CSP2103-4102: Markup Languages

Lecture 4: XML Documents

AUSTRALIA ECU EDITH COWAN UNIVERSITY

Introducing XML

- XML stands for Extensible Markup Language. A markup language specifies the structure and content of a document
- Because it is extensible, XML can be used to create a wide variety of document types
- This includes the creation of custom markup languages (i.e user-specific equivalents of HTML)

Introducing XML

- XML is a subset of the Standard Generalized Markup Language (SGML) which was introduced in the 1980s. SGML is very complex and can be costly
- These reasons led to the creation of Hypertext Markup Language (HTML), a more easily used markup language. XML can be seen as sitting between SGML and HTML – easier to learn than SGML, but more robust than HTML

The Limits of HTML

- HTML was designed for formatting text on a Web page. It was not designed for dealing with the content of a Web page. Additional features have been added to HTML, but they do not solve data description or cataloging issues in an HTML document
- Because HTML is not extensible, it cannot be modified to meet specific needs. Browser developers have added features making HTML more robust, but this has resulted in a confusing mix of different HTML standards
- CSS and JavaScript extended functionality somewhat, but were not designed to solve underlying logical structure issues

The Limits of HTML

- HTML cannot be applied consistently. Different browsers require different standards making the final document appear differently on one browser compared with another
- This is because each browser has its own internal HTML parser, or rules for processing HTML tags

The 10 Primary XML Design Goals

1. XML must be easily usable over the Internet
2. XML must support a wide variety of applications
3. XML must be compatible with SGML
4. It must be easy to write programs that process XML documents
5. The number of optional features in XML must be kept to a minimum, ideally zero

The 10 Primary XML Design Goals Continued

6. XML documents should be clear and easily understood by nonprogrammers
7. The XML design should be prepared quickly
8. The design of XML must be exact and concise
9. XML documents must be easy to create
10. Terseness in XML markup is of minimum importance (as they are not primarily meant to be human readable, though it helps)

XML Vocabularies

XML Vocabulary	Description
Channel Definition Format (CDF)	Automatic delivery of information from Web publishers to PCs, PDAs, cell phones, and other information devices
Chemical Markup Language (CML)	Coding of molecular and chemical information
Extensible Hypertext Markup Language (XHTML)	HTML written as an XML application
Mathematical Markup Language (MathML)	Presentation and evaluation of mathematical equations and operations
Musical Markup Language (MML)	Display and organization of music notation and lyrics
Open Financial Exchange (OFX)	Exchange of financial data between financial institutions, businesses, and consumers via the Internet
Real Simple Syndication (RSS)	Distribution of news headlines and syndicated columns
Synchronized Multimedia Integration Language (SMIL)	Editing of interactive audiovisual presentations involving streaming audio, video, text, and any other media type
Voice Markup Language (VoiceXML)	Creation of audio dialogues that feature synthesized speech, digitized audio, and speech recognition

Well-Formed and Valid XML Documents

- An XML document is well-formed if it contains no syntax errors and fulfills all of the specifications for XML code as defined by the W3C
- An XML document is valid if it is well-formed and also satisfies the rules laid out in the DTD or schema attached to the document

