Technische Hochschule
mannheim

XML: DTD und Schema

Technische Hochschule Mannheim | Prof. Dr. Frank Dopatka

Was ist eine DTD?

Technische Hochschule mannheim

- Eine Dokumenttypdefinition (DTD) ist ein Satz an Regeln, der benutzt wird, um Dokumente eines bestimmten Typs zu deklarieren.
- Ein Dokumenttyp ist dabei eine Klasse ähnlicher Dokumente, wie beispielsweise Telefonbücher oder Inventurdatensätze.
- Die Dokumenttypdefinition besteht dabei vorwiegend aus
  - Elementtypen,
  - Attributen von Elementen und
  - Entitäten.
- In einer DTD wird dabei die Reihenfolge, die Verschachtelung der Elemente und die Art des Inhalts von Attributen festgelegt, und damit also die Struktur des Dokuments.
- Eine DTD spezifiziert die Syntax einer Anwendung von SGML oder XML, wie die von ihnen abgeleiteten Sprachen HTML oder XHTML.
- Diese Syntax ist in einer weniger generellen Form gehalten als die SGML- oder XML-Syntax.

DTD-Elemente

Technische Hochschule mannheim

- Mit einer Elementtyp-Deklaration ELEMENT wird ein Element und sein möglicher Inhalt definiert.
- In einem gültigen Dokument dürfen nur Elemente vorkommen, die in der DTD definiert sind.
- Der Inhalt eines Elementes kann durch die Angabe anderer Elementnamen und durch einige Schlüsselwörter und Zeichen angegeben werden.

DTD-Elemente

Technische Hochschule mannheim

- #PCDATA für Zeicheninhalte (Parsed Character Data)
- EMPTY für keinen Inhalt
- ANY für beliebigen Inhalt
- , für Reihenfolgen
- | für Alternativen (im Sinne „entweder…oder“)
- () zum Gruppieren
- * für beliebig oft aufeinander folgend
- + für mindestens einmal
- ? für keinmal oder genau einmal
- Wird kein Stern, Pluszeichen oder Fragezeichen angegeben, so muss das Element genau einmal vorkommen.

DTD-Attribute

Technische Hochschule mannheim

- Attribute werden innerhalb einer Attributliste
  `<!ATTLIST Elementname Attributliste>`
  definiert.
- Die Attributliste enthält durch Leerzeichen oder Zeilenumbrüche getrennt jeweils den Attributnamen, den Typ und Vorgaben der einzelnen Attribute.
- Beispiele für Attribute:
  - ID
  - IDREF und IDREFS
  - NMTOKEN und NMTOKENS

DTD-Attribute und Beispiel

Technische Hochschule mannheim

- Mit den Attribut-Vorgaben kann angegeben werden, ob ein Attribut
  - vorkommen muss (#REQUIRED),
  - ob es optional ist (#IMPLIED) oder
  - einen festen Wert enthält (#FIXED) und welcher Wert als Standardwert benutzt wird, falls das Attribut bei einem Tag nicht angegeben wird.
- Beispiel einer Attribut-Deklaration für das img-Tag in HTML:

```
<!ATTLIST img
   id      ID        #IMPLIED
   src     CDATA     #REQUIRED
   alt     CDATA     #REQUIRED
   ismap   IDREF     #IMPLIED
>
```

DTD-Entität und ein Beispiel dazu

Technische Hochschule mannheim

- Entitäten bestehen aus Zeichenketten und können selber wieder Entitäten enthalten.
- Eine Entität ist ein Platzhalter für Text oder Daten, der innerhalb eines XML-Dokuments verwendet werden kann.
- Entitäten ermöglichen die Wiederverwendung und Modularisierung von Daten sowie die Einbindung von externen Inhalten.

```
<!ENTITY name "Benedikt">
<!ENTITY papst "&name;, der XVI.">
<!ENTITY wplink "<a href='http://de.wikipedia.org'>Wikipedia</a>">
```

Beispiel für eine XML-Datei gemäß einer DTD

Technische Hochschule mannheim

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note SYSTEM "Note.dtd">
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

Die Note.dtd passend zum Beispiel

Technische Hochschule mannheim

```
<!DOCTYPE note
[
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
```

Eine DTD zum HTML-Standard

Technische Hochschule mannheim

```
<!ELEMENT html (head, body)>
<!ELEMENT hr EMPTY>
<!ELEMENT div (#PCDATA | p | ul | ol | dl | table | pre | hr |
          h1|h2|h3|h4|h5|h6 | blockquote | address | fieldset)*>
<!ELEMENT dl (dt|dd)+>
```

Kritik zur DTD

Technische Hochschule mannheim

- Lange Zeit war der Weg über eine DTD der einzige Weg zur Beschreibung einer XML-Datenstruktur.
- Allerdings hat eine DTD gravierende Mängel:
  - Zunächst fehlt es daran, Elementen und Attributen präzisen Datentypen zuweisen zu können.
  - Eine DTD hat selbst keine XML-Syntax, sondern eher an die Sprache von regulären Ausdrücken angelehnt. Daher kann eine DTD nicht von einem gewöhnlichen XML-Parser analysiert werden.
  - Eine DTD ist nicht zu namespaces kompatibel.

