neuen Spezifikationen sind. Das soll die ganze Angelegenheit für die Autoren vereinfachen. Die Elemente werden also weiterhin unterstützt, sollten aber nach Möglichkeit nicht mehr verwendet werden.

Folgende Elemente sind nun nicht mehr vorhanden, da ihre rein auf die Präsentation bezogenen Aufgaben besser mit CSS gelöst werden:

- **basefont**
- **big**
- **center**
- **font**
- **s**
- **strike**
- **tt**
- **u**

Folgende Elemente sind nicht mehr vorhanden, da sie die Benutzerfreundlichkeit negativ beeinflussten:

- **frame**
- **frameset**
- **noframes**

Folgende Elemente sind nicht mehr vorhanden, da sie zu selten genutzt wurden:

- **acronym** hat zu viel Verwirrung gestiftet, stattdessen soll **abbr** verwendet werden.
- **applet** wurde zu Gunsten von **object** entfernt.
- **isindex** kann durch Formularelemente ersetzt werden.
- **dir** wurde zu Gunsten von **ul** entfernt.

Außerdem ist das **noscript**-Element nur in der HTML- und nicht in der XML-Syntax möglich.

Teil 5 – Geänderte Elemente

Einige Elemente wurden verändert, entweder um ihre Bedeutung deutlicher zu machen oder um ihren Nutzen zu verstärken.

Folgende Elemente wurden verändert:

- Das **a**-Element repräsentiert nun, wenn es kein **href**-Attribut besitzt, einen Platzhalter für einen Link den es an der entsprechenden Stelle gegeben haben könnte und kann nun auch fließenden Inhalt erfassen.
- Das **address**-Element wird nun mithilfe des neuen Konzepts der Einteilung erweitert, das heißt, es bezieht sich immer auf den entsprechenden Abschnitt, in dem es sich befindet.
- Das **b**-Element stellt nun eine Textspanne dar, die hervorgehoben wird, ohne dadurch an Gewichtung zu gewinnen, sondern lediglich einen Produktnamen, ein Schlüsselbegriff oder Ähnliches, welches typographisch hervorgehoben werden sollte.
- Das **cite**-Element soll nun verwendet werden, um den Titel eines Werks (Ein Buch, eine Zeitschrift, ein Gedicht, ein Lied, ein Film, eine Sendung, ein Spiel, ein Kunstwerk, etc.) hervorzuheben, anstatt wie es vorher oft verwendet wurde, zum Beispiel den Namen einer Person hervorzuheben.
- Das **hr**-Element repräsentiert nun einen thematischen Umbruch auf Absatz-Ebene.
- Das **i**-Element stellt nun eine alternative Stimme/Stimmung oder andernfalls eine taxonomische Kennzeichnung, einen technischen Ausdruck, die idiomatische Phrase einer anderen Sprache, einen Gedanken, einen Schiffsnamen oder etwas Ähnliches, deren charakteristische typografische Präsentation kursiv ausgedrückt wird, dar. Die Verwendung ist weitgehend abhängig von der Sprache.
- Für das **label**-Element soll der Browser den Fokus nun nicht mehr von der Beschriftung zum Steuerelement führen, es sei denn, dieses Verhalten ist für die zugrunde liegende Plattform üblich.
- Das **menu**-Element wurde neu definiert, damit es fortan für Kontextmenüs und Symbolleisten von Nutzen ist.
- Das **small**-Element repräsentiert nun Kleingedrucktes (für Kommentare und rechtliche Veröffentlichungen)
- Das **strong**-Element repräsentiert nun Wichtigkeit anstelle starker Betonung.
- Im **head**-Element ist die Verwendung des **object**-Elements nun nicht mehr gestattet.

