Introduction

- XML stands for e**X**tensible **M**arkup **L**anguage.
- It is a markup language (i.e. programs are written in tags)
- It was designed to carry data, not to display data.
- XML tags are not predefined. We define our own tags.
- XML is designed to be self-descriptive
- XML is a W3C Recommendation

Why XML?

- **International Standard**

XML is a document standard that is maintained by the W3C, an organization that is responsible for Web standards. XML documents are vendor-neutral, and they are not tied to one application or one company.

- **eXtensible Language**

XML uses XML elements or tags to define document structure. XML does not have a fixed number of tags or elements, as HTML does, but it is extensible, allowing the document designer to define meaningful tags. By using XML, developers can develop a markup language that is suitable for their purpose. This ability to define custom elements makes XML extremely flexible.

- **Separates Data from Display**

XML allows you to separate content from format. The formatting of the XML document is inside a separate style sheet. This separation allows you to easily maintain and update formatting as needs change. It is easy to maintain a consistent style for all documents when the content is separate from the formatting.

Difference between HTML and XML

HTML	XML
It is used to format and display the data.	It is used to transport data.
It uses a fixed, unchangeable set of tags.	It allows user to define its own tags.
HTML describes both structure (e.g. <P>, <H2>, <EM>) and appearance (e.g. , <I>).	It describes only 'content', or 'meaning'.
It is used to mark up text so it can be displayed to users.	It is used to mark up data so it can be processed by computers.
It is defined in SGML.	It is a subset of SGML.
It is not case sensitive language.	It is case sensitive language.
Browsers ignore/correct many HTML errors.	The rules are strict and errors are not allowed.
Files saved with extension '***.html***' or '***.htm***'	Files saved with extension '***.xml***'

Similarities between HTML & XML

HTML and XML both looks similar because they are both SGML (**S**tandard **G**eneralized **M**arkup **L**anguage developed by IBM) languages.

- In both the languages elements are enclosed in tags.
  Example: <body>This is an element</body>

- In both the languages attributes are used to set the behavior of tags.
  Example: <font face="Verdana" size="+1" color="red">

- Both the language makes use entities.
- Both the languages are platform independent.

Comments

- Comments are ignored by the web browser.
- Comments can be put anywhere in an XML document.
- Comments starts with **<!--** and ends with **-->**

Example:

<!-- This is a comment in both HTML and XML -->

- Comments do not have an end tag
- The blanks after <!-- and before --> are optional
- The character sequence -- cannot occur in the comment

Entities

- Five special characters must be written as entities:
  1. & for & (almost always necessary)
  2. < for < (almost always necessary)
  3. > for > (not usually necessary)
  4. " for " (necessary inside double quotes)
  5. ' for ' (necessary inside single quotes)
- These entities can be used even in places where they are not absolutely required
- These are the only predefined entities in XML

XML Declaration

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

- The XML declaration is not required by browsers, but it is required by most XML processors.
- If present, the XML declaration must be first statement, not even whitespace should precede it.
- The brackets are <? and ?>
- version="1.0" is required (this is the only version so far)
- encoding can be "UTF-8" (ASCII) or "UTF-16" (Unicode), or something else, or it can be omitted
- standalone can be “yes” or “no” it tells the XML parser whether there is a separate DTD or not.

Encoding

- XML documents can contain international characters, like Norwegian æøå, or French êèé.
- To avoid errors, we should specify the encoding used, or save your XML files as UTF-8.
- UTF = Universal character set Transformation Format.
- Unicode is standard for character encoding of text documents.
- It defines (nearly) every possible international character by a name and a number.
- Unicode has two variants: UTF-8 and UTF-16.

Processing instructions

- Processing Instructions give command to an application that is processing the XML document to handle it in a certain way
- It may occur anywhere in the XML document, (but usually first is preferred)

Syntax:
<?*target instructions*?>

Example:
<?xml version="1"?>
OR
<?xml-stylesheet type="text/css" href="mySheet.css"?>

- XML documents are typically processed by more than one program
- Programs that do not recognize a given PI should just ignore it

CDATA

- By default, all text inside an XML document is parsed.
- By using *Character DATA* we can force text to be treated as unparsed.

