XML 2010

Erik Tjong Kim Sang
erikt(at)xs4all.nl

13 september 2010

Vorige week

Introductie tot XML:

- structuur XML-documenten
- tags, attributen, entities
- prolog, commentaar, CDATA
- werken met XML-documenten
- geschiedenis van XML

Deze week

Definitie van documentstructuren voor XML: DTDs en XML-Schemas

Introductie tot Linux

DTDs

Waarom moeten we documentstructuur definiëren?

In HTML is de reeks `<body> <li> item </li> </body>` illegaal

Waarom?

Een lijstelement mag niet op zichzelf staan maar moet in een lijstomgeving staan (`<ul>` of `<ol>`)

Als we een bepaalde tagvolgorde willen uitsluiten in XML dan zullen we die expliciet moeten definiëren

DTDs: document type definitions

De toegestane relaties tussen tags in een XML-document worden gedefinieerd in een documentmodel: DTD (document type definition)

In een DTD worden drie dingen gedefinieerd:

1. de namen van mogelijke documenttags
2. de mogelijke attributen van deze tags
3. de relaties tussen de documenttags

Hoe ziet een DTD eruit?

Een DTD is een apart document dat eruit ziet als een XML-document

Maar het heeft geen prolog nodig, tenzij je wil afwijken van de standaard-XML-versie (1.0) of de standaard-encoding (utf8)

Definities van documenttags

Definities van documenttags in een DTD zien er zo uit:

```
<!ELEMENT naam definitie>
```

Hierbij is `naam` de naam van de tag en `definitie` haar definitie

Voorbeelden:

```
<!ELEMENT beschrijving (#PCDATA)>
<!ELEMENT toneelstuk (tekst, beschrijving)>
```

`#PCDATA` = tekst met entities; `(tekst, beschrijving)` = een element `tekst` gevolgd door een element `beschrijving`

Reguliere expressies

Voor de definitie van tags gebruiken we reguliere expressies:

(a)	1 keer element element a
(a+)	1 of meer keren element a
(a*)	0 of meer keren element a
(a?)	0 of 1 keer element a
(a, b)	element a gevolgd door element b
(a | b)	element a of element b

Hiermee kunnen we de elementen van de toneelstukdocumenten uit de eerste praktische opdracht definiëren

Voorbeelden van elementdefinities (1)

De toneelstukelementen bevatten willekeurige aantalen elementen `tekst` en `beschrijving` in willekeurige volgorde:

```
<!ELEMENT toneelstuk (tekst | beschrijving)*>
```

Hiervoor past de definitie: 0 keer of vaker `tekst` of `beschrijving`

De tekstelementen bevatten zowel tekst (`#PCDATA`) als beschrijvingselementen:

```
<!ELEMENT tekst (#PCDATA | beschrijving)+>
```

Hiervoor past de definitie: 1 keer of vaker tekst of `beschrijving`

Voorbeelden van elementdefinities (2)

Verschillende reguliere expressies kunnen worden gecombineerd:

```
<!ELEMENT tekst (titel, auteurs*, (kopje, sectie)+,
                 literatuurlijst?)>
```

Dus volgens deze definitie bevat een tekst:

- altijd precies 1 titel
- nul of meer auteurs
- 1 of meer secties met steeds voor elke sectie een kopje
- maximaal 1 literatuurlijst

Bijzondere elementdefinities

```
<!ELEMENT br EMPTY>
```

Dit element bevat geen inhoud: gebruik het als `<br/>`

```
<!ELEMENT alles ALL>
```

Dit element kan alles als inhoud bevatten: `#PCDATA` en willekeurige tags (niet erg nuttig)

Definities van attributen

Attributen worden apart gedefinieerd in `<!ATTLIST`-elementen

De definities moeten de volgende delen bevatten:

1. de naam van de tag
2. de naam van het attribuut
3. het type van het attribuut
4. eisen aan het attribuut

Voorbeeld: `<!ATTLIST tekst spreker CDATA #REQUIRED>`

Mogelijke typen voor attributen

`CDATA`: willekeurige reeks tekens (character data)

`NMTOKEN`: tekenreeks die begint met een letter en geen spaties bevat (name token)