Teil 6 – Neue Attribute

Für HTML5 wurden einige neue Attribute erdacht, welche natürlich nur Elemente betreffen, die bereits zu HTML4 gehörten. Folgende Attribute sind neu:

- Das **a**- und das **area**-Element besitzen nun das **media** Attribut für die Übereinstimmung mit dem **link**-Element.
- Das **a**- und das **area**-Element erhalten das neue **ping**-Attribut, welches eine durch Leerzeichen getrennte Liste von URLs definiert, welche angepingt (Kurze Aufrufe im Hintergrund) werden sollen. Dadurch wird das User-Tracking einfacher und komfortabler für die Nutzer, da sie nicht mehr über mehrere Seiten umgeleitet werden bevor sie ans Ziel gelangen. Zudem bietet es mehr Privatsphäre, da Nutzer wählen können, ob sie diese Seiten anpingen wollen oder nicht.
- Das **area**-Element erhält zur Übereinstimmung mit dem **a**- und **link**-Element die Attribute **hreflang** und **rel**.
- Das **base**-Element kann nun mit dem **target**-Attribut verwendet werden, hauptsächlich zur Übereinstimmung mit dem **a**-Element. Außerdem gilt das **target**-Attribut nicht weiter als veraltet, da es nützlich für ist Web-Applikationen, zum Beispiel im Verbindung mit dem **iframe**-Element.
- Das **value**-Attribut gilt für das **li**-Element nicht länger als veraltet, da es nicht auf die Präsentation bezogen ist. Dies gilt auch für das **start**-Attribut des **ol**-Elements.
- Das **meta**-Element erhält nun das **charset**-Attribut, welches bereits weit verbreitet war, womit es nun eine schöne Möglichkeit bietet den Zeichensatz zu definieren.
- Das neue **autofocus**-Attribut kann auf **input**- (sofern sichtbar), **select**-, **textarea**- und **button**-Elemente angewendet werden. Es bietet eine einfache Möglichkeit, ein Formular zu fokussieren, während die Seite geladen wird. Außerdem ließe sich dieses 'Feature' vom Nutzer ggf. im Browser deaktivieren.
- Das neue **placeholder**-Attribut kann mit einem **input**- oder **textarea**-Element verwendet werden, um dem Nutzer einen Tipp für möglichen Inhalt zu geben.
- Das neue **form**-Attribut für **input**-, **output**-, **select**-, **textarea**-, **button**-, und **fieldset**-Elemente erlaubt es, Steuerelemente einem Formular bzw. **form**-Element zuzuordnen, selbst, wenn diese nicht direkt im **form**-Element eingeschlossen sind. Dadurch ließen sich die Elemente eines Formulars über ein ganzes Dokument verteilen. Im **form**-Attribut muss lediglich die ID des **form**-Elements eingetragen werden und schon ist die Platzierung außerhalb möglich.
- Das neue **required**-Attribut für **input**- (sofern das type-Attribut weder hidden, noch image, noch ein 'button-type' ist) und **textarea**-Elemente ermöglicht es, dem Nutzer automatisch anzuzeigen, dass das entsprechende Element ausgefüllt werden muss.
- Das **fieldset**-Element erhält nun das **disabled**-Attribut. Es sperrt alle Steuerelemente, welche von ihm abstammen, sich also 'in' dem entsprechenden Element befinden.

