XML light

XML bearbeiten

Jörn Clausen
joern@TechFak.Uni-Bielefeld.DE

Übersicht

- Formen von XML
- Nutzen von Grammatiken
- XML-Mode des Emacs
- Beispiel-Sprache: XHTML

Daten $\neq$ Daten?

```
<atoms>
  <atom>
    <name>Hydrogen</name>
    <symbol>H</symbol>
    <anum>1</anum>
    <aweight>1.00797</aweight>
  </atom>
  <atom>
    <name>Helium</name>
    <symbol>He</symbol>
    <anum>2</anum>
    <aweight>4.0026</aweight>
  </atom>
</atoms>
```

```
<glossary>
  <glossentry id="Berzerkeley">
    <glossterm>Berzerkeley</glossterm>
    <glossdef>
      <para>Humorous distortion of
        <quote>Berkeley</quote> used
        esp. to refer to the
        practices or products of the
        <glossterm>BSD</glossterm> Unix
        hackers. See <glossterm>software
        bloat</glossterm>,
        <glossterm>Berkeley Quality
        Software</glossterm>.</para>
    </glossdef>
  </glossentry>
</glossary>
```

Daten-Typen

- *data centric*
  - regelmäßige, sich wiederholende Struktur
  - evtl. tiefe Schachtelung
  - Daten nur in den „Blättern“
- *document centric*
  - Text von Elementen unterbrochen
  - Elemente an (fast) beliebigen Positionen

    ```
    <para>... there lived a <index>Hobbit</index>.</para>
    ```
  - *mixed content*

Daten-Typen

- *data centric*
  - regelmäßige, sich wiederholende Struktur
  - evtl. tiefe Schachtelung
  - Daten nur in den „Blättern“
- *document centric*
  - Text von Elementen unterbrochen
  - Elemente an (fast) beliebigen Positionen

    ```
    <para>... there lived a <index>Hobbit</index>.</para>
    ```
  - *mixed content*
- Text $\neq$ Daten?

Grammatiken

- formale Beschreibung der Markup-Sprache
- Wo dürfen welche Elemente/Attribute vorkommen?
- XML-Parser kann *Validität* einer *Instanz* überprüfen
- nur Syntax, nicht Semantik
- verschiedene Grammatik-Sprachen
  - Document Type Definition (DTD)
  - W3C XML Schema
  - Relax NG
  - ...

Aufgaben

- Die Datei `chemicals.xml` enthält Informationen über chemische Elemente. Überzeuge Dich davon, daß es sich um eine wohlgeformte XML-Datei handelt:

  ```
  $ xmllint chemicals.xml
  ```

- Die Datei `atoms.dtd` enthält eine formale Beschreibung der in `chemicals.xml` verwendeten Markup-Sprache. Überprüfe mit

  ```
  $ xmllint --valid chemicals.xml
  ```

  ob die XML-Datei auch valide ist. Falls nicht, korrigiere sie entsprechend mit dem Emacs.

- Woher weiß `xmllint`, daß `atoms.dtd` die Grammatik zu `chemicals.xml` enthält?

Aufgaben

- Setze im Emacs den Cursor direkt hinter das letzte schließende `atom`-Tag. Drücke die Tastenkombination

  ```
  C-c C-e Return
  ```

  Was passiert?

- Vervollständige den Eintrag: Boron, B, 5, 10.811
- Speichere die Datei ab und überprüfe nochmals mit

  ```
  $ xmllint --valid chemicals.xml
  ```

  ob die Datei syntaktisch korrekt ist.

Aufgaben

- Öffne die Datei `webpage.html` mit dem Emacs und aktiviere den XML-Modus:

  ```
  M-x xml-mode
  ```

- Bewege den Cursor an das Dateiende und verwende erneut die Tastenkombination `C-c C-e Return`. Was passiert?

Hypertext Markup Language

- Struktur einer (X)HTML-Datei:

```
<html>
  <head>
    ...
  </head>
  <body>
    ...
  </body>
</html>
```

- header: Metainformationen über das Dokument
- body: das eigentliche Dokument

Aufgaben

- Lösche den Kommentar im `head`-Element und drücke die Tastenfolge

  ```
  C-c C-e TAB
  ```

  Was passiert?

- Gib im Minibuffer „`title`“ ein und drücke Return.

- Im `title`-Element sollte ein sinnvoller Titel für die Seite stehen. Dieser Eintrag wird z.B. als Bookmark verwendet. Der Eintrag sollte also nicht zu lang, aber trotzdem aussagekräftig sein.

Absätze und Überschriften

- Inhalt des Dokuments im body-Element
- Fließtext in Absätzen:

```
<p> ... </p>
<p> ... </p>
```

- Überschriften:

```
<h1> ... </h1>
<h2> ... </h2>
...
<h6> ... </h6>
```

Aufgaben

- Füge einige Absätze mit Text und einige Überschriften in das body-Element ein. Verwende die gezeigten Tastenkombinationen.

  Blindtext kann man sich mit Hilfe von

  http://www.lipsum.com

  erzeugen lassen und per cut'n'paste übernehmen.

- Speichere das Dokument ab und sieh es Dir mit einem Web-Browser an.

