UNIVERSITY OF MUMBAI

XML

BY:

PRIYUSH B.K.

NEW EDITION

2022

What is xml:

- Xml (eXtensible Markup Language) is a mark up language.
- XML is designed to store and transport data.
- Xml was released in late 90's. it was created to provide an easy to use and store self describing data.
- XML became a W3C Recommendation on February 10, 1998.
- XML is not a replacement for HTML.
- XML is designed to be self-descriptive.
- XML is designed to carry data, not to display data.
- XML tags are not predefined. You must define your own tags.
- XML is platform independent and language independent.

Features:

XML separates data from HTML:

If you need to display dynamic data in your HTML document, it will take a lot of work to edit the HTML each time the data changes.

With XML, data can be stored in separate XML files. This way you can focus on using HTML/CSS for display and layout, and be sure that changes in the underlying data will not require any changes to the HTML.

XML simplifies data sharing:

XML data is stored in plain text format. This provides a software- and hardware-independent way of storing data.

This makes it much easier to create data that can be shared by different applications.

XML simplifies data transport:

One of the most time-consuming challenges for developers is to exchange data between incompatible systems over the Internet.

Exchanging data as XML greatly reduces this complexity, since the data can be read by different incompatible applications.

XML simplifies Platform change:

Upgrading to new systems (hardware or software platforms), is always time consuming. Large amounts of data must be converted and incompatible data is often lost.

XML data is stored in text format. This makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.

XML increases data availability:

Different applications can access your data, not only in HTML pages, but also from XML data sources.

With XML, your data can be available to all kinds of "reading machines" (Handheld computers, voice machines, news feeds, etc), and make it more available for blind people, or people with other disabilities.

XML can be used to create new internet languages:

A lot of new Internet languages are created with XML.

Here are some examples:

- XHTML
- WSDL for describing available web services
- WAP and WML as markup languages for handheld devices
- RSS languages for news feeds
- RDF and OWL for describing resources and ontology
- SMIL for describing multimedia for the web.

🖉Advantages of XML:

- It supports Unicode, allowing almost any information in any written human language to be communicated.
- It can represent common computer science data structures: records, lists, and trees.
- Its self-documenting format describes structure and field names as well as specific values.
- The strict syntax and parsing requirements make the necessary parsing algorithms extremely simple, efficient, and consistent.
- XML is heavily used as a format for document storage and processing, both online and offline.
- It is based on international standards.
- It can be updated incrementally.

🖉Disadvantages of XML:

- XML syntax is redundant or large relative to binary representations of similar data, especially with tabular data.
- The redundancy may affect application efficiency through higher storage, transmission and processing costs
- XML syntax is verbose, especially for human readers, relative to other alternatives ‘text-based’ data transmission formats.

Ex.

```
1. <?xml version="1.0" encoding="ISO-8859-1"?>
2. <!-- Write your comment-->
3. <note>
4.   <to>Tove</to>
5.   <from>Jani</from>
6.   <heading>Reminder</heading>
7.   <body>Don't forget me this weekend!</body>
8. </note>
```

The first line is the XML declaration. It defines the XML version (1.0) and the encoding used (ISO-8859-1 = Latin-1/West European character set).

The next line describes the root element of the document (like saying: "this document is a note"): <note>

The next 4 lines describe 4 child elements of the root (to, from, heading, and body).

```
1. <to>Tove</to>
2. <from>Jani</from>
3. <heading>Reminder</heading>
4. <body>Don't forget me this weekend!</body>
```

And finally, the last line defines the end of the root element.

</note>

XML documents must contain a root element. This element is "the parent" of all other elements.

The elements in an XML document form a document tree. The tree starts at the root and branches to the lowest level of the tree.

