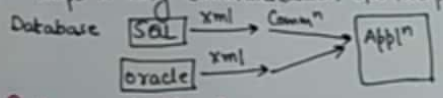
(XML)-①

Easy Engineering Classes – Free YouTube Lectures

For Engineering Students of GGSIPU, UPTU and Other Universities, Colleges of India

XML :- eXtensible Markup Language. (you can create your own tags.)

- → Used to store and transport data.
- → Self descriptive.
- → Used to carry data (Not used to display data)
- → Self-defined tags.
- → Platform and language independent.
- → Helps in easy communication b/w two platforms.

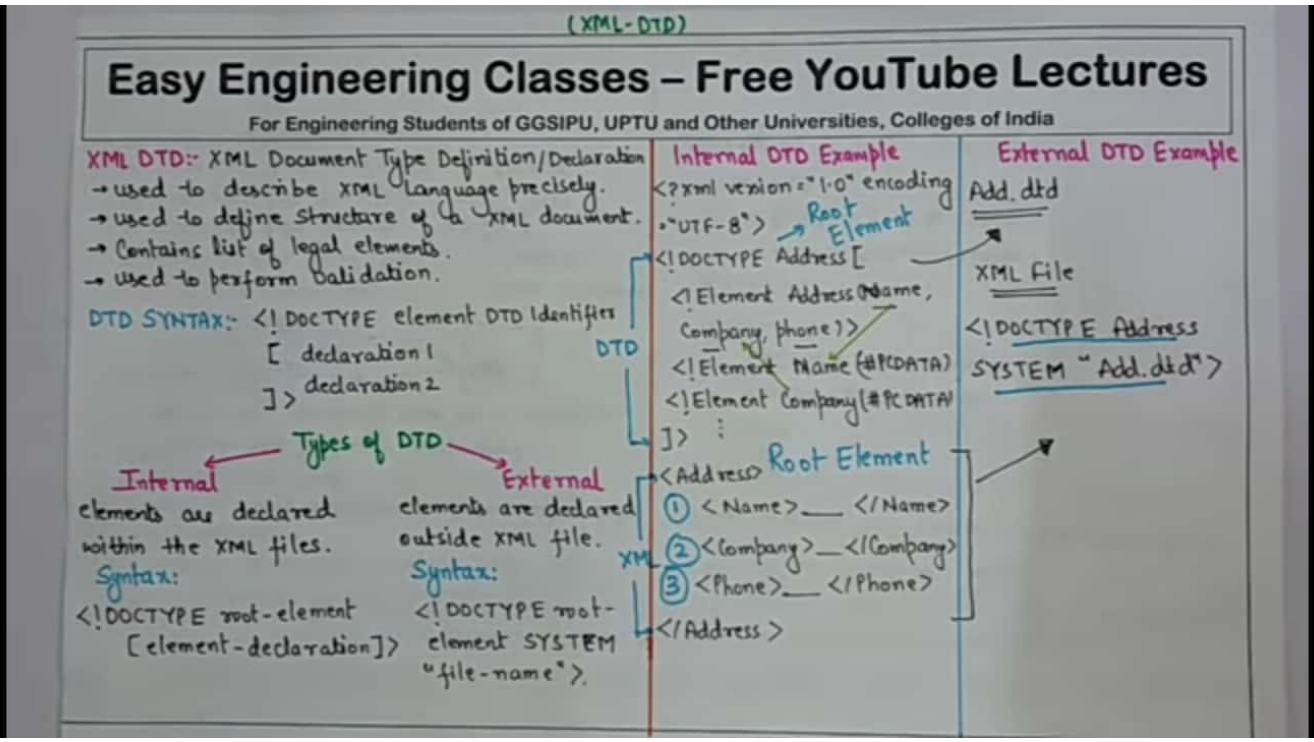


Features and Advantages:-

i) Separates data from Html.
ii) Simplifies data sharing.
iii) Simplifies data transport.
iv) Increases data availability.
v) Simplifies platform change.

