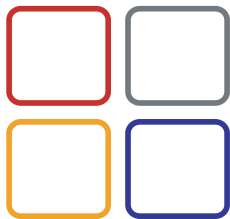


Internet ohne Hürden, Teil 2

Durchblick mit Struktur

www.best-off.org

Lösungen
Klaus Reichenbach
kr@best-off.org



best off

Inhaltsverzeichnis

1 Nicht alles ist Gold	3
2 Beim Namen nennen	4
3 Datenfütterung	5
4 Blinde oder sehbehinderte Anwender	5
5 Anwender mit motorischen Einschränkungen	5
6 Schneller surfen	6
7 Bilder für Blinde	7
8 Schmerzlose Tabellen	8
9 Werkzeugkasten	9
10 Fazit	9

Vorwort

Accessible (zugängliches) Web-Design hat keineswegs nur den Vorteil, Webseiten für Behinderte zugänglich zu machen. Die Schritte auf dem Weg dorthin führen gleichzeitig zu einer leicht zu pflegenden, wirtschaftlichen und flexibleren Internetseite.

Zunächst haben wir von Ihrer Webseite allen unnötigen Ballast entfernt, indem sie von veralteten Syntaxkonstrukten befreit und die optische Gestaltung in externe Stilvorlagen ausgelagert wurde. Mit der damit vollzogener Umstellung hat man schon einiges erreicht: Die Inhalte werden nicht mehr durch Tags durchgesetzt, die nur das optische Erscheinungsbild beeinflussen. An ihre Stelle treten Syntaxelemente mit struktureller Bedeutung - so hebt z.B. statt eines strukturell irrelevanten `jetzt Abschnitte` hervor. Diese neue, korrekte Auszeichnung erschließt sich auch einem blinden Anwender sofort, der die Seiten mit einem Screen Reader besucht. Hinzu kommt, dass sich `` je nach Wahl des Designers entweder durch Fettschrift oder durch eine auffällige Farbe hervorheben darf - das kann man im Style Sheet frei festlegen.

1 Nicht alles ist Gold

Gültiger Code bildet die Grundlage für alle nachfolgenden Schritte - dies schließt sowohl die HTML/XHTML-Syntax ein als auch die Cascading Style Sheets (CSS).

Das Standardisierungskomitee World Wide Web Consortium (W3C) bietet kostenlose Online-Dienste zur Überprüfung von HTML- und CSS-Dokumenten; ein deutschsprachiger Alternativ-Validator findet sich beim HTML-Tutorial SelfHTML. Im Unterschied zu anderen im Web verfügbaren Tools führen diese Tester eine komplette logische Analyse der Seite durch und finden so auch komplexe Fehler wie richtig formatierte Tags an einer unzulässigen Stelle.

Allerdings klopfen selbst die besten Validatoren Dokumente nur auf Grammatikfehler ab - ob die Seite semantisch sinnvoll aufgebaut ist, beurteilen sie nicht. Man kann dies analog zu einer automatischen Rechtschreibprüfung betrachten, die zwar die korrekte Schreibweise eines Briefes überprüft, nicht aber ihren Wahrheitsgehalt. Hinzu kommt, dass Validatoren gelegentlich auch fehlerhafte Verschachtelungen akzeptieren.

Mancher Web-Designer ist trotzdem der Meinung, valides (X)HTML bedeute automatisch, dass seine Seiten in Ordnung sind. Hinter diesem Vorhang finden sich dann aber oft genug semantisch sinnlose `<div>`-Tags, die nur dazu dienen, Inhalte per CSS optisch zu formatieren. Das mag zwar valides (X)HTML sein, ist dann aber alles andere als zugänglich.

Was das W3C selbst für behindertengerecht hält, hat es in seinen "Zugangsrichtlinien für Web-Inhalte" festgehalten. Vom Inhalt her entsprechen die Anforderungen grob der "Barrierefreien Informationstechnik-Verordnung" des Bundesinnenministeriums (BITV); allerdings formuliert das W3C die Richtlinien diplomatischer und einfühlsamer.

