XML

XML was designed to transport and store data.

HTML was designed to display data.

XML stands for EXtensible Markup Language

XML was designed to describe data.

XML is a software- and hardware-independent tool for carrying information.

XML tags are not predefined. You must define your own tags

XML is designed to be self-descriptive

XML was designed to carry data, not to display data

Introduction to XML

**XML was designed to describe data.**

**HTML was designed to display data.**

The Difference Between XML and HTML

**XML is not a replacement for HTML.**

**XML and HTML were designed with different goals:**

* **XML was designed to transport and store data, with focus on what data is**
* **HTML was designed to display data, with focus on how data looks**

**HTML is about displaying information, while XML is about carrying information.**

**XML is Not a Replacement for HTML**

**XML is a complement to HTML.**

It is important to understand that XML is not a replacement for HTML. In most web applications, XML is used to transport data, while HTML is used to format and display the data.

My best description of XML is this:

**XML is a software- and hardware-independent tool for carrying information.**

XML is now as important for the Web as HTML was to the foundation of the Web.

XML is the most common tool for data transmissions between all sorts of applications.

XML is used in many aspects of web development, often to simplify data storage and sharing.

**XML document is just information wrapped in tags. Someone must write a piece of software to send, receive or display it.**

**XML tags are "invented" by the author of the XML document.**

XML is used in many aspects of web development, often to simplify data storage and sharing.

**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 concentrate 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.**

**With a few lines of JavaScript code, you can read an external XML file and update the data content of your web page.**

XML Simplifies Data Sharing

In the real world, computer systems and databases contain data in incompatible formats.

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 Changes

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 Makes Your Data More Available

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 is 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

If Developers Have Sense

**If they DO have sense, future applications will exchange their data in XML.**

The future might give us word processors, spreadsheet applications and databases that can read each other's data in XML format, without any conversion utilities in between.

**XML Tree**

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

An Example XML Document

XML documents use a self-describing and simple syntax:

<?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>

The first line is the XML declaration. It defines the XML version (1.0).

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

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

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

XML Documents Form a Tree Structure

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.

All elements can have sub elements (child elements):

<root>  
  <child>  
    <subchild>.....</subchild>  
  </child>  
</root>

The terms parent, child, and sibling are used to describe the relationships between elements. Parent elements have children. Children on the same level are called siblings (brothers or sisters).

All elements can have text content and attributes (just like in HTML).



The image above represents one book in the XML below:

<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>

**Syntax Rules**

In XML, it is illegal to omit the closing tag. All elements **must** have a closing tag.

XML tags are case sensitive.

In HTML, you might see improperly nested elements:

<b><i>This text is bold and italic</b></i>

In XML, all elements **must** be properly nested within each other:

<b><i>This text is bold and italic</i></b>

XML documents must contain one element that is the **parent** of all other elements. This element is called the **root**element.

XML elements can have attributes in name/value pairs just like in HTML.

In XML, the attribute values must always be quoted.

<note date="12/11/2007">  
  <to>Tove</to>  
  <from>Jani</from>  
</note>

Entity References

Some characters have a special meaning in XML.

If you place a character like "<" inside an XML element, it will generate an error because the parser interprets it as the start of a new element.

This will generate an XML error:

<message>if salary < 1000 then</message>

To avoid this error, replace the "<" character with an **entity reference**:

<message>if salary &lt; 1000 then</message>

There are 5 predefined entity references in XML:

|  |  |  |
| --- | --- | --- |
| &lt; | < | less than |
| &gt; | > | greater than |
| &amp; | & | ampersand |
| &apos; | ' | apostrophe |
| &quot; | " | quotation mark |

**Note:** Only the characters "<" and "&" are strictly illegal in XML. The greater than character is legal, but it is a good habit to replace it.

Comments in XML

The syntax for writing comments in XML is similar to that of HTML.

<!-- This is a comment -->

White-space is Preserved in XML

HTML truncates multiple white-space characters to one single white-space:

|  |  |
| --- | --- |
| HTML: | Hello           Tove |
| Output: | Hello Tove |

With XML, the white-space in a document is not truncated.

