Semantisches HTML5: 10 Elemente, die du wahrscheinlich falsch nutzt

Autor: Mario Janschitz
Kontakt:
Datum: 14.12.2014, 15:47 Uhr

Eines der wichtigsten Features von HTML5 ist die Möglichkeit, Inhalte in einen semantischen Kontext zu setzen. Gerade Googles Hummingbird zeigt uns, wohin die Reise gehen wird. In diesem Artikel zeigen wir euch, wie semantisches HTML5 zu schreiben ist. Mit dabei: Die Unterschiede zwischen `em`, `i`, `strong` und `b`.

1 Was bedeutet „Semantik“ in HTML5?

Web 3.0 oder das „Semantic Web“ – heutzutage dreht sich fast alles um Semantik und so natürlich auch in HTML5. Nicht umsonst. Das Semantische Web ist im Rahmen des „Internet der Dinge“ eine Notwendigkeit, um der schieren Datenflut durch die Zuordnung von bloßen Daten zu einem Kontext Herr zu werden und auch Suchmaschinen wie Google bevorzugen semantische Websites. Stichwort: Googles Such-Alogrithmus Hummingbird. Schon Ende 2013 beeinflusste der neue Algorithmus schon 90 Prozent aller Google-Suchanfragen und leitete damit einen wichtigen Schritt zu Tim Burners-Lees semantischem Web ein.

In einem Satz: Ihr setzt euren Inhalt durch Code in einen Kontext. Anders gesagt: Eure Inhalte brauchen Metainformationen, damit sie interpretiert und automatisiert weiterverarbeitet werden können.

Ein Beispiel: „String“ ist für Entwickler ganz klar eine Zeichenkette, für Physiker eine Theorie und für andere Unterwäsche. Ein Wort, drei unterschiedliche Bedeutungen. Das heißt, der abstrakte „String“ wird erst in einem Kontext zur Realität. Und dasselbe Problem gibt es auch bei der Datenverarbeitung. Das Semantische Web – oder Web3.0 – soll durch HTML5 Abhilfe schaffen. Und so vermeidet ihr die häufigsten Fehler im Umgang mit HTML5:

1.1 Aufmerksamkeit erzeugen: Das `i`-Element ist nicht einfach nur „italic“

„Das gibt es doch gar nicht mehr, das wurde ja durch `em` ersetzt!“ Diesen Satz hört man inzwischen ziemlich oft, aber: Er ist falsch.

Das `i`-Element wurde nicht gestrichen, sondern es kennzeichnet Wörter, die anders betont, beziehungsweise ausgesprochen werden. Gründe dafür sind: Fremdsprachen, Transkription, Eigennamen oder um Fachbegriffe zu kennzeichnen. Wenn ihr das `i`-Element aufgrund einer Fremdsprache verwendet, vergesst nicht, die Sprache über das `lang`-Attribut anzugeben.

1.2 HTML5 und das em-Element

Dieses Element wird genutzt, um einzelner Wörter, oder ganzer Sätze herauszuheben beziehungsweise anders zu gewichten. Mit diesem Element könnt ihr die Bedeutung eines Satzes ändern, nur indem ihr einzelne Abschnitte betont:

```
<em>Bier</em> ist ein gutes Getränk

Bier <em>ist</em> ein gutes Getränk

Bier ist ein <em>gutes</em> Getränk
```

Jeder dieser Sätze ist für einen Computer identisch, durch die Betonung, die uns die Sprache ermöglicht, entsteht erst ein Unterschied. Dieses Element sorgt also dafür, dass diese drei Sätze unterschiedlich erfasst werden können, denn die Gewichtung ist bei jedem Satz eine andere.

Das `em`-Element ist also nicht ein allgemeines Element um „italic“ darzustellen. Genauso wenig wie es dazu da ist, um „Wichtigkeit“ zu vermitteln.

1.3 Das `strong`-Element

Wichtigkeit oder Dringlichkeit wird mit dem `strong`-Element gekennzeichnet. Dabei können Sätze mit `strong` gekennzeichnet werden, um wirklich Wichtiges von Unwichtigem trennen zu können. Das gleiche gilt, wenn eure Benutzer etwas zuerst lesen müssen, noch vor allem anderen.

Wichtig ist, dass die relative Wichtigkeit der einzelnen `strong`-Elemente, abhängig zu der Anzahl vorheriger `strong`-Elemente ist. Jedes `strong`-Element erhöht die Wichtigkeit seines Inhalts. Was meine ich damit? Im Gegensatz zu `em` oder `i` wird der Satz also in keinster Weise verändert.