`NMTOKENS`: reeks van NMTOKENs gescheiden door spaties

`ID`: unieke NMTOKEN om naar tag te kunnen verwijzen

`IDREF`: als `ID` maar verwijst nu naar een bestaande `ID`

`ENTITY`: een verwijzing naar een bestaande entity

Mogelijke eisen aan attrributen

`#REQUIRED`: dit is een verplicht attribuut

`#IMPLIED`: dit is een optioneel attribuut

`waarde`: dit is een optioneel attribuut dat bij weglaten de waarde `waarde` krijgt

Een complete DTD voor de toneelstukken

```
<!ELEMENT toneelstuk (tekst | beschrijving)*>
<!ELEMENT tekst (#PCDATA | beschrijving)+>
<!ELEMENT beschrijving (#PCDATA)>
<!ATTLIST tekst spreker CDATA #REQUIRED>
```

Het gebruik van een DTD (1)

Het `<!DOCTYPE`-element bevat een verwijzing naar de DTD:

```
<!DOCTYPE toneelstuk
  SYSTEM "toneelstuk.dtd"
  [ <!ENTITY euml "ë" ] >
```

Op de regel van `SYSTEM` kunnen drie dingen staan

- niks: er wordt geen DTD gebruikt
- een URL: de DTD staat op die URL
- `SYSTEM "bestand"`: de DTD zit in dit lokale bestand

Het gebruik van een DTD (2)

Een uitgebreider voorbeeld van een prolog die verwijst naar een DTD uit Wikipedia:

```
<!DOCTYPE html
  PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Hier verwijst `PUBLIC` naar een algemene naam van de DTD voor het geval dat het adres van de DTD (de url) niet meer werkt

Validatie van documenten met DTDs

Het programma `xmllint` kan XML-documenten met DTDs controleren mits bij de aanroep optie `--valid` wordt gebruikt:

```
$ xmllint --valid -noout erikt.xml
erikt.xml:237: element beschr: validity error : No
declaration for element beschr
</beschr>
         ^
erikt.xml:238: element toneelstuk: validity error : Element
toneelstuk content does not follow the DTD, expecting (tekst |
beschrijving)*, got (beschrijving beschrijving tekst beschr )
</toneelstuk>
```


Namespaces

In de prolog van een XML-document kan maar 1 DTD worden gedefinieerd

Een XML-document kan tags uit verschillende DTDs bevatten mits daarvoor gebruik wordt gemaakt van namespaces

Een namespace is een plaatselijke verwijzing naar een DTD met een lokale naam

De naam van de namespace kan dan worden gebruikt om aan te geven dat een tag bij die namespace hoort: `<namespacenaam:tagnaam>`

Voorbeeld van gebruik namespaces

Hoe ziet een toneelstukdocument eruit als we namespaces gebruiken?

```
<ts:toneelstuk xmlns:ts="toneelstuk.dtd">
  <ts:beschrijving>
    ANTONIO, SALARINO _en_ SOLANIO _komen op._
  </ts:beschrijving>
  <ts:tekst spreker="ANTONIO">
```

In de roottag definiëren we de namespace `ts` en koppelen we het aan de DTD in het bestand `toneelstuk.dtd`
Vanaf de definitie kunnen we tags uit de DTD gebruiken mits we voor de namen van de tags `ts:` plaatsen

Wat ontbreekt er aan DTDs?

Met DTDs kunnen we tagnamen, attributen en relaties tussen tagnamen definiëren

Maar we kunnen geen retricties opleggen aan elementen die tekst (`#PCDATA`) bevatten

Voorbeelden van zulke restricties zijn: beperkingen in een element van de mogelijke woorden of de mogelijke getalswaarden

Voor zulke beperkingen hebben we een ander type docmentmodel nodig: XML Schema Definities (XSDs)

XML-SCHEMAS

XML-schema's

Schema's zijn een alternatieve manier om de structuur van een document te definiëren

De structuur van XML Schema definitie's (XSD) wordt sinds 1999 ontwikkeld door het World Wide Web Consortium (W3C) als alternatief voor DTDs

Voordelen van XSDs boven DTDs zijn de mogelijkheid voor inhoudsbeperkingen en een betere samenwerking met namespaces

Een voorbeeld van een XML-schema

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
  <element name="toneelstuk">
    <complexType>
      <sequence>
        <element name="beschrijving" type="string"/>
        <element name="tekst" type="mixed"/>
      </sequence>
    </complexType>
  </element>
</schema>
```