XML Stores New Line as LF

Windows applications stores a new line as: carriage return and line feed (CR+LF).

Unix and Mac OSX uses LF.

Old Mac systems uses CR.

XML stores a new line as LF.

Well Formed XML

XML documents that conform to the syntax rules above are said to be "Well Formed" XML documents.

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.

An element can contain:

* other elements
* text
* attributes
* or a mix of all of the above...
* <bookstore>  
    <book category="CHILDREN">  
      <title>Harry Potter</title>  
      <author>J K. Rowling</author>  
      <year>2005</year>  
      <price>29.99</price>  
    </book>  
    <book category="WEB">  
      <title>Learning XML</title>  
      <author>Erik T. Ray</author>  
      <year>2003</year>  
      <price>39.95</price>  
    </book>  
  </bookstore>
* In the example above, <bookstore> and <book> have **element contents**, because they contain other elements. <book> also has an **attribute** (category="CHILDREN"). <title>, <author>, <year>, and <price> have **text content** because they contain text.

Empty XML Elements

An alternative syntax can be used for XML elements with no content:

Instead of writing a book element (with no content) like this:

<book></book>

It can be written like this:

<book />

This sort of element syntax is called self-closing.

XML Naming Rules

XML elements must follow these naming rules:

* Names can contain letters, numbers, and other characters
* Names cannot start with a number or punctuation character
* Names cannot start with the letters xml (or XML, or Xml, etc)
* Names cannot contain spaces

Any name can be used, no words are reserved.

XML Attributes

Attributes often provide information that is not a part of the data. In the example below, the file type is irrelevant to the data, but can be important to the software that wants to manipulate the element:

<file type="gif">computer.gif</file>

If the attribute value itself contains double quotes you can use single quotes, like in this example:

<gangster name='George "Shotgun" Ziegler'>

or you can use character entities:

<gangster name="George &quot;Shotgun&quot; Ziegler">

XML Elements vs. Attributes

Take a look at these examples:

<person sex="female">  
  <firstname>Anna</firstname>  
  <lastname>Smith</lastname>  
</person>

<person>  
  <sex>female</sex>  
  <firstname>Anna</firstname>  
  <lastname>Smith</lastname>  
</person>

In the first example sex is an attribute. In the last, sex is an element. Both examples provide the same information.

There are no rules about when to use attributes or when to use elements. Attributes are handy in HTML. In XML my advice is to avoid them. Use elements instead.

An expanded date element is used in the third: (THIS IS MY FAVORITE):

<note>  
  <date>  
    <day>10</day>  
    <month>01</month>  
    <year>2008</year>  
  </date>  
  <to>Tove</to>  
  <from>Jani</from>  
  <heading>Reminder</heading>  
  <body>Don't forget me this weekend!</body>  
</note>

Avoid XML Attributes?

Some of the problems with using attributes are:

* attributes cannot contain multiple values (elements can)
* attributes cannot contain tree structures (elements can)
* attributes are not easily expandable (for future changes)

Attributes are difficult to read and maintain. Use elements for data. Use attributes for information that is not relevant to the data.

XML Attributes for Metadata

Sometimes ID references are assigned to elements. These IDs can be used to identify XML elements in much the same way as the id attribute in HTML. This example demonstrates this:

<messages>  
  <note id="501">  
    <to>Tove</to>  
    <from>Jani</from>  
    <heading>Reminder</heading>  
    <body>Don't forget me this weekend!</body>  
  </note>  
  <note id="502">  
    <to>Jani</to>  
    <from>Tove</from>  
    <heading>Re: Reminder</heading>  
    <body>I will not</body>  
  </note>  
</messages>

The id attributes above are for identifying the different notes. It is not a part of the note itself.

What I'm trying to say here is that metadata (data about data) should be stored as attributes, and the data itself should be stored as elements.

**XML Namespaces**

<http://www.w3schools.com/xml/xml_namespaces.asp>

\_\_\_\_\_\_\_\_\_\_\_\_\_

<http://www.tizag.com/xmlTutorial/>

<https://m.youtube.com/watch?v=UqwGSo82cwU>