An Introduction to XML

By Suresh Kumar

Table of Contents

Introduction to XML − Extensible Markup Language

XML is an acronym for "Extensible Markup Language". XML is the latest platform−independent and content−dependent technology available for Internet development. XML is the tool of choice for distributing structured information in this age. A working group under the guidance World Wide Web Consortium (W3C) started developing XML to simplify the transmission of documents across the internet.

XML is a young meta language. In early 1998, W3C published the XML 1.0 recommendations. Content developers have started developing various applications of XML − for example Mathematical Markup Language(MathML), CML − Chemical Markup Language etc. W3C, while releasing the HTML 4.0 recommendation in early 1998, said that it would approximately take 18 months to develop this transitional language. We have time to learn the basics of XML and develop the future internet language.

XML not only fulfills the needs of web authors but also those of anyone interested in publishing. Oracle, IBM and Microsoft are coming out with XML−related software and this gives sufficient indications about the future of XML in the IT industry..

SGML, HTML and XML

SGML −Standard Generalized Markup Language

SGML is an international standard for describing electronic documents. SGML is a meta language used to write other languages. SGML helps describe text documents in a logical and structural manner. SGML is used primarily for the creation, storage, and distribution of documents and as a source for conversion to other documents.

SGML documents have been used in the US military and American aviation industries for many years. It is too complicated for web publishers and this is the reason for the growth of HTML, a simplified subset of SGML.

HTML − Hyper Text Markup Language

HTML can be considered as the simplest subset of SGML and is simple enough to have Web publishing accessible to anyone. Publishers do not necessarily need knowledge of HTML as a lot of WYSIWYG editors are available in the market.

What are the problems with HTML?

HTML is too restrictive. Standard tags are predefined by W3C, so HTML is not powerful enough to describe more complex documents. HTML is more presentation oriented than content oriented, so HTML tags do not give an indication of the meaning of the content. You may ask, why can't W3C introduce more tags to describe content? Doing just that led to another problem: browser companies have introduced new, proprietary tags to attract users to their products.

With current HTML, publishers have to do lot of adjustments to their documents to be compatible with popular browsers. Browsers do not check for bad HTML code and hence the Internet has a lot of documents with several HTML mistakes. These issues were raised by content managers and Internet publishers and this problem escalated to such an extent that W3C began to look for alternatives. What is the solution?

XML − eXtensible Markup Language

XML can be considered as a simplified version of SGML. XML is *case sensitive*. <p> is different from <P>. though in HTML both would be considered the same.
XML is extensible − You can create your own elements to meet your publishing demands. You need not wait for W3C HTML committee to release the next version of HTML to include your required tags.

XML is structured − XML documents should adhere to a specific structure. If a document is not structured properly, it is not considered to be XML.

XML is a much more accessible language than SGML. Since XML documents are well structured, programmers can easily write software for rendering the XML documents. XML has simple rules to differentiate between the document contents and the XML markup elements.

XML markup elements start with either a less than symbol(<) or an ampersand (&) character XML also uses

greater than symbol (>), single quote (') and the double quotation marks(") for markup. To use the above markup characters, one should use the corresponding general XML entity (&amp for &, &gt for >, &lt for <, &apos for ' and &quot for ").

What is DTD − Document Type Definition

A DTD can be considered the grammar for a markup language. It is a set of regulations that specifies the usage of XML markup. It defines elements, an element's attributes and its values, and contains specifications about which elements can be contained in others. DTD can also define entities.

We will consider an example DTD for email:

```
<!ELEMENT Mail (From, To, Cc?, Date?, Subject, Body)> <!ELEMENT From (#PCDATA) > <!ELEMENT To (#PCDATA) > <!ELEMENT Cc (#PCDATA) > <!ELEMENT Date (#PCDATA) > <!ELEMENT Subject (#PCDATA) > <!ELEMENT Body (#PCDATA | P | Br)* > <!ELEMENT P (#PCDATA | Br)* > <!ATTLIST P align (left | right | justify) "left" > <!ELEMENT Br EMPTY >
```

Description
A XML document conforming to the mail DTD has only one From, one To, an optional Cc, an optional Date, one Subject and one body.

- A From element has only text.
- A To element has only text.
- A Cc element has only text.
- A Date element has only text.
- A Subject element has only text.
- A Body element can have text and zero or more of P and Br elements.
- A P element can have text and zero or more of Br element
- The P element has an align attribute. The attributes possible values are left, justify or right. Its default value is left.
- The Br element is empty.

A XML parser (discussed in the software section) will use the DTD to parse the document. The DTDs enable you to publish your documents to be used by others. The XML document should have instructions to tell the XML processing programs to find out the DTD.

A <!DOCTYPE> element at the start of the XML file will instruct the program about the location of the DTD. For example:

```
<!DOCTYPE Mail system "http://infowest.com/DTDS/mail.dtd"> <Mail> .. .. .. </Mail>
```

Valid and Well−formed XML

Two levels of conformance are there in the XML recommendation: *valid* and *well−formed*.

A **well−formed** XML file must follow a few key rules:

- It should have at least one element.
- The document should conform to the XML specification
- The root element (<Mail>) should not be contained by any other element.
- Proper nesting of elements is a must.
- Attribute values should be within quotation marks.
- All the entities other than reserved entities should be declared.

