Week 3a Lecture : More About XML

Screencast of Recorded Lecture (2022):

- **https://sjeccd-edu.zoom.us/rec/play/Eq_fV5LdA8qGaLKZDlBwtUCcvrAZdqPZk7HWPeJltD9YS1zhrk5znqda7QwvOhJVErM6bPDVxBMzKvRV.wf5a_ahGvAKjT5bQ (https://sjeccd-edu.zoom.us/rec/play/Eq_fV5LdA8qGaLKZDlBwtUCcvrAZdqPZk7HWPeJltD9YS1zhrk5znqda7QwvOhJVErM6bPDVxBMzKvRV.wf5a_ahGvAKjT5bQ)**

Screencast of Recorded Lecture (2021):

- **https://sjeccd-edu.zoom.us/rec/play/uOcLnFnkv2w2pZHYaP8rt4zOCFPqtiR9SomT0H93J6kQ6apgQf8DRyZUYRmzGdlAJFd12BSrvu1XxsPL.OMCsUpiLudKzoTjM (https://sjeccd-edu.zoom.us/rec/play/uOcLnFnkv2w2pZHYaP8rt4zOCFPqtiR9SomT0H93J6kQ6apgQf8DRyZUYRmzGdlAJFd12BSrvu1XxsPL.OMCsUpiLudKzoTjM)**

I’ll start off with a review of what we have talked about so far.

What is XML?

- XML stands for eXtensible Markup Language.
- XML is a markup language much like HTML.
- XML was designed to describe data.
- XML tags are not predefined in XML. You must define your own tags.
- XML uses a Document Type Definition (DTD) or an XML Schema to describe data.
- XML with a DTD or XML Schema is designed to be self-descriptive.

The main difference between XML and HTML

- XML was designed to carry data.
- XML is not a replacement for HTML.
- XML and HTML were designed with different goals.
- XML was designed to **describe data** and to focus on what **data is**.

- HTML was designed to **display data** and to focus on how **data looks**.
- HTML is about **displaying information**, while XML is about **describing information**.

XML does not do anything

XML was not designed to do anything.

Maybe it is a little hard to understand, but XML does not do anything. XML is created to structure, store and to send information.

The following example is a note from Jan to Joe, stored as XML:

```
<note>
<to>Joe</to>
<from>Jan</from>
<heading>Reminder</heading>
<body>Don’t forget this weekend</body>
</note>
```

The note has a header and a message body. It also has sender and receiver information. But still, this XML document does not do anything.

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

XML is free and extensible

XML tags are not predefined. You must ‘invent’ your own tags.

The tags used to mark up HTML documents and the structure of HTML documents are predefined.

The author of HTML documents can only use tags that are defined in the HTML standard (like <p>, <h1>, etc.).

XML allows the author to define their own tags and their own document structure.

The tags in the example note above (like <to> and <from>) are not defined in any XML standard. These tags are invented by the author.

XML is a complement to HTML

XML is not a replacement for HTML.

It is important to understand that XML is not a replacement for HTML.

My best description of XML is this: XML is a cross-platform, software and hardware independent tool for describing and transmitting information.

XML can separate data from HTML

With XML, your data is stored outside your HTML.

When HTML is used to display data, the data is stored inside your HTML. With XML, data can be stored in separate XML files. This way you can concentrate on using HTML for data layout and display, and be sure that changes in the underlying data will not require any changes to your HTML.

XML is used to exchange data

With XML, data can be exchanged between incompatible systems.

In the real world, computer systems and databases contain data in incompatible formats. One of the most time-consuming challenges for developers has been to exchange data between such systems over the Internet.

Converting the data to XML can greatly reduce this complexity and create data that can be read by many different types of applications.

XML and B2B

With XML, financial information can be exchanged over the Internet.

XML is one of the main languages for exchanging financial information between businesses over the Internet. A lot of interesting B2B applications are have been developed and are under development.

Because retailers use different order form formats, a retailer's order data can be transformed into XML so that any supplier can recognize and process their orders.

XML can be used to share data

With XML, plain test files can be used to share data.

Since XML data is stored in plain text format, XML provides a software-independent and hardware-independent way of sharing data.

This makes it much easier to create data that different applications can work with. It also makes it easier to expand or upgrade a system to new operating systems, servers, applications, and new browsers.

XML can be used to store data

With XML, plain text files are used to store data.

XML can also be used to store data in files or in databases. Applications can be written to store and retrieve information from the store, and generic applications can be used to display the data.