Was ist ein XML-Schema?

Technische Hochschule mannheim

- Eine XML Schema Definition (XSD) ist eine Empfehlung des W3C zum Definieren von Strukturen für XML-Dokumente.
- Anders als bei den klassischen XML-DTDs wird die Struktur in Form eines XML-Dokuments beschrieben.
- Darüber hinaus wird eine große Anzahl von Datentypen unterstützt.
- XML Schema unterscheidet zwischen
  - einfachen atomaren Datentypen und
  - komplexen Datentypen.
- Datentypen in XML Schema werden weiterhin klassifiziert in
  - eingebaute bzw. vordefinierte sowie
  - benutzerdefinierte
- Datentypen.

XML-Schema: Einfache Datentypen

Technische Hochschule mannheim

- XML Schema stellt einige grundlegende atomare Datentypen bereit.
- Die atomaren Datentypen enthalten die „klassischen“ Datentypen, wie sie auch in Programmiersprachen wie C, Java oder SQL spezifiziert sind:
  - xs:string
  - xs:decimal
  - xs:integer
  - xs:float
  - xs:boolean
  - xs:date
  - xs:time

XML-Schema: Einfache Datentypen

Technische Hochschule mannheim

- Hinzu kommen weitere XML-spezifische atomare Typen, unter anderem:
  - QName: ein qualifizierter Name, ein global eindeutiger Bezeichner
  - anyURI: ein Uniform Resource Identifier (URI)
  - language: eine Sprachbezeichnung, z. B. de-DE, en-US, fr
  - ID: ein Identifikationsattribut innerhalb von XML-Elementen
  - IDREF: eine Referenz auf einen ID-Wert

XML-Schema: Ein eigener Datentyp

Technische Hochschule mannheim

- Das folgende Beispiel definiert einen neuen XML-Datentyp mit dem Namen monatInt sowie eine Liste monate dieses neuen Typs:

```
<xs:simpleType name="monatInt">
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="1"/>
    <xs:maxInclusive value="12"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="monate">
  <xs:list itemType="monatInt"/>
</xs:simpleType>
```

- Eine Instanz des neuen Typs könnte wie folgt aussehen:

```
<monate>
   1 2 3 4 5 6 7 8 9 10 11 12
</monate>
```

XML-Schema: Ein eigener Datentyp

Technische Hochschule mannheim

- Ein neuer Typ wird als Vereinigungsmenge bereits bestehender Typen definiert. Jede Instanz wählt dann ihren Typ aus dieser Menge.
- Das nachfolgende Beispiel definiert einen weiteren Typ monatsname sowie einen Union-Typ monat:

```
<xs:simpleType name="monatsname">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Jan"/>
    <xs:enumeration value="Feb"/>
    <xs:enumeration value="Mär"/>
    <!-- und so weiter … -->
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="monat">
  <xs:union memberTypes="monatsname monatInt"/>
</xs:simpleType>
```

XML-Schema: Ein eigener Datentyp

Technische Hochschule mannheim

- XML-Elemente vom Typ monat dürfen entweder ganzzahlige Werte im Bereich 1–12 enthalten oder eine der entsprechenden Monatsbezeichnungen als Zeichenkette.
- Gültige Instanzen sind beispielsweise:

```
<monat>Jan</monat>
<monat>2</monat>
```

XML-Schema: Ein eigener, komplexer Datentyp

Technische Hochschule mannheim

- Meine eigene komplexe Datenstruktur:

```
<xs:complexType name="pc-Typ">
  <xs:sequence>
    <xs:element name="name"       type="xs:string"/>
    <xs:element name="hersteller" type="xs:string"/>
    <xs:element name="prozessor"  type="xs:string"/>
    <xs:element name="mhz"        type="xs:integer" minOccurs="0"/>
    <xs:element name="kommentar"  type="xs:string"  minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:integer"/>
</xs:complexType>
```

