

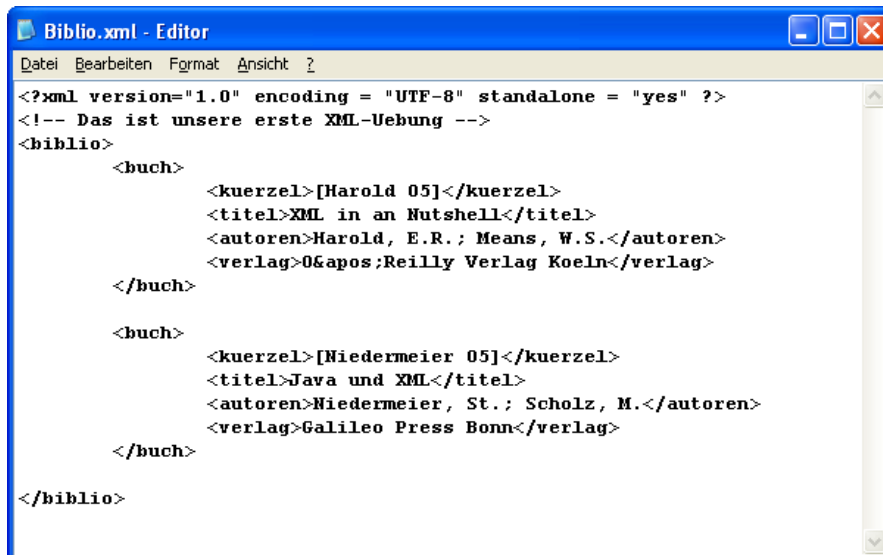
# Datenrepräsentation mit XML

## Praktikum 01

### XML-Grundlagen (wohlgeformte und validierte XML-Dokumente)

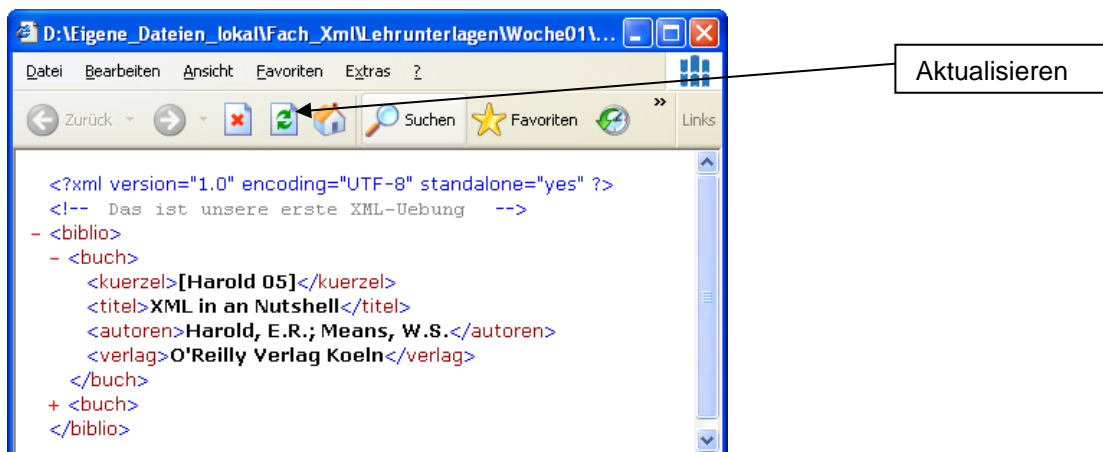
#### Übung 1.1 - Elemente kennenlernen

Erstellen Sie mittels Texteditor (z.B. Notepad, JEdit oder ähnlich) eine einfache XML-Datei, die eine kleine persönliche Literatursammlung beinhaltet:



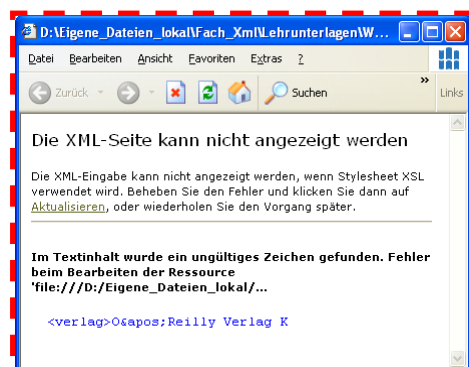
```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<!-- Das ist unsere erste XML-Uebung -->
<biblio>
  <buch>
    <kuerzel>[Harold 05]</kuerzel>
    <titel>XML in an Nutshell</titel>
    <autoren>Harold, E.R.; Means, W.S.</autoren>
    <verlag>O&apos;Reilly Verlag Koeln</verlag>
  </buch>
  <buch>
    <kuerzel>[Niedermeier 05]</kuerzel>
    <titel>Java und XML</titel>
    <autoren>Niedermeier, St.; Scholz, M.</autoren>
    <verlag>Galileo Press Bonn</verlag>
  </buch>
</biblio>
```

Die **Wohlgeformtheit** überprüfen Sie nun, indem Sie die Datei **Biblio.xml** mit einem Browser anzeigen lassen (Klicken auf Datei → rechte Maustaste → Öffnen mit ...)



Sollten Sie nun Fehler angezeigt bekommen, so müssen Sie diese korrigieren

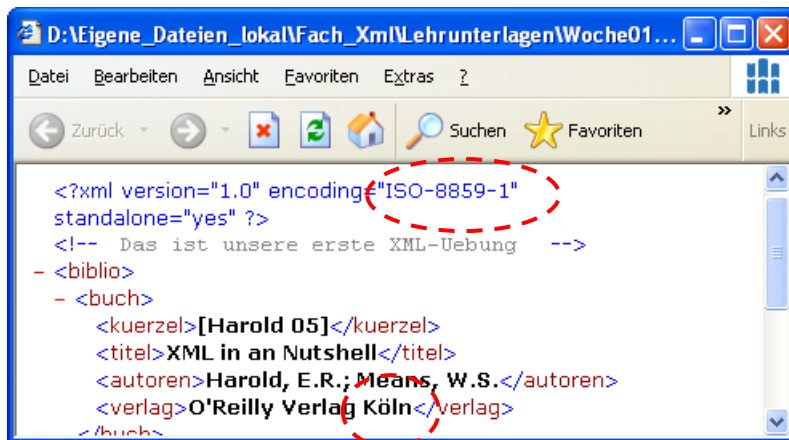
(Haben Sie etwa Köln und nicht Koeln geschrieben? → Das muss nicht zwangsläufig ein Problem sein und hängt auch vom verwendeten Editor ab).



Tipp zum Roundtrip: Lassen Sie während des Editierens den Editor offen. Speichern Sie die Datei mit *Datei* → *Speichern* ab. Aktualisieren Sie anschließend den Browser.

# Datenrepräsentation mit XML

Um das Problem mit dem deutschen Umlaut ö vorteilhaft lösen zu können, empfiehlt es sich, das Attribut encoding im XML-Prolog nachträglich auf die Unicode-Mappe "**ISO-8859-1**" zu setzen. Nun können Sie "Köln" normal schreiben und der XML-Parser im Browser stellt alles korrekt dar.



## Übung 1.2. Attribute einsetzen

Nun wollen wir jedes Buch-Element mit einer eindeutigen Identnummer belegen. Wir werden deshalb jedem Buch-Element ein Attribut **buch\_id** verpassen. Der Wert dieses Attributes wird für jeden Neueintrag um eins hoch gezählt. Der Beginn soll bei "1" liegen.

Außerdem soll beim Element Verlag ein optionales Attribut **link** eingetragen werden, wenn die Web-Adresse beachtenswert ist.



