

XML-Technologien

Tutorium im Sommersemester 2014

niels.hoppe@fu-berlin.de

6. Mai 2014

XML Grundlagen

URIs, URNs, URLs, IRIs, ...

Namensräume

Document Type Definitions

Elemente deklarieren

Attribute deklarieren

Entitäten deklarieren

Teilmengen definieren

XML Grundlagen

Beachte besonders:

- ▶ Wohlgeformtheit
- ▶ XML-Deklaration
- ▶ Syntax
- ▶ Regeln für Namen
- ▶ ...

Regeln für Namen

Gültige Namen...

- ... beginnen immer mit einem Buchstaben oder Unterstrich, jedoch nie mit `xm1` (unabhängig von der Groß- und Kleinschreibung).
- ... enthalten nur Buchstaben, Zahlen, Punkte oder Unterstriche und insbesondere keine Leerzeichen oder Doppelpunkte.

URIs, URNs, URLs, IRIs, ...

Ein verwirrendes Thema, es gibt laut W3C eine klassische und eine zeitgemäße Sichtweise dazu: <http://www.w3.org/TR/uri-clarification/>

URIs, URNs und URLs heute

- ▶ URI steht für Uniform Resource Identifier
- ▶ URI-Schemas bestimmen, wie URIs aussehen. Sie werden von IANA registriert:
`http://www.iana.org/assignments/uri-schemes`
- ▶ Eine URL ist eine URI, die eine Ressource durch eine Repräsentation ihrer primären Zugriffsmethode (z.B. HTTP) identifiziert.
- ▶ URN ist ein spezielles URI-Schema für universell eindeutige Namen, das ein Konzept von Namensräumen, z.B. ISBN, beinhaltet:
`urn:isbn:n-nn-nnnnnn-n`

Namensräume

Definition

An **XML namespace** is identified by a URI reference [RFC3986]; element and attribute names may be placed in an XML namespace using the mechanisms described in this specification.

Quelle: <http://www.w3.org/TR/REC-xml-names/#concepts>

Namensräume

Definition

An **XML namespace** is identified by a URI reference [RFC3986]; element and attribute names may be placed in an XML namespace using the mechanisms described in this specification.

Definition

An **expanded name** is a pair consisting of a *namespace name* and a *local name*.

Quelle: <http://www.w3.org/TR/REC-xml-names/#concepts>

Namensräume

Definition

An **XML namespace** is identified by a URI reference [RFC3986]; element and attribute names may be placed in an XML namespace using the mechanisms described in this specification.

Definition

An **expanded name** is a pair consisting of a *namespace name* and a *local name*.

Definition

For a name N in a namespace identified by a URI I, the **namespace name is I**. For a name N that is not in a namespace, the **namespace name has no value**. In either case the **local name is N**.

Quelle: <http://www.w3.org/TR/REC-xml-names/#concepts>

Namensräume

Definition

An **XML namespace** is identified by a URI reference [RFC3986]; element and attribute names may be placed in an XML namespace using the mechanisms described in this specification.

Definition

An **expanded name** is a pair consisting of a *namespace name* and a *local name*.

Definition

For a name N in a namespace identified by a URI I, the **namespace name is I**. For a name N that is not in a namespace, the **namespace name has no value**. In either case the **local name is N**.

Definition

A **qualified name** is a name subject to namespace interpretation.

Quelle: <http://www.w3.org/TR/REC-xml-names/#concepts>

Namensräume

Man verwendet *qualified names* anstelle von *expanded names*. Sie werden entweder als *prefixed names* oder als *unprefixed names* geschrieben.

```
<root>
  <foo
    xmlns="http://www.example.org/default-namespace"
    xmlns:pre="http://www.example.org/another-namespace">

    <bar /><!-- this is in default-namespace -->

    <pre:baz /><!-- this is in another-namespace -->

    <!-- this is in default-namespace -->
  </foo>
  <!-- this is in null-namespace -->
</root>
```

Namensräume

Attribute in XML sind sogenannte assoziierte Knoten. Sie werden nicht wie normale Kindelemente eines Elements behandelt.

