XML11

Autor: Heiko Schröder

Inhaltliches Lektorat: Dirk Frischalowski

3. überarbeitete Ausgabe, 2009

Internet: www.herdt.com

XML 1.1

Grundlagen

XML11

3 XHTML 1.1

In diesem Kapitel erfahren Sie

- wie sich XHTML vom bisherigen Standard HTML unterscheidet
- wie Sie Ihre bisherigen HTML-Dokumente für XML vorbereiten

Voraussetzungen

- HTML-Kenntnisse
- Kenntnisse im Umgang mit der MS-DOS-Eingabeaufforderung

3.1 Überblick

Ausgehend von der Meta-Auszeichnungssprache SGML entstand HTML als SGML-Applikation sowie die Metasprache XML. Problematisch ist, dass es in der Beschreibungssprache HTML bereits unterschiedliche browserinterne HTML-Tags gibt. Ein Beispiel ist der für den Internet Explorer typische Tag `marquee`. Dieser ermöglicht einen Lauftext, ähnlich einem Nachrichtenticker. Dieses Element entspricht nicht dem HTML-Standard des W3-Konsortiums, wird aber aus Gründen der Kompatibilität auch von anderen Browsern interpretiert. Die Browser Firefox und Opera interpretieren wiederum das Element `blink`, die anderen Browser nicht.

Das W3-Konsortium hat mit der HTML-Version 4.01 vom 24. Dezember 1999 die weitere Standardisierung der Sprache HTML eingestellt. Die Sprache XHTML wurde als neuer Standard für Webseiten entwickelt. XHTML (Extensible Hypertext Markup Language) vereint die Vorteile von HTML 4.0 und XML 1.0 und existiert als Empfehlung in der Version 1.1 (*http://www.w3.org/TR/xhtml11/*). Ziel dieser Entwicklung ist es, den Inhalt, die Struktur und die Formatierung eines Dokuments voneinander zu trennen.

Die wichtigste Änderung gegenüber HTML ist, dass die XHTML-Quelldateien wohlgeformt sein müssen. Dies bedeutet, dass nach den XHTML-Regeln geschriebene HTML-Dokumente keinen Fehler in der Struktur der HTML-Tags enthalten dürfen. Diese klar definierten Regeln ermöglichen es den Entwicklern beispielweise, die Ausführungsgeschwindigkeit von Browsern zu erhöhen, da sie keine Routinen mehr entwickeln müssen, die auch fehlerhaften Code verarbeiten können. Die Entwickler können den HTML-Code über eine standardisierte Dokumenttyp-Definition prüfen. Dies verringert demzufolge auch die Größe des Programms. In neueren Browser-Versionen wäre es möglich, dass fehlerhafte Dokumente nicht mehr angezeigt werden. Stattdessen könnte eine Fehlermeldung erscheinen, ähnlich den Fehlermeldungen beim Ausführen von fehlerhaftem JavaScript.

3.2 Elemente

Groß- und Kleinschreibung

Die Groß- und Kleinschreibung von XML-Elementen ist von großer Bedeutung. Schreiben Sie das Start-Tag groß, müssen Sie das schließende Ende-Tag ebenfalls großschreiben. Wenn Sie das Start-Tag kleinschreiben, müssen Sie auch das Ende-Tag kleinschreiben. In XHTML werden alle HTML-Tags kleingeschrieben.

Das folgende Beispiel ist in HTML korrekt, in XHTML jedoch falsch:

```
<H1 Align="center">nachfolgender Text</h1>
```

In XHTML müsste dieses Beispiel so geschrieben werden:

```
<h1 align="center">nachfolgender Text</h1>
```

In HTML wird die Groß- und Kleinschreibung nicht berücksichtigt. Damit Sie auf die zukünftige XHTML-Schreibweise vorbereitet sind, schreiben Sie künftig Element- und Attributnamen immer klein.