The Structure of an XML Document

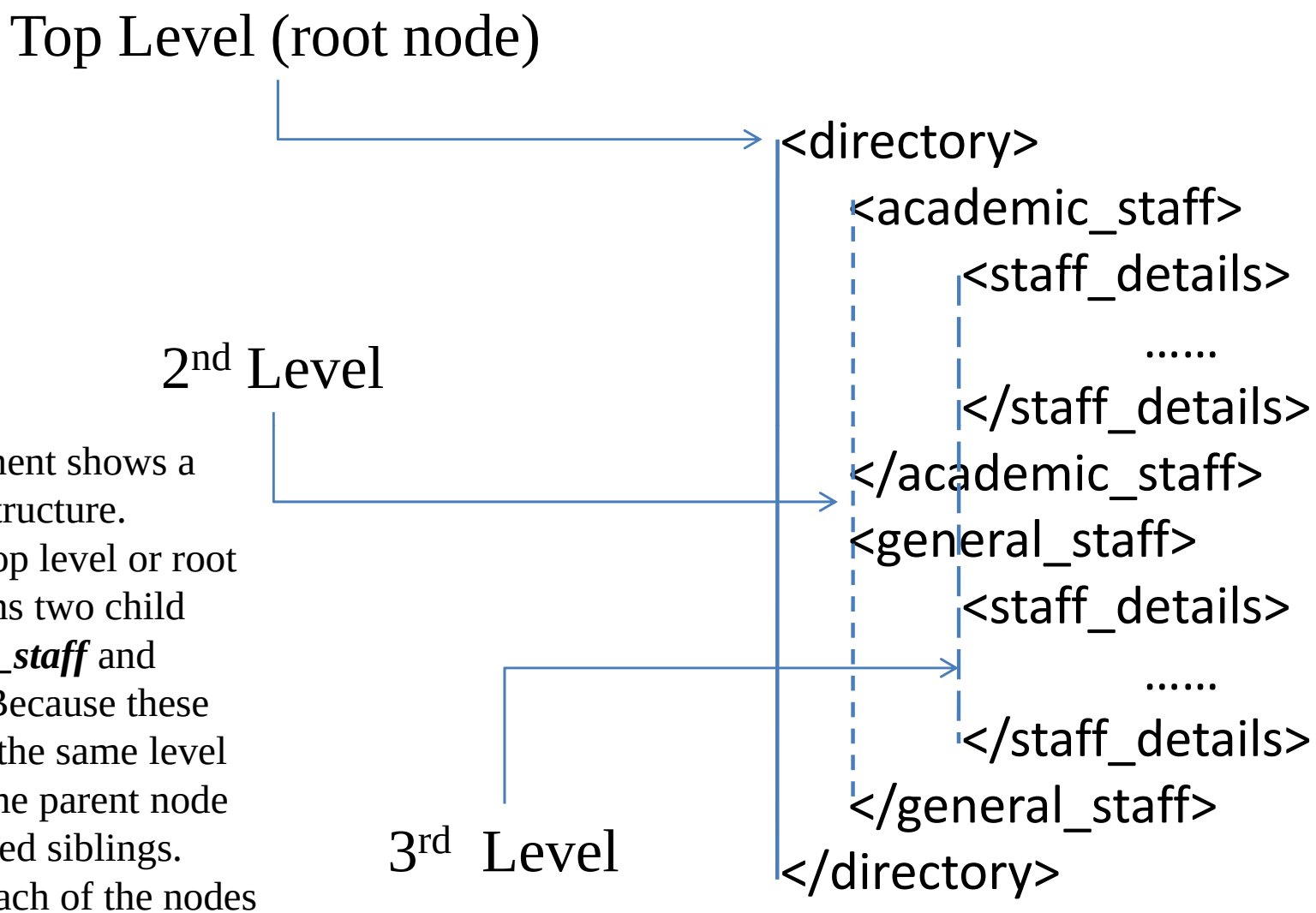
- XML documents consist of three parts
  - The prolog
  - The document body
  - The epilog

- The prolog is optional and provides information about the document itself

The Structure of an XML Document

- The document body contains the document's content in a hierarchical tree structure
- The use of structure describes the logic of the data within the XML document
- i.e – a piece of information may contained a number of related pieces of information, some of which may contain their own (typically each being one level further 'down' the tree
- The epilog is also optional and contains any final comments or processing instructions

XML Document Structure



This basic document shows a three level xml structure. ***directory*** is the top level or root node, and contains two child nodes, ***academic_staff*** and ***general_staff***. Because these two nodes are at the same level and share the same parent node they are considered siblings. ***staff_details*** in each of the nodes are grandchild nodes of ***directory***.