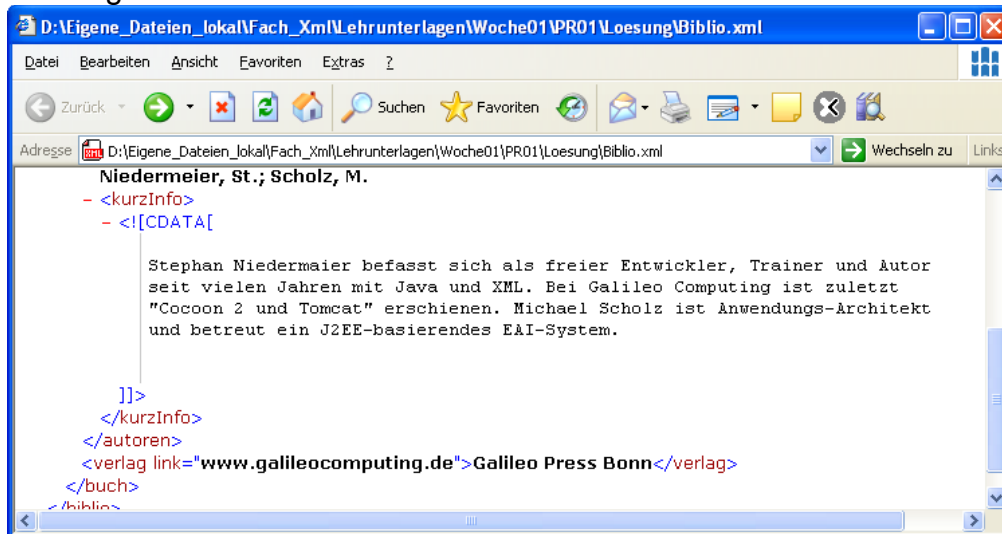
Als nächstes sollte eine kleines Autorenporträt abspeicherbar sein. Dazu sollte es bei Bedarf ein Element <kurzInfo> als Kindelement von <autoren> geben. Fügen Sie den folgenden Text per Zwischenablage mit **<![CDATA[ Portraettext ]>** an der richtigen Stelle ein.

"Stephan Niedermaier befasst sich als freier Entwickler, Trainer und Autor seit vielen Jahren mit Java und XML. Bei Galileo Computing ist zuletzt "Cocoon 2 und Tomcat" erschienen. Michael Scholz ist Anwendungs-Architekt und betreut ein J2EE-basierendes EAI-System."

Quelle: [Niedermaier 06], Bucheinband

# Datenrepräsentation mit XML

Das Ergebnis sollte so aussehen:



Dieses Beispiel zeigt einen typischen "mixed content", wie er bei **narrativen** (erzählenden, dokumentenzentrierten) Anwendungen vorkommen darf.

Bei **datensatzzentrierten** XML-Anwendungen sollte der Entwickler versuchen, diesen Zustand bald wieder abzustellen (z.B. indem für den Autoren-Namen-Text ein separates Element `<namensListe>` eingeführt wird).

## Übung 1.3 Ein gültiges XML-Dokument per DTD erstellen und validieren

Um diesem XML-Dokument schrittweise eine strengere Form zu geben, damit Anwendungen sicherer damit arbeiten können, wird anhand der bisherigen Strukturüberlegungen eine interne DTD (Document Type Definition) erstellt.

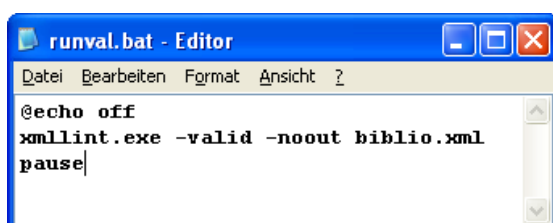
Fangen wir mit dem Erstellen einer DTD an. Um das Ergebnis auch kontrollieren zu können, brauchen wir einen XML-Parser, der außer der Wohlgeformtheit auch die Gültigkeit testen kann, d.h. er muss validieren können.

