Tidy und Co., der Syntax-Checker für Web-Dokumente

Saubermänner

Sierk Bornemann

Der W3C Markup Validation Service kann HTML- und XHTML-Seiten auf korrekte Syntax hin überprüfen. Tidy und Co. hilft im Vorfeld, den Markup Validator des W3C zufriedenzustellen.

Vortrag vor der PHP Usergroup Hannover

gehalten am 03.03.2005
Update am 23.09.2005

Inhaltsverzeichnis

Kraft-Fahrzeuge müssen hierzulande für den Straßenverkehr zugelassen sein. Das heißt nicht nur, dass sie bestimmten technischen Normen entsprechen müssen, sondern sie müssen auch üblichen Qualitäts- und Sicherheitsrichtlinien entsprechen. Diese Voraussetzungen sind notwendig, um sich im individuellen Straßenverkehr frei bewegen zu dürfen. Das Risiko der gegenseitigen Gefährdung oder gar Schädigung wäre sonst zu hoch.

> **Das Internet: „…** Aber auf dieser Autobahn – um im Bilde zu bleiben – tummelt sich eine merkwürdige Ansammlung von Fahrzeugen: Busse mit Rasenmähermotoren, Sportwagen mit Fahrradfelgen und Käfer mit Zwölfzylinder-Turbolader. Sicherheitsgurte und Airbags gibt es nur ausnahmsweise; Türschlösser, Windschutzscheiben, ja sogar Bremsen und Lenkrad gelten als entbehrliches Sonderzubehör. Zu allem Überfluss sind sämtliche Fahrzeuginsassen maskiert.“
>
> Paul Wallich, Spektrum der Wissenschaft, 05/94

1 Regeln und Standards auch im World Wide Web

Man könnte nun meinen, das World Wide Web wäre ein Freiraum, in dem sich jeder so bewegen könne, wie es ihm beliebt. Ohne Regeln, ohne Gesetze. Ohne technische Mindeststandards, ohne Qualitätsprüfungen.

Vor allem in der zweiten Hälfte der zurückliegenden 90er Jahre, der Zeit als das zunächst akademisch geborene World Wide Web die segensreiche Erfahrung machen durfte, von der Öffentlichkeit wahrgenommen und für sich entdeckt zu werden, waren derlei Qualitätsstandards noch kaum entwickelt. Die „Brot- und Buttersprache“, um Textinhalte im Web auszuzeichnen, HTML, war Mitte der 90er Jahre gerade erst vom World Wide Web Consortium (W3C) [1] veröffentlicht (das W3C ist ein Zusammenschluss von ausgesuchten Vertretern weltweit interessierter Firmen und Gruppen aus dem Umfeld der Informationstechnologie, welcher betreut und geführt wird von drei akademischen Einrichtungen: MIT/CSAIL USA, ERCIM Frankreich und KEIO University Japan).

Allgemein übliche Praxis war damals (und ist es vielfach noch heute), Web-Dokumente zu veröffentlichen ohne Rücksicht auf korrekte Syntax oder semantische Sinnhaftigkeit. Warum auch? Niemand hatte es bisher eingefordert. Die Browser-Hersteller Netscape und Microsoft hingegen haben in der Vergangenheit mehr oder minder erfolgreich mit ihren Web-Browsern versucht, dem Ersteller von Webseiten das Vokabular zu diktieren. Besonders im legendären Browser-Krieg der 90er-Jahre versuchten diese beiden Hersteller sich mit teilweise inkompatiblen Spracherweiterungen gegenseitig das Feld streitig zu machen. Microsoft versuchte mit allen Mitteln, die Hoheit des Pioniers Netscape zu brechen. Der Ausgang dieses Browser-Krieges ist bekannt, der weltweit vorherrschende Web-Browser ist seit einigen Jahren Microsofts Internet Explorer (begrüßenswerterweise hat er inzwischen mehr als überzeugende Konkurrenz bekommen).

Dem Autor von Web-Dokumenten wurde in der Vergangenheit auf diese Weise die Qual der Wahl gelassen, welchem der beiden proprietären Spracherweiterungen er sich zuwenden wollte. Vielen Autoren war es (und ist es bis heute) einerlei, ob diese von ihm gewählte Spracherweiterung nun Sinn machte bzw. von der offiziellen Sprachdefinition des W3C gedeckt war. Hauptsache, man bediente ein anvisiertes Potential an möglichen Kunden, die voraussichtlich einen bestimmten Browser-Typ bevorzugten. Die Folgen sind noch heute spürbar: es existiert heute eine Vielzahl an Web-Dokumenten, welche erstens oft nur von der eingebauten Fehlerkorrektur des Web-Browsers zusammengehalten und zweitens oft nur in *einem bestimmten* Web-Browsertyp korrekt angezeigt werden.

Mit fortschreitenden Jahren haben sich diese Probleme verschlimmert. Diese fälschliche Praxis und auch das fehlende Gefühl für Korrektheit haben sich in einer großen Zahl von Entwickler- und Autoren-Köpfen eingebrannt (die verunsichernde Wirkung der „Browser-Kriegsjahre“ hat ihr übriges dazu beigesteuert). Großenteils ist es auch Bequemlichkeit, an dieser schlechten Angewohnheit etwas zu verändern.

Eine eingebaute automatische Syntax- und Qualitätsüberprüfung seitens der Software (Editor, Web-Browser) existiert in puncto HTML nicht – dies ist ein Manko gegenüber klassischen Programmiersprachen und -werkzeugen, welche jeden kleinsten Syntax-Fehler gnadenlos aufzeigen und eine erfolgreiche Compilation verweigern, sollte der Fehler nicht beseitigt sein.

Scriptsprachen wie JavaScript wurden damals und werden heutzutage teilweise immer noch eingesetzt, um Web-Dokumente, welche in verschiedenen Web-Browser- bzw. Betriebssystem-Umgebungen Probleme bereiten, per sog. „Browserweiche“, ggf. anzupassen.