Syntax:
<![CDATA[...]]>

- Any characters can occur inside a CDATA section like &, <, whitespace, etc.
- The only real restriction is that the character sequence]]> cannot occur inside a CDATA
- CDATA is useful when your text has a lot of illegal characters (for example, if your XML document contains some HTML text)

XML-Related Technologies

Technology	Meaning	Description
XHTML	e**X**tensible **HTML**	It is a clearer and stricter version of XML. It belongs to the family of XML markup languages. It was developed to make HTML more extensible and increase inter-operability.
XML DOM	**XML** **D**ocument **O**bject **M**odel	It is a standard document model that is used to access and manipulate XML. It defines the XML file in tree structure.

Technology	Meaning	Description
XSL	e**X**tensible **S**tyle sheet **L**anguage	It contain three parts: (i) XSLT (**XSL T**ransform): It transforms XML into other format, like HTML. (ii) XSL: It is used for formatting XML to screen, paper etc. (iii) XPATH: It is a language to navigate XML documents.
XQuery	**X**ML **Query**	It is a XML based language which is used to query XML based data.
DTD	**D**ocument **T**ype **D**efinition	It is a standard which is used to define the legal elements in an XML document.
XSD	**X**ML **S**chema **D**efinition	It is an XML based alternative to DTD. It is used to describe the structure of an XML document.
XLink	**X**ML **Link**ing Language	It is used for creating hyperlinks (external and internal links) in XML documents.
XPointer	**X**ML **Pointer** Language	It is a system for addressing components of XML based internet media. It allows the XLink hyperlinks to point to more specific parts in the XML documents.
SOAP	**S**imple **O**bject **A**ccess **P**rotocol	It is XML based protocol used for accessing web services.
WSDL	**W**eb **S**ervices **D**escription **L**anguages	It is an XML based language to describe web services. It also describes the functionality offered by a web service.
RDF	**R**esource **D**escription **F**ramework	RDF is an XML based language to describe web resources. It is a standard model for data interchange on the web. It is used to describe the title, author, content and copyright information of a web page.
SVG	**S**calable **V**ector **G**raphics	It is an XML based vector image format for two-dimensional images. It defines graphics in XML format. It also supports animation.

Naming Conventions

- Name must start with an alphabet or underscore.
- It can be combination of:
- Alphabets (a-z), (A-Z)
    - Digits (0-9)
    - . (dot)
    - -(hyphen)
    - _ (underscore)
    - : (colon) should be used only for namespaces

Types of XML Documents

1. Invalid Documents
- Document that does not follow the XML tag rules.
- If document has a DTD, and it does not follow the rules defined in its DTD.

2. Well-Formed Documents
- Documents that follow the XML tag rules but does not have DTD.

3. Valid Documents
- Documents that follow both the XML tag rules and the rules defined in their DTD.

DTD (Document Type Definition)

- It 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.

CDATA

- CDATA means *C*haracter *DATA*.
- CDATA is text is NOT parsed by a parser.
- Tags inside the CDATA will not be treated as markup and entities will not be expanded.

PCDATA

- PCDATA means *P*arsed *C*haracter *DATA*.
- PCDATA is text that will be parsed by a parser.
- The text will be examined by the parser for entities and markup.
- Tags inside the text will be treated as markup and entities will be expanded.
- PCDATA should not contain any
  - &
  - <
  - >
  - “
  - ,

  These need to be represented by & < > " ' entities respectively.

Declaring Elements

In a DTD, XML elements are declared with an element declaration.

Syntax:
<!ELEMENT element-name (element-content)>

OR

<!ELEMENT element-name category>

Empty Elements

In a DTD, XML empty elements are declared with the category using keyword EMPTY.

Syntax:
<!ELEMENT element-name EMPTY>

Elements with PCDATA

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

Syntax:
<!ELEMENT element-name (#PCDATA)>

Elements with any Contents

Elements declared with the category keyword ANY, can contain any combination of parsable data.