XML can be used to create new languages

XML is the mother of WAP (Wireless Application Protocol) and WML.

The Wireless Markup Language (WML), used to markup Internet applications for handheld devices like mobile phones, is written in XML.

An example XML document

XML documents use a self-describing and simple syntax.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<to>Joe</to>
<from>Jan</from>
<heading>Reminder</heading>
<body>Don’t forget this weekend</body>
</note>
```

The first line in the document – the XML declaration – defines the XML version and the character encoding used in the document. In this case the document conforms to the 1.0 specification of XML and uses the ISO-8859-1 (Latin-1/West European) character set.

The next line describes the root element of the document (like it was saying: “this document is a note”):

```
<note>
```

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

```
<to>Joe</to>
<from>Jan</from>

<heading>Reminder</heading>

<body>Don’t forget this weekend</body>
```

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

```
</note>
```

You can easily detect from this example that the XML document contains a note from Jan to Joe. Don’t you agree that XML is pretty self-descriptive?

OK, Let’s move on to new material:

All XML elements must have a closing tag

With XML, it is illegal to omit the closing tag.

In XML all elements must have a closing tag, like this:

```
<to>some text here</to>
```

You may have noticed in an above example that the XML declaration did not have a closing tag.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

This is not an error. The declaration is not part of the XML document itself. It is not an XML element, and it should not have a closing tag.

If an HTML tag doesn’t have a closing tag, it is called an empty tag. In XHTML (a combination of both HTML & XML), the way you handle empty tags is to have the tag self-close, or self-terminate. We do this by putting in a space, and then the forward slash symbol. An example is the <br> tag, which is used in HTML to create a line break.

This would be valid HTML, but invalid XML:

```
<br>
```

This would be valid XML, but invalid HTML (there is no </br> tag):

```
<br></br>
```

This is valid XML *and* valid HTML:

```
<br />
```

XML tags are case sensitive

With XML, the tag <body> is different from the tag <Body>

Opening and closing tags must therefore be written with the same case:

```
<from>This is incorrect</From>
<from>This is correct</from>
```

Generally, I prefer to use all lowercase letters, but in XML, it is up to you.

All XML documents must have a root element

All XML documents must contain a single tag pair to define a root element. All other elements must be within this root element.

All elements can have sub elements (child elements). Sub elements must be correctly nested within their parent element, such as:

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

Comments in XML

The syntax for writing comments in XML is:

```
<!--comment text-->
```

`<!--` is the starting delimiter of the comment and `-->` is the ending delimiter of the comment.

XML elements must be properly nested

In XML all elements must be properly nested within each other. Elements may contain (and often do contain) other elements.

However, elements may not overlap. This means that if an element contains a start tag for an element, it must also contain the corresponding end tag.

This is legal XML:

```
<first><second> some text here </second></first>
```

This is not legal XML:

```
<first><second> some text here </first></second>
```

Text Content

Text, which is basically a sequence of characters, may occur between the start tag and end tag of an element; it is said to be the element’s content.

Most English alphabetic or numeric characters can simply be typed as normal. Certain characters must not be used in text content, The following example of an arithmetic expression in XML will generate an error:

```
  <expression> 1 < 2 </expression>
```

An XML processor recognizes the less than sign between 1 and 2 as the starting angle bracket of a new tag. An error results upon finding a space and then a number, which is not allowed to start an XML name.

The following characters must be escaped to use them in text content:

- < (the less than symbol) – must be written as &lt;
- > (the greater than symbol) – must be written an &gt;
- ‘ (the single quotation mark) – must be written as &apos;
- “ (the double quotation mark) – must be written as &quot;
- & (the ampersand) -- must be written as &amp;

CDATA sections

An XML document may contain information expressed in a non-XML syntax. The XML mechanism for indicating that such content is not to be parsed as XML is the CDATA section.

The starting delimiter of a CDATA section is the character sequence `<![CDATA[` and the ending delimiter is the character sequence `]]>`

CDATA sections can be used to store XML code without having to escape all characters that an XML processor would recognize as markup.

Attributes

Elements may optionally have attributes. Each attribute of an element is encoded in the start tag of the element as a name-value pair separated by an equals sign. The attribute value is enclosed in single or double quotes.

Example:

```
<GREETING language="English">
```

Here the GREETING element has a language attribute that has the value English.

Keep in mind that attribute values are just character strings.

That should do it for now.