Öffnende und schließende Tags

Alle geöffneten Tags müssen in XML auch einen schließenden Tag besitzen. Diese Regel wurde ebenfalls in XHTML übernommen. In HTML können Sie an vielen Stellen den schließenden Tag weglassen, ohne dass der Browser eine Fehlermeldung einblendet.

```
<ol>
  <li>Dies ist ein Listenelement
  <li>Dies ist ein weiteres Listenelement
</ol>
```

In XHTML müssen Sie den schließenden Tag `</li>` hinzufügen.

```
<ol>
  <li>Dies ist ein Listenelement</li>
  <li>Dies ist ein weiteres Listenelement</li>
</ol>
```

Jedes Start-Tag müssen Sie auch wieder mit dem entsprechenden Ende-Tag schließen. Ein Spezialfall sind die allein stehenden Elemente: `area`, `base`, `br`, `col`, `frame`, `hr`, `img`, `input`, `isindex`, `link`, `meta`, `option` und `param`. Um mit älteren Browsern kompatibel zu bleiben, müssen Sie vor dem Zeichen `>` ein Leerzeichen und einen Schrägstrich `/` schreiben.

```
<br />
<hr />
<img src="bild.jpg" alt="alternativer Text" />
```

Sie können diese Elemente jedoch auch als öffnenden und schließenden Tag schreiben. Es ist daher nicht falsch, wenn Sie statt der möglichen Kurzform `<br />` die ausführliche Schreibweise `<br></br>` benutzen.

Verschachtelung von HTML-Tags

Die Reihenfolge der Elemente muss logisch gegliedert sein. Das heißt, dass sich die einzelnen Elemente nicht überlappen dürfen. Das folgende Beispiel ist in XHTML falsch:

```
<b><i>fetter kursiver Text</b></i>
<p>Ein Text mit <em>einer logischen Textauszeichnung.</p></em>
```

Die korrekte Schreibweise in XHTML lautet:

```
<b><i>fetter kursiver Text</i></b>
<p>Ein Text mit <em>einer logischen Textauszeichnung.</em></p>
```

Blockelemente, wie `h1` … `h6`, `p` oder auch `hr`, erzeugen im Dokument stets einen Zeilenvorschub. Inline-Elemente tun dies nicht, sie enthalten die Textdaten und weitere Inline-Elemente. Sie erzwingen keinen Zeilenvorschub und werden verwendet, um Textelemente besonders zu kennzeichnen.

Da die Inline-Elemente innerhalb von Blockelementen verwendet werden, sind sie den Blockelementen untergeordnet. Es ist daher nicht erlaubt, dass ein Inline-Element ein Blockelement umschließt.

```
<span><p>lange Textpassagen oder einzelne Worte</p></span>
```

Das Element p ist dem Element span übergeordnet, da p ein Blockelement und span ein Inline-Element ist. Somit darf span nicht das Element p umschließen, sondern p muss span umschließen.

```
<p><span>lange Textpassagen oder einzelne Worte</span></p>
```

Kombinationen

Einige Kombinationen von HTML-Tags sind verboten. So darf beispielsweise innerhalb der physischen Auszeichnung pre in Zukunft kein Bild img angegeben werden.

Folgende Elemente dürfen die anderen angegebenen Elemente nicht enthalten:

Das Element	darf das Element ... nicht enthalten
a	a
pre	img, object, big, small, sub, sup
button	input, select, textarea, label, button, form, fieldset, iframe, isindex
label	label
form	form

3.3 Attribute und Werte

Anführungszeichen

In XHTML müssen alle Attribute auch einen Wert besitzen. Die Werte sind immer in Anführungszeichen zu setzen, auch wenn sie nur aus einer Zahl oder einer Prozentangabe bestehen. Beispielsweise ist

```
<td valign=top>
  <img alt="alternativer Text" width=300% height=25 />
</td>
```

keine gültige XHTML-Angabe. Die Anführungszeichen müssen in diesem Fall angegeben werden.

```
<td valign="top">
  <img alt="alternativer Text" width="300%" height="25" />
</td>
```

Kurzschreibweise

Die Kurzschreibweise von Attributen ist nicht mehr erlaubt. Folgende Attribute sind von dieser Regelung betroffen: compact, nowrap, ismap, declare, noshade, checked, disabled, readonly, multiple, selected, noresize, defer.

```
<input name="text" rows="20" cols="50" readonly />
<select name="auswahl" size="5" multiple>
  <option>Auswahl 1</option>
  <option>Auswahl 2</option>
  <option>Auswahl 3</option>
</select>
```

Stattdessen ist der Name des Attributs noch einmal als Wert in Anführungszeichen anzugeben.

```
<input name="text" rows="20" cols="50" readonly="readonly" />
<select name="auswahl" size="5" multiple="multiple">
  <option>Auswahl 1</option>
  <option>Auswahl 2</option>
  <option>Auswahl 3</option>
</select>
```