The Structure of an XML Document: Creating the Prolog

- The prolog consists of four parts in the following order:
  - XML declaration
  - Miscellaneous statements or comments
  - Processing instructions
  - Document type declaration

The Structure of an XML Document: The XML Declaration

- The XML declaration is always the first line of code in an XML document. It tells the processor what follows is written using XML. It can also provide any information about how the parser should interpret the code.
- The complete syntax is:

  <?xml version="*version number*" encoding="*encoding type*" standalone="*yes | no*" ?>

- A sample declaration might look like this:

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

The Structure of an XML Document: Inserting Comments

- Comments or miscellaneous statements go after the declaration. Comments may appear anywhere after the declaration
- The syntax for comments is:

<!- - *comment text* - ->

- This is the same syntax for HTML comments

Elements

- Elements are the basic building blocks of XML files
- Elements contain an opening tag and a closing tag
  - Content is stored between tags

```
<first_name>John</first_name>
<surname>Bloggs</surname>
```

Elements

- A closed element, has the following syntax:

  <element_name>Content</element_name>

- Example:

  <Artist>Miles Davis</Artist>

Element

- Element names are case sensitive
- Elements can be nested, as follows:

```
<tracks>Kind of Blue
    <track>So What ((:22)</track>
    <track>Blue in Green (5:37)</track>
</tracks>
```

Elements

- Nested elements are called child elements
- Elements must be nested correctly
- Child elements must be enclosed within their parent elements
- In the previous slide, each <track> element is a child of <tracks>
- Elements at the same level (i.e <track> as child of <tracks>) are called siblings

Elements and Attributes

- All elements must be nested within a single **document** or **root element**. There can be only **one** root element

- An **open** or **empty** element is an element that contains no content. They can be used to mark sections of the document for the XML parser

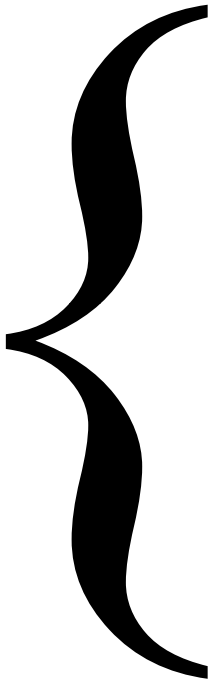
Working with Attributes

- An **attribute** is a feature or characteristic of an element. Attributes are text strings and must be placed in single or double quotes. The syntax is:

<element_name attribute="value"> …
</element_name>

Elements and Attributes

document elements and attributes



```
<item>
   <title>Kind of Blue</title>
   <artist>Miles Davis</artist>
   <tracks>
      <track length="9:22">So What</track>
      <track length="9:46">Freddie Freeloader</track>
      <track length="5:37">Blue in Green</track>
      <track length="11:33">All Blues</track>
      <track length="9:26">Flamenco Sketches</track>
   </tracks>
</item>
<item>
   <title>Cookin'</title>
   <artist>Miles Davis</artist>
   <tracks>
      <track length="5:57">My Funny Valentine</track>
      <track length="9:53">Blues by Five</track>
      <track length="4:22">Airegin</track>
      <track length="13:03">Tune-Up</track>
   </tracks>
</item>
<item>
   <title>Blue Train</title>
   <artist>John Coltrane</artist>
   <tracks>
      <track length="10:39">Blue Train</track>
      <track length="9:06">Moment's Notice</track>
      <track length="7:11">Locomotion</track>
      <track length="7:55">I'm Old Fashioned</track>
      <track length="7:03">Lazy Bird</track>
   </tracks>
</item>
```

Element and Attribute Use

- There is no set rule that requires attributes to be used, however they are excellent for 'adding on' some extra info onto an element
- Some developers use base elements and then add many attributes
- For example, instead of : <track length="9:46">Freddie Freeloader</track>
- You could have :
  <track length="9:46" title="Freddie Freeloader" />

Character References

- Special characters, such as the symbol for the British pound, can be inserted into your XML document by using a **character reference**. The syntax is:

  &#*nnn*;

- *Character* is a character reference number or name from the ISO/IEC character set
- Character references in XML are the same as in HTML

Character References

This figure shows commonly used character reference numbers

Symbol	Character Reference	Entity Reference	Description
©	&#169;		Copyright symbol
®	&#174;		Registered trademark symbol
™	&#153;		Trademark symbol
<	&#60;	&lt;	Less than symbol
>	&#62;	&gt;	Greater than symbol
&	&#38;	&amp;	Ampersand
"	&#22;	&quot;	Double quote
'	&#27;	&apos;	Apostrophe (single quote)
£	&#163;		Pound sign
€	&#128;		Euro sign
¥	&#165;		Yen sign

