**What is XML?**

* XML stands for eXtensible Markup Language
* XML is a markup language much like HTML
* XML was designed to store and transport data
* XML was designed to be self-descriptive
* XML is a W3C Recommendation

**XML Does Not DO Anything**

Maybe it is a little hard to understand, but XML does not DO anything.

**Why Study XML?**

XML plays an important role in many different IT systems.

XML is often used for distributing data over the Internet.

It is important (for all types of software developers!) to have a good understanding of XML.

**The Difference Between XML and HTML**

XML and HTML were designed with different goals:

* XML was designed to carry data - with focus on what data is
* HTML was designed to display data - with focus on how data looks
* XML tags are not predefined like HTML tags are
* HTML is a case insensitive whereas XML is case sensitive.

**XML Separates Data from HTML**

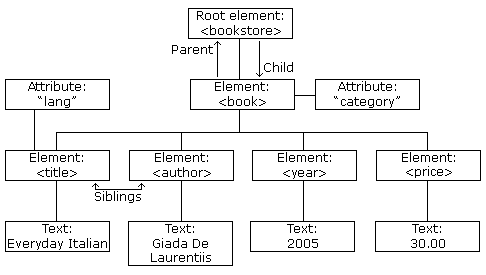
When displaying data in HTML, you should not have to edit the HTML file when the data changes.

With XML, the data can be stored in separate XML files.

With a few lines of JavaScript code, you can read an XML file and update the data content of any HTML page.

XML documents form a tree structure that starts at "the root" and branches to "the leaves".

**XML Tree Structure**



<?xml version="1.0" encoding="UTF-8**"**?>  
<bookstore>  
  <book category="cooking">  
    <title lang="en">Everyday Italian</title>  
    <author>Giada De Laurentiis</author>  
    <year>2005</year>  
    <price>30.00</price>  
  </book>  
  <book category="children">  
    <title lang="en">Harry Potter</title>  
    <author>J K. Rowling</author>  
    <year>2005</year>  
    <price>29.99</price>  
  </book>  
  <book category="web">  
    <title lang="en">Learning XML</title>  
    <author>Erik T. Ray</author>  
    <year>2003</year>  
    <price>39.95</price>  
  </book>  
</bookstore>

**XML Syntax Rules**

**XML Documents Must Have a Root Element**

**The XML Prolog**

This line is called the XML **prolog**:

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

**XML Elements**

An XML document contains XML Elements.

**What is an XML Element?**

An XML element is everything from (including) the element's start tag to (including) the element's end tag.

<price>29.99</price>

Rules for XML Document

* Every document contain a root tag

Wrong Exaple

============

<name>Ram</name>

<city>Agra</city>

Correct example

================

<emp>

<name>Ram</name>

<city>Agra</city>

</emp>

* Every tag must close
* Attribute values must be quoted
* Tag are are case sensitive.

XML Document

* Well formed document
* Valid document

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

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

**Valid XML Documents**

A "Valid" XML document is a "Well Formed" XML document, which also conforms to the rules of a DTD:

<?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>

**XML DTD**

The purpose of a DTD is to define the structure of an XML document. It defines the structure with a list of legal elements:

Note.dtd

==========  
<!ELEMENT note (to,from,heading,body)>  
<!ELEMENT to (#PCDATA)>  
<!ELEMENT from (#PCDATA)>  
<!ELEMENT heading (#PCDATA)>  
<!ELEMENT body (#PCDATA)>

The DTD above is interpreted like this:

* !DOCTYPE note defines that the root element of the document is note
* !ELEMENT note defines that the note element must contain the elements: "to, from, heading, body"
* !ELEMENT to defines the to element to be of type "#PCDATA"
* !ELEMENT from defines the from element to be of type "#PCDATA"
* !ELEMENT heading defines the heading element to be of type "#PCDATA"
* !ELEMENT body defines the body element to be of type "#PCDATA"

#PCDATA means parse-able text data.

Example 2

Abc.dtd

==========

<!ELEMENT Employee (name,phone,address,email)>

<!ELEMENT address (street,city,country)>  
<!ELEMENT name (#PCDATA)>  
<!ELEMENT phone (#PCDATA)>  
<!ELEMENT street (#PCDATA)>

<!ELEMENT city (#PCDATA)>

<!ELEMENT country (#PCDATA)>  
<!ELEMENT email (#PCDATA)>

Kp.xml

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

<!DOCTYPE Employee SYSTEM "abc.dtd">  
<Employee>

<name>Arun</name>

<phone>9034565667</phone>

<address>

<street>GauGhat</street>

<city>Mathura</city>

<country>India</country>

</address>

<email>[as@yahoo.com HYPERLINK "mailto:as@yahoo.com%3c/email"< HYPERLINK "mailto:as@yahoo.com%3c/email"/email](mailto:as@yahoo.com%3c/email)>

</employee>

Types of DTD

* External style
* Internal style

External style

DTD in separate file ( dtd file save with .dtd extension and xml file save with .xml extension)

(Above code are the example on external DTD)

Internal DTD

DTD code and xml code in single file. File must save with .xml extension.

Abc.xml

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

<!DOCTYPE note

[

<!ELEMENT note (to,from,heading,body)>  
<!ELEMENT to (#PCDATA)>  
<!ELEMENT from (#PCDATA)>  
<!ELEMENT heading (#PCDATA)>  
<!ELEMENT body (#PCDATA)>  
]>  
<note>  
<to>Tove</to>  
<from>Jani</from>  
<heading>Reminder</heading>  
<body>Don't forget me this weekend!</body>  
</note>

**XML Schema**

An XML Schema describes the structure of an XML document, just like a DTD.

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

An XML document validated against an XML Schema is both "Well Formed" and "Valid".

**XML Schema**

XML Schema is an XML-based alternative to DTD:

<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>

The Schema above is interpreted like this:

* <xs:element name="note"> defines the element called "note"
* <xs:complexType> the "note" element is a complex type
* <xs:sequence> the complex type is a sequence of elements
* <xs:element name="to" type="xs:string"> the element "to" is of type string (text)
* <xs:element name="from" type="xs:string"> the element "from" is of type string
* <xs:element name="heading" type="xs:string"> the element "heading" is of type string
* <xs:element name="body" type="xs:string"> the element "body" is of type string

**XML Schemas are More Powerful than DTD**

* XML Schemas are written in XML
* XML Schemas are extensible to additions
* XML Schemas support data types
* XML Schemas support namespaces

==============

XML Schema has a lot of built-in data types. The most common types are:

DATA TYPE IN Schema

-------------------------------

xs:string

xs:decimal

xs:integer

xs:boolean

xs:date

xs:time