Eine weitere Änderung betrifft die Tags `a`, `applet`, `form`, `frame`, `iframe`, `img` und `map`. In HTML werden sie über die Attribute `name` oder `id` eindeutig gekennzeichnet, in XHTML werden sie in Zukunft nur noch über das Attribut `id` referenziert. Es ist aus Gründen der Kompatibilität zu älteren Browsern noch möglich, zusätzlich das Attribut `name` zu belassen. Aus der Schreibweise

```
<a name="marke">Stelle, die angesprungen werden soll.</a>
```

wird ab sofort

```
<a name="marke" id="marke">Stelle, die angesprungen werden soll.</a>
```

Beachten Sie, dass das Attribut `name` zwar noch zugelassen ist, jedoch in einer späteren Version von XHTML und somit von zukünftigen Browsern nicht mehr unterstützt werden wird.

3.4 Dokumenttyp und Zeichensätze

Die Anzahl möglicher Sprachversionen, die ein Browser beim Lesen einer HTML-Datei beherrschen muss, wird immer größer. Deshalb ist es sinnvoll, wenn Sie die HTML-Version, die Sie in Ihren Dateien verwenden, an den Anfang einer jeden HTML-Datei schreiben. Die Versionsangabe ist ein Hinweis auf eine bestimmte HTML-Sprachversion. Die Angabe von

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
```

besagt zum Beispiel, dass sich die nachfolgenden Zeilen des HTML-Dokuments genau an die HTML-4.01-Definition des W3-Konsortiums halten. Bisher hatte diese Angabe keine Konsequenzen für die Darstellung der HTML-Datei und wurde deshalb von den Autoren einer Webseite vernachlässigt. Das ändert sich jedoch mit der Einführung von XHTML und XML.

Um festzulegen, dass es sich im Nachfolgenden um eine Datei nach dem Standard XHTML 1.1 handelt, ist nur noch die folgende Dokumenttyp-Definition anzugeben:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

Mit der Angabe dieses Dokumenttyps weisen Sie darauf hin, dass Ihr Dokument den strengen XHTML-Regeln der Version 1.1 genügt. Die in der Version 1.0 noch geduldeten, veralteten Angaben werden entweder durch entsprechende Stylesheet-Angaben ersetzt oder alternativ über die Standardeinstellungen des Clients formatiert. Aus HTML bekannte, aber nicht mehr empfohlene Elemente werden nicht verwendet.

Die bisherigen Dokumenttypen von XHTML 1.0 mit den Angaben von `Strict`, `Transitional` und `Frameset` sind damit ungültig.

Weiterhin müssen Sie in einer HTML-Datei den im Dokument verwendeten Zeichensatz spezifizieren. Diese Spezifikation muss ebenfalls in XHTML erfolgen.

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
```

In diesem Fall wird darauf hingewiesen, dass innerhalb des Dokuments Zeichen des westeuropäischen Zeichensatzes ISO-8859-1 verwendet werden.

3.5 Änderungen XHTML 1.1 gegenüber 1.0

Die bedeutendste Änderung ist, dass veraltete Elemente und Attribute, die bisher als "deprecated" geduldet wurden, nicht mehr verwendet werden dürfen. XHTML wird modularisiert, indem die HTML-Tags in Modulen zusammengefasst werden (z. B. Text-, Formular-, Tabellen-, Grafik-, Objektmodul). Die Definition der XHTML-Modularisierung finden Sie unter *http://www.w3.org/TR/xhtml-modularization*.

Weitere wichtige Änderungen sind:

- Die Attribut-Angabe `lang` entfällt an einzelnen Elementen und wird zentral als Attribut `xml:lang` festgelegt.
- Die Attributangabe von `name` bei den Elementen `a` und `map` entfällt und ist durch das Attribut `id` zu ersetzen. Ankerpunkte sind z. B. als `<a id="ankername" />` zu setzen.
- Hinzugekommen ist das Modul `ruby` mit verschiedenen neuen Elementen (`rbc`, `rtc`, `rb`, `rt`, `rp`). Damit können kurze Anmerkungen oder die Aussprache von Texten hinzugefügt werden. Diese Art der Darstellung ist vor allen Dingen aus dem ostasiatischen Raum bekannt. Somit können z. B. um Datumsangaben herum nähere Erläuterungen angegeben werden (siehe kleine Abbildung).

