XML Introduction

XML (Extensible Markup Language) is a markup language used for storing and transporting data.

XML doesn’t depend on the platform and the software(programming language). You can write a program in any language on any platform (Operating System) to send, receive or store data using XML.

The top line <?xml version="1.0" encoding="UTF-8"?> is called XML prolog. It is an optional line and should me mentionead as the first line in an xml.

 XML is just information wrapped in tags. Someone must write a piece of software to send, receive, store, or display it.

XML is a standard data format to store and transport data among different systems with different configurations.

XML Features

1. XML focuses on data rather than how it looks

One of the reason, XML is popular because it focuses on data rather than data presentation. The other markup language such as HTML is used for data presentation. This separates the data and its presentation part and gives us the freedom to present the data, the way we want, once we receive it using XML.

Two or more systems can receive the same data from a same XML and present it in a different way using other markup language such as HTML.

2. Easy and efficient data sharing

Since XML is software and hardware independent, it is easier to share data between different systems with different hardware and software configuration. Any system with any programming language can read and process a XML document.

3. Compatibility with other markup language HTML

It is so much easier to read the data from XML and display it on an GUI(graphical user interface) using HTML markup language.

When the data changes over time, we need not to make any changes in the HTML.

4. Supports platform transition

The main reason why changing to new systems and platform is challenging, because it involves the headache of data conversion between incompatible formats which often results in data loss. XML simplifies this process as the data is transported on new upgraded systems without any data loss.

5. Allows XML validation

A XML document can be validated using DTD or XML schema. This ensures that the XML document is syntactically correct and avoids any issues that may arise due to the incorrect XML.

6. Adapts technology advancements

The reason why XML is popular and being used from a very long time is because, it can adapt to the new technologies because of its platform-independent nature.

7. XML supports Unicode

XML supports Unicode that allows it to communicate almost any information in any written human language.

XML Advantages

1. XML is platform independent and programming language independent, thus it can be used on any system and supports the technology change when that happens.

2. XML supports unicode. Unicode is an international encoding standard for use with different languages and scripts, by which each letter, digit, or symbol is assigned a unique numeric value that applies across different platforms and programs. This feature allows XML to transmit any information written in any human language.

3. The data stored and transported using XML can be changed at any point of time without affecting the data presentation. Generally other markup language such as HTML is used for data presentation, HTML gets the data from XML and display it on the GUI (graphical user interface), once data is updated in XML, it does reflect in HTML without making any change in HTML GUI.

4. XML allows validation using DTD and Schema. This validation ensures that the XML document is free from any syntax error.

1. XML simplifies data sharing between various systems because of its platform independent nature. XML data doesn’t require any conversion when transferred between different systems.

HTML vs XML

XML and HTML were designed with different goals:

* XML was designed to carry data - with focus on what data is
* HTML was designed to display data - with focus on how data looks
* XML tags are not predefined like HTML tags.
* HTML is case insensitive unlike XML and end tag is mandatory in XML.

JSON vs XML

JSON supports arrays unlike XML and is much more readable.

JSON is short and light when compared to XML.

JSON can be parsed easily as most of programming languages contain identical datatypes of JSON.

XML is more secure than JSON.

XML sample document example

<? xml version=”1.0” encoding=”utf-8”?>

<movies>

<movie>

<actor>Sushant</actor>

<title>Dil Bechara</title>

<director>Mukesh Chhabra </director>

</movie>

<movie>

<actor>Amir</actor>

<title>Dangal</title>

<director>Nitesh Tiwari</director>

</movie>

</movies>

XML document structure

Based on the above discussion we can say that a XML document structure looks like this:

<root>

<child>

<subchild>.....</subchild>

</child>

</root>

XML Syntax

* Root element is mandatory in an XML document.
* XML is case sensitive.
* XML prolog should be the first line of an XML document.
* Elements should be nested properly.
* An opening tag in XML can have attributes, these attributes are name & value pairs.
* Attribute names are case sensitive and should not be in quotation marks.
* Attribute values should be in single or double quotation.
* XML comments are same as HTML comments.
* Closing tag is mandatory for an XML element.

