Module 6

XSL and XSLT

For Aptech Center Use Only

Module Overview

In this module, you will learn about:

- Introduction to XSL
- Working with XSL

For Aptech Center Use Only

Lesson 1 – Introduction to XSL

In this first lesson, **Introduction to XSL**, you will learn to:

- Define Extensible Stylesheet Language (XSL), Extensible Stylesheet Language Transformations (XSLT) and their purpose.
- Explain the structure and syntax of XSL.
- Distinguish between Cascading Style Sheet (CSS) and XSL.

For Aptech Center Use Only

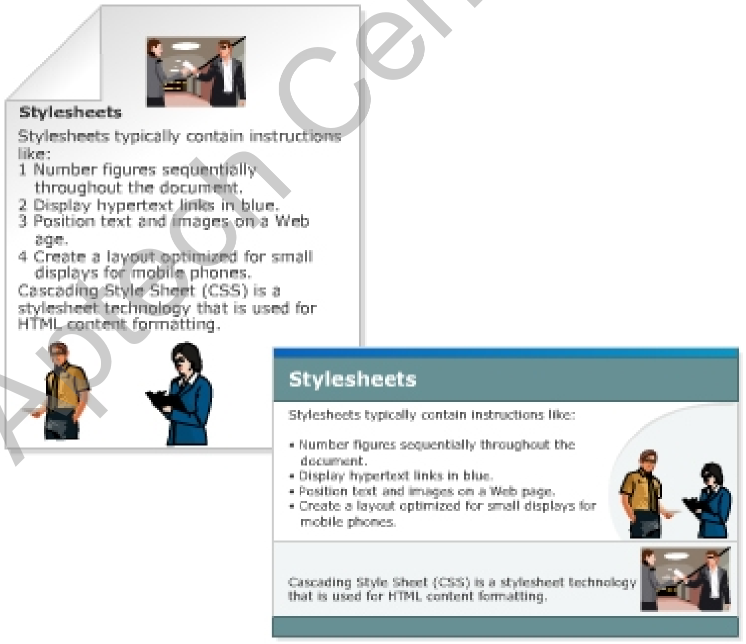
Stylesheets 1-2

- It is a collection of commands that tells a processor how to render the visual appearance of content in a web page.
- Stylesheets typically contain instructions like:
  - Number figures sequentially throughout the document.
  - Display hypertext links in blue.
  - Position text and images on a Web page.
  - Create a layout optimized for small displays for mobile phones.

For Aptech Center Use Only

Stylesheets 2-2

- CSS is used for HTML content formatting.
- XSL is used to describe how the XML document should be displayed.



Stylesheets

Stylesheets typically contain instructions like:

1 Number figures sequentially throughout the document.
2 Display hypertext links in blue.
3 Position text and images on a Web age.
4 Create a layout optimized for small displays for mobile phones.

Cascading Style Sheet (CSS) is a stylesheet technology that is used for HTML content formatting.

Stylesheets

Stylesheets typically contain instructions like:

- Number figures sequentially throughout the document.
- Display hypertext links in blue.
- Position text and images on a Web page.
- Create a layout optimized for small displays for mobile phones.

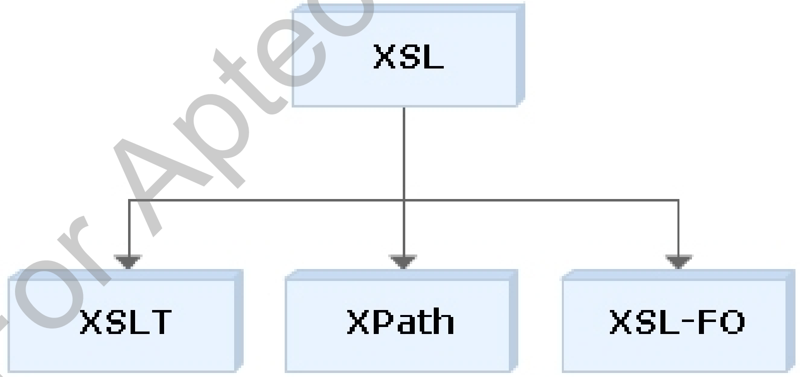
Cascading Style Sheet (CSS) is a stylesheet technology that is used for HTML content formatting.

For Aptech Center Use Only

Extensible Stylesheet Language (XSL)

- XSL Transformations (XSLT)
  - An XML language for transforming XML documents.
- XML Path Language (XPath)
  - A language for navigating the XML document.
- XSL Formatting Objects (XSL-FO)
  - An XML language for formatting XML documents.

For Aptech Center Use Only



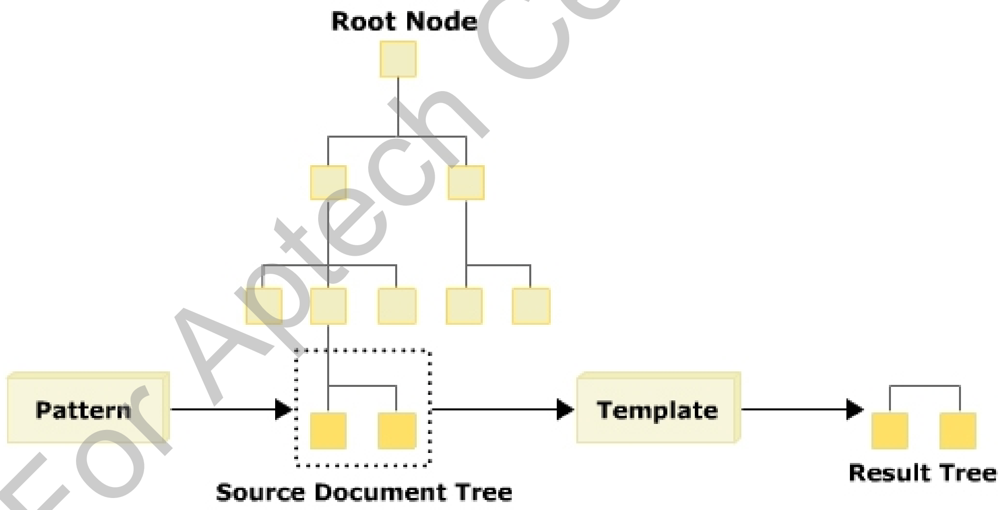
XSL Transformations

- The transformation component of the XSL stylesheet technology is XSLT.
- It describes the process of transforming an XML document, using a transformation engine and XSL.

For Aptech Center Use Only

XSL Processing Model

- It reads an XML document and processes it into a hierarchical tree.
- It starts with the root node in the tree and performs pattern matching in the stylesheet.



For Aptech Center Use Only

XSLT Structure and Syntax

- It uses a standard document introduction, matching closing tags for any opening tags that contain content, and a proper syntax for empty elements.
- The style rules are written in a file with the extension `.xsl`.

