DTD (Document Type Definition)

Introduction

- A Document Type Definition (DTD) defines the legal building blocks of an XML document.
- It defines the document structure with a list of legal elements and attributes.
- A DTD can be declared inline inside an XML document, or as an external reference.

How to include DTD in xml

- If the DTD is declared inside the XML file, it should be wrapped in a DOCTYPE definition with the following syntax:
  - <!DOCTYPE root-element [element-declarations]>
- If the DTD is declared in an external file, it should be wrapped in a DOCTYPE definition with the following syntax:
  - <!DOCTYPE root-element SYSTEM "filename">

Building blocks of XML

- XML documents (and HTML documents) are made up by the following building blocks:
  - Elements
  - Attributes
  - Entities
  - PCDATA
  - CDATA

PCDATA & CDATA

- PCDATA is text that WILL be parsed by a parser.
- Tags inside the text will be treated as markup and entities will be expanded.
- CDATA Character data is the text found between the start tag and the end tag of an XML element.
- CDATA is text that will NOT be parsed by a parser.
- Tags inside the text will NOT be treated as markup and entities will not be expanded.

DTD ELEMENTS

Declaration of Element

- Empty elements are declared with the category keyword EMPTY
- Like br, common elements are available such as bool, photo, hr, logo and so on.

```
<!ELEMENT element-name EMPTY>

Example:
<!ELEMENT br EMPTY>

XML example:
<br/>
```

Elements with PCDATA

- Elements with only parsed character data are declared with #PCDATA inside parentheses:

<!ELEMENT element-name (#PCDATA)>

Example:

  - <!ELEMENT from (#PCDATA)>

Elements with Children

- Elements with one or more children are declared with the name of the children elements inside parentheses:
  - <!ELEMENT element-name (child1)>

    or
  - <!ELEMENT element-name (child1,child2,...)>

Example:

- <!ELEMENT note to,from,heading,body)>

When children are declared in a sequence separated by commas, the children must appear in the same sequence.

Occurrence of Element

- One Occurences
- One or More Occurences -- +
- Zero or More Occurences - *
- Zero or One Occurences -- ?
- Either / Or ---------- |

Element with one occurrence

- The example below declares that the child element "message" must occur once, and only once inside the "note" element.

  - <!ELEMENT element-name (child-name)>

Example:

  - **<!ELEMENT note (message)>**

Element with one or more occurrence

- The + sign declares that the child element "message" must occur one or more times inside the "note" element.

  - <!ELEMENT element-name (child-name+)>

Example:

  - **<!ELEMENT note (message+)>**

Element with zero or more occurrence

- The * sign declares that the child element "message" must occur zero or more times inside the "note" element.
  - <!ELEMENT element-name (child-name*)>

Example:

  - **<!ELEMENT note (message*)>**

Element with either or content

- "note" element must contain either a "message" or a "body" element.

  - <!ELEMENT note (message | body)>

Example:

  - **<!ELEMENT note (to,from,header,(message | body))>**

DTD Attributes

Attribute Declaration

- Attributes are declared with an ATTLIST declaration.
  - <!ATTLIST element-name attribute-name attribute-type default-value>
- DTD example:
  - <!ATTLIST payment type CDATA "check">
- XML example:
  - <payment type="check" />

Default value

Value	Explanation
value	The default value of the attribute
#REQUIRED	The attribute is required
#IMPLIED	The attribute is not required
#FIXED value	The attribute value is fixed

Attribute Types

Type	Description
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

ID Attribute Type

- The value of the ID attribute must not appear more than once throughout the XML document.
- ID resembles the primary key concept used in databases.
  - <ATTLIST question no ID #REQUIRED>
  - <ATTLIST employee id ID #REQUIRED>
  - <ATTLIST car serial ID #REQUIRED>

IDREF Attribute

- Similar to foreign key concept used in databases. IDREF is used to establish connection between elements.
- IDREF value of the attribute must refer to an ID value declared elsewhere in the document.
- <ATTLIST question no ID #REQUIRED>
- <ATTLIST answer qno IDREF #REQUIRED>

…CONTD

```
<question no =“Q1”>
   What is the full form of DTD
</question>
<question no =“Q2”>
   What is the full form of XML
</question>
<answer qno=“Q1”>
   Document Type Definition
</answer>
```

IDREFS Attribute

- List of ID values separated by white spaces
  - <ATTLIST student roll ID #REQUIRED>
  - <ATTLIST answer sid ID #REQUIRED>
  - <ATTLIST marks ref IDREFS #REQUIRED>
- Example
  - <student rollno='r01'>Samir</student>
  - <subject sid='s1'>Web Technology</subject>
  - <marks ref='r01 s1'>82</marks>
- In this way connections among set of elements can be established with IDREF attribute

…Contd

- List of ID values separated by white spaces
  - <ATTLIST employee id ID #REQUIRED>
  - <ATTLIST managers emp IDREFS #REQUIRED>

<employee id='e01'>Rahul</employee>

<employee id='e02'>Mithra</employee>

<employee id='e03'>Sharma</employee>

<managers emp='e01 e03'/>

- Rahul and Sharma are Managers

DTD: Default value

- <!ELEMENT square EMPTY>
- <!ATTLIST square width CDATA "0">

- Valid XML:
  - <square width="100" />
- How to use the default value for square element?