Da Browser dazu grundsätzlich nicht in der Lage sind, benutzen wir ein kleines freies Kommandozeilenprogramm **xmllint.exe**.

Wenn wir später mit entsprechenden XML-Java-Classlibraries arbeiten, so können diese dort verwendeten Parser das Validieren dann selbstverständlich auch.

Entpacken Sie das kleine Programmpaket **xmllint.zip** so, dass alle enthaltenen Dateien in Ihr aktuelles Übungsverzeichnis (z.B. ...\\Praktikum1) hineinkopiert werden.

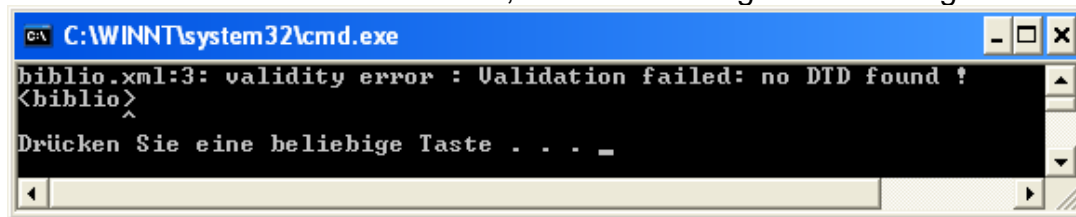
Legen Sie dann eine kleine Batch-Datei **runval.bat** an, mit der Sie die Validierung immer bei Bedarf unkompliziert starten können.



Der Schalter **-valid** schaltet das **Validieren** ein. **-noout** verhindert die Ausgabe des gesamten XML-Baumes.

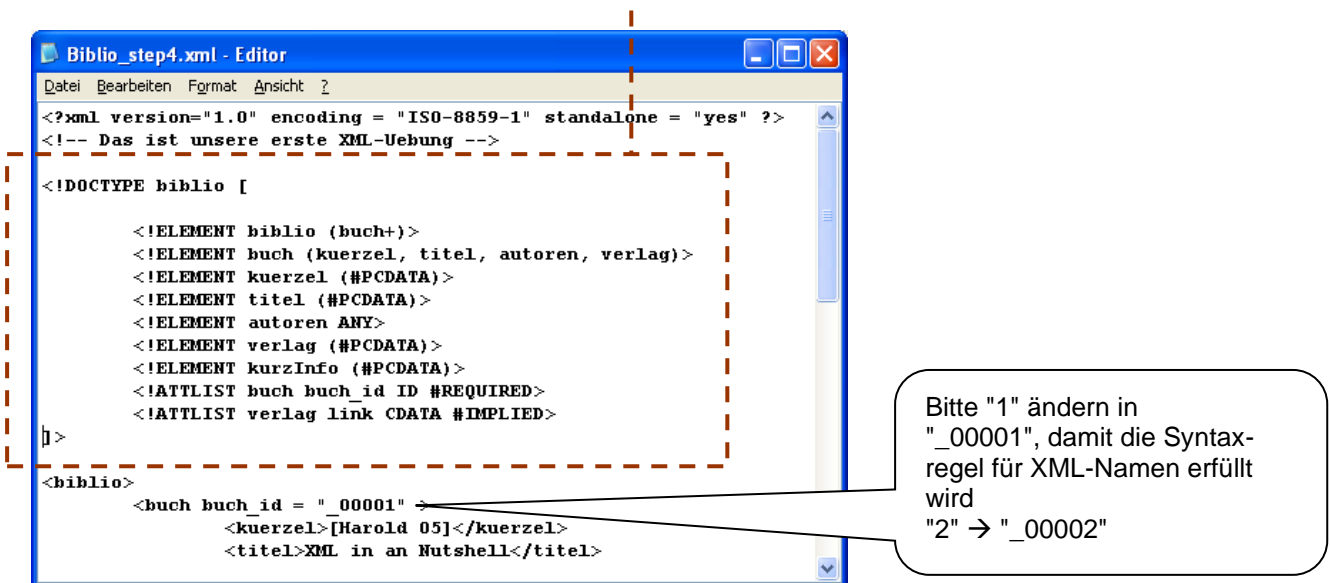
# Datenrepräsentation mit XML

Wenn Sie das erste Mal **validieren**, erhalten Sie folgende Meldung:



Diese Aussage stimmt. - Wir müssen deshalb dringend etwas tun:

Zunächst erstellen wir folgende interne DTD:



Beim wiederholten Testen, wenn alles stimmt, bringt **xmllint.exe** keine Fehlermeldung mehr.

