UNIT-2

❖ XML:

- ✓ This stands for extendable mark of language. It is a subset of SGML(standard generalized markof language).
- ✓ The main drawback of SGML is its complexicit y. it is overcome by XML.
- ✓ HTML is restricted to display data on other hand XML supports displaying data also includes specification of syntax.
- ✓ Any user can defined his own tags and parser the data along with tags.
- ✓ XML is use to create the web application describing the data, configure the data and also provide security of data.
- ✓ It is an user friendly, case sensitive language.
- ✓ Every open tag is ended with closing tag. an user can create applications in an abstraction view.

➢ **FEATURES OF XML:**

It supports the following feature

- user self describing the data.
- Designing application
- User friendly.
- Platform independent code.
- Configure setting data.
- Secure code.

GENERAL SYNTAX OF XML:

```
<? xml version=”1.0”encoding=”UTF-S” standalone=”yes”?>
<Root tag>
    < child-tag >
        < sub-child tag >
        </sub-child tag>
  </child –tag>
</root-tag>
```

Example:

```
< books >
  < book >
       <b-title>let us c</b-title>
  <b-author> yashwanth </b-author>
<b-price>250</b-price>
<b section>
     <section 1>php</section 1>
    <section 2>xml</section 2>
<b/section>
```

</book>
</books>

❖ DOCUMENT TYPE DEFINITION AND BASIC BUILDING BLOCKS OF DTD:

- *Basic building blocks of DTD:*

XML having the following basic building blocks to built XML applications.
1. Tags
2. Elements
3. Attributes
4. PC data (parsed character data)
5. C-data character data)
6. Entities.

1. TAGS:

DTD supports tags every opening tag have its respective closing tag.
Example:
<TD>bindhu</TD>

2. ELEMENTS:

Elements of DTD can be use to represent data in XML document.

Elements declaration:

The general syntax is < !ELEMENT content>
If we want create an element with no data use the following syntax.
<!ELEMENT name of content(EMPTY)>
Example:
<!ELEMENT BOOK TITLE(EMPTY)>
If we want to create an element with same data we can use the following syntax.
<!ELEMENT name of element(#pc data)>
Example:
<!ELEMENT ADDRESS(#pc data)>
An element can have the childrens that can be declared like as
<!ELEMENT kits(cse,ece,eee)>
<!ELEMENT CSE(#PC DATA)>
<!ELEMENT EEE(#PC DATA)>
<!ELEMENT ECE(#PC DATA)>
To declare only one occurance of a given element is
<!element kits(cse)>
To declare zero or more occurance of given elements we use
<!ELEMENT kits(cse)>*
To declare one or more occurances of a given element we use
<!element kits (cse+)>

3.ATTRIBUTES:

Additional information along with the element is called attribute.
Syntax:
name
Example:
bindhu
DTD attributes are declared using"ATTLIST"

Syntax:
<!ATTLIST name of element attribute attributevalue>
Example:
<!ATTLIST rectangle length c data"0">
Example:2
<!ATTLIST ram address c data # required>
Four keywords are used in DTD declaration
1.implied=>#implied
2.required=>#required
3.fixed=>#fixed
4.pc data=>#pc data
1.#implied:
It specifies that the attribute value is not essential.
2.#required:
It specifies that the attribute value is an essential.
3.#fixed:
It specifies that the attribute value cannot be changed.
4.#pc data:
It specifies that the attribute value as parsed in character data.

Pc data:

Pc data is parsed by XML process it carries some tags which are treated as elements, entities associated with the elements can be expanded.

C DATA:

C data are not parsed by XML procedures no matter of entity expansion because the tags are not treated as elements.

ENTITIES:

Special variable used by XML instead of frequently used text by XML is called an entity. It is of two types.
1.internal entity-It can be declared with in the tag.
Syntax:
<!entity name of entity "value">

EXAMPLE:

< !ENTITY BOOK "OS">

2. EXTERNAL ENTITY: *THE VALUE OF AN ENTITY ADDED THROUGH URL.*

SYNTAX:

<!ENTITY NAME OF ENTITY "URL">

EXAMPLE:

<!ENTITY BOOK "HTTP://WWW.BOOK.COM/OS BOOK.DTD">

DOCUMENT TYPE DEFINITION:

DTD carries certain list of elements which specifies rules for constructing XML documents.
1. DTD mainly used to validate XML documents.

2. DTD can be declared as inside of an XML or outside of an XML.
3. A DTD declared inside of an XML file is called internal DTD.
4. if DTD can declared outside of XML file is called external DTD.
5. A DTD file must be save DTD extension.

INTERNAL DTD:

The DTD elements are specified inside the XML file to validate XML is called internal DTD.
Example:

```
<? xml version=”1.0”encoding=”UTF-8”standalone=”yes”?>
<! DOC type book [
<! ELEMENT title (#pc data)>
<! ELEMENT author(#pc data)>
<! ELEMENT publisher(#pc data)>
]>
< book >
    < title >let us c</title>
   <author> yashwanth </author>
   <publisher> bpb </publisher>
</book>
```

EXTERNAL DTD:

The DTD elements are specified in a separate file and saved it with . DTD extension and it included into .XML file to validate the XML file.
Example:
The “book .DTD “ code is

```
<! doc type book[
<! ELEMENT title(#pc data)>
<! ELEMENT author(#pc data)>
<! ELEMENT publisher(#pc data)>
]>
```

It can be store in an external and also embedded by using the following syntax into the XML file.

```
<! doc type book system/public” dtd file” >
```

Example:
By using XML

```
<! doctype book system ”book.dtd”>
< book >
    < title >let us c</title>
   <author> yashwanth </author>
   <publisher> bpb </publisher>
</book>
```

❖ INTRODUCTION TO PARSING :

XML parser are helper programs use to perform function. A parser builds a bridge between XML document and application.

