1. XML Basics

Link : https://www.w3schools.com/xmL/xml\_syntax.asp

<?xml version="1.0" encoding="UTF-8"?>  
<note>  
  <to>Tove</to>  
  <from>Jani</from>  
  <heading>Reminder</heading>  
  <body>Don't forget me this weekend!</body>  
</note>

* version="1.0" means that this is the XML standard this file conforms to
* The **encoding** declaration identifies which **encoding** is used to represent the characters in the document. Although **XML** parsers can determine automatically if a document uses the **UTF**-**8** or **UTF**-16 Unicode **encoding**, this declaration should be used in documents that support other encodings.

XML Namespaces - The xmlns Attribute

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*".

When a namespace is defined for an element, all child elements with the same prefix are associated with the same namespace.

Namespaces can also be declared in the XML root element:

<root   
xmlns:h="http://www.w3.org/TR/html4/"  
xmlns:f="https://www.w3schools.com/furniture">

**Note:** 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.

However, companies often use the namespace as a pointer to a web page containing namespace information.

## Uniform Resource Identifier (URI)

A **Uniform Resource Identifier** (URI) is a string of characters which identifies an Internet Resource.

The most common URI is the **Uniform Resource Locator** (URL) which identifies an Internet domain address. Another, not so common type of URI is the **Universal Resource Name** (URN).

1. XSLT Basics

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

XSTL is used to transform xml into other format like html or pdf based on the API.

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.

<?xml version="1.0"?>  
  
<xsl:stylesheet version="1.0"  
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">  
  
<xsl:template match="/">  
  <html>  
  <body>  
    <h2>My CD Collection</h2>  
    <table border="1">  
      <tr bgcolor="#9acd32">  
        <th>Title</th>  
        <th>Artist</th>  
      </tr>  
      <xsl:for-each select="catalog/cd">  
        <tr>  
          <td><xsl:value-of select="title"/></td>  
          <td><xsl:value-of select="artist"/></td>  
        </tr>  
      </xsl:for-each>  
    </table>  
  </body>  
  </html>  
</xsl:template>  
  
</xsl:stylesheet>

## Link the XSL Style Sheet to the XML Document

Add the XSL style sheet reference to your XML document ("cdcatalog.xml"):

<?xml version="1.0" encoding="UTF-8"?>  
<?xml-stylesheet type="text/xsl" href="cdcatalog.xsl"?>  
<catalog>  
  <cd>  
    <title>Empire Burlesque</title>  
    <artist>Bob Dylan</artist>  
    <country>USA</country>  
    <company>Columbia</company>  
    <price>10.90</price>  
    <year>1985</year>  
  </cd>  
.  
.  
</catalog>

1. XML DTD

This is used to validate the xml format.

An XML document with correct syntax is called "Well Formed".

An XML document validated against a DTD is both "Well Formed" and "Valid".

<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE note SYSTEM "Note.dtd">  
<note>  
<to>Tove</to>  
<from>Jani</from>  
<heading>Reminder</heading>  
<body>Don't forget me this weekend!</body>  
</note>

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

1. XSD Basic

An XML Schema describes the structure of an XML document.

The XML Schema language is also referred to as XML Schema Definition (XSD).

This is similar to DTD that validate the xml document but more advance data type.

schema is a blueprint, defining the rules for creating a certain "shape" of XML for purchase orders.

<?xml version="1.0"?>  
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">  
  
<xs:element name="note">  
  <xs:complexType>  
    <xs:sequence>  
      <xs:element name="to" type="xs:string"/>  
      <xs:element name="from" type="xs:string"/>  
      <xs:element name="heading" type="xs:string"/>  
      <xs:element name="body" type="xs:string"/>  
    </xs:sequence>  
  </xs:complexType>  
</xs:element>  
  
</xs:schema>

## A Reference to an XML Schema

This XML document has a reference to an XML Schema:

<?xml version="1.0"?>  
  
<note  
xmlns="https://www.w3schools.com"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xsi:schemaLocation="https://www.w3schools.com note.xsd">  
  <to>Tove</to>  
  <from>Jani</from>  
  <heading>Reminder</heading>  
  <body>Don't forget me this weekend!</body>  
</note>

1. XML Bean : <http://xmlbeans.apache.org/overview.htm>

**What is XMLBeans?**

XMLBeans is a technology for accessing XML by binding it to Java types. XMLBeans provides several ways to get at the XML, including:

* Through XML schema that has been compiled to generate Java types that represent schema types. In this way, you can access instances of the schema through JavaBeans-style accessors after the fashion of "getFoo" and "setFoo".

The XMLBeans API also allows you to reflect into the XML schema itself through an XML Schema Object model.

* A cursor model through which you can traverse the full XML infoset.
* Support for XML DOM

# Getting Started with XMLBeans

XMLBeans provides intuitive ways to handle XML that make it easier for you to access and manipulate XML data and documents in Java.

Characteristics of XMLBeans approach to XML:

* It provides a familiar Java object-based view of XML data without losing access to the original, native XML structure.
* The XML's integrity as a document is not lost with XMLBeans. XML-oriented APIs commonly take the XML apart in order to bind to its parts. With XMLBeans, the entire XML instance document is handled as a whole. The XML data is stored in memory as XML. This means that the document order is preserved as well as the original element content with whitespace.
* With types generated from schema, access to XML instances is through JavaBean-like accessors, with get and set methods.
* It is designed with XML schema in mind from the beginning — XMLBeans supports all XML schema definitions.
* Access to XML is fast.

The starting point for XMLBeans is XML schema. A schema (contained in an XSD file) is an XML document that defines a set of rules to which other XML documents must conform. The XML Schema specification provides a rich data model that allows you to express sophisticated structure and constraints on your data. For example, an XML schema can enforce control over how data is ordered in a document, or constraints on particular values (for example, a birth date that must be later than 1900). Unfortunately, the ability to enforce rules like this is typically not available in Java without writing custom code. XMLBeans honors schema constraints.

**Note:** Where an XML schema defines rules for an XML document, an XML instance is an XML document that conforms to the schema.

You compile a schema (XSD) file to generate a set of Java interfaces that mirror the schema. With these types, you process XML instance documents that conform to the schema. You bind an XML instance document to these types; changes made through the Java interface change the underlying XML representation.

Previous options for handling XML include using XML programming interfaces (such as DOM or SAX) or an XML marshalling/binding tool (such as JAXB). Because it lacks strong schema-oriented typing, navigation in a DOM-oriented model is more tedious and requires an understanding of the complete object model. JAXB provides support for the XML schema specification, but handles only a subset of it; XMLBeans supports all of it. Also, by storing the data in memory as XML, XMLBeans is able to reduce the overhead of marshalling and demarshalling.