HTML und XHTML

Stand: 29.11.2022

HTML und XHTML

Vielleicht haben Sie bereits von **XHTML** gehört und kennen Aussage wie die, dass man kein HTML mehr verwenden solle, sondern nur noch XHTML. In diesem Zusammenhang sollten Sie ein paar grundsätzliche Hintergründe kennen.

SGML und XML

HTML ist ein wohldefinierter Standard. Die Regeln für korrektes HTML werden jedoch nicht mithilfe natürlicher Sprache, sondern in einer ebenfalls computerlesbaren Meta-Sprache namens SGML (Structured Generalized Markup Language) formuliert.
Die Computerlesbarkeit ist wichtig, damit z.B. ein Programm eine in HTML geschriebene Webseite gegen die in SGML formulierten HTML-Regeln auf syntaktische Korrektheit testen kann. Diesen Vorgang nennt man *Validierung* (Gültigkeitsüberprüfung).
SGML ist sehr leistungsfähig, aber auch recht komplex und für viele praktische Anwendungsfälle überdimensioniert. Als praxisnähere Alternative zu SGML hat sich mittlerweile XML (Extensible Markup Language) etabliert. XML ist wie SGML eine computerlesbare
Meta-Sprache zum Formulieren der Regeln von Markup-Sprachen wie HTML.

> XHTML (Version 1.0) ist zunächst einmal nichts anderes als eine Neuformulierung der Sprachregeln von HTML

(Version 4.01) in XML. Mit XHTML 1.0 können Sie also nicht mehr und nicht weniger und nichts anderes tun als mit der aktuellen HTML-Version 4.01: nämlich die Inhalte Ihrer Webseiten strukturieren. Es gibt in diesem Stadium von XHTML auch keine wesentlich
anderen Sprachbestandteile als in HTML. Da allerdings die Markup-Regeln von XML etwas anders aussehen als die von SGML, gibt es einige syntaktische Abweichungen und Besonderheiten.

Interpretation von HTML und XHTML

Ein **Parser** leistet die Verarbeitung von Markup-Sprachen entsprechend der Regeln der Markup-Sprache. Jeder Browser, der eine Webseite am Bildschirm anzeigt, muss deren HTML-Quelltext parsen. Aus dem Markup geht hervor, aus welchen Elementen
die Webseite besteht und welche Elemente innerhalb welcher anderer Elemente vorkommen. So kann beispielsweise ein einzelnes Wort wie „Mensch“ Inhalt einer Kopfzelle einer Tabelle sein, die sich in einem bestimmten Bereich innerhalb des Dateikörpers befindet.
Zum Verarbeiten von HTML verfügt jeder Browser über einen *HTML-Parser*. Bei XHTML ist die Angelegenheit etwas komplizierter: Wenn das HTML-Dokument vom Browser als HTML akzeptiert wird, verwendet der Browser seinen HTML-Parser. Wird es jedoch
als XML-Dokument akzeptiert, so wird der XML-Parser verwendet, sofern der Browser über einen

entsprechenden Parser verfügt. In der Darstellung kann dies erhebliche Auswirkungen haben. So stellen einige Browser XML-geparste Dokumente nur als Markup-Strukturbaum
dar, also letztlich als eine sauber formatierte Quelltextansicht. Andere bieten XML- geparste Dokumente möglicherweise nur zum Download an und stellen gar nichts dar.