XML Example: (college) — Hierarchial Structure. (college ← root; child ↳ class1, class2)

```
<?xml version="1.0" encoding="ISO-8859-1"?>   ↳ declaration
<college>  ← Root Element
  <class1>
    <Name> Amit </Name>
    <RollNo> 1 </RollNo>
  </class1>                                   ↳ child Element.
  <class2>
    <Name> Mohan </Name>
    <RollNo> 2 </RollNo>
  </class2>
</college>
```



(XML Namespaces)

Easy Engineering Classes – Free YouTube Lectures

For Engineering Students of GGSIPU, UPTU and Other Universities, Colleges of India

XML Namespaces:- used to avoid element name conflict in XML document.
→ It is a set of unique names.
→ Identified by URI(Uniform Resource Identifier)
→ Attribute name must start with "xmlns"

Syntax:- <element xmlns:name = "URL" >
element and Attributes names belongs to URL. Prefix → URI

Conflict: Generally conflict occurs when we try to mix XML documents from diff. XML Application.

Eg. of conflict.

1.xml
```
<class>
  <name>___</name>
</class>
```

2.xml
```
<class>
  <name>___
  </name>
</class>
```

Conflict occurs due to same element name.

Example of Namespace: 1.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<c1:class xmlns:c1="class1 ....">
  <c1:name> Aman </c1:name>
</c1:class>
```

(c1 → Prefix; UTF-8 → URI)

Now there will be no conflict due to namespace.

2.xml

```
<c2:class xmlns:c2="class2 ...">
  <c2:name> Aman </c2:name>
</c2:class>
```

XML- Schemas:- Commonly known as XML Schema Definition (XSD). It is used to describe & validate the structure and content of XML Data.

→ It is a method of expressing constraints about XML documents.

→ It is like DTD but provides more control on XML structure.

Syntax: <xs: schema xmlns: xs="____" >

Definition Types

Simple Type
used only in the content of the text.
eg: xs: int, xs: string.
<xs: element name= "Phone" type="xs: int" />

Complex Type
It is the container for other element definitions. Allows you to specify which child elements an element can contain & to provide some structure within your XML documents

eg:- of Complex Type (Add.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs: schema xmlns: xs="schema1...." >
  <xs: element name="Address" >
    <xs: complexType>
    <xs: sequence>
      <xs: element name="Name" type="xs:string" />
      <xs: element name="Phone" type="xs:int" />
    </xs: sequence>
  </xs: complexType>
</xs: element>
</xs:schema>
```

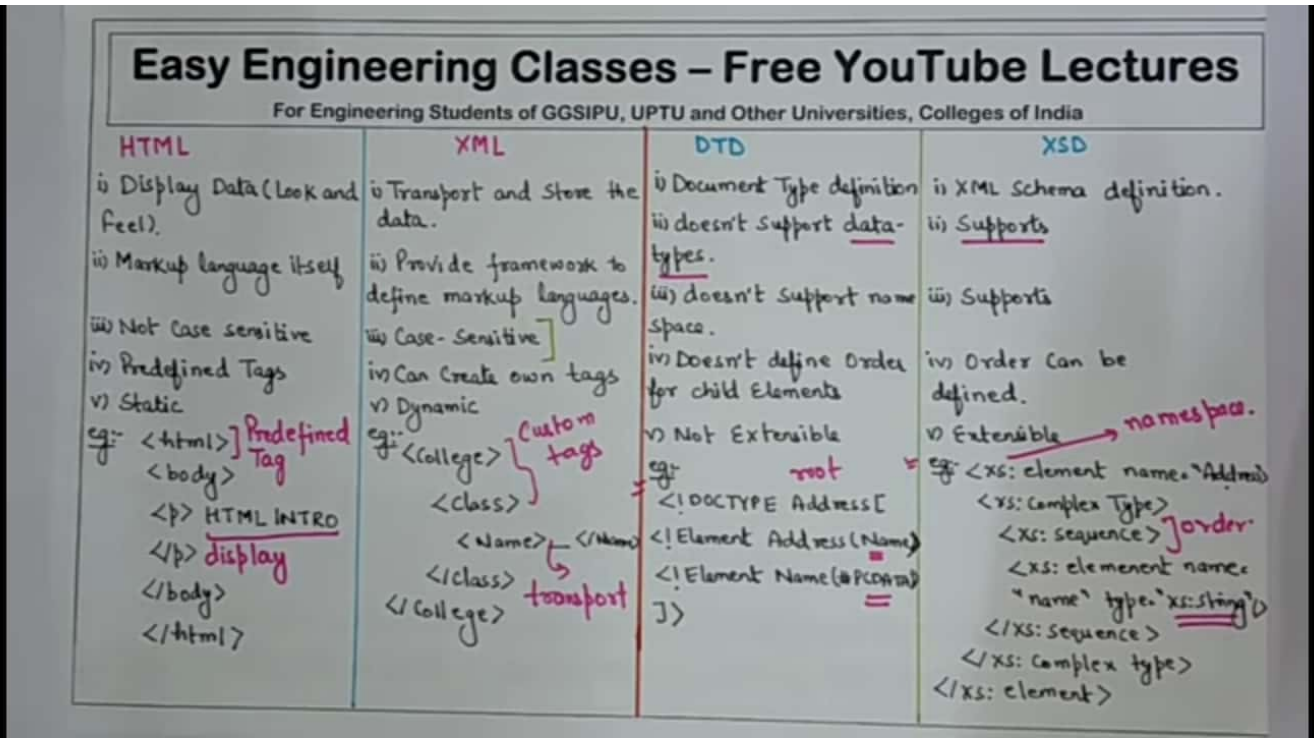
Child elements should appear in sequence.