Weil das eigene, grundlegende Verständnis des (übrigens schon immer vom W3C beabsichtigten und empfohlenen) Zusammenspiels der Strukturs prache HTML und der Stilsprache Cascading Stylesheets (CSS) von vielen Webentwicklern bzw. Webautoren oft nicht ausreicht, werden vorhandene, bisher praktizierte Kenntnisse teilweise falsch angewendet. Dann auftauchende Probleme werden erstmal der Unzulänglichkeit des Web-Browsers zugeschoben statt der eigenen fehlerhaften Umsetzung. Es werden teilweise Lösungen für möglicherweise selbst geschaffene Probleme (falsch verstandenes und angewendetes HTML) ersonnen und umgesetzt, statt nach der wirklichen Ursache des Problems zu suchen. Im schlimmsten Fall folgt ein Workaround für den Workaround des Workarounds... Welch eine Vergeudung von Zeit und Geld!

Dieser weitgehend sinnfreie Umgang mit der Brot- und Buttersprache des World Wide Web, HTML, hat sich bis heute in vielen Köpfen diverser Web-Autoren und -Entwicklern gehalten. An sie richten sich diese Ausführungen, um den Fokus auf intelligente Werkzeuge zu lenken, welche nicht nur eine qualitativ bessere Arbeit(sweise) ermöglichen, sondern darüberhinaus – richtig angewendet – sogar Nerven, Zeit und somit Geld sparen helfen.

Es ist i.A. aufwendiger, Webseiten nicht valide und nicht barrierefrei zu erstellen als von Beginn an die durchdachten Möglichkeiten von HTML und CSS auf richtige Art und Weise auszuschöpfen und umzusetzen! Valide, saubere Webdokumente schränken zudem mögliche Probleme im Zusammenspiel mit unterschiedlichen Web-Browsern auf ein Minimum ein. Es ist zu bedauern, dass gerade vielen professionellen Web-Entwicklern diese Tatsachen zu wenig bewusst sind.

2 Barrierefreiheit für Webinhalte von heute und morgen

In einer immer mehr vernetzten Welt mit seinen unterschiedlichen Ausprägungen und Entwicklungen ist es heutzutage weniger denn je vorhersehbar, wer wann mit welchem Endgerät Zugriff auf öffentlich bereitgestellte Web-Inhalte hat. Das ist auch gut so, denn das ist ganz im Sinne des Erfinders. Mobile Endgeräte mit verhältnismäßig wenig Speicher und kleineren Displays finden immer mehr Verbreitung. Das W3C trägt dieser Entwicklung Rechnung, indem es die dazu notwendigen Technologien plant und bereitstellt.

Um so wichtiger ist es, dass der Ersteller von Webinhalten seinerseits in dieser Richtung tätig ist und seine Webinhalte so anbietet, dass möglichst viele potentielle Nutzer/Kunden davon profitieren können. So etwas bezeichnet man allgemein als *Barrierefreiheit: k*ein Besucher/Kunde soll von vornherein ausgeschlossen oder wesentlich benachteiligt werden, um Webinhalte nutzen zu können. Eigentlich dürfte dieser Grundsatz sehr im Sinne aller Webinhalte-Anbieter sein – ganz besonders im Sinne derjenigen, welche über das Web Geld verdienen oder/und neue Kunden gewinnen wollen.

Valide, saubere Webinhalte sind im großen und ganzen per se weitgehend barrierefreier als nicht valide Webinhalte. Sie sind einem größeren Kreis zugänglich – nicht zu vergessen den blindesten und gehandicaptesten Nutzern: den Suchmaschinen (allen voran: Google) – eine marketingtechnisch viel beachtete Nutzergruppe, die im allgemeinen Sprachgebrauch und Denken so gar nicht in Verbindung gebracht wird mit blinden, tauben Maschinen oder Programmen, welche keine Augen zum Sehen, keine Arme zum Fühlen, keine Ohren zum Hören, und erst recht keine menschliche Intelligenz besitzen. Gerade diese Assoziation sollte man sich vor Augen führen, wenn man über Barrierefreiheit beginnt nachzudenken oder Nutzer mit Behinderung bisher sorglos nicht zum Kreis seiner potentieller Kunden zählt!

3 Was ist Markup-Validation?

Die meisten Seiten des World Wide Webs werden in Computersprachen geschrieben, die es Webautoren erlauben, Text zu strukturieren (z.B. mittels HTML), Multimediainhalt hinzuzufügen und anzugeben, welches Aussehen das Ergebnis haben sollte.

Wie bei jeder Sprache haben diese Seiten ihre eigene Grammatik, Vokabular und Syntax, und jedes mit diesen Computersprachen geschriebene Dokument sollte diesen Regeln folgen. Doch so, wie Texte in einer natürlichen Sprache Rechtschreibungs- oder Grammatik-Fehler aufweisen können, können Dokumente wel-

che Textauszeichnungssprachen (*engl. markup language*) wie HTML verwenden, ebenso fehlerbehaftet sein.

Das Überprüfen, ob ein Dokument wirklich den Regeln für die Sprache folgt, die es verwendet, wird Validation genannt, und das dafür verwendete Werkzeug ist ein Validator. Ein Dokument, das dieses Verfahren mit Erfolg passiert, wird gültig bzw. valide genannt.

Der online wie offline verfügbare Referenz-Validator für HTML- und XHTML-Dokumente ist der des World Wide Web Consortiums (W3C), der W3C Markup Validation Service [2].

4 Ist Validation eine Art Qualitätskontrolle?

Gültigkeit, auch Validität genannt, ist *eines* der Qualitätskriterien für eine Webseite, aber es gibt viele andere. Mit anderen Worten: eine *gültige* Webseite ist nicht zwangsläufig eine *gute* Webseite, aber eine *ungültige* Webseite hat eine *geringere Chance*, eine *gute* Webseite zu sein.

Die Tatsache, dass der W3C Markup Validation Service, eine Webseite als valide passieren lässt bedeutet nicht, dass das W3C befindet, es sei eine gute Seite. Es bedeutet lediglich, dass ein Werkzeug (welches per se selbst nicht fehlerfrei sein kann) herausgefunden hat, dass die Webseite einen spezifischen Satz von Regeln erfüllt. Nicht mehr und nicht weniger. Zweifelsfrei erhöht eine valide Webseite sehr die Chance, dass eine möglichst große Anzahl von Benutzer-Agenten (gemeint sind z.B. Web-Browser) diese korrekt darstellt.

