EKON 7

Consic

XML-DTD

Einführung in die Syntax der XML Document Type Definitions

Stefan Heymann
Consic Software Engineering · www.consic.de · heymann@consic.de

Inhalts-Übersicht

- DTD – was ist das?
- Deklaration
- Bausteine der DTD
- Ausblick

DTD – was ist das?

DTD

- XML definiert selbst keine Tags, Attribute oder Entities
- Wichtig für den Austausch von XML-Instanzen: Einhaltung gleicher Konventionen
  - Verwendete Elemente
  - Entities
  - Struktur des Dokuments
- DTD = Syntax zur Festlegung solcher Konventionen
- Prüfung der Einhaltung der DTD durch „Validating Parser“

DTD

- DTD = Document Type Definition
- Definiert den *Typ* der XML-Applikation
- Erst die DTD beschreibt einen speziellen XML-Dateityp

DTD-Deklaration

DTD-Deklaration

- Eine DTD steht am Anfang des XML-Dokuments in der *Document Type Declaration*
- Besteht aus speziellem Markup
- DTD kann außerhalb des Dokuments liegen [External DTD]
- Alternativ/zusätzlich dazu Deklarationen im Dokument selbst

DTD-Deklaration

Syntax

```
<!DOCTYPE name [...] >

<!DOCTYPE name SYSTEM "externe.dtd“
 [...interne Deklarationen...] >

<!DOCTYPE name PUBLIC "publicid" "externe.dtd" >
```

Position

- Nach dem XML-Prolog
- Vor dem Inhalt

DTD-Deklaration

- Eine DTD kann beliebige Kommentare und PIs enthalten
- Externe DTDs sollten mit einer „Text Declaration“ beginnen

Syntax

```
<?xml version="1.0" encoding="UTF-8" ?>
```

Interne DTDs

- Der interne Teil der DTD steht in [eckigen Klammern]
- Er ist optional
- Seine Deklarationen überschreiben bzw. ergänzen diejenigen der externen DTD

Bausteine der DTD

DTD-Bausteine

- Mögliche **Elemente**
- **Anordnung** der Elemente (Schachtelung, Reihenfolge, Anzahl)
- **Attribute** der Elemente
- **Entities** (Textbausteine)

Elemente

Syntax

```
<!ELEMENT name inhalt >
```

EMPTY	Element ist immer leer
ANY	Beliebiger Inhalt
(#PCDATA)	Nur Text
(#PCDATA | e1 | e2)	Nur Text oder e1 oder e2

(e1, e2|e3, e4*, e5+, e6?|e7)

Beispiele

```
<!ELEMENT br EMPTY>
<!ELEMENT absatz ANY>
<!ELEMENT header (#PCDATA) >
<!ELEMENT buyer (name, address+, phone?) >
```

Attribute

- Zu jedem Element wird definiert,
  - welche Attribute es gibt (Name)
  - Datentyp der Attribute
  - Default-Wert der Attribute

Syntax

```
<!ATTLIST elementname
          name typ default >
```

Attribute: Typen

CDATA	Beliebiger Text
ID	Ein XML-Bezeichner
IDREF	Bezug auf im Dokument vorhandene ID
IDREFS	Mehrere IDREF‘s, durch [SP] getrennt
ENTITY	Name einer Entity
ENTITIES	Mehrere ENTITY‘s
NMTOKEN	Mehrere Name-Token (1 2 3 4 5)
enum	Aufzählung (left | right | top | bottom)
NOTATION	Auflistung von Notations-Namen

Attribute: Defaults

'default'	Direkte Angabe des Defaults
#REQUIRED	Attribut *muss* ang. werden
#IMPLIED	Default-Wert wird von Applikation eingesetzt
#FIXED 'xxx'	Es darf nur der angegebene Wert verwendet werden

Attribute: Beispiele

```
<!ATTLIST absatz
          valign (top|center|bottom) 'top'
          halign (left|cntr|right|block) #IMPLIED
          borderleft NMTOKEN '0'>

<!ATTLIST img
          src CDATA #REQUIRED
          alt CDATA 'Image not found' >
```

Entities

- Entities sind Textbausteine
- Vom einzelnen Zeichen bis zum kompletten externen Dokument
- Aufruf durch `&name;`
- Inhalt wird in der DTD definiert
- Schachtelung möglich

Entities: Syntax

```
<!ENTITY name "inhalt">
<!ENTITY name SYSTEM "extern.xml" >
<!ENTITY name SYSTEM "img.gif" NDATA gif>
```

Beispiele

```
<!ENTITY versionno '1.0.1'>
<!ENTITY datum "2000-09-27">
<!ENTITY einleitung SYSTEM "./einl.xml">
<!ENTITY ver "&versionno; &datum;
  &build;">
```

Parameter-Entities

- Sonderfall der Entities
- Können nur innerhalb der DTD referenziert werden
- Wiederverwendung von DTD-Teilen
- Eigener Namensraum
- Eingeleitet durch Prozent-Zeichen

Parameter-Entities

Deklaration

```
<!ENTITY % generalattr "id ID #IMPLIED">
```

Referenz

```
<!ATTRLIST para
          h-align (l|c|r|b) 'b'
          %generalattr; >
```

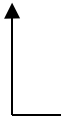
Parsed/Unparsed Entities

- Entities werden unterschieden in:
  - **Parsed** E.: Ersetzungstext ist XML
  - **Unparsed** E.: kein XML (Bilder, Videos, Textverarbeitungs-Dokumente, usw.)
- Unparsed E. bekommen eine **Notation** zugewiesen (=Typ)

Conditional Sections

- Nur in externen DTDs
- Abschnitte können ein- und ausgeblendet werden
- Schachtelung möglich

```
<![INCLUDE [ deklarationen ]]>
<![IGNORE [ deklarationen ]]>
```



Deklaration als Parameter Entity möglich

Ausblick

DTD Ausblick

- Probleme bei DTDs:
  - Zu wenig Möglichkeiten
    (z. B. keine Bereichs-Eingrenzung)
  - Keine XML-Syntax
    (eigener Parser erforderlich)
- Lösung: XML-SCHEMA
  + Viel mehr Optionen
  + XML-Syntax
  – Hohe Komplexität

Fragen?

Consic

</talk>

Stefan Heymann
Consic Software Engineering · www.consic.de · heymann@consic.de