- Und die Definition einer Liste davon:

```
<xs:element name="pc-liste">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="pc" type="pc-Typ" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

XML-Schema: Beispiel zum eigenen, komplexen Typ

Technische Hochschule mannheim

```
<pc-liste>
  <pc id="1">
    <name>Dimension 3100</name>
    <hersteller>Dell</hersteller>
    <prozessor>AMD</prozessor>
    <mhz>3060</mhz>
    <kommentar>Arbeitsplatzrechner</kommentar>
  </pc>
  <pc id="2">
    <name>T 42</name>
    <hersteller>IBM</hersteller>
    <prozessor>Intel</prozessor>
    <mhz>1600</mhz>
    <kommentar>Laptop</kommentar>
  </pc>
</pc-liste>
```

Beispiel einer XML-Bücherliste

Technische Hochschule mannheim

```
<?xml version="1.0"?>
<BookCatalogue xmlns="http://www.ppedv.de/schema/BookCatalogue"
 xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
 xsi:schemaLocation=
  "http://http://www.ppedv.de/schema/BookCatalogue/BookCatalogue.xsd">
 <Book>
  <Title>Computernetzwerke</Title>
  <Author>Tanenbaum</Author>
  <Date>1999</Date>
  <ISBN>3-8272-9568-8</ISBN>
  <Publisher>Prentice Hall</Publisher>
 </Book>
 <Book>
  <Title>1984</Title>
  <Author>George Orwell</Author>
  <Date>1995</Date>
  <ISBN>3-548-22562-4</ISBN>
  <Publisher>Ullstein</Publisher>
 </Book>
</BookCatalogue>
```

Beispiel einer XML-Bücherliste: Eine passende DTD dazu...

Technische Hochschule mannheim

```
<!?Ein BookCatalogue enthält ein oder mehrere Books -->
<!ELEMENT BookCatalogue (Book)*>
<!ELEMENT Book (Title, Author, Date, ISBN, Publisher)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Author (#PCDATA)>
<!ELEMENT Date (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT Publisher (#PCDATA)>
```

Beispiel einer XML-Bücherliste:
Eine passendes XML-Schema dazu...

Technische Hochschule mannheim

```
<?xml version="1.0" encoding="UTF-8"?>
<!--W3C Schema generated by XML Spy v3.5 NT (http://www.xmlspy.com)-->
<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
elementFormDefault="qualified">
 <xsd:element name="Author" type="xsd:string"/>
 <xsd:element name="Book">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element ref="Title"/>
    <xsd:element ref="Author"/>
    <xsd:element ref="Date"/>
    <xsd:element ref="ISBN"/>
    <xsd:element ref="Publisher"/>
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>
```

Beispiel einer XML-Bücherliste: Eine passendes XML-Schema dazu...

Technische Hochschule mannheim

- Eine Bücherliste besteht aus Büchern...

```
<xsd:element name="BookCatalogue">
 <xsd:complexType>
  <xsd:sequence>
   <xsd:element ref="Book" maxOccurs="unbounded"/>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

Beispiel einer XML-Bücherliste: Eine passendes XML-Schema dazu...

Technische Hochschule mannheim

- Author hat den einfachen Typ String.
- Title darf nur aus dieser Menge sein:

```
<xsd:element name="Title">
 <xsd:simpleType>
  <xsd:restriction base="xsd:string">
   <xsd:enumeration value="1984"/>
   <xsd:enumeration value="Computernetzwerke"/>
  </xsd:restriction>
 </xsd:simpleType>
</xsd:element>
```

Beispiel einer XML-Bücherliste: Eine passendes XML-Schema dazu...

Technische Hochschule mannheim

- Das Datum darf nur aus dieser Menge sein:

```
<xsd:element name="Date">
 <xsd:simpleType>
  <xsd:restriction base="xsd:int">
   <xsd:enumeration value="1995"/>
   <xsd:enumeration value="1999"/>
  </xsd:restriction>
 </xsd:simpleType>
</xsd:element>
```

- Die ISBN darf nur aus dieser Menge sein:

```
<xsd:element name="ISBN">
 <xsd:simpleType>
  <xsd:restriction base="xsd:string">
   <xsd:enumeration value="3-548-22562-4"/>
   <xsd:enumeration value="3-8272-9568-8"/>
  </xsd:restriction>
 </xsd:simpleType>
</xsd:element>
```

Beispiel einer XML-Bücherliste: Eine passendes XML-Schema dazu...

Technische Hochschule mannheim

- Auch der Publisher ist nur ein String:

```
<xsd:element name="Publisher" type="xsd:string"/>
```