5 Warum sollte ich meine HTML-Seiten validieren?

Eines der wichtigen Sprichworte der Computerprogrammierung ist:

> *„Sei konservativ in dem, was Du erzeugst; sei liberal in dem, was Du akzeptierst."*

Web-Browser folgen der zweiten Hälfte dieses Sprichwortes, indem sie Webseiten akzeptieren und versuchen sie anzuzeigen, selbst wenn sie in ungültigem HTML geschrieben sind. Gewöhnlich bedeutet das, dass der Web-Browser versuchen wird, definierte Annahmen darüber zu machen, was der Autor der Webseite wahrscheinlich gemeint hat. Das Problem besteht darin, dass unterschiedliche Web-Browser (oder gar unterschiedliche Versionen desselben Web-Browsers) verschiedene Annahmen über ein- und dasselbe inkorrekte Konstrukt machen; schlimmer noch: wenn das HTML des Dokuments wirklich pathologisch inkorrekt ist, könnte der Web-Browser hoffnungslos verwirrt werden und ein ziemliches Durcheinander produzieren oder sogar abstürzen.

Aus diesem Grund sollte man der ersten Hälfte des Sprichwortes folgen und sicherstellen, dass seine Webseiten gültiges HTML sind. Die beste Weise das zu tun ist, seine Dokumente durch einen HTML-Validator überprüfen zu lassen und bemängelte Fehler zu beseitigen.

6 Bedeutet Validation langweilige Websites, erstickt es Kreativität?

„Validation bedeutet langweilige Websites und erstickt Kreativität!" Aus diesem Satz sprechen Unerfahrenheit und Ignoranz (tatsächlich liegt einem Großteil spektakulärer Dot-Com-Misserfolge diese Tatsache zu Grunde). Validation ist sehr wohl vereinbar mit der breiten Auswahl an dynamischen Seiten, Multimedia-Präsentationen, Scripting, aktivem Inhalt usw.! Es ist ein Teil des Unterschieds zwischen richtigem und falschem Tun, egal ob in einer dynamischen Multimedia-Präsentation oder bei einer rein textlichen Seite.

Es ist vollkommen in Ordnung von Web-Entwicklern und Web-Autoren, wenn sie ihre Kreativität an angemessener Stelle und in passender Form im Web ausdrücken. Aber Menschen dieser Zunft mit kreativem Ehrgeiz sollten daran denken, dass jedes künstlerische Feld ein gründliches Verstehen der Regeln voraussetzt, bevor man beginnen kann, sie zu brechen. Ansonsten sieht man nämlich dumm aus.

7 Valides HTML ist Pflicht

Valides, wohlgeformtes HTML sollte keine Last sondern selbstverständliche Pflicht für einen Web-Entwickler und -Autor sein, damit aus technischer Sicht her gewährleistet ist, dass die zu veröffentlichen Web-Dokumente den darauf zugreifenden Benutzer-Agenten (z.B. Web-Browser, Suchmaschinen, Sprach-Browser, ...) möglichst keine Probleme bereiten.

Die u.U. eingebaute Fehler-Korrektur dieser Benutzer-Agenten sollte so wenig wie möglich in Anspruch genommen werden müssen — am Besten überhaupt nicht. Nur so ist gewährleistet, dass das Dokument am Ende so dargestellt wird, wie es der Autor vorgesehen hat. Syntaktisch korrekte Web-Dokumente werden zügiger dargestellt, eben weil *keine* interne Fehlerkorrektur eingreifen muss, welche für den Web-Browser immer auch vermeidbare Rechenzeit bedeutet. Der Nutzer/Kunde möchte schließlich schnell an seine Informationen gelangen. Zudem ist bei invaliden Web-Dokumenten die Gefahr hoch, dass sich der Web-Browser bei der Darstellung verhaspelt oder sogar instabil wird, weil einfach zu viele Fehler vorhanden sind, die ihn an die Grenzen seiner Leistungsfähigkeit treiben. Es ist sehr bedauerlich, dass das Web immer noch voll von derartig fragwürdigen Konstrukten ist. Für viele Menschen, die ihre Inhalte im Web publizieren, scheint es völlig egal zu sein, wie es um die technische Qualität ihrer veröffentlichten Web-Dokumente bestellt ist.

8 Tidy

Tidy (engl. s*auber, ordentlich*) ist ein kleines, handliches und sehr wirkungsvolles Werkzeug, welches dazu dient, HTML- und XHTML-Dokumente so zu prüfen und auf Wunsch automatisch aufzubereiten (zu „säubern“), dass sie am Ende einer Validation des W3C Markup Validation Service standhalten.

Tidy, ursprüngl. HTML Tidy [3], wurde von Dave Raggett, einem Hewlett-Packard-Mitarbeiter, welcher auch für das W3C arbeitet (u.a. Co-Autor von HTML 4.01), vor wenigen Jahren ins Leben gerufen. Weil das Projekt ihm mit der Zeit über den Kopf wuchs, hat er den Kreis erweitert und es als Open-Source-Projekt freigegeben. Das Tidy-Projekt [4] war geboren. Maßgeblich wird das Tidy-Projekt inzwischen von den Personen Terry Teague, Charles Reitzel und Björn Höhrmann in ganz enger Abstimmung mit dem W3C vorangetrieben und gesteuert. Einige Entwickler des Tidy-Projekts, wie z.B. Björn Höhrmann, arbeiten zugleich sehr engagiert auch an anderen W3C-Projekten mit, u.a. am W3C Markup Validation Service. Die Projekt-Teams von Tidy und des W3C-Validators arbeiten untereinander koordiniert und ergänzen sich gegenseitig.

Ursprünglich war Tidy ein alleinstehendes, monolithisches Programm. Mit der Zeit hat es sich jedoch gezeigt, dass diese Erscheinungsform die Integration in andere Software erschwert, weshalb man sich entschloss, Tidy als leichter zu integrierende Programm-Bibliothek zu entwicklen, die TidyLib. Sie bildet also das zentrale Herz aller Tidy-Erscheinungsformen.

Mit Tidy lassen sich (X)HTML-Dokumente aufräumen. Tidy erkennt in einem (X)HTML-Dokument fehlerhafte Syntax, falsche oder redundante Auszeichnungselemente und beseitigt auf Wunsch (je nach Konfiguration) die Fehler. Tidy kann erstaunlich viel und ist sehr konfigurierbar. Auf Wunsch konvertiert es HTML-Dokumente in XHTML-Dokumente. In HTML eingeschlossene Schrifteffekte und Farbangaben kann Tidy bei Bedarf (je nach Konfiguration) in entsprechende CSS-Angaben umwandeln und die unerwünschten bisherigen HTML-Elemente (z.B. das `<font>`-Element) entfernen.