- Das **input**-Element hat mehrere neue Attribute erhalten, welche Zwangsbedingungen definieren: **autocomplete**, **min**, **max**, **multiple**, **pattern** und **step**. Wie bereits erwähnt, enthält es ebenfalls das neue **list**-Attribut, welches zusammen mit dem **datalist**-Element verwendet werden kann.
- Das **form**-Element erhält das **novalidate**-Attribute, welches die Formular-Überprüfung beim Verschieken abschaltet. Das Formular kann dann immer abgeschickt werden.
- Die **input**- und **button**-Elemente können nun die Attribute **formaction**, **formenctype**, **formmethod**, **formnovalidate**, und **formtarget** als neue Attribute verwenden. Falls vorhanden, werden damit die Attribute **action**, **enctype**, **method**, **novalidate**, und **target** des Formular-Elements überschrieben.
- Das **menu**-Element hat zwei neue Attribute: **type** und **label**. Diese erlauben dem Element sich in ein Menü zu verwandeln, wie es in typischen Benutzeroberflächen zu finden ist, sowie die Bereitstellung von Kontextmenüs in Verbindung mit dem globalen **contextmenu**-Attribut
- Das **style**-Element besitzt ein neues Attribut: **scoped**, welches verwendet werden kann, um Style-Sheets mit begrenztem Bereich zu ermöglichen. Innerhalb eines solchen Elements treffen die Stilregeln nur auf den entsprechenden Baum zu.
- Das **script**-Element erhält das neue **async**-Attribut, welches das Laden und Ausführen des Scripts beeinflusst.
- Das **html**-Element hat ein neues Attribut namens **manifest**, welches in Verbindung mit der API für offline Webapplikationen auf ein Anwendungs-Cache-Manifest zeigt.
- Das **link**-Element hat ein neues Attribut namens **sizes**. Es kann in Verbindung mit der **icon** Zuordnung (gesetzt durch das **rel**-Attribut) die Größe des angegebenen Icons vorgeben.
- Das **ol**-Element erhält das **reversed**-Attribut, welches für eine umgekehrt nummerierte Liste sorgt.
- Das **iframe**-Element hat drei neue Attribute: **sandbox**, **seamless**, und **srcdoc**, welche für Sandbox-Inhalt zugelassen sind (zum Beispiel Blog-Kommentare)

Einige Attribute von HTML4 sind nun für alle Elemente verwendbar. Diese nennt man globale Attribute: **class**, **dir**, **id**, **lang**, **style**, **tabindex** und **title**.

Außerdem gibt es einige neue globale Attribute:

- Das **contenteditable**-Attribut zeigt an, dass es sich um ein editierbares Element handelt. Der Nutzer kann Inhalte ändern und das Element umdefinieren.
- Das **contextmenu**-Attribut kann benutzt werden um auf ein Kontextmenü zu zeigen, welches vom Autor bereitgestellt wird.
- Die **data-*** Zusammenstellung der vom Autor definierten Attribute. Autoren können jedes beliebige Attribut definieren, solange sie dem Namen des Attributs ein 'data-' voransetzen, um Übereinstimmungen mit zukünftigen Versionen von HTML zu vermeiden. Die einzige Bedingung ist, dass diese Attribute nicht für Erweiterungen der Browser verwendet werden.
- Das **draggable**-Attribut kann zusammen mit der neuen drag&drop-API verwendet werden.
- Das **hidden**-Attribut zeigt an, das ein Element noch nicht oder nicht mehr relevant ist.
- Die **role** und **aria-*** Zusammenstellungsattribute können verwendet werden um technische

Hilfsmittel zu beauftragen.

- Das **spellcheck-Attribut** empfiehlt dem Browser die Rechtschreibüberprüfung aus das jeweilige Element anzuwenden oder nicht.

Teil 7 – Verschwundene Attribute

Für HTML wurden auch einige Attribute 'eingestampft'. Sie stehen nun also nicht mehr zu Verfügung:

- **rev** und **charset** stehen für **link** und **a** nicht mehr zur Verfügung.
- **shape** und **coords** stehen für **a** nicht mehr zur Verfügung.
- **longdesc** steht für **img** und **iframe** nicht mehr zur Verfügung.
- **target** steht für **link** nicht mehr zur Verfügung.
- **nohref** steht für **area** nicht mehr zur Verfügung.
- **profile** steht für **head** nicht mehr zur Verfügung.
- **version** steht für **html** nicht mehr zur Verfügung.
- **name** steht für **img** nicht mehr zur Verfügung (stattdessen soll id verwendet werden).
- **scheme** steht für **meta** nicht mehr zur Verfügung.
- **archive**, **classid**, **codebase**, **codetype**, **declare** und **standby** stehen für **object** nicht mehr zur Verfügung.
- **valuetype** und **type** stehen für **param** nicht mehr zur Verfügung.
- **axis** und **abbr** stehen für **td** und **th** nicht mehr zur Verfügung.
- **scope** steht für **td** nicht mehr zur Verfügung.