Syntax

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
………
………
</xsl:styleheet>
```

`where,`

`<xsl:stylesheet>:` Root element of the stylesheet.

`xmlns:xsl="http://www.w3.org/1999/XSL/Transform":` refers to the official W3C XSLT namespace. You must include the attribute version="1.0" if you use this namespace.

For Aptech Center Use Only

Top Level XSLT Elements

- An element occurring as a child of an `xsl:stylesheet` element is called a top-level element.
- It can occur directly inside the `xsl:stylesheet` element.

Element Name	Description
`xsl:attribute-set`	Adds a list of attributes to the output node tree
`xsl:import`	Used to import contents of one stylesheet into another. The importing stylesheet takes precedence over the imported stylesheet
`xsl:namespace-alias`	Replaces source document Namespace with a new Namespace in the output tree node
`xsl:output`	Specifies the output for the result tree. It contains a list of attributes. The most important one is the `method` attribute which dictates if the type of output is HTML, text, or XML
`xsl:template`	Used to define a template that can be applied to a node to produce a desired output display
`xsl:variable`	Defines a `variable` in a stylesheet or template, and to assign it a value

For Aptech Center Use Only

CSS and XSL

- They are two different style languages recommended by the World Wide Web Consortium (W3C).
- XSL is more powerful and complex than CSS.

CSS	XSL
Stylesheet language to create a style for HTML and XML documents	Stylesheet language to create a style for XML documents
Determines the visual appearance of a page, but does not alter the structure of the source document	Provides a means of transforming XML documents
Does not support decision structures and it cannot calculate quantities or store values in variables	Supports decision structures and can calculate quantities or store values in variables
Uses its own notation	Uses an XML notation
Highly effective and easy to learn for simple applications	Designed to meet the needs of more complex applications for richer style sheets

For Aptech Center Use Only

Lesson 2 – Working with XSL

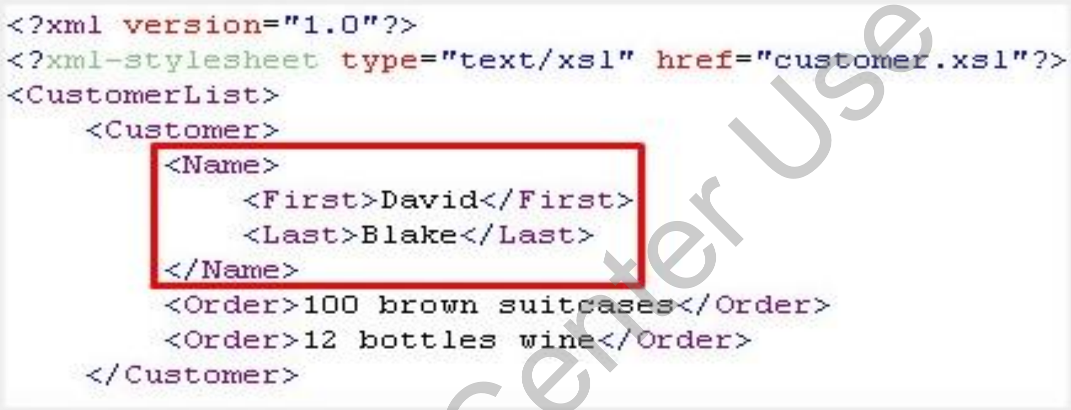
In this last lesson, **Working with XSL**, you will learn to:

- Explain XSL templates.
- Describe the use of `select` attribute.
- State how to use `xsl:value-of` element.
- Describe how to use `xsl:for-each` element.
- Explain briefly how to use `xsl:text` element.
- Describe how to use `xsl:number` element.
- Describe how to use `xsl:if` element.
- Describe how to use `xsl:choose` element.
- Explain how to perform sorting using XSL.

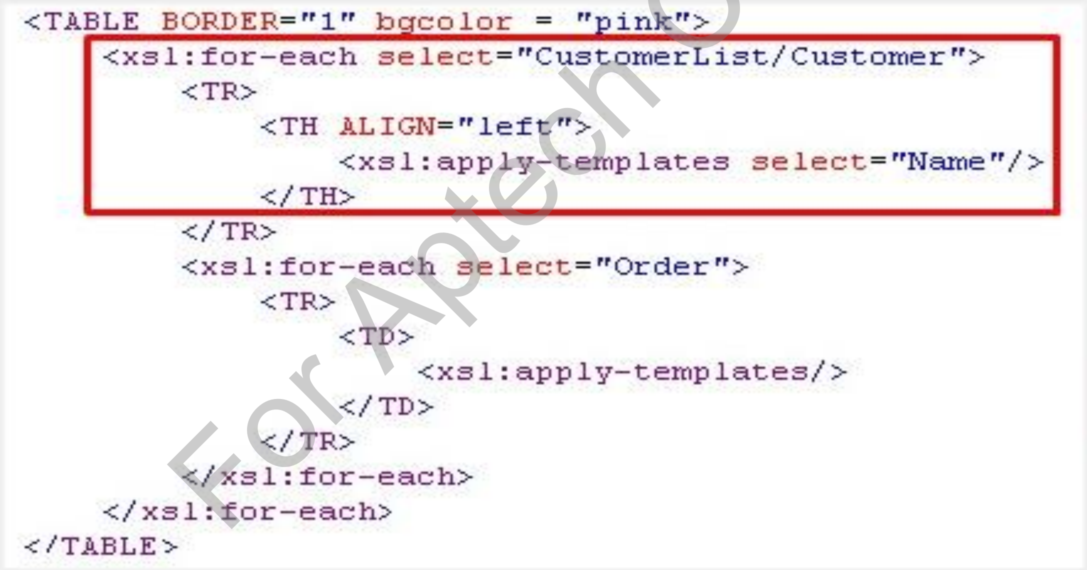
For Aptech Center Use Only

XSL Templates

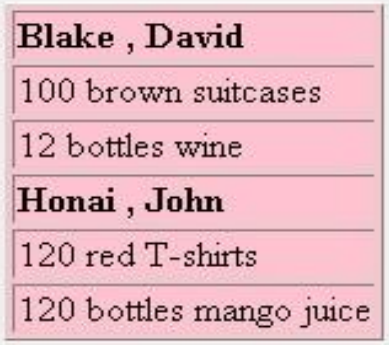
For Aptech Center Use Only



```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="customer.xsl"?>
<CustomerList>
    <Customer>
        <Name>
            <First>David</First>
            <Last>Blake</Last>
        </Name>
        <Order>100 brown suitcases</Order>
        <Order>12 bottles wine</Order>
    </Customer>
```