> Ob ein Browser bei einem XHTML-Dokument den HTML-Parser oder den XML-Parser verwendet,

hängt in erster Linie vom zugewiesenen **Mime-Type** ab. Der Mime-Type legt den Datentyp fest. Jeder Browser hat lokal eine eigene Liste von Mime-Typen gespeichert, mit deren Hilfe er z.B. Dateiendungen und Datentypen zuordnen kann. Beim
MS Internet Explorer ist diese Liste eng verzahnt mit den Dateiendungsverknüpfungen des Betriebssystems. Wird eine beliebige Datei im Browser lokal geöffnet, kann dieser an Hand seiner Mime-Type-Liste entscheiden, wie er mit der Datei verfahren soll.
Anders sieht es aus, wenn der Browser Daten von einem Webserver anfordert und empfängt. In diesem Fall übermittelt der Webserver Kopfdaten an den Browser, in denen unter anderem eine Angabe zum Mime-Type der übertragenen Daten stehen kann, die dem Browser
zur Orientierung dient. Für HTML-Dokumente lautet der Mime-Type `text/html`. Für XHTML sollte jedoch laut Spezifikation nicht dieser Mime-Type verwendet werden, sondern `application/xhtml+xml`. Alternativ dazu sind auch die Angaben
`text/xml` und `application/xml` möglich. Dies führt jedoch je nach Server- und Browser-Einstellungen zu Problemen. Der immer noch am weitesten verbreitete MS Internet Explorer kann mit dem Mime-Type `application/xhtml+xml`sogar bei angepasster Server-Konfiguration nichts anfangen und bietet ein Dokument mit diesem Mime-Type zum Download an. Daher wird in der Praxis doch noch der Mime-Type `text/html` verwendet. Wenn Sie sichergehen wollen, dass Ihre XHTML-Seiten
als Webseiten im Browser erscheinen, dann weisen Sie statischen XHTML-Dateien am besten ebenso wie HTML-Dateien die Dateiendung **.htm** oder **.html** zu.

Argumente für XHTML

Bei neu zu erstellenden Webseiten sprechen durchaus einige Gründe dafür, von vorneher- ein XHTML statt „herkömmlichem“ HTML zu verwenden. Diese Gründe hängen letztlich alle mit der Integration von XHTML in der XML-Welt zusammen:

- **Kombination mit anderen XML-Sprachen**: In einem XHTML-Dokument können Sie beispielsweise direkt mathematisch/naturwissenschaftliche Formeln in MathML oder Vektorgrafiken in SVG notieren. Der Grund ist, dass in XML-basierende Dokumente,
  zu denen XHTML-Dokumente ja gehören, Daten aus beliebigen anderen XML-basierenden Sprachen eingebettet werden können.
- **Konvertierung in andere XML-Sprachen oder Ausgabeformate**: Mithilfe von XSLT, einer standardisierten, XML-basierten Markup-Sprache zum Übertragen von Inhalten aus XML-basierten Sprachen in andere XML-basierten Sprachen oder beliebige
  andere Ausgabeformate ist XHTML optimal für die Konvertierung in andere Ausgabeformate gerüstet. Dies ist beispielsweise für Inhalte sinnvoll, die nicht nur als Webseite, sondern z.B. auch als Print-Dokument veröffentlicht werden sollen.

- **Zukunftssicherheit**: Die Spezifikation zu XHTML wird in jedem Fall weiterentwickelt. So ist XHTML 2.0 bereits in Entwicklung. Eine parallele Weiterentwicklung von herkömmlichem HTML ist dagegen nicht vorgesehen. Zwar ist XHTML 2.0
  für die aktuelle Praxis noch irrelevant, doch kann sich dies in wenigen Jahren ändern. Eine Konvertierung von XHTML-1.0-Dokumenten nach XHTML 2.0 ist vermutlich unproblematischer zu bewerkstelligen als eine Konvertierung von herkömmlichem HTML
  4.01 auf XHTML 2.0.

Andererseits ist sauberes HTML auch nicht zwangsläufig schlechter konvertierbar als sauberes XHTML. Die Einbettung anderer XML-Formate ist ebenfalls nur schöne Theorie, die in den heutigen Browsern höchstens ansatzweise funktioniert. So betrachtet spricht
aus Basiswissen HTML und CSS praktischer Sicht nichts gegen die Verwendung von HTML. Die Quelltexte auf dieser Webseite verwenden ebenfalls „gewöhnliches“ HTML.