Zudem enthält HTML5 keine auf Präsentation bezogene Attribute, deren Eigenschaften besser mit CSS umgesetzt werden:

- **align** steht für **caption**, **iframe**, **img**, **input**, **object**, **legend**, **table**, **hr**, **div**, **h1**, **h2**, **h3**, **h4**, **h5**, **h6**, **p**, **col**, **colgroup**, **tbody**, **td**, **tfoot**, **th**, **thead** und **tr** nicht mehr zur Verfügung.
- **alink**, **link**, **text** und **vlink** stehen für **body** nicht mehr zur Verfügung.
- **background** steht für **body** nicht mehr zur Verfügung.
- **bgcolor** steht für **table**, **tr**, **td**, **th** und **body** nicht mehr zur Verfügung.
- **border** steht für **table** und **object** nicht mehr zur Verfügung.
- **cellpadding** und **cellspacing** stehen für **table** nicht mehr zur Verfügung.
- **char** und **charoff** stehen für **col**, **colgroup**, **tbody**, **td**, **tfoot**, **th**, **thead** und **tr** nicht mehr zur Verfügung.
- **clear** steht für **br** nicht mehr zur Verfügung.
- **compact** steht für **dl**, **menu**, **ol** und **ul** nicht mehr zur Verfügung.
- **frame** steht für **table** nicht mehr zur Verfügung.
- **frameborder** steht für **iframe** nicht mehr zur Verfügung.
- **height** steht für **td** und **th** nicht mehr zur Verfügung.
- **hspace** und **vspace** stehen für **img** und **object** nicht mehr zur Verfügung.
- **marginheight** und **marginwidth** stehen für **iframe** nicht mehr zur Verfügung.

- **noshade** steht für **hr** nicht mehr zur Verfügung.
- **nowrap** steht für **td** und **th** nicht mehr zur Verfügung.
- **rules** steht für **table** nicht mehr zur Verfügung.
- **scrolling** steht für **iframe** nicht mehr zur Verfügung.
- **size** steht für **hr** nicht mehr zur Verfügung.
- **type** steht für **li**, **ol** und **ul** nicht mehr zur Verfügung.
- **valign** steht für **col**, **colgroup**, **tbody**, **td**, **tfoot**, **th**, **thead** und **tr** nicht mehr zur Verfügung.
- **width** steht für **hr**, **table**, **td**, **th**, **col**, **colgroup** und **pre** nicht mehr zur Verfügung.

Teil 8 – Geänderte Attribute

Die folgenden Attribute sind für Autoren zulässig, aber nicht empfohlen. Es sollen nach Möglichkeit andere Lösungswege benutzt werden um diese Attribute zu umgehen:

- Das **border**-Attribut für **img** muss den Wert 0 tragen, wenn es verwendet wird. Stattdessen soll CSS benutzt werden.
- Das **language**-Attribut für **script** muss den Wert "JavaScript" (case-insensitive) tragen, wenn es verwendet wird und darf nicht in Konflikt mit dem **type**-Attribut geraten. Autoren können es einfach weglassen, da es keine nützliche Funktion hat.
- Das **name**-Attribut bei **a**. Es stattdessen das **id**-Attribut verwendet werden.
- Das **summary**-Attribut für **table**. HTML5 bietet zahlreiche andere Mittel und Wege.

Teil 9 – Neue Steuerelemente

Um mehr Benutzerfreundlichkeit zu gewährleisten wurden zusätzliche, neue Werte für das `type`-Attribut des `input`-Elements eingeführt. So sind nun nicht mehr nur noch Text-, Passwort-, Checkbox- und ähnliche Formularelemente möglich, sondern auch:

Der Wert...	Erzeugt ein Steuerelement für...
<code>tel</code>	Telefonnummern
<code>search</code>	Eine Suchanfrage
<code>url</code>	Eine URL
<code>email</code>	Eine oder mehrere Email-Adressen
<code>datetime</code>	Ein Datum und/oder eine Zeitangabe
<code>date</code>	Ein Datum
<code>month</code>	Ein Monat
<code>week</code>	Eine Woche
<code>time</code>	Eine Zeitangabe
<code>datetime-local</code>	Ein locales Datum und/oder Zeitangabe
<code>number</code>	Eine Zahlenfolge
<code>range</code>	Eine Zahl in einem vorgegebenen Bereich
<code>color</code>	Eine Farbangabe im Hexadezimal-System