Character References

character reference

```
<item>
   <title>Kind of Blue</title>
   <priceus>US: $11.99</priceus>
   <priceuk>UK: &#163;8.39</priceuk>
   <artist>Miles Davis</artist>
   <tracks>
      <track length="9:22">So What</track>
      <track length="9:46">Freddie Freeloader</track>
      <track length="5:37">Blue in Green</track>
      <track length="11:33">All Blues</track>
      <track length="9:26">Flamenco Sketches</track>
   </tracks>
</item>
<item>
   <title>Cookin'</title>
   <priceus>US: $7.99</priceus>
   <priceuk>UK: &#163;5.59</priceuk>
   <artist>Miles Davis</artist>
   <tracks>
      <track length="5:57">My Funny Valentine</track>
      <track length="9:53">Blues by Five</track>
      <track length="4:22">Airegin</track>
      <track length="13:03">Tune-Up</track>
   </tracks>
</item>
<item>
   <title>Blue Train</title>
   <priceus>US: $8.99</priceus>
   <priceuk>UK: &#163;6.29</priceuk>
   <artist>John Coltrane</artist>
   <tracks>
      <track length="10:39">Blue Train</track>
      <track length="9:06">Moment's Notice</track>
      <track length="7:11">Locomotion</track>
      <track length="7:55">I'm Old Fashioned</track>
      <track length="7:03">Lazy Bird</track>
   </tracks>
</item>
```

Parsed Character Data

- **Parsed character data**, or **pcdata** consists of all those characters that XML treats as parts of the code of XML document
  - The XML declaration
  - The opening and closing tags of an element
  - Empty element tags
  - Character or entity references
  - Comments

CDATA Sections

- A CDATA section is a large block of text the XML processor will interpret only as text.

- The syntax to create a CDATA section is:

```
<! [CDATA [
    Text Block
] ]>
```

CDATA Sections

- In this example, a CDATA section stores several HTML tags within an element named HTMLCODE:

```
<htmlcode>
  <![CDATA[
    <h1>The Jazz Warehouse</h1>
    <h2>Your Online Store for Jazz Music</h2>
  ] ]>
</htmlcode>
```