Modularisierung von XHTML

HTML 4.0 und XHTML 1.0 sind – auch wenn sie von unwissenden Schwätzern gerne wegen ihrer Schlichtheit belächelt werden – eigentlich sehr umfangreiche Sprachen mit einem nicht ganz einfachen Regelwerk aus über 80 Elementen, ihren möglichen Attributen,
Verschachtelungsregeln und einer ganzen Reihe benannter Zeichen. Beide Sprachen bieten damit ein brauchbares Arsenal für Hypertext-Dokumente an. Egal ob einfache Porträt-Homepages oder wissenschaftliche Abhandlungen – der Strukturvorrat von *HTML/XHTML* ist durchaus reichhaltig und fur die meisten Zwecke ausreichend. Allerdings nicht für alle Zwecke.

Der Umfang beider Sprachen und ihre nicht ganz trivialen Regeln stellen andererseits hohe Anforderungen an die auslesende Software und deren Darstellungsfähigkeiten. Von einem modernen Webbrowser, der auf einem modernen PC läuft, kann man diese Fähigkeiten
verlangen. Aber von Software, die beispielsweise in einem Handy einem Pocket-Computer oder einem simplen Sprachwiedergabegerät läuft, kann man nicht unbedingt die gleichen Fähigkeiten erwarten.

> HTML/XHTML decken also einerseits trotz ihres beträchtlichen Sprachumfangs nicht alle denkbaren Wünsche ab,

und andererseits sind sie fur manche Zwecke schlichtweg überdimensioniert. Aus diesem Grund gehen die überlegungen des W3-Konsortiums in die Richtung, zumindest bei XHTML zu ermöglichen, dass sowohl abgespecktere als auch erweiterte Sprachvarianten davon
möglich sind. Handy-Software könnte dann beispielsweise eine abgespecktere Sprachvariante interpretieren, während ein Mathematiker, der seine Arbeit publizieren möchte, XHTML um Elemente aus einer speziellen Auszeichnungssprache für mathematische Formeln
erweitern kann. Das Konzept zum Abspecken und Erweitern von XHTML wird als **Modularisierung** bezeichnet. Dass es nur bei XHTML und nicht bei HTML angewendet werden soll, liegt daran, dass für die Modularisierung XML-typische Techniken angewendet
werden sollen. Die Modularisierung stellt nämlich bestimmte formale Regein zur Verfügung, die abgespeckte oder erweiterte Varianten von XHTML einhalten müssen.

Das W3-Konsortium bietet Beschreibungen an, wie Sie zum Erstellen regelkonformer eigener Module vorgehen müssen. über die W3-Einstiegsseite zu HTML/XHTML können Sie entsprechende aktuelle Beschreibungen aufrufen. Um die Beschreibungen zu verstehen, benötigen Sie allerdings gute Kenntnisse im Erstellen eigener DTDs.Neben der Möglichkeit, XHTML für eigene Zwecke zu modularisieren, macht das *W3-Konsortium* aber auch selber von dieser Möglichkeit Gebrauch. Dazu hat man zunächst die normativen Grundlagen zur Modularisierung geschaffen. Das entsprechende Dokument mit dem Titel Modularization of XHTML liegt ais Empfehlung (Recommendation) des W3-Konsortiums vor. Ausgehend davon hat man **XHTML 1.1** als modulbasiertes Sprachensystem entworfen. Auch XHTML 1.1 liegt als Empfehlung des W3-Konsortiums vor, unter dem Titel XHTML 1.1 – Module-based XHTML. XHTML 1.1 trennt sich endgültig von den
in XHTML 1.0 und HTML 4.0 als deprecated (missbilligt) eingestuften Elementen und Attributen. Es entspricht nur noch der Sprachvariante „`Strict`„. Damit fallen zugleich auch die Frames wieder aus dem Konzept.

> In wieweit sich das Konzept der Modularisierung durchsetzen wird, bleibt abzuwarten.