```
<TABLE BORDER="1" bgcolor = "pink">
    <xsl:for-each select="CustomerList/Customer">
        <TR>
            <TH ALIGN="left">
                <xsl:apply-templates select="Name"/>
            </TH>
        </TR>
        <xsl:for-each select="Order">
            <TR>
                <TD>
                    <xsl:apply-templates/>
                </TD>
            </TR>
        </xsl:for-each>
    </xsl:for-each>
</TABLE>
```



Blake , David
100 brown suitcases
12 bottles wine
Honai , John
120 red T-shirts
120 bottles mango juice

The xsl:template Element 1-2

- It is used to define a template that can be applied to a node to produce desired output.

Syntax

```
<xsl:template
   match=”pattern”
   mode=”mode
   name=”name”
   priority=”number”
>
</xsl:template>
```

where,

`match`: Is a pattern that is used to define which nodes will have which template rules applied to them. If this attribute is omitted there must be a `name` attribute.

`mode`: Allows the same nodes to be processed more than once.

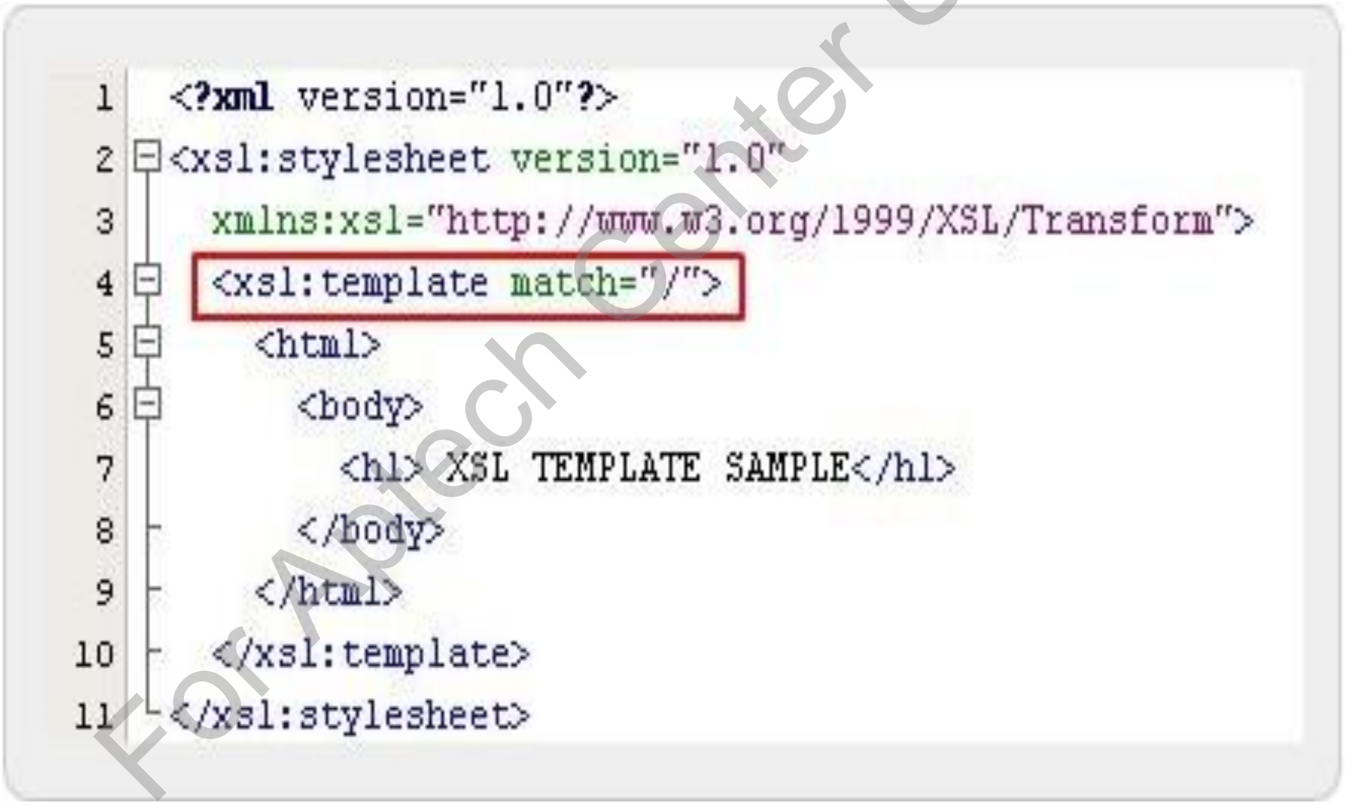
`name`: Specifies a name for the template. If this attribute is omitted there must be a `match` attribute.

`priority`: Is a real number that sets the priority of importance for a template. The higher the number, the higher the priority.

For Aptech Center Use Only

The xsl:template Element 2-2

Code Snippet



```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <h1> XSL TEMPLATE SAMPLE</h1>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

For Aptech Center Use Only

The xsl:apply-templates element 1-4

- It defines a set of nodes to be processed.

Syntax

```
<xsl:apply-templates
  select="expression"
  mode="name"
>
</xsl:apply-templates>
```