SYNTAX XML:

```
1) <root>
2) <child>
3) <subchild>.....</subchild>
4) </child>
5) </root>
```

No.	HTML	XML
1)	HTML is used **to display data** and focuses on how data looks.	XML is a software and hardware independent tool used **to transport and store data**. It focuses on what data is.
2)	HTML is a **markup language** itself.	XML provides a **framework to define markup languages**.
3)	HTML is **not case sensitive**.	XML is **case sensitive**.
4)	HTML is a presentation language.	XML is neither a presentation language nor a programming language.
5)	HTML **has its own predefined tags**.	You **can define tags according to your need**.
6)	In HTML, it is **not necessary to use a closing tag**.	XML **makes it mandatory to use a closing tag**.
7)	HTML is **static** because it is used to display data.	XML is **dynamic** because it is used to transport data.
8)	HTML **does not preserve whitespaces**.	XML **preserve whitespaces**

XML Attributes:

XML elements can have attributes. By the use of attributes, we can add the information about the element.

EX.

```
1. <book publisher="Tata McGraw Hill"></book>
```

XML Comments:

XML comments are just like HTML comments. We know that the comments are used to make codes more understandable other developers.

XML Comments add notes or lines for understanding the purpose of an XML code. Although XML is known as self-describing data but sometimes XML comments are necessary.

EX. .

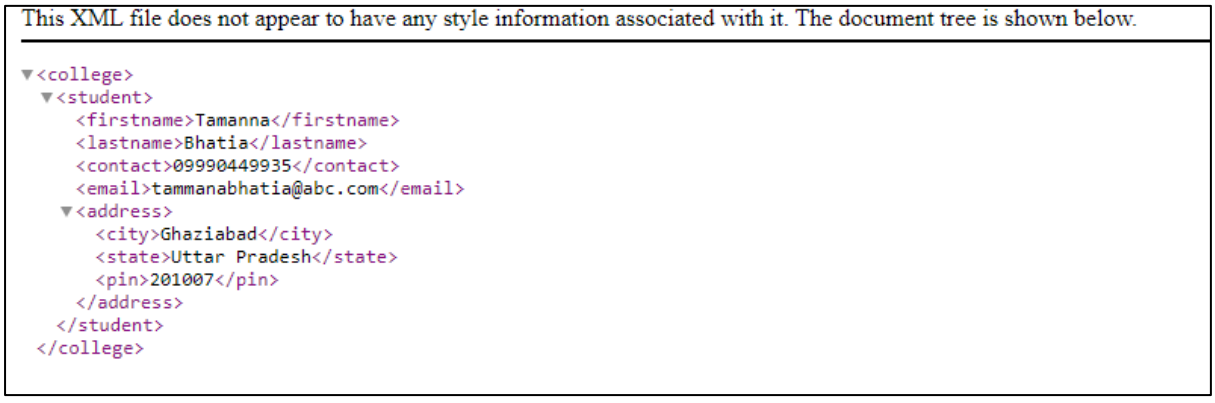
☑XML Tree Structure:

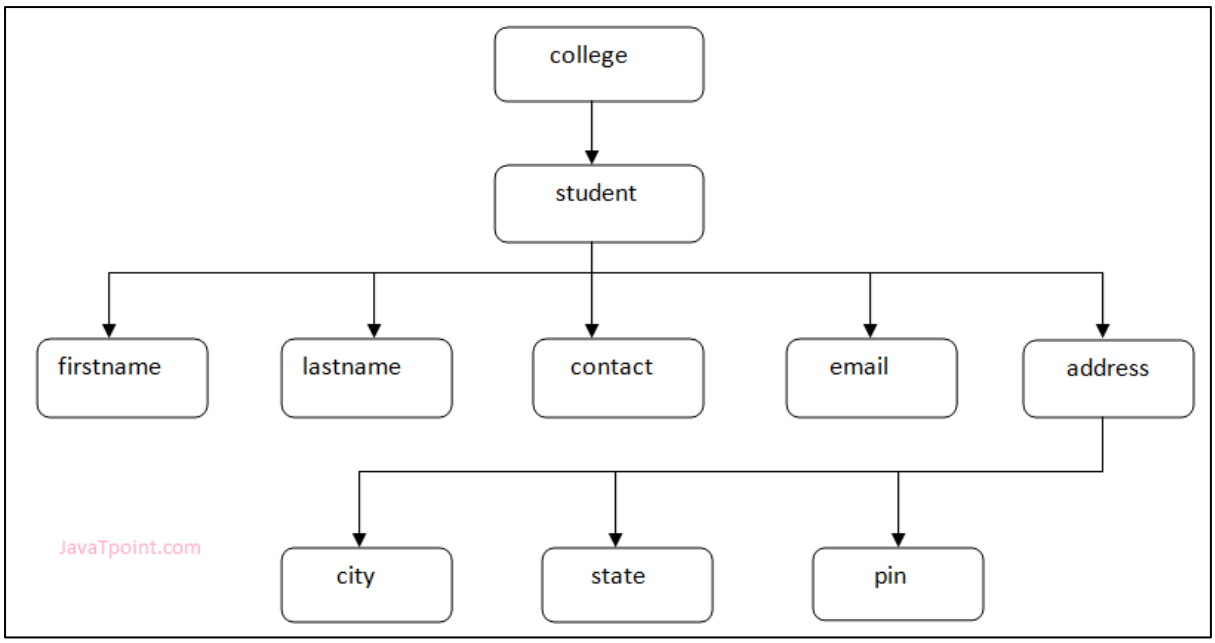
An XML document has a self-descriptive structure. It forms a tree structure which is referred as an XML tree. The tree structure makes easy to describe an XML document.

