

XML & Allied Technologies



Writing XML and Designing DTD's



XML Syntax Contains

- **Elements**
 - ✓ XML tags for markup.
- **Attributes**
 - ✓ Tuple information of elements.
- **Declarations**
 - ✓ Instructions to XML processor.
- **Processing Instructions**
 - ✓ Instructions to external applications.



Processing Instructions

- Information required by an external application.
- Processing Instructions
 - Format => `<? ... ?>`
 - XML PI => `<?xml-stylesheet?>`



XML Document

```
<booklist title="Some XML Books">
  <book>
    <author>
      <name>St. Laurent</name>
      <initial>S</initial>
    </author>
    <date>1998</date>
    <title edition="Second">XML: A Primer</title>
    <publisher>MIS Press</publisher>
    <website href="http://www.simonstl.com" />
    <rating stars="4"/>
  </book>
</booklist>
```



What is a DTD?

- A template for document markup :
 - ✓ A file which contains a **formal definition** of a particular type of document.
 - ✓ A file that **constrains** or **restricts** certain elements and attributes to exist in XML document.
- A DTD Specifies :
 - ✓ **Document hierarchy.**
 - ✓ What **names** can be used for element types.
 - ✓ Where & how element types can **occur.**
 - ✓ Names and types of element **attributes.**



Why have a DTD?

- DTD can be a mechanism for **standardization**
- DTD declares all allowed components of XML document
- Validating XML parser
 - ✓ Check that the file is **well-formed** .
 - ✓ Check that its structure is **valid against a DTD**.



Internal DTD Definition

- Included in the DOCTYPE declaration

```
<?xml version="1.0"?>  
<!DOCTYPE root _ elem  
[  
  <!-- DTD appears here -->  
  <... >  
  < ... >  
>  
<!-- Rest of XML file -->
```



Internal DTD Definition

```
<?xml version="1.0"?>
<!DOCTYPE note [
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend</body>
</note>
```




External DTD Definition

- **Document type declaration:** links the DTD file with the XML document.

```
<!DOCTYPE rootelem SYSTEM "file.dtd">
```



External DTD Definition

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```



Designing a DTD

- **Identify features of the data that need markup.**
- Usually we define **the DTD first** , then do the xml that validate the rules in the DTD.
- We can write **1 DTD file to validate more than 1 xml document.**
- We **can't validate 1 xml document against more than 1 DTD file.**



Element Declarations

- Used to define new elements and their content
⇒ `<name>my name</name>`
`<!ELEMENT name (#PCDATA)>`
- Empty element has **no content** ⇒ `<name/>`
`<!ELEMENT name EMPTY>`
- When children are allowed, use model groups
`<!ELEMENT person (name, e-mail)>`
- When declaring an element that can contain any type of content declared within the DTD
`<!ELEMENT DOCUMENT ANY>`



Model Group Quantity Indicators

- Describe constraints on elements in DTD:

A? May occur **[0 or 1]**

A+ **Must** occur **[1..*]**

A* May occur **[0..*]**

A, B A followed by B

A | B Either A or B

- <!ELEMENT father (daughter, son*)>**



Mixed content Declarations

- Used to define Mixed content elements:

<!ELEMENT name (#PCDATA | choice2 | choice3)*>

⇒ <name>my_name

<choice2>my_choice2</choice2>

<choice3>my_choice2</choice3>

</name>



Mixed content Declarations (cont.)

- **When using mixed content :**
 - **Sub-elements** *may and may not occur*.
 - **Can't specify** the *number of occurrences* or *the order* of appearance of the child elements .
 - **Can't specify** the place of *the text* (PCDATA).

So Mixed Content is to kept as minimum as possible .



Attribute Declarations

- Attributes can be attached to elements.
- Declared separately in an “**ATTLIST**” declaration.
- Attribute name
 - **<!ATTLIST** tag attr_name type default>
 - **<!ATTLIST** tag first_attr ...
 secon_attr ...
 third_attr ... >
- Attribute types include: **CDATA**, **Enumeration**, **NMTOKEN**, **NMTOKENS** and **ID**.



Default Attribute Values

- Can specify a default attribute value for when it is missing from XML document, or state that value must be entered
 - ✓ **#REQUIRED** Must be specified.
 - ✓ **#IMPLIED** May be specified.
 - ✓ *"default "* Default value if unspecified.
 - ✓ **#FIXED** Only one value allowed.
- `<!ATTLIST son sports (football|basketball) "football">`
- `<!ATTLIST name age CDATA #REQUIRED>`



Attribute types

- **CDATA** : simple character data .
- **Enumeration** : list of values (**case sensitive**)

```
<!ATTLIST son sport( football | tennis)  
           "tennis">
```

```
<!ATTLIST son sport( football | tennis)  
           #REQUIRED>
```

- **NMTOKEN** :

```
<!ATTLIST son job NMTOKEN #IMPLIED>
```



Attribute types (cont.)

➤ ID :

- Declaring attribute type ID we can **uniquely identify** certain element.
(**Note:** attribute doesn't have to be named id)
- Attribute of type ID should
 - ✓ have to be unique.
 - ✓ contain only characters permitted for NMTOKEN.
 - ✓ start with a letter. (ex: C101)
- No element type may have more than one ID.

<!ATTLIST invoice num ID #REQUIRED>



Entities

- XML document may be distributed among units of information:
 - ✓ Each **unit of information** is called an entity.
 - ✓ Each entity has a **name** to identify it.
 - ✓ Defined using an entity declaration.
 - ✓ Used by calling an entity reference.



General Entities

- Declared in 'Document Type Definition'

<!ENTITY entity_name *"replacement text">*

<!ENTITY xml *"eXtensible Markup Language">*

<!ENTITY copyright *"Copyright ITI.">*

✓ **In xml document:**

The **&xml;** and **©right;** includes entities.

✓ **Equivalent to???**



Restrictions on Entities

- **General text entities**

- ✓ Can appear in **element content**

`<para> ... &ent; ... </para>`

- ✓ Can appear in **attribute value**

`<para name="&ent;"> ... </para>`

- ✓ Can appear in **internal entity content**

`<!ENTITY cod "&ent;">`



Putting it all together...

- Have now been introduced to **the main components and rules** of XML and DTD's.
- Entities, elements, declarations, processing instructions, attribute lists.
- Use all these components in the 'Document Type Definition' (DTD) to specify the rules about the format of the XML document.



Disadv. Of DTDs

- DTD has its **own syntax**.
- **Can't precise** number of element repetitions.
- Limited number of **data types**.
- **Only 1 DTD** can be referenced from within the XML document.



XML Break.....

- Write a DTD for a sports training markup language. The following are the major pieces of information that are associated with each individual training session:





XML Break (cont.)

- **Date** : The date and time of the training session.
- **Type** : the type of training session (running, swimming, cycling, and so on).
- **Heart rate**: the average heart rate sustained during the training session.
- **Duration** : the duration of the training session.
- **Distance**: the distance covered in the training session (measured in miles or kilometers)
- **Location**: the location of the training session
- **Comments**: general comments about the training session