Dit schema definieert een toneelstuk met precies 1 beschrijving gevolgd door precies 1 uitgesproken tekst

Structuur van XML-schema's

XML-schema's zijn XML-bestanden en worden in aparte bestanden geplaatst

Deze bestanden beginnen met een XML-prolog (bijvoorbeeld `<?xml version="1.0"?>`)

De roottag in XML-schemabestanden is `<schema>`

Bij deze tag horen verschillende attributen waarvan we op dit moment alleen de namespace zullen gebruiken:
`xmlns="http://www.w3.org/2001/XMLSchema"`

Elementdefinities in schema's

Tags (elementen) worden gedefinieerd met de tag `<element>` en het attribuut `name`. Er zijn hiervoor twee varianten:

Inhoudsdefinitie door verwijzing naar type:

```
<element name="beschrijving" type="string"/>
```

Lokale inhoudsbeschrijving, bijvoorbeeld met andere elementsdefinities:

```
<element name="toneelstuk">
   <complexType>
   </complexType>
</element>
```


Welke elementstypen zijn er?

In XML-schema's bestaan de volgende elementstypes:

- string: te vergelijken met `#PCDATA` in DTDs
- zelfgefinieerde types (zie hierverderop)

De definitietag `complextype`

Met de definitietag `complextype` kunnen reeksen van elementen worden gedefinieerd met de volgende contentmodellen:

- `<sequence>`: een reeks elementen in vaste volgorde
- `<choice>`: 1 van de genoemde elementen
- `<all>`: een reeks elementen in vrije volgorde

Het attribuut `mixed="true"` geeft aan dat tussen de genoemde elementen ook tekst voor kan komen

Herhaalde elementen

In een DTD wordt herhaling van elementen aangegeven door middel van reguliere expressies: *, + en ?

In schema's wordt herhaling aangegeven door middel van twee attributen in de elementsdefinities:

- `minOccurs`: minimum aantal keren dat het element mag voorkomen
- `maxOccurs`: maximum aantal keren dat het element mag voorkomen

Het default aantal waarden voor beide attributen is 1

Voorbeelden van `minOccurs` en `maxOccurs`

```
<element name="toneelstuk" minOccurs="2" maxOccurs="3">
```

Dit element moet 2 of 3 keer voorkomen

```
<sequence minOccurs="0" maxOccurs="unbounded">
   <choice>
      <element name="a">
      <element name="b">
   </choice>
</sequence>
```

Een willekeurig aantal keren elementen `<a>` en `<b>` in willekeurige volgorde

Definities van attributen

Attributen worden gedefinieerd met een `attribuut`-tag binnen een `complexType`-element, bijvoorbeeld:

```
<element name="tekst">
  <complexType mixed="true">
    ...
    <attribute name="spreker" type="string"/>
  </complexType>
</element>
```

De definitie bevat de naam en het type van het attribuut

Voorbeeld van een compleet schema

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
  <element name="toneelstuk">
    <complexType>
      <choice maxOccurs="unbounded">
        <element name="beschrijving" type="string"/>
        <element name="tekst">
          <complexType mixed="true">
            <choice minOccurs="0" maxOccurs="unbounded">
              <element name="beschrijving" type="string"/>
            </choice>
            <attribute name="spreker" type="string"/>
          </complexType>
        </element>
```


Achtergrondliteratuur

Tim Bray, Jean Paoli, C.M. Sperberg-McQueen, Eve Maler and François Yergeau, *Extensible Markup Language (XML) 1.0* (Fifth Edition). W3C, 26-11-2008 http://www.w3.org/TR/2008/REC-xml-20081126/

David Hunter, Jeff Rafter, Joe Fawcett and Eric van der Vlist, *Beginning XML (Programmer to Programmer)*. John Wiley & Sons, 2004 (chapter 6)

Erik T. Ray, *Learning XML*, O'Reilly, 2001 (chapter 5)

W3schools, *DTD Tutorial*, http://www.w3schools.com/dtd/

W3schools, *Schema Tutorial*, http://www.w3schools.com/schema/

INTRODUCTIE TOT LINUX