A tree structure contains root element (as parent), child element and so on. It is very easy to traverse all succeeding branches and sub-branches and leaf nodes starting from the root.

EX.

```
1.  <?xml version="1.0"?>
2.  <college>
3.    <student>
4.      <firstname>Tamanna</firstname>
5.      <lastname>Bhatia</lastname>
6.      <contact>09990449935</contact>
7.      <email>tammanabhatia@abc.com</email>
8.      <address>
9.          <city>Ghaziabad</city>
10.         <state>Uttar Pradesh</state>
11.         <pin>201007</pin>
12.     </address>
13.   </student>
14. </college>
```





In the above example, first line is the XML declaration. It defines the XML version 1.0. Next line shows the root element (college) of the document. Inside that there is one more element (student). Student element contains five branches named <firstname>, <lastname>, <contact>, <Email> and <address>.

<address> branch contains 3 sub-branches named <city>, <state> and <pin>.

XML Tree Rules:

These rules are used to figure out the relationship of the elements. It shows if an element is a child or a parent of the other element.

Descendants: If element A is contained by element B, then A is known as descendant of B. In the above example "College" is the root element and all the other elements are the descendants of "College".

Ancestors: The containing element which contains other elements is called "Ancestor" of other element. In the above example Root element (College) is ancestor of all other elements.

☑ XML Validation:

A well formed XML document can be validated against DTD or Schema.

A well-formed XML document is an XML document with correct syntax. It is very necessary to know about valid XML document before knowing XML validation.

Valid XML document:

- It must be well formed (satisfy all the basic syntax condition)
- It should be behave according to predefined DTD or XML schema

Rules for well formed XML:

- It must begin with the XML declaration.
- It must have one unique root element.
- All start tags of XML documents must match end tags.
- XML tags are case sensitive.
- All elements must be closed.
- All elements must be properly nested.
- All attributes' values must be quoted.

XML DTD:

What is DTD:

DTD stands for Document Type Definition. It defines the legal building blocks of an XML document. It is used to define document structure with a list of legal elements and attributes.

Purpose of DTD:

Its main purpose is to define the structure of an XML document. It contains a list of legal elements and define the structure with the help of them.

Valid and well-formed XML document with DTD:

Let's take an example of well-formed and valid XML document. It follows all the rules of DTD.

employee.xml

```
1. <?xml version="1.0"?>
2. <!DOCTYPE employee SYSTEM "employee.dtd">
3. <employee>
4.   <firstname>vimal</firstname>
5.   <lastname>jaiswal</lastname>
6.   <email>vimal@javatpoint.com</email>
7. </employee>
```

In the above example, the DOCTYPE declaration refers to an external DTD file. The content of the file is shown in below paragraph.

employee.dtd

```
<!ELEMENT employee (firstname,lastname,email)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT email (#PCDATA)>
```

Description of DTD:

- **<!DOCTYPE employee** : It defines that the root element of the document is employee.
- **<!ELEMENT employee**: It defines that the employee element contains 3 elements "firstname, lastname and email".
- **<!ELEMENT firstname**: It defines that the firstname element is #PCDATA typed. (parse-able data type).
- **<!ELEMENT lastname**: It defines that the lastname element is #PCDATA typed. (parse-able data type).
- **<!ELEMENT email:** It defines that the email element is #PCDATA typed. (parse-able data type).