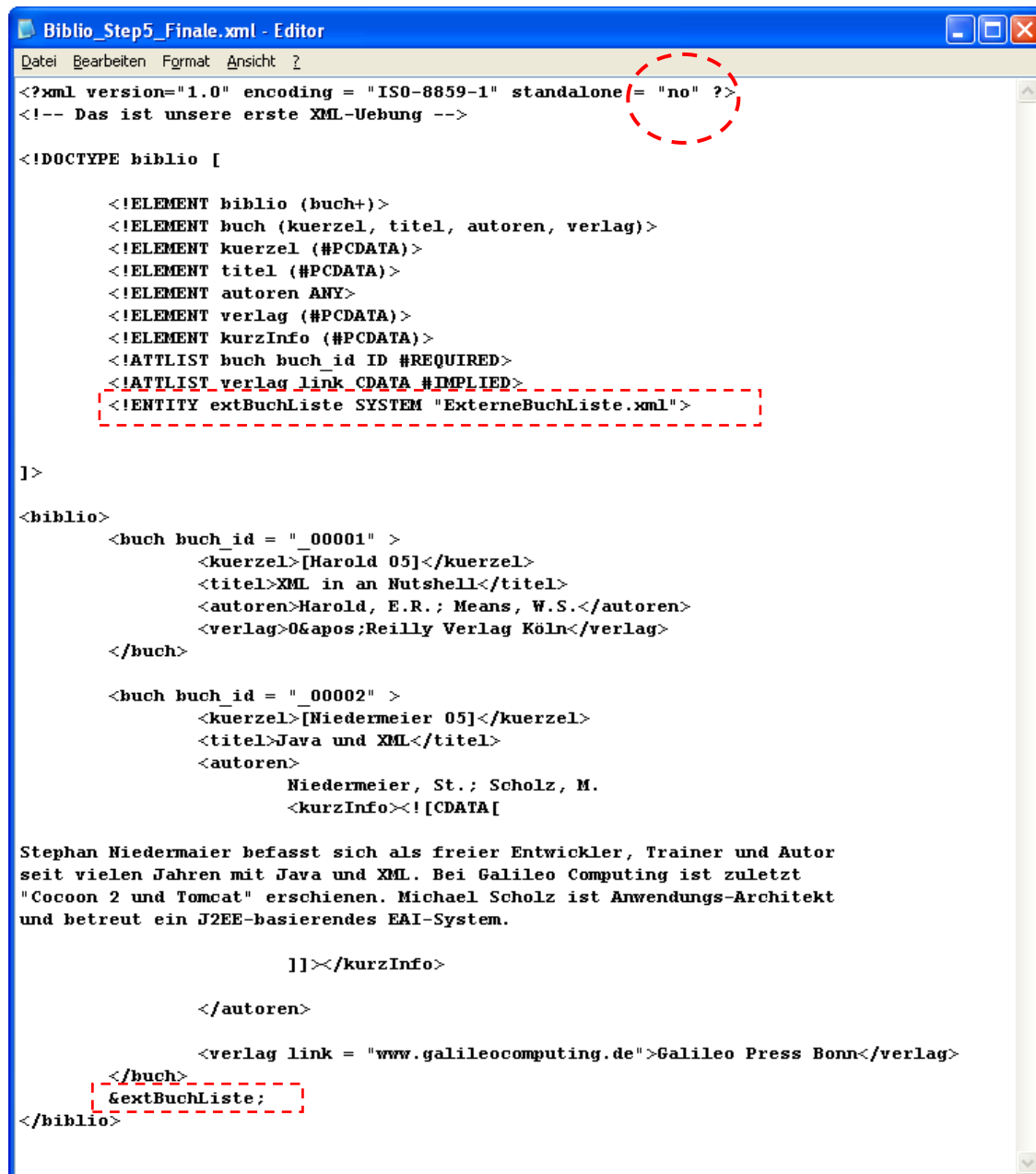
## Übung 1.4. Externes Dokument über Entity-Deklaration in der DTD einbeziehen