CDATA Sections

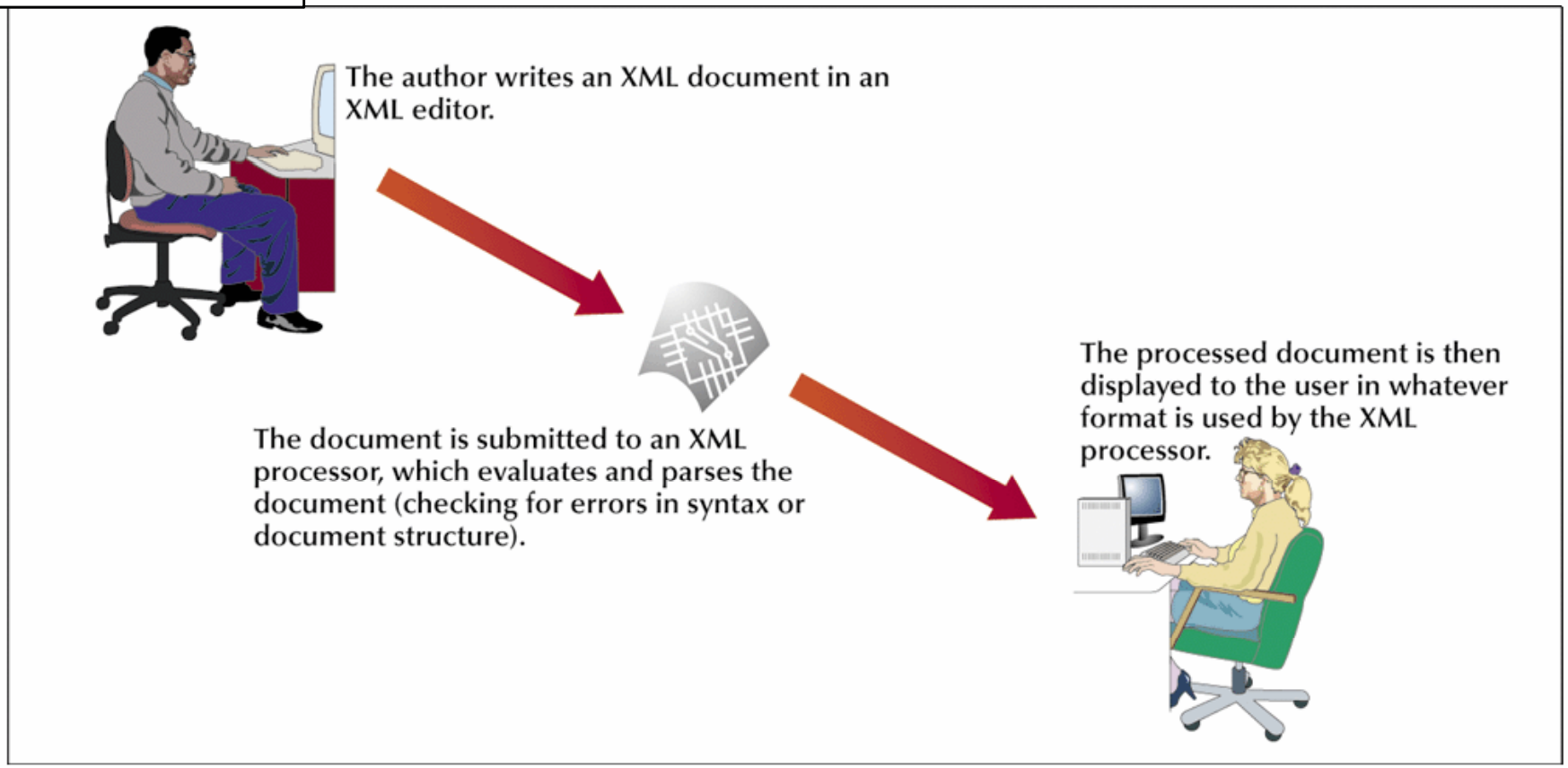
CDATA section →

```
<items>
    <message>
    <![CDATA[
        Here are some of the latest specials from the Jazz Warehouse.
        Please note that all Miles Davis & John Coltrane CDs will be
        on sale for the month of March.
    ]]>
    </message>
    <item>
        <title>Kind of Blue</title>
        <priceus>US: $11.99</price>
        <priceuk>UK: &#163;8.39</priceuk>
        <artist>Miles Davis</artist>
        <tracks>
            <track length="9:22">So What</track>
            <track length="9:46">Freddie Freeloader</track>
            <track length="5:37">Blue in Green</track>
            <track length="11:33">All Blues</track>
            <track length="9:26">Flamenco Sketches</track>
        </tracks>
    </item>
```

Parsing an XML Document

From the Textbook

Parsing an XML document



Displaying an XML Document in a Web Browser

- XML documents can be opened in Internet Explorer or in Mozilla Firefox (vers 2 onwards)
- If there are no syntax errors. IE will display the document's contents in an expandable/collapsible outline format including all markup tags
- In the end, this just shows the XML document and content, but nothing else
- To make the content of the XML document visually interesting, XSLT is used to transform it to HTML (or other document formats)