- Wie hätte man Überschriften auch realisieren können? Was könnten Gründe gewesen sein, sich für die gewählte Form zu entscheiden?

logisches Markup

- Elemente, um Bedeutung zuzuweisen:
  - Hervorhebung:

    ```
    ... das sollte man nicht <em>so</em> genau nehmen.
    Das hat <strong>keinerlei</strong> Bedeutung.
    ```

  - Code-Fragmente:

    ```
    ... verwendet man die Anweisung <code>printf</code>.
    ```

  - Tastatureingaben:

    ```
    Drücken Sie die Taste <kbd>X</kbd>.
    ```

Aufgaben

- Markiere mit der Maus ein einzelnes Wort in einem Absatz und drücke die Tastenfolge

  ```
  C-c C-r TAB
  ```

  Wähle eines der gezeigten Elemente (`em`, `strong`, `code`, `kbd`) aus.

- Wiederhole den Vorgang mit anderen Worten und anderen Elemente. Wie werden die vier Elemente im Web-Browser dargestellt?

Aufgaben

- Markiere ein bereits mit „`code`“ ausgezeichnetes Wort und füge `strong`-Tags hinzu. Verwende die Reload-Funktion Deines Web-Browsers.
- Setze den Cursor in das Start-Tag des gerade eingefügten `strong`-Elements und drücke die Tastenfolge

  ```
  C-c -
  ```

- Ist es mit Hilfe der gezeigten Tastenkombinationen möglich, das folgende, nicht-wohlgeformte Stück XML zu erzeugen?

  ```
  Dies <em>ist <strong>ein</em> kleiner</strong> Test.
  ```

  Wie reagiert Dein Web-Browser auf diese Zeile?

physikalisches Markup

- Elemente, um Schriftarten auszuwählen:
  - Kursiv:

    ```
    ... nennt sich <i>Instanz</i>.
    ```
  - Fett:

    ```
    Scheibe <b>kräftig</b> eindrücken.
    ```
  - Typewriter:

    ```
    Der Befehl <tt>xmllint</tt> ...
    ```

Aufgaben

- Setze den Cursor in das Start-Tag eines `em`-Elements und drücke die Tastenfolge

  `C-c = TAB`

  Wähle das Element „`b`“ aus. Sieh Dir das Ergebnis im Web-Browser an. Probiere die beiden anderen Schriftarten aus.

- Was ist der Unterschied zwischen den Elementen

  `em`, `strong`, `code`, `kbd`

  und

  `i`, `b`, `tt`

Listen

- HTML kennt mehrere Arten von Listen:

```
<ul>
  <li> ... </li>
  <li> ... </li>
  ...
</ul>
```

```
<ol>
  <li> ... </li>
  <li> ... </li>
  ...
</ol>
```

- wie Absatz, also nur außerhalb von „p“ erlaubt
- Absatz/Absätze innerhalb eines list items möglich:

```
<li>kurzer Text</li>
<li>
  <p>ein Absatz</p>
  <p>noch’n Absatz</p>
</li>
```

Aufgaben

- Wie unterscheiden sich `ul`- und `ol`-Listen in der Darstellung?
- Ist es möglich, Listen in Listen zu erzeugen?

Tabellen

- zur Darstellung von tabellarischen Daten
- Anordnung von Zeilen und Spalten
- Ursprung: CALS (Continuous Acquisition and Life-Cycle Support)
- häufig zur Formatierung „mißbraucht“

table

- Aufbau: Zeilen von Spalten
  - `tr`: table row
  - `th`: table head
  - `td`: table data

```
<table>
  <tr>
    <th>Land</th>
    <th>Hauptstadt</th>
  </tr>
  <tr>
    <td>Deutschland</td>
    <td>Berlin</td>
  </tr>
  <tr>
    <td>Großbrittanien</td>
    <td>London</td>
  </tr>
</table>
```

Aufgaben

- Erzeuge die folgende Tabelle mit HTML:

Autor	Titel	Erscheinungsjahr
J.R.R. Tolkien	The Lord of the Rings	1954
Douglas Adams	The Hitch-Hiker's Guide to the Galaxy	1979
Michael Ende	Die unendliche Geschichte	1979

- Wie breit werden die einzelnen Tabellenspalten und die ganze Tabelle gesetzt?
- Wende das Programm „`tidy`“ auf die XHTML-Datei an. Was hat tidy zu der Tabelle zu sagen?

Aufgaben

- Setze den Cursor in das öffnende `table`-Tag und drücke die Tastenfolge

  `C-c + TAB`

  Gib „`summary`“ ein und drücke Return. Gib eine kurze Beschreibung der Tabelle an.

Aufgaben

- Mit den Attributen `frame` und `rules` des `table`-Elements kann man festlegen, ob und welche Linien in und um die Tabelle gezeichnet werden. Die möglichen Belegungen der Attribute sind nicht frei wählbar, sondern durch XHTML festgelegt.

  Füge die Attribute in die Tabelle ein. Drücke die TAB-Taste, wenn Du im Minibuffer den Wert angeben sollst. Wähle einen der gezeigten Werte aus. Probiere verschiedene Kombinationen für die beiden Attribute aus.

Was fehlt?

- diverse Elemente und Attribute
- Bilder und Verweise (Hypertext)

→ Hausaufgabe: SELFHTML

http://selfhtml.teamone.de

- Farben, (mehr) Schriften, Layout

→ Cascading Style Sheets (CSS): nächste Stunde

- Formatierung von (X)HTML und XML
- Trennung von Inhalt und Formatierung