3.6 XHTML-Dokument deklarieren

Folgende Angaben sind innerhalb eines Dokuments vorzunehmen, um es eindeutig als XHTML-Dokument zu deklarieren.

- Der Dokumenttyp ist anzugeben.
- Das Hauptelement ist wie bisher das Element `html`.
- Das Hauptelement muss eine `xmlns`-Deklaration für XHTML-Namensräume enthalten. Der Namensraum für XHTML ist definiert als *http://www.w3.org/1999/xhtml*. Auf die Bedeutung von Namensräumen wird im weiteren Verlauf des Buches näher eingegangen.

Das Grundgerüst eines XHTML-Dokuments sieht z. B. folgendermaßen aus:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title> </title>
  </head>
  <body>
  </body>
</html>
```

3.7 JavaScript und Stylesheets

In HTML können JavaScript- und Stylesheet-Anweisungen im Dokument angegeben werden. Diese sind vom Typ PCDATA und können somit eine beliebige Zeichenfolge enthalten. Damit müssen die Daten vor der Verwendung überprüft werden. So wird sichergestellt, dass die Angaben beim Seitenaufbau korrekt verwendet werden können. In XHTML müssen die internen Stylesheet-Angaben abgewandelt werden, da sie nicht vom Programm überprüft werden sollen.

Aus der internen Stylesheet-Angabe

```
<style>
  .Klasse { <!-- Angabe der Stylesheets --> }
</style>
```

wird in der XHTML 1.1-Spezifikation

```
<style type="text/css">
<![CDATA[
  .Klasse { <!—Angabe der Stylesheets --> }
]]>
</style>
```

Die einzelnen Stylesheet-Klassen werden von der Angabe <![CDATA[...]]> umschlossen. CDATA steht für Character Data und wird im Unterschied zu PCDATA (Parsed Character Data) nicht vor der Verwendung überprüft. Dies ist wichtig, da die Stylesheet-Angabe keine XML-Elemente enthält, sondern die Formatierungsdefinitionen für das Dokument.

Ähnlich verhält es sich mit der internen Angabe von JavaScript-Anweisungen. Auch hier muss der bisherige JavaScript-Programmabschnitt

```
<script type="text/javascript">
   // JavaScript-Anweisungen
</script>
```

abgewandelt werden, damit diese nicht als XML-Anweisungen ausgewertet werden.

```
<script type="text/javascript">
<![CDATA[
  // JavaScript-Anweisungen
]]>
</script>
```

Eine Alternative ist es, externe JavaScript- und Stylesheet-Dokumente zu verwenden. An der Einbindung dieser Daten ändert sich gegenüber HTML 4.0 nichts.

3.8 Automatisches Konvertieren

HTML Tidy

Auf den Webseiten des W3-Konsortiums können Sie ein Programm herunterladen und verwenden, das den Quellcode eines HTML-Dokuments analysiert und aufgetretene Syntaxfehler kenntlich macht. Dieses Programm heißt HTML Tidy und wurde von Dave Raggett, einem Mitarbeiter des W3-Konsortiums, entwickelt. Sie können das Programm kostenlos von der Internetseite *http://tidy.sourceforge.net* downloaden. Hier finden Sie ebenfalls die englischsprachige Anleitung zum Programm.

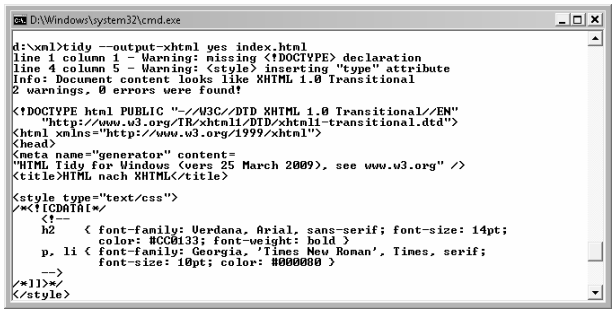
- ▶ Speichern Sie die ausführbare Datei *tidy.exe* in einem Verzeichnis Ihrer Wahl.

Das Programm wird über die Kommandoeingabe der Eingabeaufforderung (Konsole) gesteuert.

- ▶ Öffnen Sie die Eingabeaufforderung.
- ▶ Wechseln Sie mit dem Kommando `cd` in das Verzeichnis, in dem sich die Datei *tidy.exe* befindet.
- ▶ Geben Sie die nachfolgende Anweisung ein.