Viele WYSIWYG-Editoren (Macromedia Dreamweaver und Adobe Golive sind die bekanntesten unter ihnen) produzieren heute immer noch teilweise unsauberes oder sogar falsches HTML bzw. CSS. Diesem bedauernswerten Umstand kann begegnet werden durch eine nachträgliche Korrektur der betreffenden Webseite(n) mittels Tidy.

Auch HTML-Dokumente, welche von der Textverarbeitung Microsoft Word erzeugt wurden (Microsoft Word produziert i.d.R. unnötig verkompliziertes HTML – eine Mischung aus unsauberem HTML und für die meisten Web-Browser nutzlosen Word-eigenen Auszeichnungselementen), können auf diese Weise in sehr saubere, valide Web-Dokumente überführt werden, deren Chance damit steigt, auch von anderen Web-Browsern als dem Internet Explorer korrekt dargestellt werden zu können.

Der Einsatz von Tidy ist nicht nur auf (X)HTML-Dokumente beschränkt, sondern bezieht – in beschränktem Umfang – auch ASP-, JSTE- und PHP-Dokumente mit ein.

Tidy ist sehr konservativ, was die Auslegung der Spezifikationen des W3Cs angeht. Diese (so auch bspw. die Spezifikation von HTML 4.01) enthalten Formulierungen, welche mehr oder weniger absichtlich ungenau gehalten sind, um Interpretationsspielraum für zukünftige Entwicklungen oder unklare Sachlagen zu schaffen. Tidy hält sich sehr genau an das, was als sicher spezifiziert ist und ist bzgl. des Interpretationsspielraums teilweise sehr restriktiv.

In Einzelfällen bemängelt Tidy Fehler, welche der W3C Validator toleriert bzw. als richtig ausweist. Das mag sich in Zukunft ändern, weil die Entwickler beider Projekte eng zusammenarbeiten und außerdem der Druck auf das W3C wächst, Klarheit in missverständliche Formulierungen der Spezifikationen zu bringen. Wahrscheinlich ist sogar, dass der W3C-Validator in naher Zukunft der konservativen Strenge von Tidy folgen wird.

9 Erscheinungsformen von Tidy bzw. TidyLib

Tidy bzw. TidyLib ist für viele Plattformen (Windows, Linux/Unix) in unterschiedlichen Ausführungen erhältlich. TidyLib, das eigentliche Herz des Programms Tidy, ermöglicht die Zusammenarbeit mit vielen Programmiersprachen, darunter C, C++, Python, etc. Mehr Information zu TidyLib sind unter [5] zu finden.

9.1 Tidy, das Programm

Die wohl verbreitetste und ursprünglichste Erscheinungsform von Tidy dürfte die als eigenständiges Programm sein. Einige Programme zur professionellen Erstellung von Webseiten, wie z.B. Macromedias HTML-Editor Homesite oder Uli Meybohms Phase5-Editor sind so vorbereitet, dass sie Tidy einbinden können. Manchmal wird auch eine Tidy-Version gleich kostenlos mitgeliefert. Allerdings ist sie nicht immer auf dem neuesten Stand, lässt sich jedoch ggf. leicht durch eine aktuelle Version, erhältlich auf der Projekt-Webseite, ersetzen. Einigen derzeitigen Linux-Distributionen liegt das Programm Tidy als Paket bei.

9.2 Tidy als Internet Explorer-Erweiterung

Gleich in Verbindung mit einer lokalen Offline-Version des W3C Markup Validators gibt es Tidy als Erweiterung für den Internet Explorer. Björn Höhrmann hat hier dankenswerterweise beide Werkzeuge gleich in eine Browser-Erweiterung gepackt: Tidy, um eine geladene Webseite zu säubern und eine offline benutzbare Version des W3C Markup Validators, um die Webseite anschließend offline validieren zu können. Leider funktioniert diese Kombination ausschließlich unter neueren Windows-Versionen, weil sie auf Windows-Bibliotheken zurückgreift. Die notwendigen Voraussetzungen sind auf der Projekt-Webseite [6] beschrieben.

9.3 Tidy als Mozilla-Erweiterung

Für Mozilla-basierte Web-Browser gibt es ebenfalls eine Browser-Erweiterung: Html Validator (based on Tidy). Strenggenommen dürfte sich diese Erweiterung nicht als Validator bezeichnen, denn Tidy ist kein Validator. Die Mozilla-Extension bietet an, die aktuell geladene Webseite Tidy vorzulegen, wenn man die Quelltext-Anzeige des Web-Browsers verwendet. Man kann diese Browser-Erweiterung unter [7] beziehen.

10 Automatisierter Einsatz von Tidy

Ein sehr sinnvoller Weg ist es, Tidy so einzusetzen, dass dieses Werkzeug dem Entwickler/Webautor möglichst wenig Aufwand abverlangt und ihm die Arbeit erleichtert statt sie zu verkomplizieren. Gerade bei größeren Web-Projekten mit vielen Webseiten kann der manuelle Einsatz von Tidy für jede einzelne Webseite relativ umständlich sein und wird wahrscheinlich deswegen von vielen Web-Entwicklern und Web-Autoren gemieden und als verhältnismäßig unwichtig angesehen. Automatisierte Lösungen, welche im Idealfall im Hintergrund werkeln, sind da gefragt.

Setzt man Tidy als Programm ein, scheint auf den ersten Blick die Einschränkung zu bestehen, dass es immer nur ein HTML-Dokument zur Zeit bearbeiten kann. Hat man viele Dokumente, wie dies in größeren Web-Projekten die Regel ist, welche man von Tidy prüfen/bearbeiten lassen möchte, so kann das schnell sehr mühevoll und unproduktiv werden, wenn jedes Dokument einzeln Tidy zur Bearbeitung vorgelegt werden muss. Doch keine Sorge, auch dafür gibt es Möglichkeiten – auf der Kommandozeile unter Windows bzw. der Shell unter Linux/Unix können bspw. folgende Anweisungen helfen:

```
a) tidy *.html
b) tidy 1.html 2.html 3.html
c) for %a in (*.html) do tidy %a
```

Doch es geht noch viel einfacher (und produktiver)! Viele Web-Entwickler nutzen für ihre Projekte einen lokalen Entwicklungs-Webserver, auf dem die Projekte gespiegelt liegen und auf dem gearbeitet wird. Idealerweise spiegelt er entweder die Umgebung bzw. die Bedingungen der späteren Server-Umgebung für die jeweiligen Web-Projekte wider, oder er ist wenigstens so eingerichtet, dass eine Portierung auf ein späteres Live-System, welches über das World Wide Web erreichbar ist, mit möglichst wenig Aufwand machbar ist.