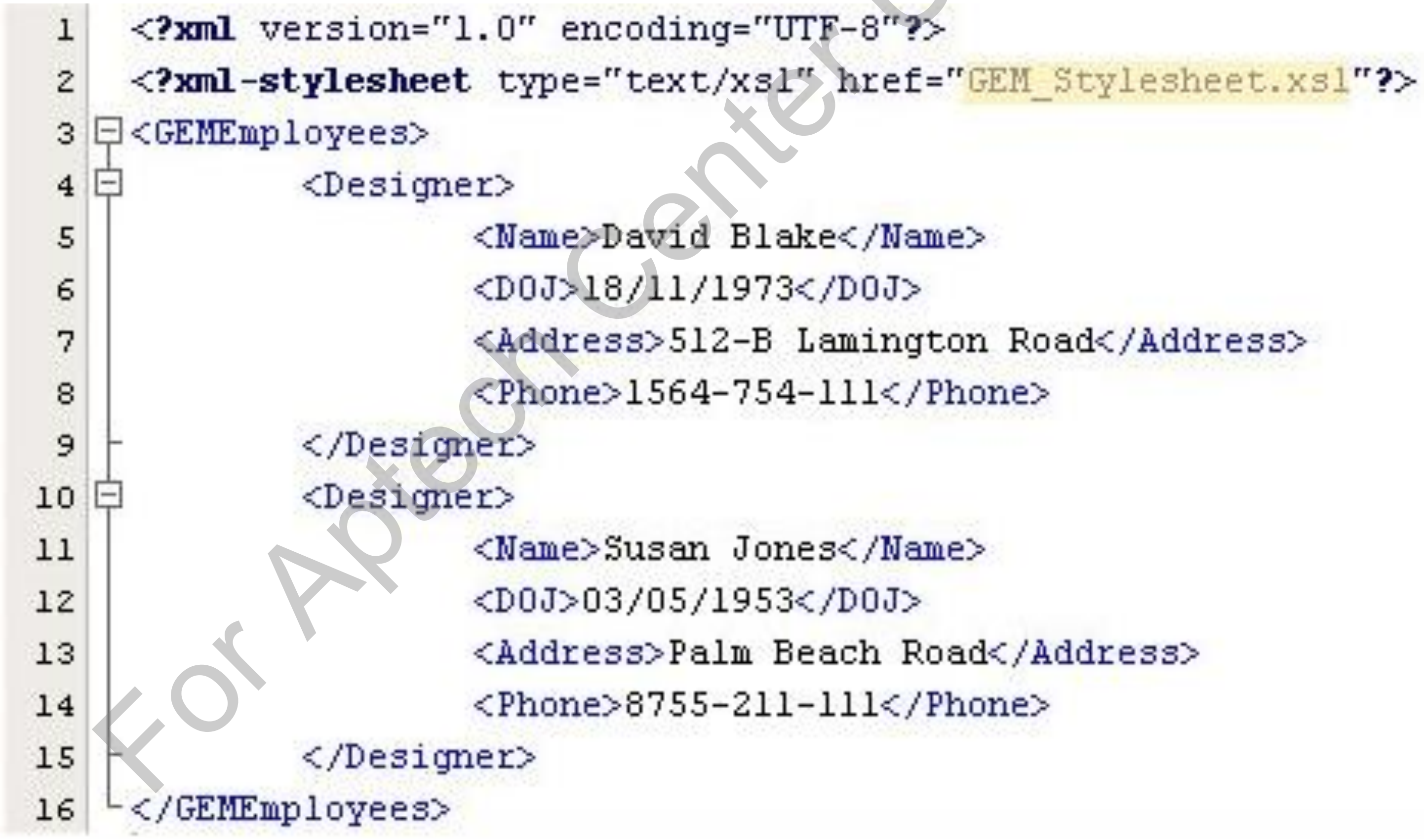
where,
  `select`
  Used to process nodes selected by an expression.
  `mode`
  Allows the same nodes to be processed more than once. Each time the nodes are processed, they can be displayed in a different manner.

For Aptech Center Use Only

The xsl:apply-templates element 2-4

For Aptech Center Use Only

Code Snippet

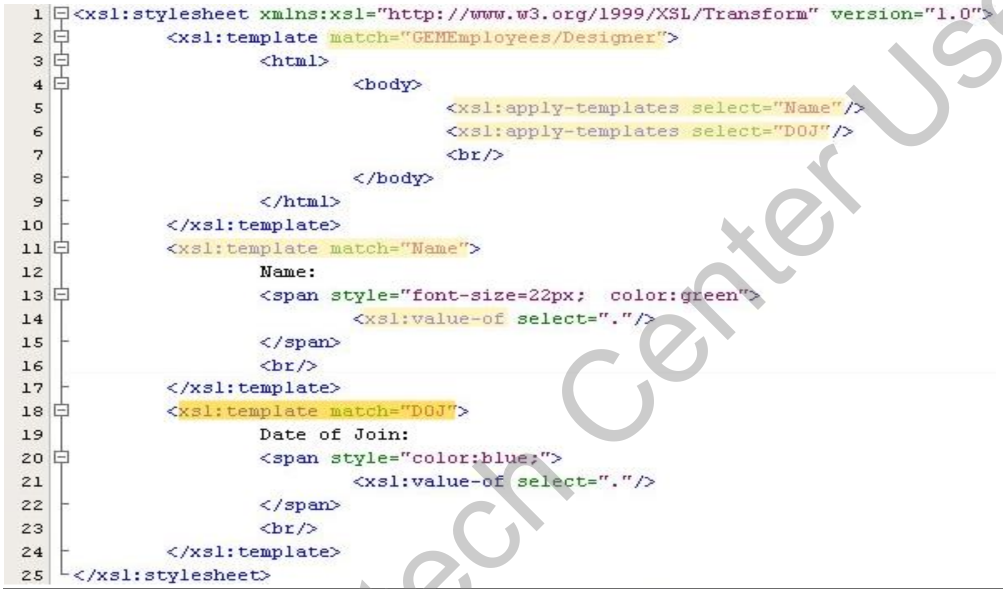


```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="GEM_Stylesheet.xsl"?>
<GEMEmployees>
        <Designer>
                <Name>David Blake</Name>
                <DOJ>18/11/1973</DOJ>
                <Address>512-B Lamington Road</Address>
                <Phone>1564-754-111</Phone>
        </Designer>
        <Designer>
                <Name>Susan Jones</Name>
                <DOJ>03/05/1953</DOJ>
                <Address>Palm Beach Road</Address>
                <Phone>8755-211-111</Phone>
        </Designer>
</GEMEmployees>
```

The xsl:apply-templates element 3-4

For Aptech Center Use Only

Style Sheet



```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
        <xsl:template match="GEMEmployees/Designer">
                <html>
                        <body>
                                <xsl:apply-templates select="Name"/>
                                <xsl:apply-templates select="DOJ"/>
                                <br/>
                        </body>
                </html>
        </xsl:template>
        <xsl:template match="Name">
                Name:
                <span style="font-size=22px;  color:green">
                        <xsl:value-of select="."/>
                </span>
                <br/>
        </xsl:template>
        <xsl:template match="DOJ">
                Date of Join:
                <span style="color:blue;">
                        <xsl:value-of select="."/>
                </span>
                <br/>
        </xsl:template>
</xsl:stylesheet>
```

where,

`match="GEMEmployees/Designer"`
Represents the `Designer` child element by specifying the `GEMEmployees` parent element.

`xsl:apply-templates select="Name"`
Applies the template on Name element.

`xsl:apply-templates select="DOJ"`
Applies the template on DOJ element.

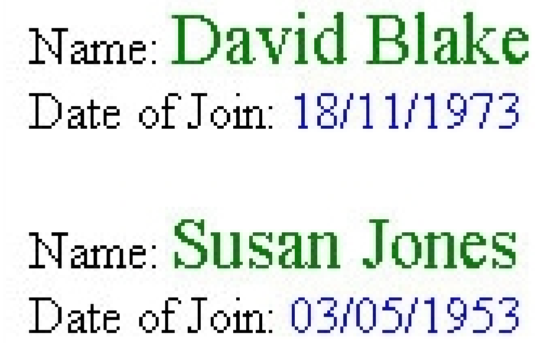
`xsl:value-of`
Used to extract the value of a selected node.

`xsl:template match="Name"`
If the template is matched with Name element, the value of the Name element is displayed in green color with font size 22 pixels.

`xsl:template match="DOJ"`
If the template is matched with DOJ element, the value of the DOJ element is displayed in blue color.