Aus dem bisherigen Bestand gibt es weitere Buch-Daten, die wir mit in unsere Gesamtbibliographie einbeziehen möchten. Damit können wir die Wartung und Pflege der Daten auf mehrere Schultern verteilen.

Entpacken Sie dazu die vorgegebene Datei **externeBuchListe.xml** und kopieren Sie diese in Ihr Übungsverzeichnis.

# Datenrepräsentation mit XML

Nun müssen Sie Ihre DTD um eine ENTITY-Deklaration erweitern. Weiterhin müssen Sie das XML-Prolog-Attribut **standalone = "no"** in der ersten Zeile setzen. Damit wird der Parser informiert, dass es weitere externe Dokumente zwingend mit zu verarbeiten gilt. Schließlich können Sie an passender Stelle die Entity **&extBuchListe;** in Ihr XML-Dokument einfügen. Der Parser löst an dieser Stelle den Inhalt der externen Datei auf und baut alle Teile der externen Datei (außer den Prolog) in Ihr Hauptdokument ein, als ob es aus einem "Guss" wäre.



```
<?xml version="1.0" encoding = "ISO-8859-1" standalone = "no" ?>
<!-- Das ist unsere erste XML-Uebung -->

<!DOCTYPE biblio [

    <!ELEMENT biblio (buch+)>
    <!ELEMENT buch (kuerzel, titel, autoren, verlag)>
    <!ELEMENT kuerzel (#PCDATA)>
    <!ELEMENT titel (#PCDATA)>
    <!ELEMENT autoren ANY>
    <!ELEMENT verlag (#PCDATA)>
    <!ELEMENT kurzInfo (#PCDATA)>
    <!ATTLIST buch buch_id ID #REQUIRED>
    <!ATTLIST verlag link CDATA #IMPLIED>
    <!ENTITY extBuchListe SYSTEM "ExterneBuchListe.xml">

]

>

<biblio>
  <buch buch_id = " 00001" >
    <kuerzel>[Harold 05]</kuerzel>
    <titel>XML in an Nutshell</titel>
    <autoren>Harold, E.R.; Means, W.S.</autoren>
    <verlag>O&apos;Reilly Verlag Köln</verlag>
  </buch>

  <buch buch_id = " 00002" >
    <kuerzel>[Niedermeier 05]</kuerzel>
    <titel>Java und XML</titel>
    <autoren>
      Niedermeier, St.; Scholz, M.
      <kurzInfo><![CDATA[
Stephan Niedermaier befasst sich als freier Entwickler, Trainer und Autor
seit vielen Jahren mit Java und XML. Bei Galileo Computing ist zuletzt
"Cocoon 2 und Tomcat" erschienen. Michael Scholz ist Anwendungs-Architekt
und betreut ein J2EE-basierendes EAI-System.

]]></kurzInfo>

</autoren>

      <verlag link = "www.galileocomputing.de">Galileo Press Bonn</verlag>
    </buch>
    &extBuchListe;
</biblio>
```

## ***Datenrepräsentation mit XML***

---

Mit dem Browser können Sie sich das Endergebnis leider nicht anschauen, da externe Dokumente aus Sicherheitsgründen nicht geladen werden. Aber Sie können zumindest mit der Validierung überprüfen, ob Ihr Dokument gültig ist.