Teil 10 – Web-Storage

HTML5 bietet zwei neue Möglichkeiten Daten zu speichern. Bisher wurden dafür oft Cookies verwendet. Diese waren aber unpraktisch, wenn es um größere Datenmengen ging, da sie jedes mal übertragen wurden und so die Performance der Website schwächten. Nun stehen zwei neue Methoden bereit. Beide werden über JavaScript verwendet:

- **localStorage** (lokalen Speicher)– speichert Daten ohne Zeitlimit

Folgendes Script erstellt und verwendet einen 'localStorage':

```
<script type="text/javascript">
    localStorage.username="Mueller112";
    document.write(localStorage.username);
</script>
```

- **sessionStorage** – speichert Daten für eine Sitzung (Session)

Folgendes Script erstellt und verwendet einen 'sessionStorage':

```
<script type="text/javascript">
    sessionStorage.username="Mueller112";
    document.write(sessionStorage.username);
</script>
```

Die so gespeicherten Daten können nur von der Website abgerufen werden, die sie auch erstellt hat.

Teil 11 – Browser

Eine wichtige Frage ist jetzt natürlich: **Was davon unterstützen die Browser denn schon?**

Was die neuen Formular-Elemente (email, url, etc.) betrifft, liegt Opera klar vorne. Dieser unterstützt bis auf search und color bereits alle anderen seit dem vorletzten Versionsprung (9.0). Die anderen Browser unterstützen diese Elemente bisher kaum.

Firefox, Opera und Chrome unterstützen bereits Videos im OGG-Format, IE und Safari noch nicht. Dafür unterstützen nur Chrome und Safari Videos im MPEG-4-Format.

Die Audio-Formate Ogg-Vorbis und wav werden von Firefox und Opera unterstützt. Chrome unterstützt Ogg-Vorbis und Mp3, Safari Mp3 und Wav. Der IE unterstützt in der momentanen Version keins dieser Formate.

Die neuen 'echten' Formular-Elemente datalist, keygen und output werden bisher nur von Opera unterstützt. Lediglich Chrome erkennt auch das keygen-Element.

Von den neuen Form-Attributen unterstützt Opera die meisten, dann folgt Chrome. Safari unterstützt einige, Firefox und der IE fast keine.

Leider liegen nicht über alle Funktionen Testergebnisse vor. Aber wenigstens die wichtigsten sind bekannt. Nach diesen liegt Opera wohl vorne. Dicht gefolgt von Firefox. Der IE liegt noch weit zurück. Es ist jedoch zu beachten, dass sowohl Firefox und der IE an neuen Versionen arbeiten (4 und 9) in denen dann wohl reichlich Unterstützung für die neuen Elemente gegeben wird.

Teil 12 – Links

Natürlich habe ich mir das alles nicht aus den Fingern gezogen. Um diesen kleinen Einblick in HTML5 zu gewähren, habe ich mich viel mit einigen Dokumenten auseinandergesetzt. Da ich natürlich nur einen Einblick verschaffen wollte, sind viele Fragen unbeantwortet geblieben. Wer nun mehr erfahren will, kann sich seine Fragen nun mit folgenden Links selbst beantworten:

<http://dev.w3.org/html5/html-author/> - Die momentane Referenz des W3C. Die wohl umfangreichste überhaupt.

<http://dev.w3.org/html5/html4-differences/> - Zeigt die Unterschiede zwischen HTML4 und HTML5 auf und ist leicht verständlich.

<http://www.w3schools.com/html5/> - Vermittelt einen netten Einstieg mit Codebeispielen.

<http://diveintohtml5.org/> - Das wohl beste Werk, das bis jetzt über HTML5 erschienen ist (auch als Buch). Es zeigt mit Humor und vielen Codes anschaulich den englischsprachigen Lesern, was wie mit HTML5 möglich ist. Besser ist meiner Meinung nach schwer denkbar.