```
tidy --output-xhtml yes Dateiname.html
```

Mit der Option `--output-xhtml yes` geben Sie an, dass die Datei auf die Einhaltung der XHTML-Regeln zu testen ist. Die Angabe der zu testenden Datei *Dateiname.html* ist durch den Namen der entsprechenden HTML-Datei zu ersetzen.



Das Programm testet die angegebene Datei und teilt Ihnen aufgetretene Fehler direkt in der Eingabeaufforderung mit. Das Programm gibt sehr viele Informationen aus, sodass Sie eventuell nicht in der Lage sind, alle Fehlermeldungen zu lesen.

Für diesen Fall gibt es eine Option, mit der Sie die Ausgabe in eine lokale Textdatei umleiten können.

- ▶ Geben Sie die nachfolgende Anweisung in die Eingabeaufforderung ein.

```
tidy --output-xhtml yes -f error.txt Dateiname.html
```

Die zusätzlich Angabe von `-f error.txt` bewirkt, dass die Fehlerausgabe nicht am Bildschirm erfolgt, sondern in die Datei *error.txt* geschrieben wird. Die Datei *error.txt* können Sie in einem beliebigen Texteditor öffnen. Sie finden Hinweise und Warnungen zu möglichen Fehlern, die bei der Umsetzung der HTML-Datei in das XHTML-Format aufgetreten sind.

- ▶ Bereinigen Sie die angegebenen Fehler in Ihrer HTML-Datei.
- ▶ Testen Sie die Datei erneut mithilfe des Programms HTML Tidy.
- ▶ Wiederholen Sie diesen Vorgang, bis keine Fehlermeldungen mehr ausgegeben werden.

Haben Sie die Struktur der HTML-Datei so angepasst, dass das Programm keine weiteren Verstöße gegen die XHTML-Regeln findet, können Sie die HTML-Datei in eine XHTML-Datei konvertieren.

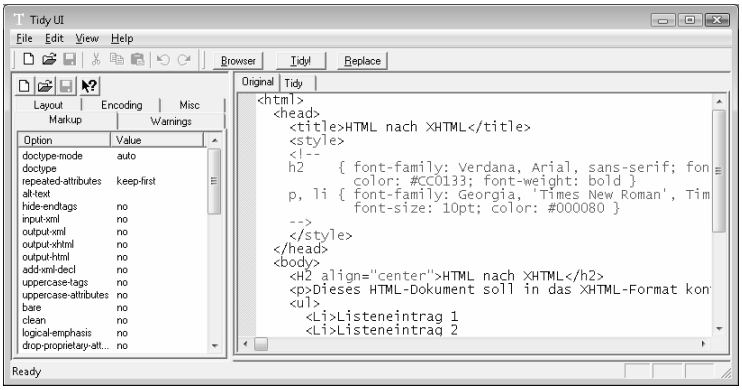
Legen Sie immer eine Sicherheitskopie der HTML-Datei an, die Sie in das XHTML-Format umwandeln lassen möchten.

- ▶ Öffnen Sie die Konsole (Eingabeaufforderung).
- ▶ Wechseln Sie mit dem Kommando `cd` in das Verzeichnis, in dem sich das Programm *tidy.exe* befindet.
- ▶ Geben Sie die nachfolgende Anweisung in die Konsole ein.