The xsl:apply-templates element 4-4

Output



```
Name: David Blake
Date of Join: 18/11/1973

Name: Susan Jones
Date of Join: 03/05/1953
```

For Aptech Center Use Only

The select attribute 1-4

- It can be used to process nodes selected by an expression instead of processing all children.

Syntax

```
<xsl:template match = "element">
   <xsl:apply-templates select ="name of the element"/>
</xsl:template>
```

where,

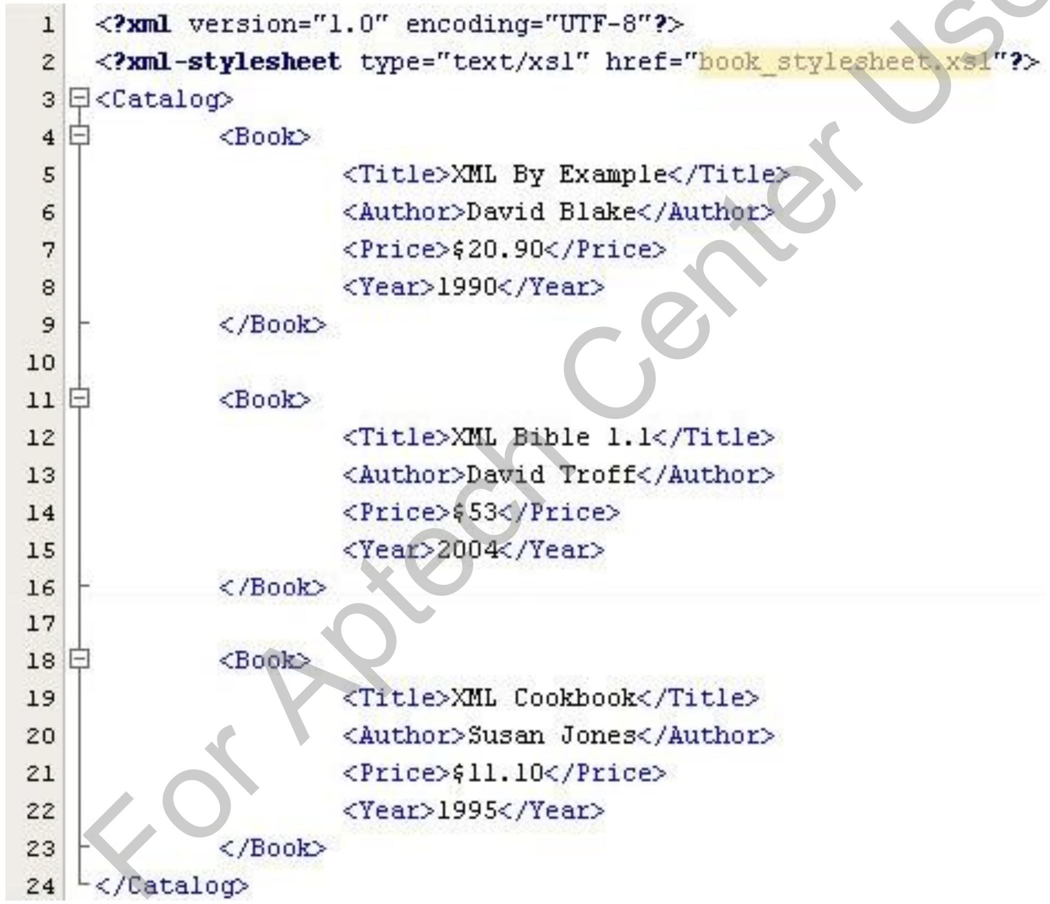
`select`

Uses the same kind of patterns as the `match` attribute of the `xsl:template` element. If `select` attribute is not present, all child element, comment, text, and processing instruction nodes are selected.

For Aptech Center Use Only

The select attribute 2-4

Code Snippet

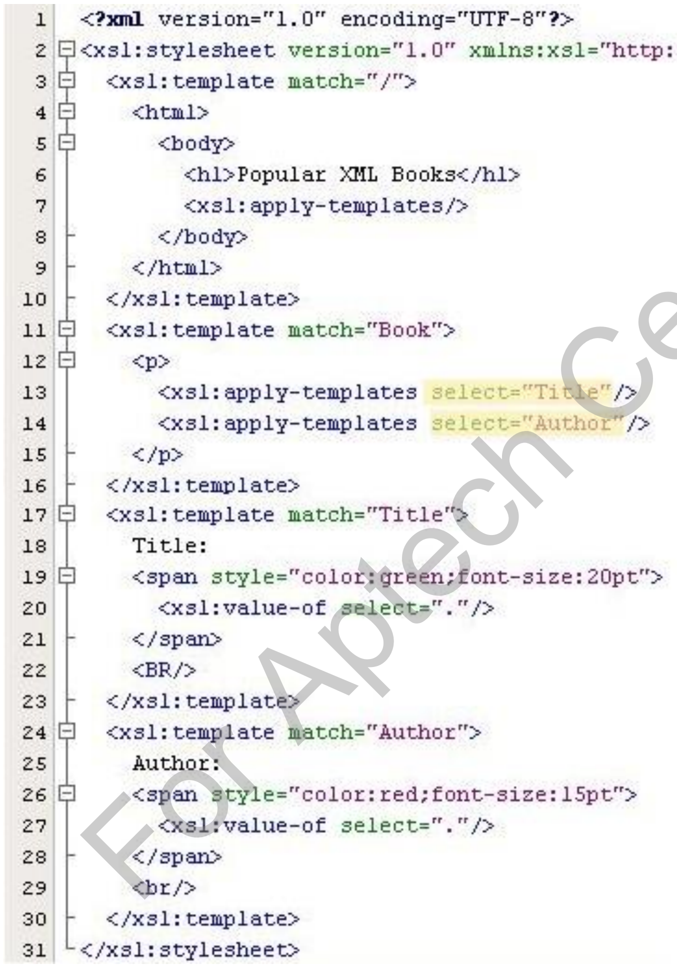


```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="book_stylesheet.xsl"?>
<Catalog>
        <Book>
                <Title>XML By Example</Title>
                <Author>David Blake</Author>
                <Price>$20.90</Price>
                <Year>1990</Year>
        </Book>

        <Book>
                <Title>XML Bible 1.1</Title>
                <Author>David Troff</Author>
                <Price>$53</Price>
                <Year>2004</Year>
        </Book>

        <Book>
                <Title>XML Cookbook</Title>
                <Author>Susan Jones</Author>
                <Price>$11.10</Price>
                <Year>1995</Year>
        </Book>
</Catalog>
```

For Aptech Center Use Only

The select attribute 3-4

Style Sheet



```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <h1>Popular XML Books</h1>
        <xsl:apply-templates/>
      </body>
    </html>
  </xsl:template>
  <xsl:template match="Book">
    <p>
      <xsl:apply-templates select="Title"/>
      <xsl:apply-templates select="Author"/>
    </p>
  </xsl:template>
  <xsl:template match="Title">
    Title:
    <span style="color:green;font-size:20pt">
      <xsl:value-of select="."/>
    </span>
    <BR/>
  </xsl:template>
  <xsl:template match="Author">
    Author:
    <span style="color:red;font-size:15pt">
      <xsl:value-of select="."/>
    </span>
    <br/>
  </xsl:template>
</xsl:stylesheet>
```

where,

`select="Title"`

Applies the template on `Title` element.