(Add.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<Address
  xsi: schemaLocation="____Add.xsd" >
1<Name> Aman </Name>
2<Phone> 9810 </Phone>
</Address>
```

Path of xml schema defn.

123 abcd Error



(XML with CSS)

Easy Engineering Classes – Free YouTube Lectures

For Engineering Students of GGSIPU, UPTU and Other Universities, Colleges of India

XML with CSS:- CSS Stands for Cascading Style Sheets. It Can be used to add Style and display info" to an XML document.

Syntax:- `<?xml-stylesheet type="text/css" href="xml1.css"?>`

Example:- xml1.css

```
College
{
  background-Color: Blue;
}
Name
 { Color: Black;
 }
Roll
 { Color: Green;
 }
```

```
<elements>
  ↓
<College>
  <Name>__</Name>
  <roll>__</roll>
</College>
```

```
<?xml-stylesheet
type="text/css"
href="xml1.css"?>
```

XML with XSLT:- "XSL" → Extensible Stylesheet Language. "XSL" is a Styling language for XML.

XSLT:- XML Transformations.

↳ It is used to transform an XML document into another XML document, or another document type.

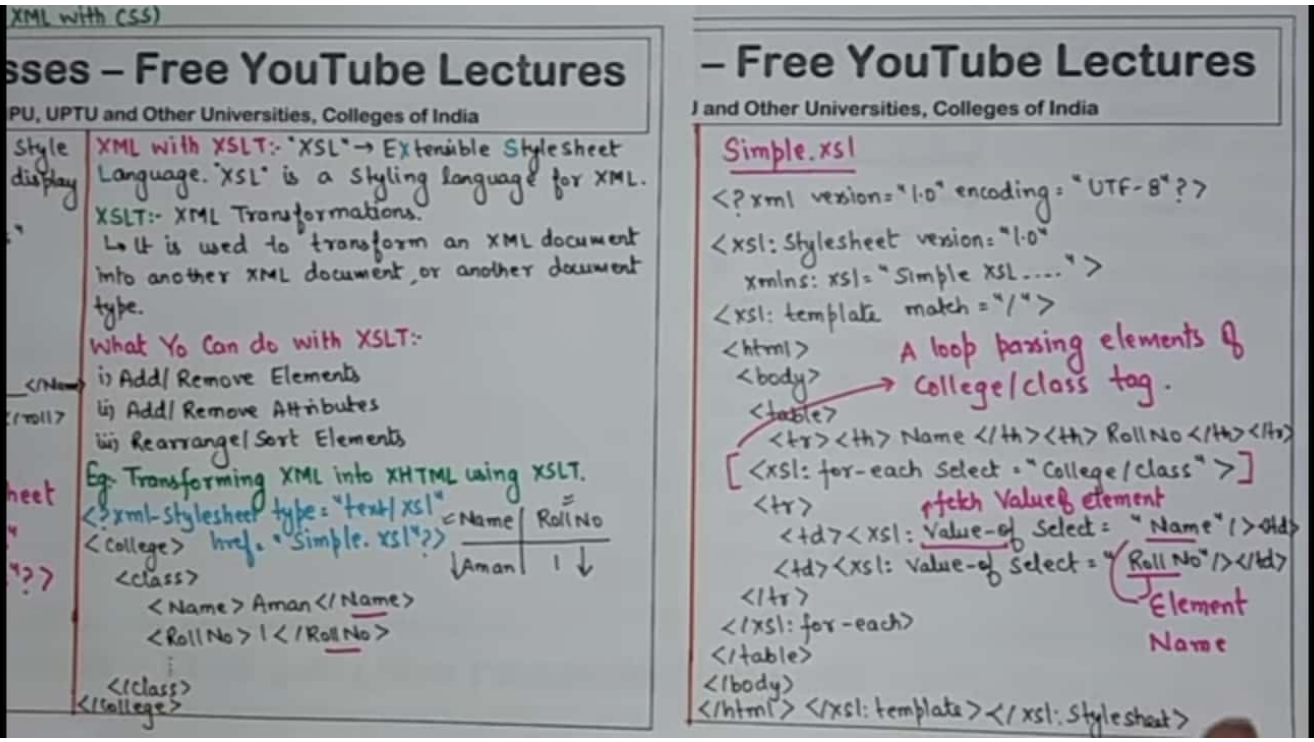
What Yo Can do with XSLT:-

i) Add/ Remove Elements
ii) Add/ Remove Attributes
iii) Rearrange/ Sort Elements

Eg: Transforming XML into XHTML using XSLT.

Name	Roll No
Aman	1