```
tidy --output-xhtml yes -f error.txt -m Dateiname.html
```

Mit der Option `-m` fordern Sie das Programm auf, die angegebene Datei *Dateiname.html* in das XHTML-Format umzuwandeln. Beachten Sie, dass mit dieser Option die Originaldatei überschrieben wird.

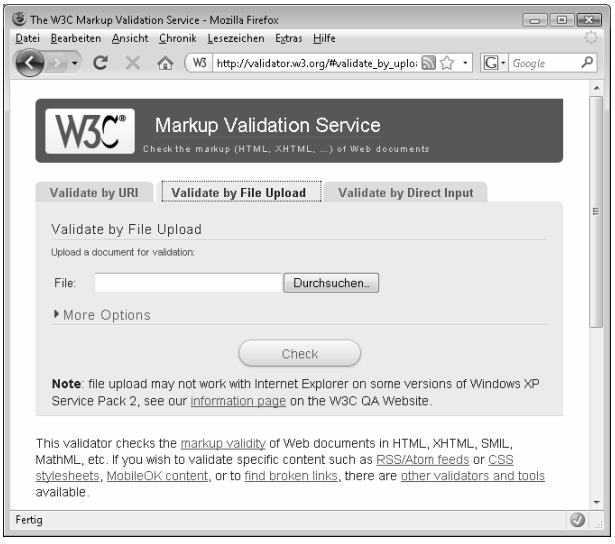
Für die Nutzung weiterer Einstellungsoptionen lesen Sie die englischsprachige Online-Hilfe des Programms unter: *http://tidy.sourceforge.net/docs/quickref.html*. Für das Betriebssystem Windows existiert außerdem eine grafische Benutzeroberfläche Tidy UI, die Sie unter *http://users.rcn.com/creitzel/tidy.html* herunterladen und in ein Verzeichnis entpacken können. Diese Software ermöglicht Ihnen die Verwendung von HTML Tidy, ohne auf die Eingabeaufforderung zurückgreifen zu müssen.



W3C MarkUp Validation Service

Nachdem Sie die Konvertierung in das XHTML-Format vorgenommen haben, können Sie die Datei vom MarkUp Validation Service des W3-Konsortiums auf Gültigkeit testen lassen.

- ▶ Öffnen Sie in Ihrem Browser die Internetadresse *http://validator.w3.org*.



Webseite zum Testen eines XHTML-Dokuments

Erfolgreich getestetes XHTML-Dokument

- ▶ Klicken Sie neben dem Eingabefeld FILE auf die Schaltfläche DURCHSUCHEN.
- ▶ Wählen Sie Ihre XHTML-Datei aus.
- ▶ Betätigen Sie die Schaltfläche CHECK.

Ihr Dokument wird auf den Server des W3-Konsortiums übertragen. Dort wird es mithilfe eines Skripts auf die korrekte Verwendung von XHTML-Elementen und Attributen getestet. Verstößt Ihr Dokument gegen keine XHTML-Regel, erscheint die Ausgabe: *This Page Is Valid XHTML1.1!* In diesem Fall können Sie das offizielle Zeichen des W3-Konsortiums auf Ihrer Webseite integrieren.

Damit zeigen Sie Ihren Besuchern, dass Ihre Webseite XHTML-1.1-kompatibel ist und Sie den ersten Schritt in Richtung XML getan haben.

3.9 Übung

HTML-Datei in XHTML-Datei umwandeln

Übungsdatei: *kap03\vorgabe.html* **Ergebnisdatei: *kap03\uebung.html***

① Konvertieren Sie die nachfolgende HTML-Datei in das XHTML-Format. Welche HTML-Tags sind nach den XHTML-Richtlinien falsch?

```
<html>
  <head>
    <title>HTML nach XHTML</title>
    <style>
    <!--
    h2    { font-family: Verdana, Arial, sans-serif; font-size: 14pt;
            color: #CC0133; font-weight: bold }
    p, li { font-family: Georgia, 'Times New Roman', Times, serif;
            font-size: 10pt; color: #000080 }
    -->
    </style>
  </head>
  <body>
    <H2 align="center">HTML nach XHTML</h2>
    <p>Dieses HTML-Dokument soll in das XHTML-Format konvertiert werden.
    <ul>
      <Li>Listeneintrag 1
      <Li>Listeneintrag 2
      <Li>Listeneintrag 3
    </ul>
    <form method="POST" action="mailto:">
      <p><input type="text" name="Eingabe" size=20 readonly>
      <p><textarea rows=2 name="Memo" cols=20 readonly></textarea>
      <p><input type="submit" value="Abschicken">
         <input type="reset" value="Zurücksetzen">
    </form>
  </body>
</html>
```

② Benutzen Sie zur Auflistung der fehlerhaften XHTML-Angaben das Konvertierungsprogramm HTML Tidy.

③ Kontrollieren Sie mithilfe des HTML Validation Services des W3-Konsortiums die Korrektheit Ihres XHTML-Dokuments. Nehmen Sie Änderungen an dem XHTML-Code vor, bis Sie das offizielle XHTML-Zeichen des W3-Konsortiums erhalten.