Syntax:
<!ELEMENT element-name ANY>

Elements with Children (sequences)

Elements with one or more children are declared with the name of the children elements inside parentheses.

Syntax:
<!ELEMENT element-name (child1)>
OR
<!ELEMENT element-name (child1,child2,...)>

Declaring Only One Occurrence of an Element

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

Declaring Minimum One Occurrence of an Element

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

Declaring Zero or More Occurrences of an Element

Syntax:
<!ELEMENT element-name (child-name*)>

Declaring either/or Content

Example:
<!ELEMENT note (to,from,header,(message|body))>

The example above declares that the "note" element must contain a "to" element, a "from" element, a "header" element, and either a "message" or a "body" element

Types of DTD

There are two types of DTD:

- Internal DTD
- External DTD

Internal DTD

- Internal DTD should be wrapped in a DOCTYPE definition with the following syntax.

Syntax:

```
<!DOCTYPE root-element [element-declarations]>
```

Example:

```
<?xml version="1.0" encoding="UTF-16" standalone="yes"?>
<!DOCTYPE Message [
<!ELEMENT Message (Title, To, From, Subject)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT To (#PCDATA)>
<!ELEMENT From (#PCDATA)>
<!ELEMENT Subject (#PCDATA)>
]>
<Message>
<Title>Title: Birthday Wishes</Title>
<To>To: you@youraddress.com</To>
<From>From: me@myaddress.com</From>
<Subject>Subject : Many Many Happy Returns of the Day </Subject>
</Message>
```

Explanation of Example:

- !DOCTYPE Message
  It defines that the root element of this document is Message.
- !ELEMENT Message (Title, To, From, Subject)>
  It defines that the Message element contains four elements: “Title, To, From, Subject"
- !ELEMENT Title (#PCDATA)>
  It defines the Title element to be of type "#PCDATA"
- !ELEMENT To (#PCDATA)>
  It defines the To element to be of type "#PCDATA"
- !ELEMENT From (#PCDATA)>
  It defines the From element to be of type "#PCDATA"
- !ELEMENT Subject (#PCDATA)>
  It defines the Subject element to be of type "#PCDATA”

External DTD

- External DTD should be wrapped in a DOCTYPE definition with the following syntax.

Syntax:

```
<!DOCTYPE root-element SYSTEM "filename">
```

Example:

Example.xml

```
<?xml version="1.0" encoding="UTF-16" standalone="no"?>
<!DOCTYPE Message SYSTEM "Mydtd.dtd">
<Message>
<Title>Title: Birthday Wishes</Title>
<To>From: you@youraddress.com</To>
<From>From: me@myaddress.com</From>
<Subject>Subject : Many Many Happy Returns of the Day</Subject>
</Message>
```

Mydtd.dtd

```
<!ELEMENT Message (Title, To, From, Subject)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT To (#PCDATA)>
<!ELEMENT From (#PCDATA)>
<!ELEMENT Subject (#PCDATA)>
```

Explanation of Example:

- !ELEMENT Message (Title, To, From, Subject)>
  It defines that the Message element contains four elements: “Title, To, From, Subject"
- !ELEMENT Title (#PCDATA)>
  It defines the Title element to be of type "#PCDATA"
- !ELEMENT To (#PCDATA)>
  It defines the To element to be of type "#PCDATA"
- !ELEMENT From (#PCDATA)>
  It defines the From element to be of type "#PCDATA"
- !ELEMENT Subject (#PCDATA)>
  It defines the Subject element to be of type "#PCDATA”

DTD Validation

- DTD checks the validity of structure and XML tag rules.
- Some methods to validate the DTD are the followings:
  1. Using XML DTD validation tools
     - XML Spy (not free)
     - XML Starlet (open source)
  2. Using XML DTD on-line validators
     - https://validator.w3.org/
     - http://www.xmlvalidation.com/
  3. Using own XML validators with XML DTD validation API

XML parsing

XML Parsing means going through XML document to access data or to modify data.