```
<College>
  <class>
    <Name> Aman </Name>
    <Roll No> 1 </Roll No>
    ⋮
  </class>
</College>
```



(XML-DOM)

Easy Engineering Classes – Free YouTube Lectures

For Engineering Students of GGSIPU, UPTU and Other Universities, Colleges of India

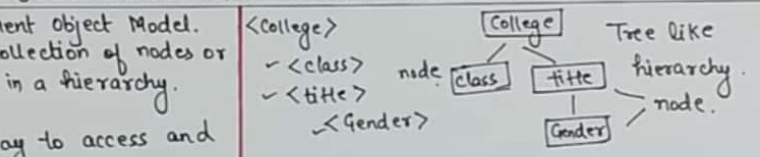
XML-DOM:- DOM is Document object Model.
↳ DOM document is a collection of nodes or pieces of info" organized in a hierarchy.
→ Tree-Based.
→ Defines a standard way to access and manipulate documents.
→ Programmer can modify / delete their content and can also create new elements.
→ Elements, their content (text & Attributes) are all known as Nodes.

Example: <h2 id="demo"> </h2>

HTML DOM.

```
<button type="button" onclick="
document.getElementById('demo).
Inner HTML="Hello"> click
</button>
```

```
<College>
 - <class>
 - <title>
   <Gender>
```



Tree like hierarchy.
node.

Example of XML DOM:-

//College.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<College>
  <Branch Category="IT">
    <Students>60</Students>
    <HOD> ABC </HOD>
  </Branch>
  <Branch Category="ECE">
    <Students>100</Students>
    <HOD> XYZ </HOD>
  </Branch>
</College>
```

Get Value of XML element

```
t = xmlDoc.getElementsByTagName("Students")
[0].childNodes[0].
nodeValue;
```

o/p = 60

```
[1].childNodes[1].
nodeValue;
```

o/p:- 100

Easy Engineering Classes – Free YouTube Lectures

For Engineering Students of GGSIPU, UPTU and Other Universities, Colleges of India

XML DOM Properties (x = node object)

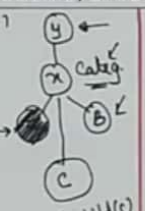
i) x.nodeName → name of x

ii) x.nodeValue → Value of x

iii) x.parentNode → Parent node of x

iv) x.childNodes → Child nodes of x

v) x.attributes → attributes of x.



x.appendChild(c)

x.removeChild(z);

XML DOM Methods

i) x.getElementsByTagName(name) → get all elements with "name".

(replace name with Tag name)

ii) x.appendChild(node) → insert a child node to x

iii) x.removeChild(node) → removes a child node from x.