Um die vollständige Zugänglichkeit zu überprüfen, gibt es auch den Validator "Bobby", der Seiten entweder nach den Kriterien des W3C oder den in den USA geltenden Accessibility-Anforderungen testet. Das Werkzeug gibt es auch als Windows-Programm für 300 US-Dollar. Seine Testergebnisse gibt Bobby in drei Prioritätsstufen gegliedert wieder - wer alle meistert, darf sich mit dem Logo "AAA Bobby Approved" auszeichnen.

In der Praxis ist Bobby trotz seines imposanten Auftritts aber nur begrenzt hilfreich. So berücksichtigt der Dienst keine barrierefreien Alternativseiten und scheint bei der Einschätzung der logischen Struktur gelegentlich zu "raten". So erkennt Bobby etwa nicht, dass ein mit JavaScript arbeitender Hyperlink auch ohne das Attribut "onkeypress" tastaturfähig ist. Andererseits schluckt der Validator einige Seiten mit semantischen Fehlern ohne Klage.

So hilft Bobby bestenfalls geübten Entwicklern dabei, aus der Mängelliste diejenigen herauszu-

picken, die richtig erkannt wurden. Einsteiger verunsichern die falschen Fehlermeldungen eher, als dass sie helfen würden.

Das gilt so oder ähnlich auch für andere Produkte von der Sorte "Klick deine Sorgen weg". Jetzt, wo die Deadline für den obligatorischen Umstieg auf barrierefreie Seiten immer näher kommt, bieten viele Unternehmen Werkzeuge an, die bestehende Seiten behindertengerecht machen sollen. Die meisten dieser Tools grasen die vorhandenen Seiten aber lediglich stur nach häufigen Formfehlern, wie etwa fehlenden alt-Attributen, ab. Das mag für den Anfang eine hilfreiche Stütze sein, macht eine Seite aber noch lange nicht barrierefrei.

2 Beim Namen nennen

Auf semantisch unstrukturierten Seiten erwartet sehbehinderte Besucher eine chaotische Abfolge von Inhalten, die stellenweise zudem falsch vorgelesen werden. Damit ein Screen Reader die Zeichenfolge "USA" nicht als "Usa" ausspricht, sondern durchbuchstabiert, muss sie als Abkürzung gekennzeichnet sein, am besten mit einer Erklärung im title-Attribut.

```
<acronym title="United States Of America">USA</acronym>
```

Texte bestehen nicht nur aus Worten und Sätzen; sie folgen auch einer logischen Struktur. Das beginnt mit der Unterscheidung von Überschriften (<h1>bis <h6>) und Absätzen und setzt sich fort mit Aufzählungen und Listen mit Gliederungspunkten.

Für kurze und längere Zitate, Adressangaben, Code-Schnipsel und Definitionen sehen HTML und XHTML spezielle Syntaxelemente vor. Diese Möglichkeit sollte man auch nutzen - sowohl im Sinne der Zugänglichkeit, als auch um diese Elemente optisch gezielt per Style Sheet auszeichnen zu können.

Hat man sich erst einmal an die Trennung von Form und Inhalt gewöhnt, muss man im nächsten Schritt auch das HTML-Dokument selbst in seine Bestandteile zerlegen. So können Navigationsleisten, Kopf- und Fußzeilen und sonstige Elemente von den eigentlichen Inhalten getrennt bearbeitet werden.

Zur Umsetzung dieses Vorhabens gibt es mehrere Ansätze. Der bei weitem schlechteste besteht darin, mit Frames zu arbeiten - je ein Frame für den Seitenkopf und -fuß, ein dritter für die Navigationsleiste und der letzte für den Inhalt. Für Suchmaschinen und Screen Reader sind Frames jedoch Gift. Eine bessere Methode besteht darin, die Seitenstruktur nach ihrer Funktion in separate Dateien aufzuteilen, die erst später wieder zusammengeführt werden.

Die einfachste Methode dazu sind Server Side Includes (SSI). Dies bedeutet in Kurzfassung: SSI ist eine von allen gängigen Web-Servern angebotene Methode, Dateien in andere einzubinden. Der Server jagt dazu jede Seite mit der Endung ".shtml" durch einen Parser, der die darin enthaltenen SSI-Befehle auswertet und ausführt. Eine vollständige Liste der Befehle findet sich unter anderem bei SelfHTML.