XML Tree Structure

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

<company>

<employee>

<name>Negan</name>

<age>40</age>

<email>imnegan@twd.com</email>

<address>

<city>Noida</city>

<state>Uttar Pradesh</state>

<pin>201301</pin>

<landmark>Near hill top</landmark>

</address>

</employee>

</company>

Root elements:company

Child elements:name,age,address and email

Sub child elements:city,state,pin and landmark

XML Attributes

XML attributes are a way to add additional data to the XML element. Attributes contain data in form of name & value pairs

Attribute values must always be quoted. Either single or double quotes can be used.

<person gender="female">

Multiple attributes

<person gender=”female” Adult=”yes”>

Attributes are name & value pairs. The attribute name should not be in quotes, however the attribute value must always be in quotes (single or double)

XML element can have more than one attributes but one attribute cannot have more than a single value.

Elements over attributes

Some things to consider when using attributes are:

* attributes cannot contain multiple values (elements can)
* attributes cannot contain tree structures (elements can)
* attributes are not easily expandable (for future changes)
* attributes values are difficult to test against a DTD
* elements are easy to be handled by the programming language compared to the attributes.

Generally metadata is stored as attributes.

XML Namespaces

XML Namespaces provide a method to avoid element name conflicts.

Name conflicts in XML can easily be avoided using a name prefix.

<h:table>

<h:tr>

<h:td>Apples</h:td>

<h:td>Bananas</h:td>

</h:tr>

</h:table>

<f:table>

<f:name>African Coffee Table</f:name>

<f:width>80</f:width>

<f:length>120</f:length>

</f:table>

When using prefixes in XML, a **namespace** for the prefix must be defined.

The namespace can be defined by an **xmlns** attribute in the start tag of an element.

The namespace declaration has the following syntax. xmlns:*prefix*="*URI*".

<root>

<h:table xmlns:h="http://www.w3.org/TR/html4/">

<h:tr>

<h:td>Apples</h:td>

<h:td>Bananas</h:td>

</h:tr>

</h:table>

<f:table xmlns:f="https://www.w3schools.com/furniture">

<f:name>African Coffee Table</f:name>

<f:width>80</f:width>

<f:length>120</f:length>

</f:table>

</root>

Namespaces can also be defined in the root element.

The namespace URI is not used by the parser to look up information.

The purpose of using an URI is to give the namespace a unique name.

Defining a default namespace for an element saves us from using prefixes in all the child elements.

xmlns:”URI”

Raw XML files can be viewed in all major browsers.

XML documents do not carry information about how to display the data.

Since XML tags are "invented" by the author of the XML document, browsers do not know display associated with a tag.

There are two document type definitions that can used with XML document to check whether the XML document is valid.

1. XML DTD (Document Type Definition)

2. XML Schema – An XML-based alternative to the Document Type Definition

1. XML DTD

DTD defines the structure of XML document that can be validated against the XML document to check for the syntax errors. XML DTD defines the structure by mentioning the XML elements in such a way so that the complete structure of XML document can be understood. The DTD file has .dtd extension.

2. XML Schema

The Schema does the same thing that a DTD can do. It also defines the structure of the XML document but unlike DTD it is an XML file, in addition to that Schema supports data types and namespaces.

XML HttpRequestObject

The XMLHttpRequest object can be used to request data from a web server.It can be used to:

* Update a web page without reloading the page
* Request data from a server - after the page has loaded
* Receive data from a server  - after the page has loaded
* Send data to a server - in the background

A common JavaScript syntax for using the XMLHttpRequest object

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() {

if (this.readyState == 4 && this.status == 200) {

// Typical action to be performed when the document is ready:

document.getElementById("demo").innerHTML = xhttp.responseText;

}

};

xhttp.open("GET", "filename", true);

xhttp.send();

The first line in the example above creates an ****XMLHttpRequest**** object.