- ✓ A parser checks the structure and content of the document and checks the validity of code against DTD and schemas.
- ✓ Schema constrains templates it stores the type of elements and attributes.

- ✓ DTD content of all elements used in the XML document. A parser checks if all the elements declares in the XML document match with those of the DTD or not.
- ✓ Parsers are two types

1.non-validating parser
2.validating parser

1. non-validating parser:

It checks for the syntax whether the XML document is well formed and non-validating parser add elements to the xml document based on the elements declared in the DTD.

2. validating parser:

It works similar to the non-validating parser. expect that is compare the rules of the DTD with XML document.

- ❖ The choice of the parser depends on the XML document.
- ❖ If XML document is simple application that does not require matching with the schema and DTD and a non-validating parser is used for simple document.
- ❖ If XML document is used to exchange the information between the organisation. the format and content of the application is provided by an organisation. Hence the validating parser is required.
- ❖ The most commonly used parsers are API, XML(SAX) simple API for XML.

THE DIFFERENCES BETWEEN DOM AND SAX:

DOM parser	SAX parser
1.It imports org.w3c.dom 2. It occupies memory. 3. It can be use to read XML document. 4. It is tree based model. 5. inserting & deleting elements in XML document is possible. 6. It parses in bidirectional. 7. It preserves the comments. 8. It loads entire XML document into the memory before process.	1. It imports org.xml.SAX 2. no memory overhead. 3. It can be use to read XML document. 4. It is event based model. 5. It is not possible to insert, delete elements into XML document. 6. It parses in uni direction. 7. It cannot preserve comments. 8. It reads node by node.

- ❖ **Xml schema:**

Xml schema recommended by the world wide consortium (wwc) specifies how to formally describe the elements in an extendable mark of language. It is used to verify the piece of code in a document.

Purpose of xml schema:

- ✓ Is to define the legal building blocks of an xml document.
- ✓ Elements and attributes can be appeared in the document.
- ✓ To define the data type for elements and attributes.
- ✓ It is more powerful than dtd.
- ✓ It can be easily convert to one data type to another data type.
- ✓ It is used to validate the data.
- ✓ It is used to describe the data.
- ✓ Xml schema secure the data communication.

Limitations of dtd:

To overcome the dtd limitations xml schema was introduced.

- ✓ Dtd treats context of element as a child element which is not so beneficial.
- ✓ Dtd specifies their own syntax in definition elements and attribution.
- ✓ These drawbacks are overcome by xml schema.

Facts about xml schema:

Xml schema is capable of defining elements and their attributes. Xml schema can define an element with no data(empty).

Advantages of xml schema:

- It allows user to learn new language.
- It directly uses xml editors and procedures to generate xml schema.

Xml name space:

The purpose of using xml name space is to avoid confusion of names.

Example:

<course>CCNA</course>

<course>java</course>

- ✓ It can be seen that name of the courses are given same.
- ✓ First one is network related course.
- ✓ Second is software related course but it is difficult to identify.
- ✓ To avoid confusion xml name space uses prefix before the name.

example 2:

< networking :course> CCNA </networking :course>

<software :course>java</software :course>

- ✓ Creation of xml name space is done using "XMLNS" along with uniform resource identifier(URI)
- ✓ Example: to demonstrate the xml name space.

```
<?xml version="1.0"?>
<prefix= "school" XMLNS=school= href :"http//……">
<prefix="college" XMLNS :college= href:"http//…….">
<education>
<school :subject> telugu </school :subject>
<college :subject>wt</college :subject>
</education>
```

- ➢ Example 2:

```
<?xml version="1.0"?>
<prefix="movie" xmlns :movie="href :http/……">
<prefix="new" xmlns :new="href :http/……..">
<entertainment>
<movie :broadcasting>bahubali</movie :broadcasting>
<new :broadcasting>tvg</new :broadcasting>
```

</entertainment>

❖ *Xml schema data types:*

Xml schema support the following data types.

- ➢ Binary data type :

It is of two types used for defining the image objects.
1.hexa decimal data type:
It supports hexa encoded arbitary data.
2.base 64 data type:
It supports base 64 encoded arbitary binary data.

- ➢ Logical data type:

Used for specifies the TRUE or FALSE.

- ➢ Number data type:

User for defining the different types of numbers which are used for develop the application these are 4 types
1.float data type:
It contains 32 bit floating point values.
2.double data type:
It contains 64 bit floating point values.
3.decimal data type:
It is of two types.

I. Integer data type:it contains 32 bit integer values.
II. Long data type:it contains 64 bit integer values.

- ➢ Data and time data type:
- ➢ Used for defining the data and time formats.

It is 3 types.

i. Time data type:H:M:S(it supports hours,minutes,seconds)
ii. Duration data type:user for specifies the period of the duration.
Example:3yrs,2months.
iii. Date data type:user for define different date formats year/month/day.
Example:2018/02/12

- ➢ Text data type:

User for define the character variable.

- ➢ URI data type:(uniform resource identifier)

It specifies the path of resource.