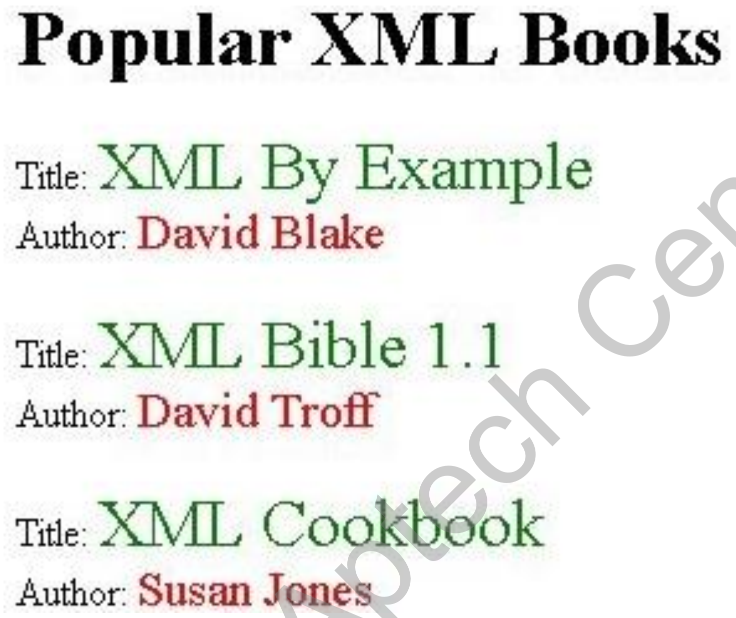
`select="Author"`

Applies the template on `Author` element.

For Aptech Center Use Only

The select attribute 4-4

Output



Popular XML Books

Title: XML By Example
Author: David Blake

Title: XML Bible 1.1
Author: David Troff

Title: XML Cookbook
Author: Susan Jones

For Aptech Center Use Only

The xsl:value-of element 1-4

- It is used to write or display text string representing the value of the element specified by the `select` attribute.

Syntax

```
<xsl:value-of
  select="expression"
  disable-output-escaping="yes" | "no"
/>
```

where,

`select`

A mandatory attribute that assigns the node (by name) to the element.

`disable-output-escaping`

Specifies how special characters should appear in the output string.

`yes`

Indicates that special characters should be displayed as is (for example, a < or >).

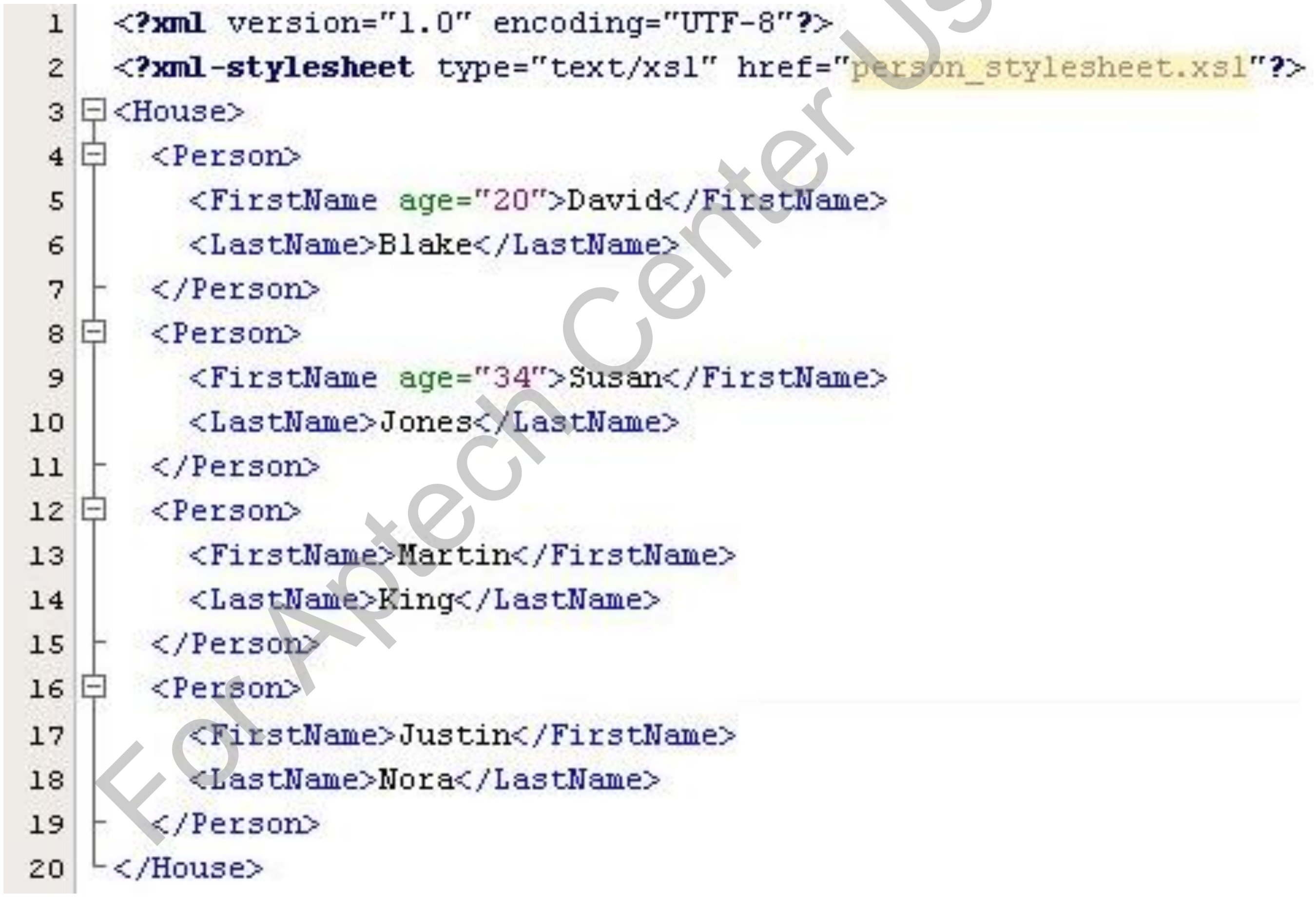
`no`

Indicates that special characters should not be displayed as is (for example, a > is displayed as &gt;).