Zwei herausragende, bereits existierende Möglichkeiten bieten sich für solche Umgebungen an:

- Tidy als PHP-Erweiterung
- Tidy als Webserver-Erweiterung (mod_tidy)

In beiden Fällen durchläuft der (X)HTML-Output automatisch zuerst die Tidy-Bibliothek und wird von ihr geprüft bzw. auf Wunsch überarbeitet, bevor das Web-Dokument im Web-Browser dargestellt wird. Empfehlenswert ist es, Tidy so zu konfigurieren, dass es zwar gemachte Fehler aufzeigt, ein manuelles Beheben derselben jedoch vom Entwickler/Autor noch vorgenommen werden muss. Auf diese Weise hat er noch die volle Kontrolle über seinen Code.

Im Web-Browser werden dann auf diese Weise Dokumente angezeigt, welche valide sind und mühelos einer Validitätsprüfung des W3C Markup Validation Service standhalten können. So benötigt man — vor allem in der Entwicklungsphase — für Webseiten bzw. Web-Projekte keine separate Applikation mehr (z.B. Tidy als Programm), um deren Syntax zu überprüfen. Dieses ist eine große Arbeitserleichterung für den Autor und Entwickler und unterstützt somit ein zügiges und kostensparendes Erstellen fehlerfreier Webseiten.

Tidy in einer solchen Umgebung so zu konfigurieren, dass Fehler zwar aufgezeigt, jedoch von Hand korrigiert werden müssen, birgt abgesehen von der größeren Kontrolle über den Code noch einen weiteren, in meinen Augen unschätzbaren Vorteil in sich: der Entwickler/Autor wird gezwungen, sich mit seinen Fehlern auseinanderzusetzen. Der Lerneffekt ist höher, auf längere Sicht produziert der Entwickler/Autor besseren Code.

Da Tidy als PHP-Erweiterung nur in einem PHP-Framework funktionieren kann, halte ich den Einsatz als Apache2-Modul nutzbringender, weil auf diese Weise theoretisch jede Art von Web-Dokument automatisch überprüft werden kann und der Einsatz von Tidy nicht auf PHP-Seiten beschränkt bleibt. Wie immer entscheiden hier der persönliche Geschmack und die Bedürfnisse des Entwicklers bzw. Web-Autors. :-)

Im Folgenden möchte ich mich deshalb vorzugsweise dem Einsatz von mod_tidy in Verbindung mit dem Apache2-Webserver widmen.

10.1 Tidy als PHP-Erweiterung

Dank John Coggeshall existiert Tidy auch als PECL-Erweiterung sowohl für PHP 4.3+ als auch für PHP 5. Mehr Info dazu unter [8] und [9]. Unter [10] gibt es die Erweiterung zum Download. Voraussetzung, damit die PHP-Erweiterung funktioniert ist es, dass TidyLib (libtidy) installiert ist.

10.2 Tidy als Apache2-Modul mod_tidy

mod_tidy ist dank Sebastian Tusk ein Modul für den Apache2-Webserver, welches dessen (X)HTML-Ausgabe automatisch prüft bzw. ggf. bearbeitet, bevor sie im Web-Browser dargestellt wird, so dass sie am Ende einer Validitätsprüfung des W3C Markup Validation Service standhalten kann.

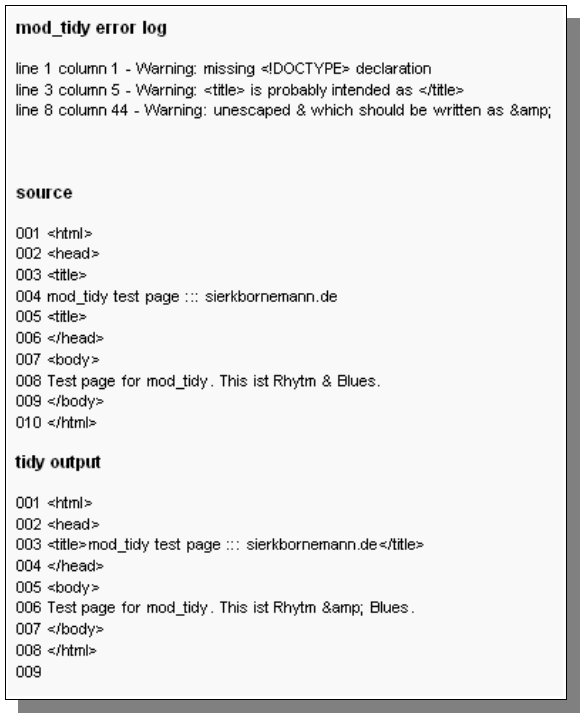
Eine Zeit lang stagnierte die Weiterentwicklung von mod_tidy, weil sich Sebastian nicht mehr darum kümmern konnte oder wollte. Als Folge ist seit wenigen Monaten seine ursprüngliche Projekt-Webseite nicht mehr erreichbar. Weil das mod_tidy-Projekt zu verschwinden drohte und um es weiterzuführen, habe ich es im Sommer 2005 in Absprache mit Sebastian Tusk offiziell übernommen. Ich habe die Sourcen aktualisiert, einer Open-Source-Lizenz unterstellt und mod_tidy unter dem Dach des Open-Source-Sponsors Sourceforge eine neue Heimat gegeben. Die neue Projekt-Webseite ist unter [11] zu finden.

Es gibt Tidy auch als Modul für den Apache-Webserver der 1.x-Generation. Die Entwicklung dieses Projekts scheint jedoch eingestellt zu sein, jedenfalls ist seit Jahren keine aktuelle Version mehr veröffentlicht worden. Die Projekt-Webseite verweist auch explizit auf die Existenz von mod_tidy für den Apache2-Webserver und empfiehlt den Einsatz dieser Kombination. Deshalb werde ich hier nicht näher darauf eingehen.