Die Folgen der Befehle bekommt man erst zu sehen, wenn sie den Server-Parser durchlaufen haben. Daher empfiehlt es sich, zu Testzwecken einen lokalen Server zu installieren wie z.B. die Freeware-Pakete LAMP oder XAMPP.

Per SSI lassen sich ohne weiteres Kopf- und Fußzeilen sowie Navigationsleisten in zentrale Dateien auslagern, die nur das relevante Code-Stückchen enthalten. Wie bei CSS muss man dann nur noch ein Dokument bearbeiten, um eine Änderung in allen SHTML-Dateien durchzuführen, die es einbinden. Die Zeitersparnis ist enorm; der Aufwand bei der Pflege der Seite schrumpft drastisch.

Server Side Includes sind auch sehr suchmaschinenfreundlich, da deren Robots die SSI-Befehle selbst nicht zu sehen bekommen, sondern nur das Ergebnis - so nehmen sie die Seite als ganz normale HTML-Dokumente wahr.

Einen Schritt weiter gehen Skriptsprachen wie ASP oder PHP. Damit lassen sich komplexere Anweisungen umsetzen wie etwa Hyperlinks, die in der hierarchischen Baumstruktur eines Server-Verzeichnisses selbstständig von einem Dokument zum nächsten führen. Ähnlich funktionieren auch Content-Management-Systeme (CMS) - die meisten sind letztendlich nicht viel mehr als Skript-Frontends.

3 Datenfütterung

Mit ASP, PHP oder Python arbeitende Websites beziehen üblicherweise die Inhalte und Navigationsstrukturen aus einer Datenbank. Ein Klick des Besuchers startet eine Abfrage, aus der das Skript oder CMS eine Seite erzeugt, die dann an den Browser des Anwenders geschickt wird. Die Funktion der Skriptsprache besteht also darin, Seiten aus verschiedenen Datensätzen zusammenzusetzen. Im Extremfall bestehen skriptgestützte Websites aus einer einzigen Seite, die stets dynamisch gefüllt wird - die Konsequenz aus der Trennung von Form und Inhalt.

Streng genommen ist die Trennung von Form und Inhalt nur eine Vorstufe zum zugänglichen Web-Design da. Will man aber konsequent auf ein zugängliches (accessible) Design umstellen, sind die vorangegangenen Schritte unvermeidlich.

4 Blinde oder sehbehinderte Anwender

Zugängliches Web-Design beginnt schon mit der Wahl geeigneter Farben - auch farbenblinde Internet-Nutzer gehören zur Gruppe der Sehbehinderten. Die meisten von ihnen haben Probleme bei der Unterscheidung von Rot und Grün - diese Farben sollte man daher nicht kombinieren.

Stärker fehlsichtige Anwender haben dagegen Probleme mit schwachen Kontrasten - einige ziehen sogar negative Kontraste vor. Diesem Publikum kann man zwei Varianten des Site-Designs mit starken Kontrasten anbieten: eine mit schwarzer Schrift auf weißem Hintergrund, die andere mit weißer Schrift und schwarzem Hintergrund.

Geschwächten Augen kommt auch die Möglichkeit zur Skalierung der Schriftgröße entgegen. Am einfachsten geht dies, wenn der Web-Designer ganz auf eine Festlegung von Font-Größen verzichtet - dann kann der Besucher den gewünschten Schriftgrad in den Browser-Einstellungen selbst definieren. Dabei muss man aber berücksichtigen, dass viele Web-Surfer gar nicht wissen, dass ihr Browser überhaupt eine Einstellungsmöglichkeit für den Schriftgrad bietet. Hier ist ein entsprechender Hinweis für den Anwender sinnvoll.