XML parser

- XML parser provides a way how to access or modify data present in an XML document.

- XML Parser is designed to read XML and create a way for programs to use XML.
- A XML parser is a package that provides interfaces for client applications to work with an XML document.
- XML parser validates the document and check that the document is well formatted.

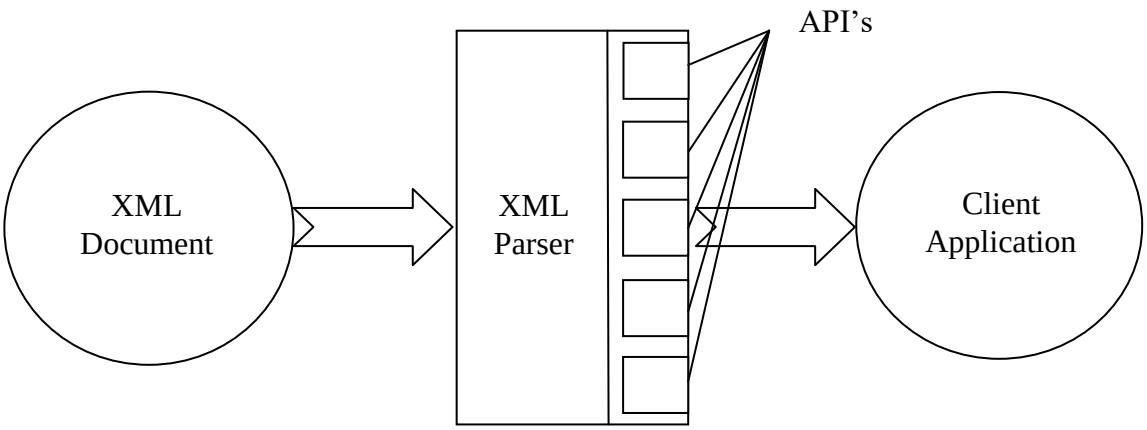

Fig. XML Parsing

Types of Parser

There are various types of XML parser. Some commonly used parsers are:

- DOM Parser
- SAX Parser
- JDOM Parser
- StAX Parser
- XPath Parser
- DOM4J Parser

DOM

- DOM stands for Document Object Model.
- A DOM document is an object which contains all the information of a document.
- The DOM is separated into three different parts / levels:
  - Core DOM
    - Standard model for any structured document.
  - XML DOM
    - Standard model for XML documents.
  - HTML DOM
    - Standard model for HTML documents

DOM Parser

- It defines a standard for accessing and manipulating XML documents.
- In simple words, XML DOM is a standard for how to get, change, add, or delete XML elements.
- It allows programs dynamically access and update the content, structure, and style of a document.
- It defines the objects and properties of all document elements, and the methods (interface) to access them.

- It parses the document by loading the complete contents of the XML document.
- It creates a complete hierarchical tree structure of XML document in memory.
- The DOM is a W3C standard.
- It is platform independent
- It is language independent

Advantages of DOM parser

- It supports both read and writes operations.
- The API is very simple to use.
- It is preferred when random access to widely separated parts of a document is required.

Disadvantages of DOM parser

- It is memory inefficient
  - It consumes more memory because the whole XML document needs to load into memory.
- It is comparatively slower than other parsers.

XML SAX

- SAX stands for Simple API for XML.
- It is an event-based sequential access parser.
- It operates on each piece of the XML document sequentially.
- It provides a mechanism for reading data from an XML document.

Advantages of SAX parser

- It is simple and memory efficient.
- It is very fast and works for huge documents.

Disadvantages of SAX parser

- It is event-based so its API is less intuitive.
- Clients never know the full information because the data is broken into pieces.

XML Schema

- XML Schema language is also known as XML Schema Definition (XSD).
- XML Schema is XML-based alternative to DTD.
- It describes the structure of an XML document.