Displaying an XML Document in a Web Browser

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<!--  This document contains data on Jazz Warehouse special offers  -->
- <items>
  - <message>
    - <![CDATA[
                Here are some of the latest specials from the Jazz Warehouse.
                Please note that all Miles Davis & John Coltrane CDs will be
                on sale for the month of March.
      ]]>
    </message>
  - <item>
      <title>Kind of Blue</title>
      <priceus>US: $11.99</priceus>
      <priceuk>UK: £8.39</priceuk>
      <artist>Miles Davis</artist>
    - <tracks>
        <track length="9:22">So What</track>
        <track length="9:46">Freddie Freeloader</track>
        <track length="5:37">Blue in Green</track>
        <track length="11:33">All Blues</track>
        <track length="9:26">Flamenco Sketches</track>
      </tracks>
    </item>
  - <item>
      <title>Cookin'</title>
      <priceus>US: $7.99</priceus>
      <priceuk>UK: £5.59</priceuk>
      <artist>Miles Davis</artist>
    - <tracks>
        <track length="5:57">My Funny Valentine</track>
        <track length="9:53">Blues by Five</track>
        <track length="4:22">Airegin</track>
        <track length="13:03">Tune-Up</track>
      </tracks>
    </item>
  - <item>
      <title>Blue Train</title>
      <priceus>US: $8.99</priceus>
      <priceuk>UK: £6.29</priceuk>
      <artist>John Coltrane</artist>
    - <tracks>
        <track length="10:39">Blue Train</track>
        <track length="9:06">Moment's Notice</track>
        <track length="7:11">Locomotion</track>
        <track length="7:55">I'm Old Fashioned</track>
        <track length="7:03">Lazy Bird</track>
      </tracks>
    </item>
</items>
```

Linking to a Style Sheet

- Link the XML document to a style sheet to format the document. The XML processor will combine the style sheet with the XML document and apply any formatting codes defined in the style sheet to display a formatted document
- There are two main style sheet languages used with XML:
  - Cascading Style Sheets (CSS) and Extensible Style Sheets (XSL)

Linking to a Style Sheet

- There are some important benefits to using style sheets:
  - By separating content from format, you can concentrate on the appearance of the document
  - Different style sheets can be applied to the same XML document
  - Any style sheet changes will be automatically reflected in any Web page based upon the style sheet

Applying a Style to an Element

- To apply a style sheet to a document, use the following syntax:

selector {attribute1:value1; attribute2:value2; …}

- selector is an element (or set of elements) from the XML document.
- *attribute* and *value* are the style attributes and attribute values to be applied to the document

Applying a Style to an Element

- For example:

```
artist {color:red; font-weight:bold}
```

- will display the text of the artist element in a red boldface type

Creating Processing Instructions

- The link from the XML document to a style sheet is created using a processing statement

- A **processing instruction** is a command that gives instructions to the XML parser

Creating Processing Instructions

- For example:

<?xml-stylesheet type="*style*" href="*sheet*" ?>

- *Style* is the type of style sheet to access and *sheet* is the name and location of the style sheet

Style Sheet Example

```
message           {display:block; width:400px; color:blue; text-align: center;
                   font-size: 10pt; font-family: Arial, Helvetica, sans-serif;
                   border: 3px solid blue; background-color: ivory;
                   margin: 10px; padding: 15px}

item              {display:block; font-size:14pt; color:red;
                   font-family: Arial, Helvetica, Sans-serif;
                   margin: 20px}

title             {display:block; font-size: 16pt; color:blue;
                   font-weight:bold;
                   font-family: Arial, Helvetica, sans-serif}

priceus, priceuk  {color:black; font-size: 12pt; font-weight: bold;
                   font-family: Times New Roman, Times, Serif;
                   margin-left: 20px;}

artist            {display:block; font-size: 12pt; color:black;
                   font-style:italic; font-weight: bold;
                   font-family: Times New Roman, Times, Serif;
                   margin-left: 20px}

track             {display:list-item; font-size: 9pt; color: black;
                   list-style-type: circle;
                   font-family: Arial, Helvetica, sans-serif;
                   margin-left: 35px}
```

Linking to a Style Sheet

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<!-- This document contains data on Jazz Warehouse special offers -->
<?xml-stylesheet type="text/css" href="jw.css" ?>

<items>
    <message>
    <![CDATA[
        Here are some of the latest specials from the Jazz Warehouse.
        Please note that all Miles Davis & John Coltrane CDs will be
        on sale for the month of March.
    ]]>
    </message>
```

processing instruction to access the a style sheet

An XML Document Formatted with a Style Sheet

- In this example, an XML document has been visually enhanced using a CSS style sheet
- While a CSS sheet can quickly and easily add some look and feel to raw XML data, it has no intrinsic processing capability
- Extensible Style Language (XSL) and Extensible Style Language for Transformations (XSLT) allows for both visual transformation of XML data and conditional processing based on the content of the XML elements

Conclusion

- As you can see, XML has few rules outside of those dictating structure and order
- XML users are essentially defining their own ‘language’ (assuming the data is to be used rather than just stored)
- Document Type Declarations provide the rules for the language
- XSLT provides the processing instructions (or DOM) for the language