XML CSS:

Purpose of CSS in XML

CSS (Cascading Style Sheets) can be used to add style and display information to an XML document. It can format the whole XML document.

How to link XML file with CSS

To link XML files with CSS, you should use the following syntax:

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

XML CSS Example:

cssemployee.css

```
employee
{
background-color: pink;
}
firstname,lastname,email
{
font-size:25px;
display:block;
color: blue;
margin-left: 50px;
}
```

Let's create the DTD file.

employee.dtd

```
<!ELEMENT employee (firstname,lastname,email)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT email (#PCDATA)>
```

Let's see the xml file using CSS and DTD.

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/css" href="cssemployee.css"?>
<!DOCTYPE employee SYSTEM "employee.dtd">
<employee>
  <firstname>vimal</firstname>
  <lastname>jaiswal</lastname>
  <email>vimal@javatpoint.com</email>
</employee>
```

✳XML schema:

What is XML schema

XML schema is a language which is used for expressing constraint about XML documents. There are so many schema languages which are used now a days for example Relax- NG and XSD (XML schema definition).

An XML schema is used to define the structure of an XML document. It is like DTD but provides more control on XML structure.

Checking Validation:

An XML document is called "well-formed" if it contains the correct syntax. A well-formed and valid XML document is one which have been validated against Schema.

XML Schema Example:

Let's create a schema file.

employee.xsd

```
1. <?xml version="1.0"?>
2. <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
3. targetNamespace="http://www.javatpoint.com"
4. xmlns="http://www.javatpoint.com"
5. elementFormDefault="qualified">
6. <xs:element name="employee">
7.   <xs:complexType>
8.     <xs:sequence>
9.       <xs:element name="firstname" type="xs:string"/>
10.      <xs:element name="lastname" type="xs:string"/>
11.      <xs:element name="email" type="xs:string"/>
12.    </xs:sequence>
13.  </xs:complexType>
14. </xs:element>
15. </xs:schema>
```

Let's see the xml file using XML schema or XSD file.

employee.xml

```
<?xml version="1.0"?>
<employee
xmlns="http://www.javatpoint.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.javatpoint.com employee.xsd">