- ▶ Für Namensräume bedeutet das, dass Attribute nicht im Namensraum des Elements stehen, in dem sie notiert sind, sondern im Null-Namensraum.
- ▶ Will man den Namensraum eines Attributes ändern, muss es mit einem Präfix versehen werden:

```
<svg xmlns="http://www.w3.org/2000/svg"  
      xmlns:xlink="http://www.w3.org/1999/xlink">  
  <a xlink:href="grafik2.svg">Link zur Grafik</a>  
</svg>
```

- ▶ Es gibt für Attribute keine Möglichkeit, sie ohne Präfix in einen bestimmten Namensraum zu setzen.

Document Type Definitions

- ▶ Elemente deklarieren
- ▶ Attribute deklarieren
- ▶ Entitäten deklarieren
- ▶ Teilmengen definieren
- ▶ DTDs einbinden

Anwendung Beschreibung der Struktur von XML-Dokumenten.

Anwendung Beschreibung der Struktur von XML-Dokumenten.

Prinzip Beschreibe Struktur von Elementen und ihren Attributen mittels einfacher regulärer Ausdrücke.

Elemente deklarieren

Elemente können verschiedene Arten von Inhalten haben:

```
<!ELEMENT my-element EMPTY>
```

```
<!ELEMENT my-element ANY>
```

```
<!ELEMENT my-element (#PCDATA)>
```

```
<!ELEMENT my-element (specified, contents)>
```

Elemente deklarieren

Die Häufigkeit enthaltener Elemente lässt sich festlegen:

```
<!ELEMENT my-element (  
    zero-or-once?,  
    zero-or-more*,  
    once-or-more+,  
    exactly-once  
)>
```

Elemente deklarieren

Man kann Alternativen angeben:

```
<!ELEMENT my-element (  
    either-this  
    | or-this  
    | or-that  
)>
```

Elemente deklarieren

Man kann Gruppierungen vornehmen:

```
<!ELEMENT my-element (  
    alice  
    | (bob, charlie)  
)>
```

Elemente deklarieren

Definition

```
elementdecl '<!ELEMENT' Name contentspec '>'  
contentspec 'EMPTY' |'ANY' |Mixed |children  
quantifier ('?' |'*' |'+')  
children (choice |seq) quantifier?  
    cp (Name |choice |sequence) quantifier?  
choice '(' cp ('|' cp)+ ')'  
seq '(' cp (',' cp)* ')'  
Mixed '(' '#PCDATA' ('|' Name)* ')'* | '(' '#PCDATA' ')'
```

Attribute deklarieren

Attribute eines Elementes werden nacheinander aufgelistet:

```
<!ATTLIST my-element some-attribute CDATA "default-value">
```

Attribute deklarieren

Attribute eines Elementes werden nacheinander aufgelistet:

```
<!ATTLIST my-element some-attribute CDATA "default-value">
```

```
<!ATTLIST my-element  
  required-attribute CDATA #REQUIRED>  
  optional-attribute CDATA #IMPLIED>
```

Attribute deklarieren

Attribute eines Elementes werden nacheinander aufgelistet:

```
<!ATTLIST my-element some-attribute CDATA "default-value">
```

```
<!ATTLIST my-element  
  required-attribute CDATA #REQUIRED>  
  optional-attribute CDATA #IMPLIED>
```

```
<!ATTLIST my-element fixed-attribute  
  CDATA #FIXED "always-this-value">
```


Attribute deklarieren

Attribute eines Elementes werden nacheinander aufgelistet:

```
<!ATTLIST my-element some-attribute CDATA "default-value">
```

```
<!ATTLIST my-element  
  required-attribute CDATA #REQUIRED>  
  optional-attribute CDATA #IMPLIED>
```

```
<!ATTLIST my-element fixed-attribute  
  CDATA #FIXED "always-this-value">
```

```
<!ATTLIST my-element choice-attribute (  
  either-this-value  
  | or-this-value  
  | or-that-value) "either-this-value">
```