Eine Alternative besteht im Einsatz eines so genannten Style Switcher. Dabei handelt es sich um ein Server-Skript, das die Seiten alternativ mit unterschiedlichen Style Sheets anzeigt. Der im ersten Teil erwähnte CSS Zen Garden nutzt beispielsweise ein solches Skript. Besucher finden im Navigationsbereich der Seite einen Link zum Style Switcher, stellen dort das von ihnen vorgezogene Aussehen ein und nutzen von da an das durch die Auswahl gewählte Style Sheet.

Über einen Style Switcher lassen sich unkompliziert verschiedene optische Varianten der Seite mit unterschiedlichen Schriftgrößen und Farbschemata bereitstellen. Spätestens dann zahlt sich die konsequente Auslagerung aller optischen Veränderungen in separate Stilvorlagen aus. Das dazu benötigte Skript kann man entweder selbst entwickeln oder aus dem Web herunterladen - eine Google-Suche führt schnell zu reicher Ausbeute in ASP oder PHP.

5 Anwender mit motorischen Einschränkungen

Besucher mit motorischen Einschränkungen stellen dagegen ganz andere Anforderungen an das Web-Design. Viele von ihnen arbeiten mit technischen Hilfsmitteln, die eine virtuelle Tastatur auf dem

Bildschirm einblenden, die mit einem großen Stick bedient wird. Andere navigieren mit Großfeldtastaturen. Eine unüberwindbare Zugangshürde stellen hier JavaScript-Menüs dar, die erst bei genauer Mausplatzierung ausklappen.

6 Schneller surfen

Lässt man sich einmal mit einer Testversion von JAWS oder Window-Eyes die eigene Homepage vorlesen, entwickelt sich das schnell zur wahren Geduldsprobe. Ach, gäbe es doch eine Möglichkeit, schnell zum eigentlichen Inhalt der Seite zu springen, oder einen fixen Weg zu den umliegenden Seiten. Die gute Nachricht: Gibt es alles. Jetzt die schlechte: Nur, sofern es der Entwickler vorgesehen hat.

Zwei konkrete Möglichkeiten gibt es zur Erleichterung der Navigation: spezielle Tastenkürzel zum direkten Zugriff auf vordefinierte Seiten sowie die Steuerung der Link-Anwahl über die Tabulatortaste.

Vielen sehenden Anwendern ist unbekannt, dass die Tabulatortaste in allen gängigen Browsern von einem Hyperlink des Dokuments zum nächsten wechselt - bei Frames allerdings stets nur innerhalb des aktuellen Rahmens. Normalerweise geht der Browser dabei die Links in der Reihenfolge durch, in der sie im Quelltext vorkommen. Benutzen sie also Frames besser gar nicht erst.

Mit dem `tabindex`-Attribut kann der Web-Designer diese Reihenfolge allerdings auch manipulieren, um direkt zu bestimmten Seitenbereichen zu führen. Das Attribut lässt sich bei allen `<a>`-Tags und Formularfeldern einsetzen. Ein Beispiel:

```
<a href="dateiname.html" tabindex="5">Linktext</a>
```

Eine noch direktere Verlinkungsmöglichkeit bieten Accesskeys. Über diese können Besucher bestimmte Links gezielt und direkt ansteuern, ohne sich mit der Tabulator-Taste durch alle Verweise der Seite zu kämpfen.

Alle modernen Browser unterstützen Accesskeys: Der Internet Explorer kennt sie unter Windows seit Version 4, unter Mac OS seit der Revision 5.5, Netscape seit Version 6 und Mozilla von Anfang an. Opera beherrscht die Funktion erst seit Revision 7.02 und zeigt in der Implementierung noch einige Schwächen.

Unterschiede gibt es allerdings bei der konkreten Umsetzung der Funktion: Der Internet Explorer fokussiert den mit einem Accesskey belegten Link und öffnet ihn erst nach Bestätigung mit der Eingabetaste. Mozilla und Opera führen den Link dagegen sofort aus.

Im Internet Explorer und Mozilla dient die Alt-Taste zur Aktivierung der Accesskeys; bei Opera ist es die etwas unhandliche Kombination Umschalt+Escape. Die Lösung der beiden großen Browser-Hersteller kollidiert hingegen mit einer Eigenschaft des Windows-Betriebssystems: Alle Anwendungsmenüs werden mit Alt-Tastenkombinationen aufgerufen, darunter Alt+D für Datei, Alt-Leertaste für Fenster und viele andere.