10.3 Wie arbeitet mod_tidy?

Das Apache-Modul mod_tidy arbeitet als Filter zwischen Webserver und (X)HTML Ausgabe.

Die ursprüngl. Ausgabe von (X)HTML leitet es weiter zu der Bibliothek TidyLib, welche die eigentliche Arbeit verrichtet und die eingereichte Webseite prüft und ggf. überarbeitet (letzteres nur, wenn per Konfigurationsoption eingeschaltet; die Standard-Einstellung ist, dass mod_tidy keine Korrekturen an der Original-Webseite unternimmt, sondern lediglich Fehler und Warnungen ausspricht). Der Entwickler/Anwender merkt außer einer leichten zeitlichen Verzögerung nichts von alledem. Findet TidyLib Fehler, dann sendet sie (in der voreingestellten Standard-Konfiguration) dem anfragenden Client (i.d.R. ein Web-Browser) statt der ursprüngl. Webseite eine Liste aller gefundenen Fehler nebst zugehöriger Zeilennummer (siehe nebenstehender Screenshot). Gibt es dagegen für TidyLib nichts zu beanstanden, dann leitet sie die Webseite wie gewohnt weiter an den anfragenden Client/Browser.



mod_tidy eignet sich deshalb ideal dafür, von Anfang an eines Web-Projektes valide Webseiten zu erzeugen, welche einer Prüfung durch den Referenz-Validator W3C Markup Validation Service standhalten.

Hinweis: Wegen der durch das Filtern und Parsen des Ausgabestroms des Webservers auftretenden zeitlichen Verzögerung sollte mod_tidy nicht auf Webservern eingesetzt werden, welche für den direkten Zugriff aus dem World Wide Web gedacht sind. Der Einsatz von mod_tidy sollte auf lokale Entwicklungsumgebungen beschränkt sein, bei denen leicht erhöhte Antwortzeiten des Webservers vertretbar sind.

10.4 Installation von mod_tidy

Für diverse Linux-Distributionen und auch BSD-Unixe existieren bereits fertige mod_tidy-Pakete:

- **BSD Unix:** mod_tidy ist in einer Ports-Collektion vorhanden.
- **Debian GNU/Linux:** Ein fertiges Debian Binär-Paket konnte ich im WWW nicht finden, ich bin jedoch optimistisch, dass ein solches irgendwann existiert.
- **Mandrake Linux:** hier empfehle ich die Download-Quellen [12] und [13].
- **Red Hat Fedora Linux:** dank Ville Skyttä gibt es unter [14] entsprechende RPMs und SRPMs.
- **SuSE Linux:** Seit kurzem existieren für SuSEs aktuelle Linux-Distributionen dank meines Zutuns unter [15] und [16] ebenfalls passende RPM- und SRPM-Pakete.

Für andere Unixe/Linux-Distributionen gibt es die Möglichkeit, mod_tidy aus dem Tarball zu erstellen.

10.4.1 Technische Voraussetzungen

- Apache 2.0.47 oder höher
- Apache2 Header- und Include-Dateien — bei SuSE Linux befinden sich diese im Paket apache2-devel
- Apache2-Modul mod_tidy

10.4.2 Vorbereitungen

Die hier gemachten Programm- und Pfadangaben beziehen sich auf SuSE Linux ab Version 9.x und den Tarball von der mod_tidy-Projektseite. Andere Unix/Linux-Systeme können teilweise abweichende Bezeichnungen und Pfadangaben benötigen. Obwohl es seit kurzem offizielle SuSE-Pakete gibt, sei im Folgenden die traditionelle Methode beschrieben, mod_tidy manuell aus den Sourcen zu erstellen. Sie ist unter jedem unixartigen Betriebssystem anwendbar.

1. Besorgen Sie sie sich die aktuelle Version von mod_tidy (mind. Version 0.5).
2. Kopieren Sie diese Datei in ein temporäres Verzeichnis (z.b. nach `/tmp`) oder nach `/usr/local/src/` (gemäß LSB bzw. FHS dürfte dieser Ort korrekter sein)
3. Wechseln Sie auf der Kommandozeile in dieses Verzeichnis.

10.4.3 Bau und Installation

Ggf. muss das `Makefile` von Tidy angepasst werden, welches (bis einschließlich mod_tidy Version 0.5) in einem Unterverzeichnis der entpackten mod_tidy-Sourcen liegt (z.B. `mod_tidy-0.5/tidy/build/gmake/Makefile`): die dort zu Beginn der Datei angegebenen Pfade für die Installation bzw. für die einzubindenen Header sind evtl. andere als die der Original Source-Datei. Sie sind ggf. anzupassen.

Um mod_tidy erfolgreich bauen zu können, ist es notwendig zu wissen, welchen absoluten Pfad das Programm apxs (Bestandteil des Pakets apache2-devel) hat. Unter SuSE Linux ist das z.B. `/usr/sbin/apxs2`.

Sie sollten sich nun in dem Verzeichnis befinden, in welchem Sie den mod_tidy-Download abgelegt haben. Geben Sie auf der Kommandozeile nun folgende Anweisungen:

```
$ tar xfz mod_tidy-0.5.tar.gz
$ cd mod_tidy-0.5
$ ./configure --with-apxs=/usr/sbin/apxs2
$ make
$ su root
# make install
```

Auf diese Weise wird mod_tidy kompiliert und im Apache2 Modul-Verzeichnis installiert.

Hinweis: Seit Version 0.3 enthält mod_tidy die TidyLib-Quellen. Während die in einem separaten Projekt entwickelte TidyLib-Bibliothek durch ihre Entwickler regelmäßige Updates erfährt, ließ die Aktualität der mod_tidy beigefügten TidyLib zuletzt zu wünschen übrig. Mit der Übernahme des mod_tidy-Projekts durch mich hat sich das geändert: ich bin stets bemüht, die mit mod_tidy mitgelieferten TidyLib-Bibliotheken auf dem aktuellen Stand zu halten. Für die Zukunft ist geplant, diese Abhängigkeit zugunsten einer größeren Unabhängigkeit vom Entwicklungsstand der TidyLib zu entkoppeln, gegen die TidyLib also dynamisch zu linken statt sie fest einzubinden. Das setzt natürlich voraus, dass die TidyLib im System schon vorhanden ist.