The reader should take a look at the XML recommendation itself before attempting to create an XML document..

Refer http://www.w3c.org/xml

Even without a DTD, an XML parser should parse a well−formed XML document. If it is not well−formed it can't be called an XML document. This aspect is good for web applications because the applications need not know the DTD structure used to create the XML document.

Valid XML

Valid XML files are those which have a DTD reference and conforms to the DTD. A valid XML file must also be well−formed. The availability of DTD along with the document facilitates the XML processing programs and rendering of the document by XML−enabled browsers.

Example XML Documents and analysis

Example 1 − A well−formed XML document:

```
<?xml version="1.0" standalone="no"?> <Mail> <From>Author</From>
<To>Receiver</To> <Date> Thu, 7 Oct 1999 11:15:16 −0600</Date> <Subject>XML
Introduction</Subject> <body><p>Thanks for reading<Br/> this article</p> <br/> <p>Hope
you enjoyed this article</p> </body> </Mail>
```

The first line is the XML declaration and it identifies what follows as XML code. It is called the prolog. The attribute version indicates the version of the XML standard. The statement standalone="no" indicates that markup declarations are external to the document. The XML declaration can be considered as a "processing instruction". Though this declaration is not compulsory, it is better to include such declaration. This will increase the portability of the document.

Example 2 − A valid XML document conforming to mail.dtd. Date element is missing because it is optional in the mail DTD The element P has the attribute justify. After the Body and before P Comments text is allowed, because DTD allows the use of plain text in the Body element.

```
<?xml version="1.0" standalone="no"?> <!DOCTYPE Mail system
"http://infowest.com/DTDS/mail.dtd"> <Mail> <From>Author</From> <To>Receiver</To>
<Cc>Receiver2</Cc> <Subject>XML Introduction</Subject> <body>Comments:<p
align="justify">Thanks for reading<Br/> this article</p> <br/> <p>Hope you enjoyed this
article</p> </body> </Mail>
```

Example 3 − A valid XML document conforming to "mail.dtd". Date element and Cc elements are present The element P has the attribute right.

```
<?xml version="1.0" standalone="no"?> <!DOCTYPE Mail system
"http://infowest.com/DTDS/mail.dtd"> <Mail> <From>Author</From> <To>Receiver</To>
<Cc>Receiver2</Cc> <Date> Thu, 7 Oct 1999 11:15:16 −0600</Date> <Subject>XML
Introduction</Subject> <body>Comments:<p align = "right" >Thanks for reading<Br/> this
article</p> <br/> <p>Hope you enjoyed this article</p> </body> </Mail>
```

An XML document can have comments. XML's comment syntax is similar to that of HTML. Any text, except double hyphen, "−−", can be placed between <−− and −−> tags. Processing instruction(PI) can be embedded in the documents. The data components of the PI should be recognized by the processing applications.

Publishers may want to include some codes that should not be parsed by the parsers. Those codes can be put in to the ignored sections. An ignored section will have the syntax like this:

```
<[CDATA[Any text to be ignored]]>
```

In simple words, any ignored sections start with <[CDATA[and end with]]>

Software for XML

XML related softwares can be broadly classified in to three categories:
XML Browsers
XML Parsers/applications
XML Editors

XML Browsers
Few XML browsers are there now. The XML style sheet specification (XSL)is in the evolution stage. Features like XLink and XPoint are still emerging and has to be incorporated in the browsers.

XML Parsers and Applications
A parser is a program to check the well−formedness of an XML document. If the document is well−formed, the parser has to read the document's DTD and should check the conformance of the document to the rules in the DTD.

DXP − A parser written in Java
Many parsers are written using Java. DXP can be used to check the well−formedness and validity of an XML document. The software, it's installation and operational documents can be downloaded from http://www.datachannel.com/

MSxml − A parser written in JAVA by Microsoft
Download and install form Microsoft http://www.microsoft.com/workshop/XML/parser/xmldl.asp

Perl Module − A perl module XML parser is available
Larry Wall prototyped this Perl module for processing XML documents in Perl scripts. It relies on James Clark's expat XML parser. This module also requires Perl 5.004 or better.

Download and install it from the Perl site. Clark Cooper has provided some sample scripts that demonstrate how to process XML in Perl.

Visit the site http://www.XML.com/pub/Guide/XML_Parsers to have more information about the XML parser softwares and latest developments.

XML Editors
XML editors give users an environment to create their own tags. One of the popular editors is XML<PRO> from Vervet Logic http://www.vervet.com/.

Few features of XML<PRO>

- Document validation − XML<PRO> can be used as a validator to test the document validity
- Entity Palette − A floating palette is available for inserting defined entities
- DTD can be associated with the document.

Visit the site http://www.XML.com/pub/Guide/XML_Editors for more information about the XML editors and the latest developments.

Conclusion

XML is still under development stage. It is evolving beyond the draft specifications put forward by w3c. These changes may be rapid and keep yourself updated by visiting related sites. Discussion about XML namespaces, schema, style sheets and XQL − a developing query language are beyond the scope of this introductory article.

You can reach the author, Suresh Kumar, at kalakad@yahoo.com.