Erschwerend kommt hinzu, dass die konkreten Alt-Kürzel je nach Sprache des Betriebssystems variieren - was hierzulande Alt-D ist, ist für Amerikaner Alt-F (File) und für Spanier Alt-A (Archivo). Da Accesskeys eine höhere Priorität haben als das Dateimenü, würde ein Accesskey "D" das Dateimenü lahm legen. Es bleiben effektiv also wenig "sichere" Tasten übrig, eigentlich fast nur die Zahlen 0 bis 9.

```
<a href="dateiname.html" accesskey="1">Linktext</a>
```

Die Auswahl der Hyperlinks für die zehn international verfügbaren Accesskeys sollte wohlüberlegt sein: Um z.B. Behinderten tatsächlich eine Orientierungshilfe zu bieten, müssen sie für die gesamte Site gelten.

Ein paar unverbindliche Vorschläge: "1" verlinkt auf die Hauptseite, "2" zu einer Auflistung aller Accesskeys, "0" führt bei mehrteiligen Beiträgen zur folgenden Seite, "9" dagegen zur vorangegangenen Seite.

Damit der behinderte Anwender überhaupt erfährt, dass die Seite Accesskeys bietet, müssen diese im Title-Attribut der Hyperlinks aufgeführt werden. Diese Schaltfläche führt beispielsweise zu einem Impressum:

```
<a href="impressum.html" accesskey="1" title="Unser Unternehmen - Accesskey 1"></a>
```

Der Button ist eine Grafik mit einem alt-Attribut, das den von der Grafik angezeigten Text wiedergibt. Das erklärende title-Attribut steht im a-Tag. Bei abgeschalteten Bildern erscheint also "Impressum" als Platzhalter. Schwebt der Mauszeiger über dem Hyperlink, erscheint eine QuickInfo mit dem Inhalt des Title-Attributs ("Unser Unternehmen - Accesskey 1"). Diesen gibt auch der Screen Reader aus.

Hier zählt sich erneut eine saubere Trennung von Inhalten und dem Drumherum aus: Wer seine Navigation per SSI oder PHP ausgelagert hat, muss die Accesskeys nur einmal in der Datei mit der Navigation definieren. Dynamische Verweise auf die jeweils vorangehende oder folgende Seite lassen sich nur bei dynamisch oder automatisch generierten Seiten umsetzen - ob direkt per PHP oder aber mit einem Content Management Systems.

7 Bilder für Blinde

Hartnäckig hält sich bei einigen Web-Designern die falsche Überzeugung, zugängliches Web-Design bedeute einen vollständigen Verzicht auf Grafiken. Diese Einschränkung bietet lediglich den einfachsten Weg, die Lesbarkeit einer Seite zu gewährleisten und macht nicht allzu viel Sinn. Richtig ist jedoch: der bloße Verzicht auf Grafiken garantiert noch lange nicht, dass die Seite behindertengerecht ist.

Zugänglichkeit (Accessibility) bedeutet vielmehr, dass alle Bildelemente so aufbereitet sind, dass sich deren Inhalt auch für Behinderte erschließt. Dies beginnt bei richtig und klar lesbar eingesetzten alt- und title-Attributen und führt bis zu ausführlichen Beschreibungen in separaten HTML-Dokumenten. Viele Web-Designer stecken Klöpsle wie "Klicken Sie hier für unsere aktuelle Angebotsliste" oder "©2000 Beispielverlag" ins alt-Attribut und freuen sich dann, wenn der Internet Explorer (und zwar nur dieser!) ein QuickInfo-Kästchen mit dem gewünschten Inhalt zeigt.