Somit ist das folgende Beispiel durchaus sinnig, wenn „Kapitel 1:“ nur eine Formalität und der eigentliche und wichtige Inhalt „Die Praxis“ ist.

```
<h1>Chapter 1: <strong>The Praxis</strong></h1>
```

1.4 HTML5 und das `mark`-Element

Dieses Element wird benutzt, wenn etwas markiert werden muss, dass weder wichtig noch besonders zu lesen ist. Das `mark`-Element stellt den Text in einen anderen Kontext, als vom ursprünglichen Autor beabsichtigt. Wenn ihr also das Augenmerk auf eine besondere Textstelle legen wollt, die vom Autor nicht als besonders erachtet worden oder nur in einem bestimmten Kontext wichtig ist, dann muss die Wahl auf das `mark`-Element fallen.

Ein Beispiel: Es wird ein Text nach dem Wort „Bier“ durchsucht, das auch gefunden wird. Dieses Wort muss jetzt gekennzeichnet werden, um die Aufmerksamkeit des Lesers auf dieses Wort zu lenken. Dabei bleiben natürlich Inhalt und Sinn unverändert. Hier ist das `mark`-Element richtig eingesetzt. Oder: Eine Zahlenangabe in einem Textblock bezieht sich auf eine Zahl in einem Bild. Das Element zeigt also Relevanz an – nicht Wichtigkeit.

1.5 `b`-Element in HTML5

Das letzte Mittel, um Aufmerksamkeit zu steuern, ist das `b`-Element. Dabei werden Textstellen gekennzeichnet, die weder die Aussprache verändern noch Wichtigkeit oder Relevanz kennzeichnen. Somit ist der Einsatz des `b`-Elements richtig, wenn die Aufmerksamkeit des Lesers gelenkt werden soll, semantisch aber keine Wichtigkeit vorliegt. Das liegt zum Beispiel vor, wenn ihr die erste Zeile eures Absatzes hervorheben wollt (zum Beispiel mit einer Fettung).

1.6 Der Zeilenumbruch mit `br`

Gerade von WYSIWYG-Editoren wird und wurde das `br`-Element regelmäßig falsch eingesetzt. Dabei handelt es sich um einen Zeilenumbruch, um Inhalte wie zum Beispiel Gedichte strukturieren zu können. Um Abstände zu generieren, sollte CSS verwendet werden, das heißt: Wenn ihr `br`-Elemente nur benutzt, um den Abstand zwischen zwei DOM-Objekten zu vergrößern, dann nutzt ihr das Element nicht der Dokumentation entsprechend.

1.6.1 Bonus: ` `

Zugegeben, bei ` ` handelt es sich nicht um ein Element, sondern um eine „Character Entity“. Trotzdem wird sie häufig – ähnlich wie das `br`-Element – als Zeilenumbruch eingesetzt, was falsch ist.

Dabei handelt es sich eigentlich um ein geschütztes Leerzeichen. Wenn ihr also zwei Informationen durch eine Leertaste trennen wollt, da das eine ohne das andere keinen Sinn ergibt, dann solltet ihr diese Entity einsetzen. Beispiele für die Nutzung dieses Leerzeichen sind Werte und dazugehörige Maßeinheiten wie zum Beispiel `12 m` oder 200 US-Dollar.

1.7 Das Kleingedruckte: `small`-Element

Dieses Element ist ausschließlich dazu da, um rechtliche Hinweise oder kurze Disclaimer zu kennzeichnen. Ein praktisches Beispiel ist, unter welcher Lizenz ein Stockfoto steht oder die Kennzeichnung von Preisen, die keine Umsatzsteuer enthalten. Dabei wäre der Hinweis so zu schreiben: `EUR 20<small>exkl. USt.</small>`

Das Impressum oder die AGB sollten nicht mit dem `small`-Element gekennzeichnet werden, denn in diesem Fall sind die AGB der eigentliche Hauptinhalt der Seite.

1.8 Das `hr`-Element

Semantisch gesehen repräsentiert das `hr`-Element einen thematischen Bruch auf Absatz-Ebene. Bei einem Szenenwechsel innerhalb eines Theaterstücks kann dieses Element eingesetzt werden, um diesen thematischen Bruch semantisch darzustellen.

Der Grund, warum bei so einem Wechsel nicht das `section`-Element benutzt wird ist, dass aus der Perspektive der Semantik der Inhaltsabschnitt unverändert bleibt. Das Theaterstück ist immer noch dasselbe Theaterstück – der Wechsel erfolgte „nur“ auf thematischer Ebene, was im Sinne der Semantik zu vernachlässigen ist.

Wichtig zu wissen ist, dass das `hr`-Element sich nicht auf den Outline-Algorithmus auswirkt.