Wer zum jetzigen Zeitpunkt schon jetzt eine andere Version von TidyLib verwenden will als die im Tarball von mod_tidy enthaltene, der sollte eine solche separat installieren und die Datei `Makefile.in` von Hand entsprechend so abändern, dass statt gegen die migelieferte TidyLib gegen die separat installierte TidyLib gelinkt wird.

Die RPM-Pakete für Red Hat Fedora Linux und SuSE Linux setzen dieses z.B. schon um (durch entspr. Patches). Für die Zukunft ist dies auch für die Sourcen von mod_tidy geplant, sodass dieser Schritt seitens des Distributors nicht mehr notwendig ist. Zum jetzigen Zeitpunkt (Anfang September 2005) ist dies von mir jedoch noch nicht fertig umgesetzt. Nicht nur deshalb lohnt es sich, bei Interesse die Projektseite von mod_tidy nach entsprechenden Updates im Auge zu behalten.

10.5 Konfiguration von mod_tidy

Jetzt geht's an die Konfiguration des Webservers. Dazu muss die Datei `httpd.conf` ein wenig geändert werden (die Datei `httpd.conf` liegt bei SuSE Linux im Verzeichnis `/etc/apache2` und ist mitunter, je nach Aufbau und Konfiguration des Webservers, in mehrere inkludierbare Dateien aufgeteilt).

10.5.1 httpd.conf

Im Abschnitt der beim Start des Webservers zu ladenden Module sollte folgende Direktive hinzugefügt werden (hier am Beispiel unter SuSE Linux):

```
LoadModule tidy_module /usr/lib/apache2/mod_tidy.so
```

Desweiteren muss folgende Direktive der Webserver-Konfiguration hinzugefügt werden (empfehlenswert ist es hier, die folgenden Angaben im entspr. `<VirtualHost>`-Abschnitt zu machen, wenn vorhanden):

```
<IfModule mod_tidy.c>
  AddOutputFilterByType TIDY text/html application/xhtml+xml
  TidyOption char-encoding utf8
  TidyOption indent auto
  TidyOption indent-spaces 4
  TidyOption tab-size 4
  TidyOption wrap 0
</IfModule>
```

Auf diese Weise lädt der Apache2-Webserver das Modul mod_tidy gleich beim Start und schickt jede Webseite durch den Tidy-Filter, bevor er sie ausliefert. Als MIME-Typ der von Tidy zurückgegebenen Seiten ist der empfohlene MIME-Typ für XHTML-Dokumente, `application/xhtml+xml`, voreingestellt. Das bezieht sich vor allem auf die im Bedarfsfall von mod_tidy generierten Fehlerseiten. Die zu prüfende Webseite wird mit dem MIME-Typ ausgeliefert, den der Entwickler bzw. Autor der Webseite vorgesehen hat.

10.5.2 YaST

Wird unter Linux (SuSE Linux) das Administrationswerkzeug YaST eingesetzt, so sollte es für die vorliegende Aufgabe verwendet werden, da es im Regelfall auch die Apache-Konfigurationsdateien verwaltet. Hier fügt man einfach im entspr. Konfigurations-Abschnitt der Variablen APACHE_MODULES den Wert „`tidy`“ als weiteres zu ladendes Modul hinzu. Anschließend sollte YaST beendet oder auf der Shell

```
SuSEconfig --module apache2
```

aufgerufen werden, falls `/etc/sysconfig/apache2` zuvor von Hand edidiert worden ist. YaST durchläuft an dieser Stelle seine Konfigurations-Skripte und erstellt in unserem Falle selbständig die notwendigen Einträge in der Apache2-Konfiguration im Verzeichnis `/etc/apache2/`.

11 Tidy Konfigurations-Optionen

Tidy ist sehr konfigurierbar und hält eine Menge an Optionen bereit. Im WWW sind sie unter [17] übersichtlich aufgelistet und erklärt.

Wird Tidy auf der Kommandozeile bzw. der Shell verwendet, so kann man sich alle Konfigurationsoptionen mit

```
tidy -help
```

anzeigen lassen. Konfigurationsoptionen können dem Kommando direkt mitgegeben werden. Z.B. wird mit

```
tidy --output-xhtml true --doctype strict test.html
```

ein fiktives HTML-Dokument in ein XHTML-Dokument mit einem Strict-Dokumententyp umgewandelt.

Auch lassen sich mehrere Konfigurationsoptionen in eine Datei (z.B. `config.txt`) schreiben, die dann der Einfachheit halber referenziert wird, wenn die Beispiel-Datei `test.html` bearbeitet wird:

```
tidy -config config.txt test.html
```

Eine mögliche, auf diese Weise von Hand erstellte Konfigurations-Datei könnte bspw. so aussehen:

```
// Beispiel-Konfiguration config.txt
indent: auto
indent-spaces: 4
wrap: 80
output-xml: yes
output-xhtml: true
doctype: strict
show-warnings: yes
numeric-entities: yes
quote-marks: yes
quote-nbsp: yes
quote-ampersand: no
break-before-br: no
uppercase-tags: no
uppercase-attributes: no
char-encoding: utf8
```

12 Konfigurations-Optionen in der `httpd.conf`-Datei

Um mod_tidy zu konfigurieren, gibt es die `<TidyOption>`-Direktive. Sie dient dazu, die schon erwähnten Tidy Konfigurations-Optionen umzusetzen.

Um z.B. den Zeichensatz (Character Encoding) des Tidy-Outputs auf Unicode (UTF-8) zu setzen, ergänzt man in der `httpd.conf`-Datei die gemachten Angaben für mod_tidy um

```
TidyOption char-encoding utf8
```

Mit entsprechenden Konfigurationsoptionen lässt sich außerdem auch ein anderes Verhalten beim Parsen bestimmen, z.B. die automatische Korrektur von Fehlern. Die automatische Korrektur sollte jedoch zugunsten der manuellen Korrektur nicht verwendet werden, damit eine bessere Kontrolle durch den Autor bzw. Entwickler möglich ist. Außerdem kann diese Einstellung eine zügige Darstellung der betreffenden Webseite verlangsamen, weil bis zu 2 Dateien abgespeichert werden: die Original-Datei und die korrigierte Datei, bevor diese im Browser dargestellt wird.