Für diesen Zweck sehen HTML und XHTML jedoch das title-Attribut vor: Es funktioniert gleichermaßen unter Linux, Mac OS und Windows bei Internet Explorer, Netscape und Opera. Wenn es um die title-Anzeige geht, sind sogar K-Meleon und Safari mit vollem Einsatz dabei.

alt steht hingegen für "alternate" - also alternativ. Der Inhalt dieses Attributs soll also nur ausgegeben werden, sofern der Browser das Bild nicht anzeigt. Entsprechend sollte hier also etwas stehen, was als Platzhalter dienen kann. Bei einem Button mit der Aufschrift "Angebotsliste" wäre also alt="grünes Rechteck mit Schrift" ebenso deplatziert wie alt="angebotsl.gif". Attribute mit dem Dateinamen des Bildes stammen oft von Web-Editoren, die das Attribut übereifrig vollautomatisch ausfüllen - schließlich muss laut XHTML-Spezifikation jedes Bild ein solches Attribut erhalten. Angemessen wäre also nur alt="Angebotsliste". Dient ein Bild lediglich als Platzhalter, sollte es ein leeres alt-Attribut enthalten (alt="") - Screen Reader interpretieren dies dann einfach als "Dieses Bild hat keine Bedeutung" und überspringen es einfach.

Ein paar Beispiele für sinnvolle alt-Attribute:

- Button: Beschriftung (alt="Angebotsliste")
- Logo: Beschriftung; bei Logos ohne Text der Herstellername (alt="Beispielverlag")

- Grafik ohne Link, zum Beispiel Wegbeschreibung: kurze Beschreibung (alt= "Wegbeschreibung")
Sinnlos sind hingegen folgende Attribute:
- Dateinamen oder -größen (alt = "Bild, 45 kByte")
- lange Bildbeschreibungen (alt= "Hier sehen Sie das wunderschöne ...")
- Leerzeichen (alt= "")

Problematisch wird es bei Bildern mit viel Text - etwa dem Scan eines Presseberichts oder ähnlichem Material. Das führt nicht zuletzt dazu, dass Leser immer wieder anfragen, wie man im alt-Attribut einen Zeilenumbruch hinbekommt. Doch für lange Beschreibungen ist dieses Syntaxelement weder gedacht noch geeignet.

Ausführliche Bildbeschreibungen für Screen Reader müssen daher mit einer längeren Beschreibung verlinkt werden, die in einer separaten HTML-Datei unterkommt. Dazu dient das Attribut longdesc (für "Long Description"), das auf die URL mit der Beschreibung verweist.

Ein Beispiel:

```

```

In Zusammenarbeit mit den Mozilla-Browsern werten Screen Reader das longdesc-Attribut schon heute aus - nicht aber beim Internet Explorer. Damit auch Anwender des Microsoft-Browsers die Seite mit der ausführlichen Erläuterung finden, kann man neben das Bild einen kleinen Link unterbringen:

```
<a href="beschreibung.html" title="Verweis zur Bildbeschreibung">[D]</a>
```

Long Descriptions ergeben aber nur dann Sinn, wenn das Bild darin anschaulich in allen Einzelheiten beschrieben wird. Blinde und sehbehinderte Anwender werden die Mühe zu schätzen wissen.

Besser ist es allemal, auf eingescannte Texte in Bildform zu verzichten und diese als richtigen Text auf die Webseite zu bringen.

8 Schmerzlose Tabellen

Trotz aller Einwände gegen den Einsatz von Tabellen in HTML-Seiten gibt es durchaus Bereiche, in denen Tabellen kein notwendiges Übel sind, sondern genau das richtige Syntaxelement: zur Darstellung tabellarischer Daten. Dazu gehören nicht nur Quartalsergebnisse und Fußballtabellen, sondern auch Kalender und ähnliche Strukturen.

Hier kann man einiges falsch machen - denn Screen Reader können Tabellen erst dann strukturiert ausgeben, wenn sie korrekt aufgebaut sind. Tabellenauszeichnungen helfen aber auch dem Web-Designer dabei, den Überblick zu behalten. Dazu empfiehlt sich zunächst das title-Attribut. Sehbehinderten hilft das summary-Attribut, in dem der Inhalt der Tabelle kurz und prägnant zusammengefasst wird.