- DTD:
  - <!ATTLIST person number CDATA #REQUIRED>
- Valid XML:
  - <person number="5677" />
- Invalid XML:
  - <person />

- DTD:
  - <!ATTLIST contact fax CDATA #IMPLIED>
- Valid XML:
  - <contact fax="555-667788" />
- Valid XML:
  - <contact />
- DTD:
  - <!ATTLIST sender company CDATA #FIXED "Microsoft">
- Valid XML:
  - <sender company="Microsoft" />
- Invalid XML:
  - <sender company="W3Schools" />

Attribute with enumerated values

- Syntax:
- <!ATTLIST element-name attribute-name (en1|en2|..) default-value>
- DTD:
  - <!ATTLIST payment type (check|cash) "cash">
- XML example:
  - <payment type="check" />

    OR
  - <payment type="cash" />

Valid XML document

```
<?xml version="1.0" ?>
<!DOCTYPE course SYSTEM
"C:\dtds\course.dtd">
<course>
<code>FYS355</code>
<title>Space Physics II</title>
<credit>3 credits</credit>
<tuition>
<day>Tuesday</day>
<place>lecture room V204</place>
<lecturer>Vega Thronfield</lecturer>
<start>First lecture: 22 January</start>
</tuition>
</course>
```

course.xml

```
<!ELEMENT course (code, title, credit,
tuition)>
<!ELEMENT code (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT credit (#PCDATA)>
<!ELEMENT tuition (day, time, place,
        lecturer, start)>
<!ELEMENT day (#PCDATA)>
<!ELEMENT time (#PCDATA)>
<!ELEMENT place (#PCDATA)>
<!ELEMENT lecturer (#PCDATA)>
<!ELEMENT start (#PCDATA)>
```

course.dtd

DTD Entities

- Entities are variables used to define shortcuts to standard text or special characters.
- Entity references are references to entities
- Entities can be declared internal or external

Internal Entity Declaration

- An entity has three parts: an ampersand (&), an entity name, and a semicolon (;)
  - Syntax: <!ENTITY entity-name "entity-value">
- DTD Example:
  - <!ENTITY writer "Donald Duck.">
  - <!ENTITY copyright "Copyright W3Schools.">
- XML example:
  - <author>&writer;&copyright;</author>
- This will be interpreted as
  - <author> Donald Duck Copyright W3Schools</author>

External Entity Declaration

- This is used for the long replacement of text that is kept in another file
- Useful in creating common references that can be shared across multiple documents’
- Changes made in the external entities are automatically updated in the documents they are referenced
  - <!ENTITY entity-name SYSTEM ‘URI’>
  - <!ENTITY author SYSTEM ‘author.xml’>

- Author.xml
  - <firstname>Uttam</firstname>
  - <middlename>Kumar</middlename>
  - <lastname>Roy</lastname>
- Another.xml
  - <book>&author;</book> will produde the following
- <book>
  - <firstname>Uttam</firstname>
  - <middlename>Kumar</middlename>
  - <lastname>Roy</lastname>
- </book>

Internal DTD Declarartion

```
<?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 Declaration

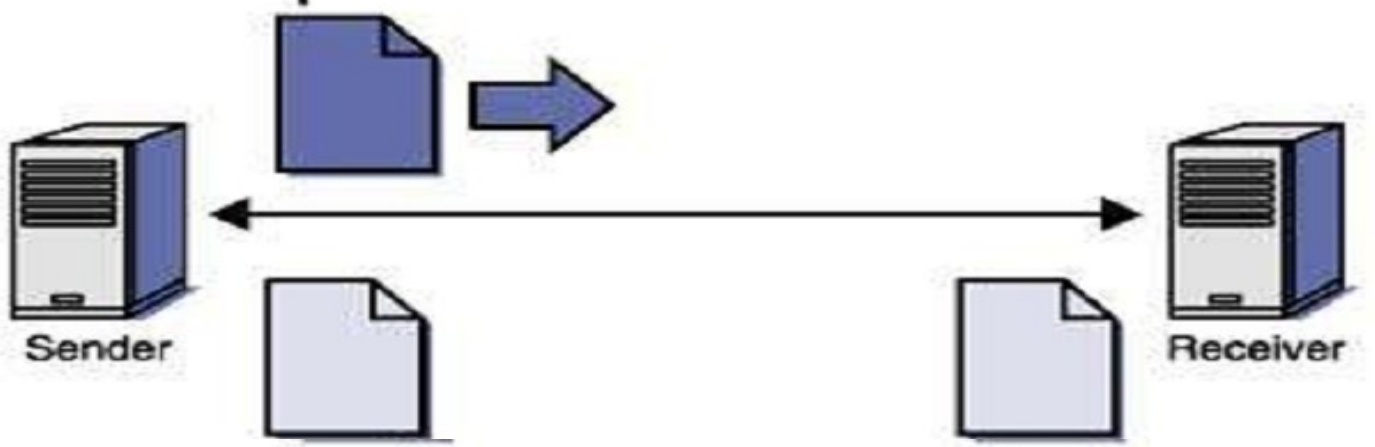
```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

```
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

- DTD can be used by both the sender and receiver.
- Sender – uses the DTD to create valid XML documents.
- Receiver – uses the DTD to check the validity of incoming
- XML document and determine if the incoming XML data is valid with respect to DTD.

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<book>
    <isbn>0-596-00058-8</isbn>
    <title>XML in a Nutshell</title>
    <author>Harold, Elliotte Rusty</author>
</book>
```



Sender uses the DTD to prepare a valid XML document

Receiver uses the DTD to check the validity of incoming XML

DTD

```
<!ELEMENT book (isbn, title, author)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT isbn (#PCDATA)>
<!ELEMENT title (#PCDATA)>
```

Limitations of DTD

- DTDs are written using a different syntax from XML.
- DTDs are not able to define distinctions about data types.
- A DTD is limited to declaring that an element must contain text.
  - Example: no distinctions among numeric or alphabet
- Solution:
  - XML Schema is evolved