  <firstname>vimal</firstname>
  <lastname>jaiswal</lastname>
  <email>vimal@javatpoint.com</email>
</employee>
```

Description of XML Schema:

<xs:element name="employee"> : It defines the element name employee.

<xs:complexType> : It defines that the element 'employee' is complex type.

<**xs:sequence>** : It defines that the complex type is a sequence of elements.

<xs:element name="firstname" type="xs:string"/> : It defines that the element 'firstname' is of string/text type.

<xs:element name="lastname" type="xs:string"/> : It defines that the element 'lastname' is of string/text type.

<xs:element name="email" type="xs:string"/> : It defines that the element 'email' is of string/text type**.**

XML Schema Data types:-

There are two types of data types in XML schema.

- ✓ Simple Type
- ✓ Complex Type

Simple Type:

The simple Type allows you to have text-based elements. It contains less attributes, child elements, and cannot be left empty.

Complex Type:

The complex Type allows you to hold multiple attributes and elements. It can contain additional sub elements and can be left empty.

No.	DTD	XSD
1)	DTD stands for **Document Type Definition**.	XSD stands for XML Schema Definition.
2)	DTDs are derived from **SGML** syntax.	XSDs are written in XML.
3)	DTD **doesn't support datatypes**.	XSD **supports datatypes** for elements and attributes.
4)	DTD **doesn't support namespace**.	XSD **supports namespace**.
5)	DTD **doesn't define order** for child elements.	XSD **defines order** for child elements.
6)	DTD is **not extensible**.	XSD is **extensible**.
7)	DTD is **not simple to learn**.	XSD is **simple to learn** because you don't need to learn new language.
8)	DTD provides **less control** on XML structure.	

XML Parsers:

An XML parser is a software library or package that provides interfaces for client applications to work with an XML document. The XML Parser is designed to read the XML and create a way for programs to use XML.

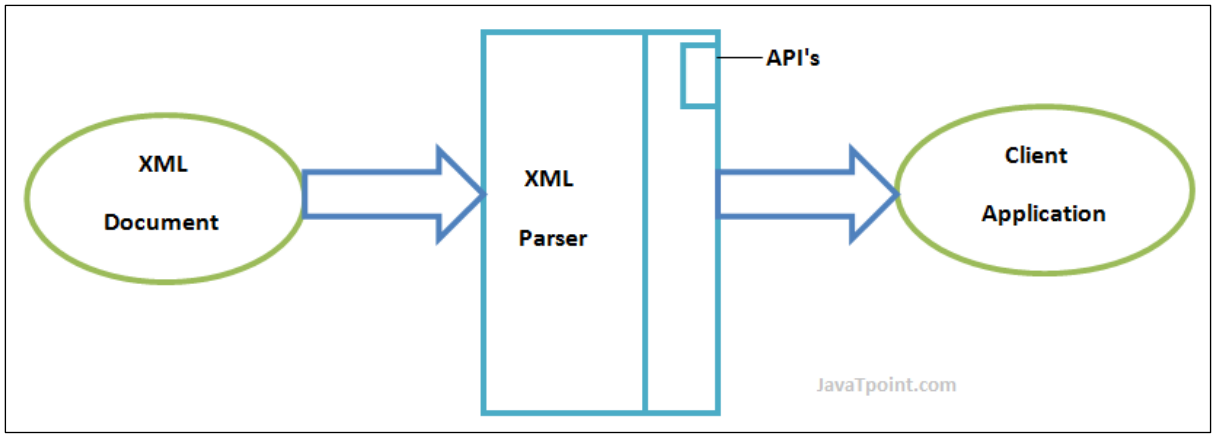
XML parser validates the document and check that the document is well formatted.

Types of XML Parsers:

These are the two main types of XML Parsers:

- ✓ DOM
- ✓ SAX

Let's understand the working of XML parser by the figure given below:



DOM (Document Object Model):

A DOM document is an object which contains all the information of an XML document. It is composed like a tree structure. The DOM Parser implements a DOM API. This API is very simple to use.

Features of DOM Parser:

A DOM Parser creates an internal structure in memory which is a DOM document object and the client applications get information of the original XML document by invoking methods on this document object.

DOM Parser has a tree-based structure.

Advantages:

1) It supports both read and write operations and the API is very simple to use.
2) It is preferred when random access to widely separated parts of a document is required.

Disadvantages:

1) It is memory inefficient. (Consumes more memory because the whole XML document needs to loaded into memory).
2) It is comparatively slower than other parsers.

SAX (Simple API for XML):

A SAX Parser implements SAX API. This API is an event-based API and less intuitive.

Features of SAX Parser:

It does not create any internal structure.

Clients does not know what methods to call, they just override the methods of the API and place his own code inside method.

It is an event-based parser, it works like an event handler in Java.

Advantages:

1) It is simple and memory efficient.
2) It is very fast and works for huge documents.

Disadvantages:

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

What is XML DOM:

DOM is an acronym stands for Document Object Model. It defines a standard way to access and manipulate documents. The Document Object Model (DOM) is a programming API for HTML and XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated.

What does XML DOM

- ✓ The XML DOM makes a tree-structure view for an XML document.
- ✓ We can access all elements through the DOM tree.
- ✓ We can modify or delete their content and also create new elements. The elements, their content (text and attributes) are all known as nodes.

XML DOM Example : Load XML File:

Let's take an example to show how an XML document ("note.xml") is parsed into an XML DOM object.

This example parses an XML document (note.xml) into an XML DOM object and extracts information from it with JavaScript.

Let's see the XML file that contains message.

note.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
  <to>sonoojaiswal@javatpoint.com</to>
  <from>vimal@javatpoint.com</from>
  <body>Hello XML DOM</body>
</note>
```

Let's see the HTML file that extracts the data of XML document using DOM.