Um den Tabelleninhalt für blinde und sehende Besucher gleichermaßen zusammenzufassen, sieht die (X)HTML-Spezifikation das <caption>-Tag vor. Ob dessen Inhalt als Über- oder Unterschrift angezeigt wird, entscheidet das align-Attribut. Prinzipiell funktioniert die Positionierung auch über CSS (z. B. caption side: bottom); leider ignoriert der Internet Explorer diese Auszeichnung.

Innerhalb der Tabelle unterscheiden die Tags <th> und <td> Zellen mit Überschriften von Zellen mit Daten. Angenehmer Nebeneffekt: So lassen sich Überschriften und Daten auch zentral über ein Style Sheet getrennt voneinander formatieren. So heben die Browser den Inhalt von <th>-Tags durch Fettschrift von sich aus hervor.

Bei der Entschlüsselung von Abkürzungen in Tabellenüberschriften hilft das abbr-Attribut. So verhindert <th abbr= "Standardabkürzung " >Std-Abk.</th>, dass der Screen Reader dem sehbehinderten Besucher nur "Std Strich Abk Punkt " entgegenspuckt.

9 Werkzeugkasten

Die meisten grafischen Web-Editoren sind nur sehr begrenzt hilfreich, wenn es um die Erstellung zugänglicher Webseiten geht. Dreamweaver MX bietet zwar einen Accessibility-Check, der sich an den US-amerikanischen Vorgaben orientiert. Doch fehlt meist schon den Dialogen zum Einfügen von Grafiken die Option, überhaupt ein title-Attribut zu setzen, geschweige denn einen longdesc-Verweis. Da der Weg zur Barrierefreiheit stets in die Quelltext-Ansicht zurückführt, kann man auf den grafischen Bearbeitungsmodus daher auch gleich ganz verzichten.

Bei einem Design, das die optische Gestaltung der Seite komplett in Style Sheets auslagert, verliert ein WYSIWYG-Editor seinen Sinn. Sehr viel nützlicher ist ein brauchbarer CSS-Editor mit einer guten grafischen Vorschau der getroffenen Stilentscheidungen. Den HTML-Teil kann man ebenso gut mit einem HTML-Texteditor schreiben. Wer viel Text vor sich hat, kann diesen in einem einfachen grafischen HTML-Editor wie den mit Mozilla ausgelieferten Composer verfassen und die relevanten Bereiche dann per Cut & Paste in den Quelltext der eigentlichen HTML-Datei überführen. Diese Aufgabe kann natürlich auch ein anderer WYSIWYG-Editor übernehmen, sofern dieser den Quelltext nicht selbstständig mit Syntaxelementen anreichert, die dort nicht hingehören.

10 Fazit

Auf den ersten Blick mag zugängliches (accessible) Web-Design wie eine neue Plackerei erscheinen, die viel Aufwand für eine kleine Zielgruppe bedeutet. Diese Betrachtungsweise greift aber erheblich zu kurz; Accessibility (Zugänglichkeit) ist kein Selbstzweck. Standardkonform gestaltete Seiten erreichen nicht nur eine größere Zielgruppe, sie sind wirtschaftlicher, bieten langfristige Rechtssicherheit und sind auch viel leichter zu pflegen. So profitieren Anwender und Entwickler gleichermaßen von zugänglichem Web-Design. Das sollte die Umstellung allemal wert sein.

Literatur

- [TR-BITV] Verordnung zur Schaffung barrierefreier Informationstechnik nach dem Behindertengleichstellungsgesetz. 24. Juli 2002. (<http://www.gesetze-im-internet.de/bitv/>)
- [TR-XHTML1] XHTMLTM1.0 The Extensible HyperText Markup Language (Second Edition). XHTML Working Group und andere. 01. August 2002. (<http://www.w3.org/TR/xhtml1/>)
- [TR-Css2] Cascading Style Sheets, level 2 (Css2 Specification). CSS Arbeitsgruppe und andere. 12. Mai 1998. (<http://www.w3.org/TR/Css2/>)
- [W3-WAI] Web Accessibility Initiative (WAI). W3C und andere. 1994. (<http://www.w3.org/WAI/>)
- [TR-WCAG20] Web Content Accessibility Guidelines 2.0 (W3C Working Draft). 19. November 2004. (<http://www.w3.org/TR/WCAG20/>)