Validation - How to make XML documents meaningful and consistent

Simon Mahony
From an original document by Susan Hockey

This document is part of a collection of presentations and exercises on XML. For full details of this and the rest of the collection see the cover sheet at:

OPEN EDUCATIONAL RESOURCES FOR THE DIGITAL HUMANITIES

XML Document Structure

A Well-formed XML document is a nested structure (tree) of elements

- Elements can contain
  - attributes
  - other elements
  - text (the leaves of the tree) - including entities
  - mixed content – text and/or other elements
- Elements can be repeated
- Elements can be optional

XML Document Definition

- XML Declaration:

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

- Document Type Declaration (DTD)
  - is an instruction that associates a particular XML document with a Document Type Definition (DTD)

    <!DOCTYPE memo […….] >

- Document Type Definition (DTD)
  - What goes into the Document Type Declaration!
  - Can be within the XML document or a reference to an external file

Valid Documents

- The Document Type Definition (DTD) forces tighter control on an XML document
- The DTD is a formal specification of the document structure
- The DTD defines which elements are allowed in the document and where they can occur
- Documents are validated against the DTD by a special program, often called a parser
- The DTD eliminates some types of errors

Other Ways to Validate XML Documents

- XML Schema
- RelaxNG (REgular LAnguage for XML Next Generation)
- Schematron
- DSDL – Document Schema Definition Languages.
  - Umbrella standard for different validation schemes

DTD Syntax

- A language for document definition
- Origins in SGML
- Declare every element and every attribute
- Specify the content model for every element

Content Model – What is Inside the Element

- The content model gives the order of the nested elements
- Specifies whether they can be repeated
- Whether mandatory/optional
- Attributes and values
- An element can also be empty

Element Declaration Example 1

Element with two elements directly inside it

```
<!DOCTYPE memo [
<!ELEMENT memo (heading,bodytext)>
]>
```

A <memo> consists of one <heading> followed by one <bodytext>

Document Skeleton Example 1

<heading> and <bodytext> are the two elements immediately inside <memo>.

<memo>
<heading> … contents … </heading>
<bodytext> ... contents… </bodytext>
</memo>

The contents of <heading> and <bodytext> must also be defined

Element Declaration Example 2

Element with three elements directly inside it

<!DOCTYPE book [

<!ELEMENT book (frontmatter,body,backmatter)>

]>

A <book> consists of one <frontmatter> followed by one <body> followed by one <backmatter>

Document Skeleton Example 2

<book>
<frontmatter> … content … </frontmatter>
<body> … content … </body>
<backmatter> … content … </backmatter>
</book>

Element Declaration for Text Content

<!ELEMENT name (#PCDATA)>
<!ELEMENT title (#PCDATA)>

#PCDATA ("Parsed Character DATA") is text only, with no further elements inside it, i.e. the leaves of the tree

<name>Jane Austen</name>
<title>Pride and Prejudice</title>

Empty Elements

Use the keyword EMPTY in the definition

<!ELEMENT pagebreak EMPTY>

<!ELEMENT image EMPTY>

<pagebreak n="34"/>

<image filename="picture.jpg"/>

Note: can be written as:

<pagebreak n="34" /> to be compatible with old browsers

Occurrence Indicators

- Whether and how elements can repeat
- Follow the name of the element
- No occurrence indicator means the element must appear once and only once

Occurrence Indicators

+ the element can appear one or more times

* the element can appear zero or more times

? the element is optional; it can appear once or not at all

Occurrence Indicator Example

<!ELEMENT section (p+)>

The element <section> contains one or more instances of the element . This means that immediately inside <section> are one or more s.

<section>
 some text or more elements
 some text or more elements
</section>

Occurrence Indicator Example

<!ELEMENT memo (heading?,bodytext)>

The element <memo> contains an optional <heading> followed by one <bodytext>

Another Example

<!ELEMENT chapter (heading*,p+)>

The element <chapter> contains zero or more instances of <heading> followed by one or more instances of <p>

Markup Examples

<!ELEMENT chapter (heading*,p+)>

Would allow this markup

<chapter>
<heading>this is a heading</heading>
paragraph 1
</chapter>

Markup Example

<!ELEMENT chapter (heading*,p+)>

Would allow this markup

<chapter>
paragraph 1
</chapter>

Markup Example

<!ELEMENT chapter (heading*,p+)>

Would allow this markup

<chapter>
<heading>this is a heading</heading>
paragraph 1
paragraph 2
</chapter>

Markup Example

<!ELEMENT chapter (heading*,p+)>

Would allow this markup

<chapter>
paragraph 1
paragraph 2
paragraph 3
</chapter>

Markup Example

<!ELEMENT chapter (heading*,p+)>

Would allow this markup

<chapter>
<heading>heading 1</heading>
<heading>heading 2</heading>
paragraph 1
</chapter>

Connectors – Sequences of Elements

- Separate the elements in the content model
- Indicate the order in which the elements must appear

, the elements before and after the comma must appear in the order given

| specifies alternatives

Examples of Sequences of Elements

<!ELEMENT salutation (sender,recipient,date)>

The element <salutation> consists of one <sender> followed by one <recipient> followed by one <date>

Sequence Example 1 - CORRECT

<!ELEMENT salutation (sender,recipient,date)>

<salutation>
<sender> Janet</sender>
<recipient>John</recipient>
<date>27 January 2011</date>
</salutation>

Sequence Example 1- INCORRECT

```
<!ELEMENT salutation (sender,recipient,date)>

<salutation>
<sender> Janet</sender>
<date>27 January 2011</date>
<recipient>John</recipient>
</salutation>
```