xmldom.html

```
<!DOCTYPE html>
<html>
<body>
<h1>Important Note</h1>
<div>
<b>To:</b> <span id="to"></span><br>
<b>From:</b> <span id="from"></span><br>
<b>Message:</b> <span id="message"></span>
</div>
<script>
if (window.XMLHttpRequest)
  {// code for IE7+, Firefox, Chrome, Opera, Safari
  xmlhttp=new XMLHttpRequest();
  }
else
  {// code for IE6, IE5
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
xmlhttp.open("GET","note.xml",false);
xmlhttp.send();
xmlDoc=xmlhttp.responseXML;
document.getElementById("to").innerHTML=
xmlDoc.getElementsByTagName("to")[0].childNodes[0].nodeValue;
document.getElementById("from").innerHTML=
xmlDoc.getElementsByTagName("from")[0].childNodes[0].nodeValue;
```

```
26. document.getElementById("message").innerHTML=
27. xmlDoc.getElementsByTagName("body")[0].childNodes[0].nodeValue;
28. </script>
29. </body>
30. </html>
```

Programming Interface:

The DOM models XML as a set of node objects. The nodes can be accessed with JavaScript or other programming languages. In this tutorial we use JavaScript.

The programming interface to the DOM is defined by a set standard properties and methods.

Properties are often referred to as something that is (i.e. nodename is "book").

Methods are often referred to as something that is done (i.e. delete "book").

XML DOM Properties:

These are some typical DOM properties:

- x.nodeName - the name of x
- x.nodeValue - the value of x
- x.parentNode - the parent node of x
- x.childNodes - the child nodes of x
- x.attributes - the attributes nodes of x
- Note: In the list above, x is a node object.

XML DOM Methods:

x.getElementsByTagName(name) - get all elements with a specified tag name

x.appendChild(node) - insert a child node to x

x.removeChild(node) - remove a child node from x

XSLT:

XSL stands for EXtensible Stylesheet Language. It is a styling language for XML just like CSS is a styling language for HTML.

XSLT stands for XSL Transformation. It is used to transform XML documents into other formats (like transforming XML into HTML).

“In HTML documents, tags are predefined but in XML documents, tags are not predefined. World Wide Web Consortium (W3C) developed XSL to understand and style an XML document, which can act as XML based Stylesheet Language.”

Main parts of XSL Document:

XSLT: It is a language for transforming XML documents into various other types of documents.

XPath: It is a language for navigating in XML documents.

XQuery: It is a language for querying XML documents.

XSL-FO: It is a language for formatting XML documents

Advantage of XSLT:

A list of advantages of using XSLT:

- XSLT provides an easy way to merge XML data into presentation because it applies user defined transformations to an XML document and the output can be HTML, XML, or any other structured document.
- XSLT provides Xpath to locate elements/attribute within an XML document. So it is more convenient way to traverse an XML document rather than a traditional way, by using scripting language.
- XSLT is template based. So it is more resilient to changes in documents than low level DOM and SAX.
- By using XML and XSLT, the application UI script will look clean and will be easier to maintain.
- XSLT templates are based on XPath pattern which is very powerful in terms of performance to process the XML document.
- XSLT can be used as a validation language as it uses tree-pattern-matching approach.

Ex .

XML Namespaces

XML Namespace is used to avoid element name conflict in XML document.

XML Namespace Declaration

An XML namespace is declared using the reserved XML attribute. This attribute name must be started with "xmlns".

Let's see the XML namespace syntax:

<element xmlns:name = "URL">

Here, namespace starts with keyword **"xmlns"**. The word **name** is a namespace prefix. The **URL** is a namespace identifier.

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <cont:contact xmlns:cont="http://sssit.org/contact-us">
3.   <cont:name>Vimal Jaiswal</cont:name>
4.   <cont:company>SSSIT.org</cont:company>
5.   <cont:phone>(0120) 425-6464</cont:phone>
6. </cont:contact>
```

Namespace Prefix: cont

Namespace Identifier: http://sssit.org/contact-us

It specifies that the element name and attribute names with cont prefix belongs to http://sssit.org/contact-us name space.