Denn wer sich mit dem Design eigener Auszeichnungssprachen befasst, kann schließlich auch direkt die Möglichkeiten von XML nutzen, um entsprechende Sprachen zu entwerfen. Und „einfacher“ als XML ist das Konzept der Modularisierung sicher nicht – im Gegenteil:
Es setzt im Grunde die Beherrschung von XML voraus und reizt dessen Möglichkeiten aus.

Die Module von XHTML 1.1

Die folgende Tabelle listet die Module auf, aus denen XHTML 1.1 besteht. Auf die Praxis als XHTML-Anwender hat diese Modularisierung keinen Einfluss. Sie können Ihre **XHTML-Dokumente** schreiben wie gewohnt – mit der Einschränkung, dass
Sie nichts mehr verwenden dürfen, was noch aus den HTML-Varianten „`Transitional`“ und „`Frameset`“ stammt.

Name des Moduls	Elemente	Erläuterung
Structure	`body head html title`	Modul für die Elemente, die das Grundgerüst eines XHTML-Dokuments bilden.
Text	`abbr acronym address blockquote br cite code dfn div em h1 h2 h3 h4 h5 h6 kbd p pre q samp span strong var`	Modul für alle Block- und Inline-Elemente, die Text enthalten können und logische Auszeichnungen darstellen, also keinen Hinweis auf ihre visuelle Darstellung enthalten.
Hypertext	`a`	Modul für Anker und Verweise innerhalb des Dokuments.
List	`dl dt dd ol ul li`	Modul für nummerierte Listen, Aufzählungslisten und Definitionslisten (Glossarlisten).

Name des Moduls	Elemente	Erläuterung
Object	`object param`	Modul für Multimedia-Referenzen.
Presentation	`b big hr i small sub sup tt`	Modul für Elemente, die Text enthalten können und physische Auszeichnungen darstellen, also einen Hinweis auf ihre visuelle Darstellung enthalten.
Edit	`del ins`	Modul für änderungsmarkierungen im Text.
Bidirectional Text	`bdo`	Modul für mehrsprachige Dokumente, die Text in Sprachen mit unterschiedlicher Schreibrichtung enthalten.
Forms	`button fieldset form input label legend select optgroup option textarea`	Modul für Formulare und Formularelemente.
Table	`caption col colgroup table tbody td tfoot th thead tr`	Modul für Tabellen und Tabellenelemente.
Image	`img`	Modul für Grafikreferenzen.
Client-side Image Map	`area map`	Modul für Grafiken mit verweis-sensitiven Flächen, vom Browser allein darzustellen.
Server-side Image Map	`ismap` (bei `img`)	Modul für Grafiken mit verweis-sensitiven Flächen, vom Browser in Kommunikation mit dem Web-Server darzustellen.
Intrinsic Events	`on*` (Event-Handler-Attribute)	Modul für alle Event-Handler, die als Attribut in Elementen notiert werden können und eine Schnittstelle für Script-Sprachen bilden.
Meta-Information	`meta`	Modul für Meta-Daten für Suchmaschinen und Browser.
Scripting	`noscript script`	Modul für Script-Bereiche und Bereiche für Browser, die keine Scripts ausführen können.
Stylesheet	`style` (Element)	Modul für Bereiche, in denen zentrale Stylesheet-Formate definiert werden.

Name des Moduls	Elemente	Erläuterung
Style Attribute	`style` (Attribut)	Modul für das style-Attribut.
Link	`link`	Modul für logische Beziehungen zu anderen Dateien.
Base	`base`	Modul für Adressbasis und Zielfensterbasis.
Ruby Annotation	`ruby rbc rtc rb rt rp`	Modul für Ruby-Text. Die entsprechenden Elemente sind neu. Ruby-Text ist eine Textform, die in fernöstlichen Sprachen verwendet wird, um Silben- oder Wortzeichen mit zusätzlichen Bedeutungshinweisen zu versehen.