1.9 Adressen mit dem `address`-Element?

Dieses Element wird häufig dazu benutzt, um die postalische Anschrift auf der Kontakt-Seite zu kennzeichnen. Das kann richtig sein, muss es aber nicht, denn in der Regel ist das `p`-Element ausreichend für postalische Anschriften.

Das `address`-Element bedeutet mehr – nämlich die semantische Darstellung einer Kontaktmöglichkeit des im DOM naheliegendsten `article`- oder `body`-Eltern-Elements. Ist also das `address`-Element im `body`-Element, gelten die Kontaktinformationen für dieses eine body-Element beziehungsweise diese Seite – sie könnten sich aber auch auf ein `article`-Element beziehen. Üblicherweise befindet sich das `address`-Element in einem `footer`-Element.

Wichtig dabei ist, dass dieses Element nicht nur postalische Informationen beinhalten kann, sondern auch Links. Es ist also nicht definiert, welche Art von Kontaktinformation angegeben werden kann, sondern nur DASS welche, in Relation zu anderen DOM-Objekten, angegeben werden können.

1.10 Zitate mit `blockquote`, `cite` und `q`

Beim Zitieren merkt man, dass das Web der akademischen Welt entsprungen ist. Das `cite`-Element wird benutzt, um eine Referenz auf ein kreatives Werk setzen zu können. Es kann den Titel des Werks, den Namen des Autors oder eine URI beinhalten. Dabei gilt als Werk:

*Creative works include a book, a paper, an essay, a poem, a score, a song, a script, a film, a TV show, a game, a sculpture, a painting, a theatre production, a play, an opera, a musical, an exhibition, a legal case report, a web site, a web page, a blog post or comment, a forum post or comment, a tweet, a written or oral statement, etc.*W3.org

Das heißt, das `cite`-Element bildet die Herkunft beziehungsweise eine Referenz ab, und dient somit nicht zur Darstellung des eigentlichen Zitats. Folglich ist dieses Codefragment falsch:

```
<p><cite>This is wrong!, said Hillary.</cite> is a quote from the popular
daytime TV drama When Ian became Hillary.</p>
```

Für Zitate ist das `q`-Element zu verwenden. Das obige Beispiel in der semantisch richtigen Form sieht so aus:

```
<p><q>This is correct, said Hillary.</q> is a quote from the popular
daytime TV drama <cite>When Ian became Hillary</cite>.</p>
```

Somit wird die eigentliche Phrase des Zitats mit dem `q`-Element umschlossen, wobei der Herkunft durch das `cite`-Element beschrieben wird. Allerdings muss nicht immer ein `cite`-Element angegeben werden, und `q`-Elemente sind auch ineinander verschachtelbar.

Das `blockquote`-Element ist semantisch nicht auf dem Text-Level, sondern gruppiert Inhalte. Da es keine einheitliche Regelung gibt, wie Zitate auf Code-Ebene mit `blockquote` dargestellt werden sollen, wird zur Referenzangabe aber die Verwendung des `footer`- oder `cite`-Elements vorgeschlagen.

```
<blockquote>
    <p>I contend that we are both atheists. I just believe in one fewer
    god than you do. When you understand why you dismiss all the other
    possible gods, you will understand why I dismiss yours.</p>
    <footer>— <cite>Stephen Roberts</cite></footer>
</blockquote>
```

```
<p>His next piece was the aptly named <cite>Sonnet 130</cite>:</p>
<blockquote cite="http://quotes.example.org/s/sonnet130.html">
<p>My mistress' eyes are nothing like the sun,<br>
Coral is far more red, than her lips red,<br>
...
```

```
<p>He began his list of "lessons" with the following:</p>
<blockquote>One should never assume that his side of
the issue will be recognized, let alone that it will
be conceded to have merits.</blockquote>
<p>He continued with a number of similar points, ending with:</p>
<blockquote>Finally, one should be prepared for the threat
of breakdown in negotiations at any given moment and not
be cowed by the possibility.</blockquote>
<p>We shall now discuss these points...
```

Alle drei oben genannten Varianten sind richtig und möglich. Das `blockquote`-Element kann aber auch Überschriften, Listen und sogar Bilder enthalten und ist somit für umfangreichere Zitate konzipiert, kann aber das `q`-Element auch ersetzen.

Wenn ihr mehr über `article`-, `section`-, `aside`-, `nav`- und `header`-Elemente lesen wollt, dann werdet ihr im Artikel „Semantik im Web 3.0: Wir zeigen euch, wie HTML5 dem Web mehr Bedeutung verleiht“ fündig. Oder ihr kämpft euch durch die HTML5-Dokumentation.

Quelle: http://t3n.de/news/semantik-html5-2-584087/