For Aptech Center Use Only

The xsl:value-of element 2-4

Code Snippet

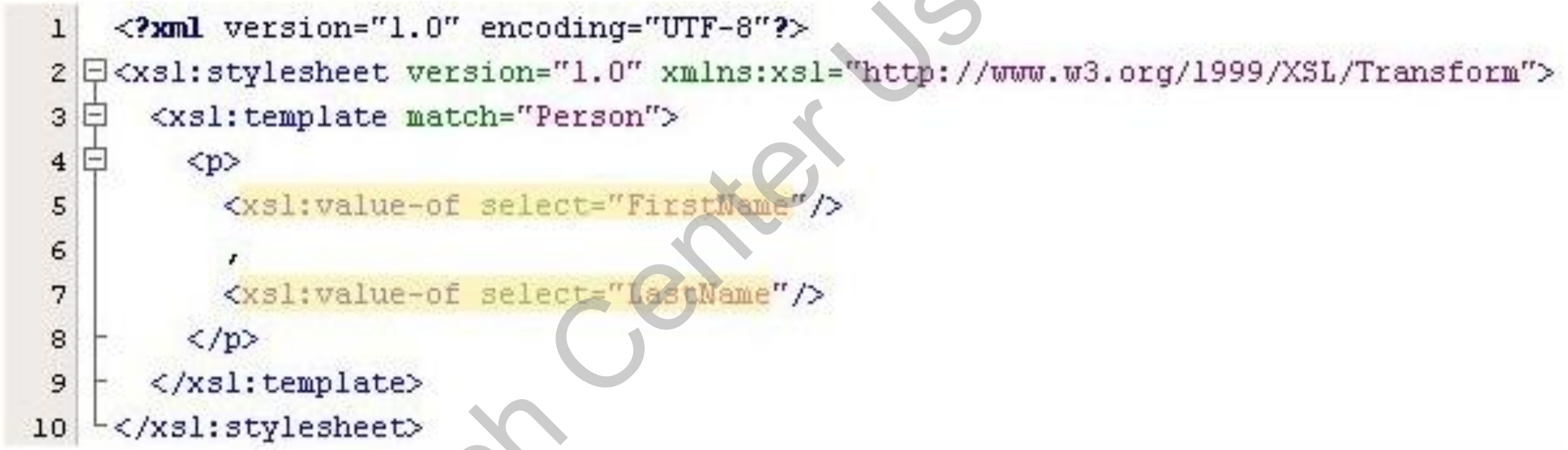


```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="person_stylesheet.xsl"?>
<House>
  <Person>
    <FirstName age="20">David</FirstName>
    <LastName>Blake</LastName>
  </Person>
  <Person>
    <FirstName age="34">Susan</FirstName>
    <LastName>Jones</LastName>
  </Person>
  <Person>
    <FirstName>Martin</FirstName>
    <LastName>King</LastName>
  </Person>
  <Person>
    <FirstName>Justin</FirstName>
    <LastName>Nora</LastName>
  </Person>
</House>
```

For Aptech Center Use Only

The xsl:value-of element 3-4

Style Sheet



```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="Person">
    <p>
      <xsl:value-of select="FirstName"/>
      ,
      <xsl:value-of select="LastName"/>
    </p>
  </xsl:template>
</xsl:stylesheet>
```

where,

`xsl:value-of select="FirstName"`

Display the value of the element `FirstName`.

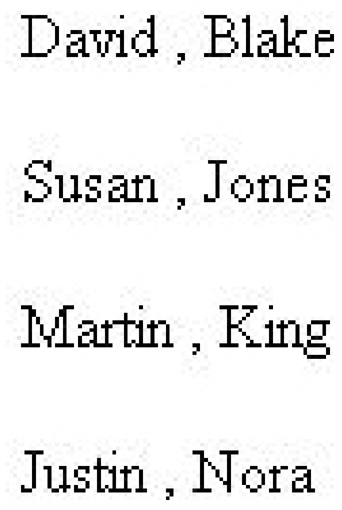
`xsl:value-of select="LastName"`

Display the value of the element `LastName`.

For Aptech Center Use Only

The xsl:value-of element 4-4

Output



```
David , Blake

Susan , Jones

Martin , King

Justin , Nora
```

For Aptech Center Use Only

The xsl:for-each element 1-4

- It can be used to iterate through the XML elements of a specified node set.

Syntax

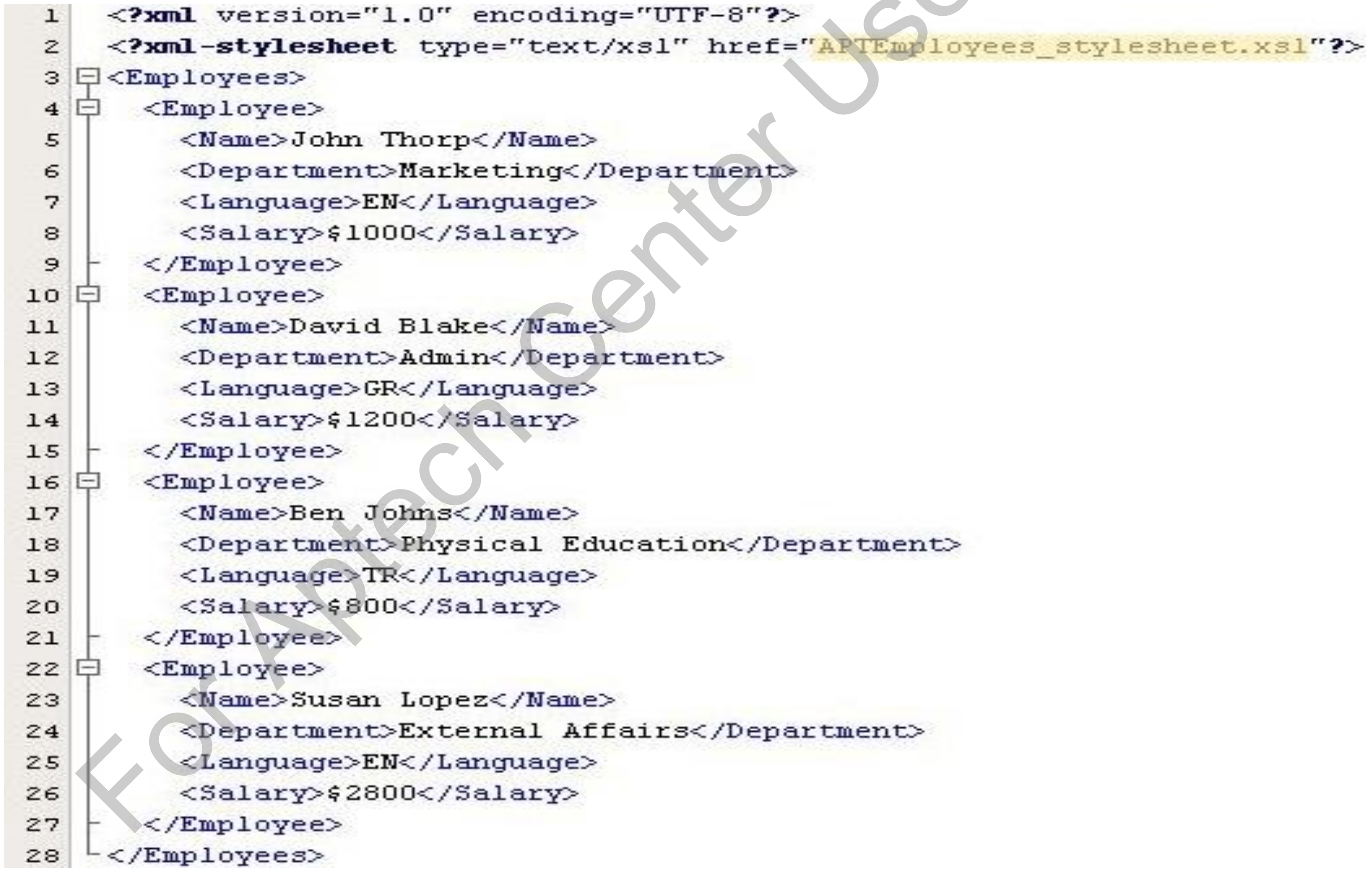
```
<xsl:for-each select="expression">
</xsl:for-each>
```