Attribute deklarieren

Es gibt eine ganze Menge von Attributtypen:

- CDATA** The value is character data
- (en1 |en2 |..)** The value must be one from an enumerated list
 - ID** The value is a unique id
 - IDREF** The value is the id of another element
 - IDREFS** The value is a list of other ids
- NMTOKEN** The value is a valid XML name
- NMTOKENS** The value is a list of valid XML names
- ENTITY** The value is an entity
- ENTITIES** The value is a list of entities
- NOTATION** The value is a name of a notation
 - xml:** The value is a predefined xml value

Quelle: http://www.http://w3schools.com/dtd/dtd_attributes.asp

Attribute deklarieren

Definition

AttlistDecl '`<!ATTLIST` *Name* *AttDef** `>`'

AttDef *Name* *AttType* *DefaultDecl*

AttType *StringType* | *TokenizedType* | *EnumeratedType*

StringType 'CDATA'

TokenizedType 'ID' | ... | 'NMTOKENS'

EnumeratedType *NotationType* | *Enumeration*

NotationType 'NOTATION' '(' *Name* ('|' *Name*)* ')'

Enumeration '(' *Nmtoken* ('|' *Nmtoken*)* ')'

DefaultDecl '#REQUIRED' | '#IMPLIED' | ((' #FIXED')? *AttValue*)

Entitäten deklarieren

Entitäten sind Konstanten, die innerhalb von PCDATA-Blöcken ausgewertet werden. Es gibt verschiedene Arten von Entitäten:

- ▶ Built-in Entities (z.B. `&` und `<t;`;))
- ▶ Character Entities (z.B. `ó` und `ó`;))
- ▶ General Entities, selbst definiert
- ▶ Parameter Entities, selbst definiert

Entitäten können sowohl innerhalb der DTD deklariert, als auch extern referenziert werden.

Entitäten deklarieren

Beachte bei General Entities:

- ▶ Der Name muss ein gültiger XML-Name sein.
- ▶ Der Wert interner Entitäten muss wohlgeformt sein.
- ▶ Der Wert externer Entitäten muss nicht wohlgeformt sein, wohl aber das Dokument, nachdem sie ausgewertet wurden.
- ▶ Der Wert kann andere Entitäten enthalten.

Entitäten deklarieren

Beachte bei General Entities:

- ▶ Der Name muss ein gültiger XML-Name sein.
- ▶ Der Wert interner Entitäten muss wohlgeformt sein.
- ▶ Der Wert externer Entitäten muss nicht wohlgeformt sein, wohl aber das Dokument, nachdem sie ausgewertet wurden.
- ▶ Der Wert kann andere Entitäten enthalten.

Parameter Entities ermöglichen es, Strukturen wieder zu verwenden.

Entitäten deklarieren

Und so sehen sie aus:

```
<!ENTITY internal-entity "entity-value">
```

```
<!ENTITY external-entity SYSTEM  
    "http://www.example.org/entities.dtd">
```

Entitäten deklarieren

Und so sehen sie aus:

```
<!ENTITY internal-entity "entity-value">
```

```
<!ENTITY external-entity SYSTEM  
    "http://www.example.org/entities.dtd">
```

```
<!ENTITYY %parameter-entity "(some,(complex|construct))">
```


Teilmengen definieren

Es ist möglich, in einer internen DTD eine Teilmenge einer externen DTD zu definieren:

```
<!DOCTYPE BookStore SYSTEM "book.dtd" [  
    <!ELEMENT BookStore (Book | BookShelf)*>  
>
```

Teilmengen definieren

Es ist möglich, in einer internen DTD eine Teilmenge einer externen DTD zu definieren:

```
<!DOCTYPE BookStore SYSTEM "book.dtd" [  
    <!ELEMENT BookStore (Book | BookShelf)*>  
>
```

Beachte dabei:

- ▶ Elemente und Attribute können nicht verändert werden.
- ▶ Entitäten können überschrieben werden.