Weitere Tidy-Optionen sind in der Dokumentation von Tidy zu finden. In der Standardeinstellung sind bei mod_tidy recht vernünftige Optionen voreingestellt, sodass bei durchschnittlichem Einsatz kaum die Notwendigkeit besteht, wesentlich an der Konfigurations-Schraube zu drehen. Obige, als Beispiel genannte Konfigurationsmöglichkeit könnte in neueren Web-Dokumenten jedoch notwendig sein, wenn deren Zeichensatz UTF-8-kodiert sein soll.

13 Tidy auf der Kommandozeile/Shell

Die vielfältigen Einsatz-Möglichkeiten von Tidy im Detail näher zu beschreiben, würde den Rahmen sprengen. Tidy kann entweder auf der Kommandozeile (Windows) bzw. Shell (Linux/Unix) eingesetzt werden oder auch eingebettet in eine GUI. Das bisher populärste Projekt, Tidy eine eigenständige GUI zu verpassen, ist leider eingestellt. Doch das ist nicht so schlimm, denn wenn andere, GUI-basierte Programme den Einsatz von Tidy vorsehen, so bieten sie bisher alle eigenständige Bedienelemente an, um Tidy zu konfigurieren und zu steuern.

14 ToDo-Liste und Wünsche der Tidy-Entwickler

Die ToDo-Liste des Tidy-Projects hält einige größere Aufgaben bereit, und die Bugliste von Tidy auf [18] ist mindestens ebenso umfangreich. Tidy ist nicht perfekt, ebensowenig wie der W3C Validator es ist. Jedoch vollbringen beide Werkzeuge trotz kleinerer Unzulänglichkeiten Erstaunliches.

Wichtige Punkte auf der ToDo-Liste für TidyLib sind u.a.:

- breitere Internationalisierung, um Dokumente in anderen Zeichenkodierungen (z.B. aus Russland, Asien) besser verarbeiten zu können
- verbessertes Nachrichtensystem für verständlichere Warnungs- und Fehlermeldungen (gleiches Vorhaben auch beim W3C-Validator); Kategorisierung von Fehlermeldungen
- Unterstüzung für neue Markup-Sprachen wie z.B. XHTML 2.0
- Plugin-System, um Tidy leicht um zusätzliche Funktionalität erweitern zu können

15 Fazit

Tidy in seinen verschiedenen Erscheinungsformen ist ein äußerst nützliches Werkzeug bei der Erstellung von sauberen, standardkonformen und damit webtauglichen Dokumenten. Validator wie auch Tidy sollten – egal in welcher Erscheinungsform – zum alltäglichen Werkzeug eines jeden Web-Entwicklers oder Web-Autors gehören, ganz gleich, ob Anfänger oder routinierter Profi.

Die Vorteile und die Nützlichkeit von Tidy liegen auf der Hand. Der regelmäßige, routinierte Einsatz erleichtert dem Ersteller von Web-Dokumenten die Arbeit enorm und schult gleichzeitig seine Fähigkeiten, sauberes (X)HTML zu schreiben bzw. zu denken.

Ein Web-Autor, der lediglich ab und zu einzelne Webseiten erstellt, ist mit Tidy als Standalone-Programm oder der Version als Browser-Erweiterung am besten bedient. Für solche Fälle lohnt sich der Aufbau eines Webserver-Frameworks meistens nicht bzw. er dürfte damit überfordert sein.

Für Web-Autoren und -Entwickler, die regelmäßig größere Web-Projekte zu bearbeiten haben, sind die Eigenschaften der Tidy PHP-Erweiterung und vor allem die des Apache2-Moduls mod_tidy bestechend interessant und sicher sehr hilfreich im Alltag.

16 Über den Autor

Sierk Bornemann ist Jahrgang 1970 und arbeitet seit Mitte der 80er Jahre mit Computern.

Nach der Schulzeit und dem Abitur diente er zwei Jahre lang der Deutschen Bundesmarine und hat in dieser Zeit erfolgreich die Ausbildung zum Reserveoffizier gemacht.

Anschließend folgten einige Semester ein Studium der Elektrotechnik an der Universität Hannover.

Seitdem arbeitet er haupt- und freiberuflich im IT-Umfeld als Web-Entwickler und -Autor. Seit mehreren Jahren beschäftigt er sich im Allgemeinen wie auch im Speziellen mit verschiedenen Web-Standards und tritt für deren Um- und Durchsetzung ein.

Kontakt:
Sierk Bornemann
Website: http://sierkbornemann.de
email: mail@sierkbornemann.de

17 Literatur und Download-Adressen

[1] World Wide Web Consortium (W3C): www.w3.org
[2] W3C Markup Validation Service: validator.w3.org
[3] Dave Ragget's Tidy Page www.w3.org/People/Raggett/tidy
[4] Tidy Project Page: tidy.sourceforge.net
[5] Introduction to TidyLib: tidy.sourceforge.net/libintro.html
[6] Tidy und W3-Validator als Erweiterung für den Internet Explorer: ieqabar.sourceforge.net/
[7] Tidy als Erweiterung für Mozilla-Browser: users.skynet.be/mgueury/mozilla/
[8] Tidy PHP Extension: www.coggeshall.org/oss/tidy/index.php/4/
[9] PHP- Tidy Functions - Manual: de.php.net/tidy
[10] PECL – Package – tidy: pecl.php.net/package/tidy
[11] mod_tidy Project Page: mod-tidy.sourceforge.net
[12] RPM-Repository RPMfind.net: rpmfind.net
[13] RPM-Repository PBone.net: rpm.pbone.net
[14] Fedora Downloads: download.fedora.redhat.com/pub/fedora/linux/extras/3/RPMS/
[15] SuSEs Tidy: ftp.suse.com/pub/projects/apache/tidy/
[16] SuSEs mod_tidy: ftp.suse.com/pub/projects/apache/mod_tidy/
[17] HTML Tidy Configuration Options Quick Reference: tidy.sourceforge.net/docs/quickref.html
[18] Tidy Bugliste: sourceforge.net/tracker/?group_id=27659&atid=390963
[19] Wolfgarten, Sebastian; Apache Webserver 2.0. Installation, Konfiguration, Programmierung., 2. Auflage 2003; Addison Wesley Verlag; Kapitel 10.3.1 Dynamische Syntaxüberprüfung von HTML-Dateien mit mod_tidy