Attributes & Values

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:

Session 5

- More on:
- Attributes
- Values

- Looking ahead
- Case study as an example

Attributes

- Attributes store information about the text not the text itself

- Required: #REQUIRED
- Implied (optional): #IMPLIED
- Has default value: "value"
- Has fixed value: #FIXED "value"
- Has choices: (… | …)

Required: #REQUIRED (mandatory)

DTD

```
<!ELEMENT letter (addresse, date, body)
<!ATTLIST letter type CDATA #REQUIRED>
```

XML

```
<letter type="accept">
        <addressee> address</addressee>
        <date> date</date>
        <body> body of the letter </body>
</letter>
```

Implied: #IMPLIED (optional)

DTD

<!ELEMENT ingredient_amnt (#PCDATA) >

<!ATTLIST ingredient_amnt unit CDATA #IMPLIED >

XML

<ingredient-amnt unit="oz"> 6 </ingredient-amnt>

<ingredient-amnt unit="grams"> 175 </ingredient-amnt>

or nothing (fine if you have no data for that value)

Has default value:

DTD

<!ELEMENT ingredient_amnt (#PCDATA) >
<!ATTLIST ingredient_amnt
unit CDATA "grams" >

Valid XML:

<ingredient-amnt unit="grams"> 175 </ingredient-amnt>
<ingredient-amnt unit="oz"> 6 </ingredient-amnt>
<ingredient-amnt > 175 </ingredient-amnt>

- If no value is set, the parser will assume grams

Has fixed value:

DTD

```
<!ELEMENT ingredient_amnt (#PCDATA) >
 <!ATTLIST ingredient_amnt
   unit CDATA #FIXED "grams" >
```

Valid XML:

```
<ingredient-amnt unit="grams"> 175 </ingredient-amnt>
<ingredient-amnt > 175 </ingredient-amnt>
```

Invalid XML:

```
<ingredient-amnt unit="oz"> 6 </ingredient-amnt>
```

Attributes with choices

DTD

```
<!ELEMENT ingredient_amnt (#PCDATA) >
<!ATTLIST ingredient_amnt
        unit (oz | gram) CDATA #REQUIRED>
```

Valid XML

```
<ingredient-amnt unit="grams"> 175 </ingredient-amnt>
<ingredient-amnt unit="oz"> 6 </ingredient-amnt>
```

Invalid XML:

```
<ingredient-amnt unit="kilo"> 1 </ingredient-amnt>
<ingredient-amnt > 175 </ingredient-amnt>
```

REPEATED ATTRIBUTES
but not ="values"

```
<cooking>
      <timing>
        <time>65 minutes</time>
    <oven-temp unit="gas mark">8</oven-temp>
    <oven-temp unit="fahrenheit">450</oven-
   temp>    <oven-temp
   unit="centigrade">230</oven-temp>
      </timing>
</cooking>
```

Attribute values

- Always use matching quotes
- ie both " … " and ' … ' are legal
- Recommend always using " … "
- Saves having to remember which was used
- Unless value contains " then must use ' … '

<module class="simon's class">

Not <module class='simon's class'>

Attributes with unique values (IDs)

Note: can be either #REQUIRED or #IMPLIED

(#REQUIRED makes more sense!)

Allows you to apply a unique identifier (unique value)

DTD

```
<!ELEMENT name  (#PCDATA)>
<!ATTLIST  name ID CDATA #REQUIRED>
```

XML

```
<name ID="101">Simon</name>
```

Remember (cf XHTML):

- Attribute follows rules for XML names
- No spaces in attribute names and case sensitive

- Value of ID attribute must be unique within document
- Only ONE attribute ID per element
- Attribute declaration for an ID attribute must be:
  - #REQUIRED
  - #IMPLIED
  - (#FIXED makes no sense)
- Can be used to identify a part of the document

CDATA

- Character DATA: text non parsed by processor
- #PCDATA: Parsed Character Data
- CDATA used for attribute values
  and text that you don't want parsed by the processor

Display XML elements as text

To display an example of XML in the XML (create a CDATA section):

```
<! [ CDATA [
    <recipes>
    <recipe>
        <title ></title>
        <author>
    <last_name></last_name>
        </author>
        <ingredients>
    <ingredient_name></ingredient_name>
        </ingredients>
        <process></process>
    </recipe>
    </recipes>
] ] !>
```

Problems with DTDs

- Not written using XML syntax: require parsing
- Poor data typing – to ensure correct data (integer, date, string etc)
- Limited capacity to define content model
  - branches of the tree / leaves
- Poor support for XML Namespaces

XML Namespace

Potential problem when combining XML docs

<ELEMENT title #PCDATA>

<title> Pride and Prejudice</title>

<title> Mr</title> <lastname>Darcy</lastname>

<title> Document name</title>

Elements with same name but different meaning and semantics. We understand the difference from the context. To the parser they all are the same.

Namespaces

Concerned with VOLCABULARY not document type

- A conceptual grouping of terms
- Categories

- A means to distinguish one XML vocabulary from another
- Ability to uniquely identify a specific vocabulary
- Usually using a URL (not to point to but as an ID)

<recipes xmlns="http://www.simon.mahony.org/names">

- We will see this later in XSLT

Recap

- Document analysis
  - tree structure
    - trunk / branch(es) / leaves
    - parent / child / sibling
- Well-formed XML
- Valid XML
- Comments: annotate your XML (also XSLT & CSS)
  - Document the structure of your XML doc
  - Help find problems (by excluding chunks from parsing)

Second half module

- XML Schema
- Transformations: XSLT
- CSS: for appearance and layout
- XPath: address a specific part of XML document
- More transformations with XSLT
- Other XML standards and applications

A case study

- The Inscriptions of Roman Tripolitania
- XML Repository
  - All XML files in ZIP archive
  - EpiDoc DTD
- Individual inscription (search box)
  - Web display
  - Print preview
  - View Source
  - EpiDoc XML and EpiDoc DTD to validate
- All available for reuse under CC license