Alternatives

<!ELEMENT sender (fullname | lastname)>

The element <sender> consists of <fullname> or <lastname>

Alternatives Example

```
<sender>
<fullname>Jane Austen</fullname>
</sender>

<sender>
<lastname>Austen</lastname>
</sender>
```

Grouping Elements

( ) indicates a grouping

<!ELEMENT anthology (intro?, (poem|essay)+)>

The element <anthology> contains an optional <intro>, followed by <poem> or <essay> repeated one or more times

Grouping Example 1

<!ELEMENT anthology (intro?, (poem|essay)+)>

<anthology>
<intro> … </intro>
<poem> … </poem>
<poem> … </poem>
<essay> … </essay>
</anthology>

Grouping Example 2

<!ELEMENT anthology (intro?, (poem|essay)+)>

<anthology>
<intro> … </intro>
<poem> … </poem>
<essay> … </essay>
<poem> … </poem>
</anthology>

Grouping Example 3

```
<!ELEMENT anthology (intro?, (poem|essay)+)>

<anthology>
<poem> … </poem>
<poem> … </poem>
<poem> … </poem>
<poem> … </poem>
</anthology>
```

Mixed Content

Define as repeatable alternatives with #PCDATA always as the first alternative

```
<!ELEMENT p (#PCDATA|title|author)*>
```

```
<p>The novel <title>Pride and Prejudice</title> by
  <author>Jane Austen</author> is a good
  read.</p>
```

Simple Attributes

Attributes are declared with !ATTLIST associated with an element declaration

```
<!ELEMENT title (#PCDATA)>
<!ATTLIST title type CDATA #IMPLIED>
```

The element <title> has an attribute type whose value is any string – CDATA

```
<title type="article">Article about XML</title>
```

Element with More Than One Attribute

```
<!ELEMENT title (#PCDATA)
<!ATTLIST title type CDATA #IMPLIED
                dateofpub CDATA #IMPLIED>
```

The element <title> has attributes type and dateofpub

Example Markup - Attributes

```
<title type="article" dateofpub="1998">Article about
   XML</title>

<title type="chapter" dateofpub="2003">Another
   article about XML</title>
```

Required Attribute

An attribute can be declared as required and must not then be omitted

```
<!ELEMENT title (#PCDATA)>
<!ATTLIST title type CDATA
  dateofpub CDATA #REQUIRED>
```

```
<title type="article">Article about XML</title>
```

would be then an error (ie invalid)

Default Attribute Value

A default attribute value can be set

```
<!ELEMENT title (#PCDATA)>
<!ATTLIST title type CDATA
                dateofpub CDATA "1998">
```

The attribute dateofpub on the element <title> is assumed to be 1998 if not given

DOCTYPE Declaration

- The DOCTYPE declaration identifies the DTD
- The syntax of the DOCTYPE declaration depends on where the DTD is stored
- The name of the DOCTYPE is the same as the outer **(root)** element of the document.

Storing DTDs

- The DTD can be embedded at the beginning of an XML document (an internal DTD)
- The DTD can be a file on your own computer

```
<!DOCTYPE memo SYSTEM "memo.dtd">
```

- The DTD can also be a file stored on the network

```
<!DOCTYPE memo SYSTEM "file://x:/memo.dtd">
```

- Can also be a ‘public’ location e.g.

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

Skeleton Example with an Internal DTD

```
<?xml version="1.0"? >
<!DOCTYPE memo [
    ….element declarations
]>
<memo>
        …. contents of memo
</memo>
```

The name of the DOCTYPE is the same as the name of the root element

Example 3.1: DTD in a document (a memo)

```
<?xml version="1.0"?>
<!DOCTYPE memo [
<!ELEMENT memo (heading, body)>
<!ATTLIST memo type CDATA  #REQUIRED>
<!ELEMENT heading (sender,recipient,date)>
<!ELEMENT sender (#PCDATA)>
<!ELEMENT recipient (#PCDATA)>
<!ELEMENT date (#PCDATA)>
<!ELEMENT body (title,p)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT p (#PCDATA)>
]>
<memo type="business">
        <heading>
                <sender>Janet</sender>
                <recipient>John</recipient>
                <date>2 Feb</date>
        </heading>
         <body>
                <title>title of memo</title>
                <p>Paragraph</p>
         </body>
</memo>
```

Example 3.2: DTD in document (some books)

```
<?xml version="1.0"?>
<!DOCTYPE books [
<!ELEMENT books (book+)>
<!ELEMENT book (author+, title,datepub, placepub, publisher)>
<!ELEMENT author (lastname, firstname)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT datepub (#PCDATA)>
<!ELEMENT placepub (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
]>
<books>
        <book>
                <author>
                <lastname>Hockey</lastname>
                <firstname>Susan</firstname>
                </author>
                <title>Electronic Texts in the Humanities</title>
                <datepub>2000</datepub>
                <placepub>Oxford</placepub>
                <publisher>Oxford University Press</publisher>
        </book>
        <book>
                <author>
                <lastname>Peek</lastname>
                <firstname>Robin P.</firstname>
                </author>
                <author>
                <lastname>Newby</lastname>
                <firstname>Gregory B.</firstname>
                </author>
                <title>Scholarly publishing: the electronic frontier</title>
                <datepub>1996</datepub>
                <placepub>Cambridge: Mass</placepub>
                <publisher>MIT Press</publisher>
        </book>
</books>
```

Validating Documents

- Internet Explorer or Firefox does not validate a document against a DTD
- Go to http://www.stg.brown.edu/pub/xmlvalid
  for a free validator
- Or use an XML Editor
  - Helps you create your markup
  - Reads the DTD and prompts for markup