Purpose of an XML Schema

- The purpose of XML Schema is to define the legal building blocks of XML document, just like a DTD.
- By using XML schema, we can define the following:
  - Elements that can appear in a document
  - Attributes that can appear in a document

- Which elements are child elements
- The number of child elements
- The order of child elements
- Whether an element is empty or can include text
- Data types for elements and attributes
- Default and fixed values for elements and attributes

XML Schemas Data Types

- XML Schemas support various data types.
- With support for data types:
  - It is easier to describe allowable document content
  - It is easier to validate the correctness of data
  - It is easier to work with data from a database
  - It is easier to define restrictions on data
  - It is easier to define data formats (data pattern)
  - It is easier to convert data between different data types
- There are lot of built in data types supported by XML Schema.
- Some common data types are:
  - xs:string
  - xs:decimal
  - xs:integer
  - xs:boolean
  - xs:date
  - xs:time

XML Schemas are Extensible

- Reuse our Schema in other Schemas
- Create our own data types derived from the standard types
- Reference multiple schemas in the same document

XML Schema Data Communication

- When sending data from a sender to a receiver, it is essential that both parties have the same "expectations" about the content.
- With XML Schemas, the sender can describe the data in a way that the receiver will understand.
- A date like: "03-11-2004" will, in some countries, be interpreted as 3.November and in other countries as 11.March.
- However, an XML element with a data type like this:

  <date type="date">2004-03-11</date>

  It ensures a mutual understanding of the content, because the XML data type "date" requires the format "YYYY-MM-DD".

Questions asked in semester paper

Question-What is XML? Discuss the significance of XML. How is XML different from HTML? Explain the process of XML parsing. How are they useful?

[2015-2016] [2016-2017]

Question-What is DTD? What are the differences between external and internal DTD? Use suitable example.

[2015-2016]

Question-What is XML? What is the reason behind development of XML?

[2014-2015]

Question-Discuss various types of DTDs (Document Type Definition) in XML. Which type of DTD is preferable and why?

[2014-2015]

Question-What is HTML? Differentiate between XML and HTML.

[2013-2014]

Question-How is XML defined? Write down the XML syntax and structure rules.

[2012-2013]

Question-List the rules used for writing XML.

[2012-2013]

Question-What is XML Schema? Compare XML Schema and XML DTD with neat example.

[2012-2013]

Question-What are the differences between XML and HTML?

[2012-2013]

Question-Explain the term Document Type Definition (DTD) with the help of suitable examples.

[2011-2012]

Question-What is XML? Create an XML document of 10 students of first year. Add their roll numbers, marks obtained in 5 subjects, total marks and percentage. Save this XML document at the server, write a program that accepts student's roll number as input and returns the students marks, total and percentage by taking the student's information for the XML document.

[2011-2012]

Question-What is XML? Create an XML document of 10 students of first MCA. Add their roll numbers, marks obtained in 5 subjects, total marks and percentage. Save this XML document at the server, write a program that accepts student's roll number as input and returns the students marks, total and percentage by taking the student's information for XML document.

[2011-2012]

Question-Explain the term Document Type Definition (DTD) with the help of suitable examples.

[2011-2012][2010-2011]

Question-What is XML? Discuss the significance of XML. How is XML different from HTML? Explain the process of XML parsing. How are they useful?

[2010-2011]

Question-Explain the role of SAX and DOM in XML document verification.

[2010-2011]

Question-Compare and contrast HTML and XML showing their capabilities, limitations etc.

[2010-2011]

Question-Describe the mechanism of DOM and SAX parser with its architecture.

[2010-2011]

Question-What is the difference between external and internal DTD? Create a XML file which explains the attributes of phones using DTD.

[2010-2011]

Question-Write an XML document to store the details of Employees. Also design a web page to get the input of Employee id and retrieve the full details from the XML document.

[2007-2008]

Question-Explain the role of SAX and DOM in XML document verification.

[2007-2008]

Question-Differentiate between the following pair: HTML and XML

[2006-2007]

Question-Describe various types of Document Type Definition (DTD) in XML. Which type of DTD is more preferable and why?

[2006-2007]