The **onreadystatechange** property specifies a function to be executed every time the status of the XMLHttpRequest object changes.

When ****readyState**** property is 4 and the ****status**** property is 200, the response is ready.

The ****responseText**** property returns the server response as a text string.The text string can be used to update a web page.

All major browsers have a built-in XML parser to access and manipulate XML.

XML Parser

The XML DOM (Document Object Model) defines the properties and methods for accessing and editing XML.

However, before an XML document can be accessed, it must be loaded into an XML DOM object.

All modern browsers have a built-in XML parser that can convert text into an XML DOM object.

XML DOM

The XML DOM defines a standard way for accessing and manipulating XML documents. It presents an XML document as a tree-structure.

Create a variable txt containing xml

now parse the text into a xmldom using parser

parser=new DOMParser(); //Creating a new parser object

xmlDoc=parser.parseFromString(txt,”text/xml”);

This loads the parsed xml into the xmlDoc

Now we can access the respective tag using the syntax

xmlDoc.getElementsByTagName(“tag”)[ind].childNode[ind].nodeValue;

Where index starts from 0.

innerHTML can also be used in case of a nodeValue

XPath

XPath is a major element in the XSLT standard.

XPath can be used to navigate through elements and attributes in an XML document.

* XPath is a syntax for defining parts of an XML document
* XPath uses path expressions to navigate in XML documents
* XPath contains a library of standard functions
* XPath is a major element in XSLT and in XQuery

|  |  |
| --- | --- |
| **XPath Expression** | **Result** |
| /bookstore/book[1] | Selects the first book element that is the child of the bookstore element |
| /bookstore/book[last()] | Selects the last book element that is the child of the bookstore element |
| /bookstore/book[last()-1] | Selects the last but one book element that is the child of the bookstore element |
| /bookstore/book[position()<3] | Selects the first two book elements that are children of the bookstore element |
| //title[@lang] | Selects all the title elements that have an attribute named lang |
| //title[@lang='en'] | Selects all the title elements that have a "lang" attribute with a value of "en" |
| /bookstore/book[price>35.00] | Selects all the book elements of the bookstore element that have a price element with a value greater than 35.00 |
| /bookstore/book[price>35.00]/title | Selects all the title elements of the book elements of the bookstore element that have a price element with a value greater than 35.00 |

XML XLST

XSLT (eXtensible Stylesheet Language Transformations) is the recommended style sheet language for XML.

XSLT is far more sophisticated than CSS. With XSLT you can add/remove elements and attributes to or from the output file. You can also rearrange and sort elements, perform tests and make decisions about which elements to hide and display, and a lot more.

XSLT uses XPath to find information in an XML document.

XQUERY

XQuery was designed to query XML data.

XQuery can be used to:

* Extract information to use in a Web Service
* Generate summary reports
* Transform XML data to XHTML
* Search Web documents for relevant information

XML Links

* XLink is used to create hyperlinks within XML documents
* Any element in an XML document can behave as a link
* With XLink, the links can be defined outside the linked files
* The XLink namespace is declared at the top of the document (xmlns:xlink="http://www.w3.org/1999/xlink")
* The xlink:type="simple" creates a simple "HTML-like" link
* The xlink:href attribute specifies the URL to link to (in this case - an image)
* The xlink:show="new" specifies that the link should open in a new window

XML DTD

Internal Declaration

<!DOCTYPE document\_name [

<!ELEMENT document\_name (to,from,subject,message)>

<!ELEMENT to (#PCDATA)>

<!ELEMENT from (#PCDATA)>

<!ELEMENT subject (#PCDATA)>

<!ELEMENT message (#PCDATA)>

]

External reference

<!DOCTYPE document\_name SYSTEM "bb.dtd">

XML Schema

XML schema is an alternative to DTD. An XML document is considered “well formed” and “valid” if it is successfully validated against XML Schema. The extension of Schema file is .xsd.