where,
`select="expresion"`
The expression is evaluated on the current context to determine the set of nodes to iterate over.

For Aptech Center Use Only

The xsl:for-each element 2-4

Code Snippet



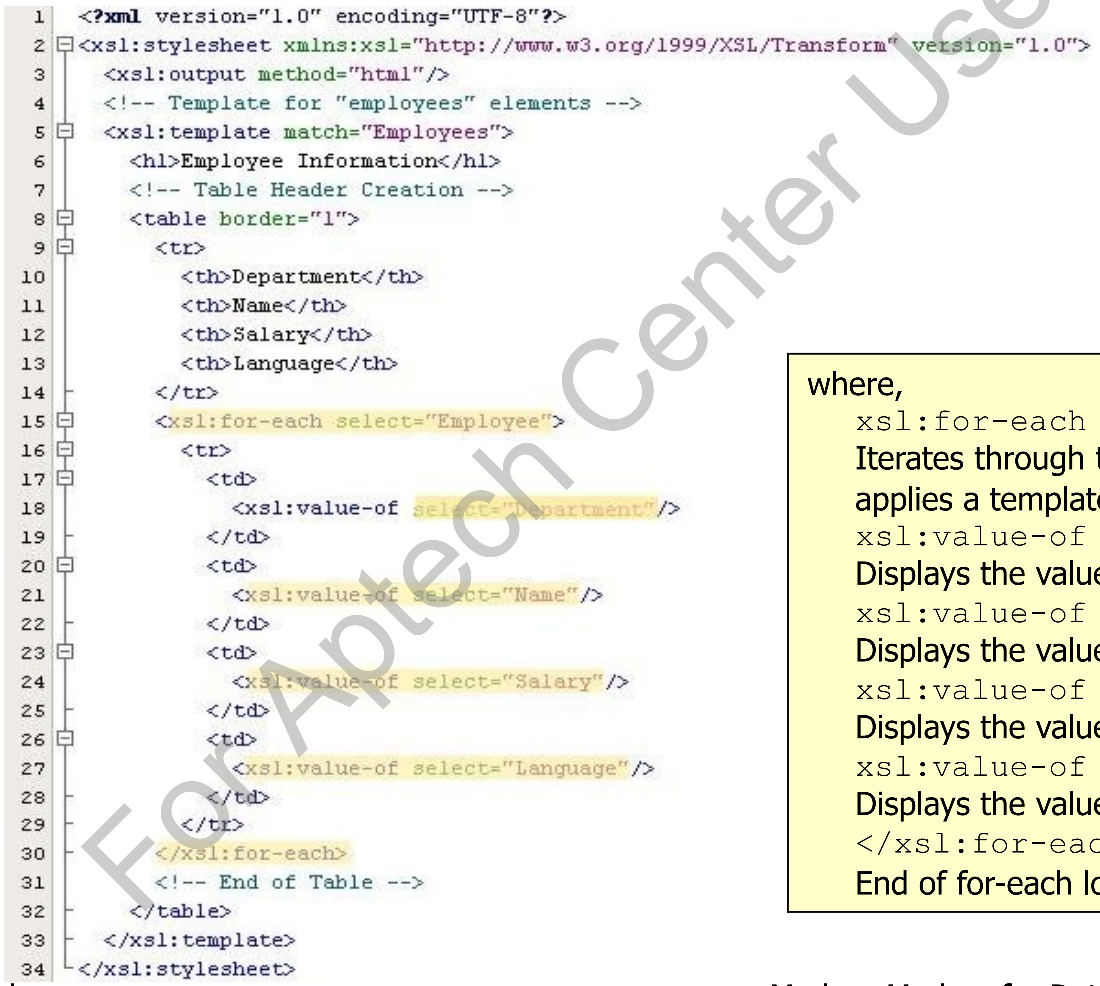
```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="APTEmployees_stylesheet.xsl"?>
<Employees>
  <Employee>
    <Name>John Thorp</Name>
    <Department>Marketing</Department>
    <Language>EN</Language>
    <Salary>$1000</Salary>
  </Employee>
  <Employee>
    <Name>David Blake</Name>
    <Department>Admin</Department>
    <Language>GR</Language>
    <Salary>$1200</Salary>
  </Employee>
  <Employee>
    <Name>Ben Johns</Name>
    <Department>Physical Education</Department>
    <Language>TR</Language>
    <Salary>$800</Salary>
  </Employee>
  <Employee>
    <Name>Susan Lopez</Name>
    <Department>External Affairs</Department>
    <Language>EN</Language>
    <Salary>$2800</Salary>
  </Employee>
</Employees>
```

For Aptech Center Use Only

The xsl:for-each element 3-4

For Aptech Center Use Only

Style Sheet



```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html"/>
  <!-- Template for "employees" elements -->
  <xsl:template match="Employees">
    <h1>Employee Information</h1>
    <!-- Table Header Creation -->
    <table border="1">
      <tr>
        <th>Department</th>
        <th>Name</th>
        <th>Salary</th>
        <th>Language</th>
      </tr>
      <xsl:for-each select="Employee">
        <tr>
          <td>
            <xsl:value-of select="Department"/>
          </td>
          <td>
            <xsl:value-of select="Name"/>
          </td>
          <td>
            <xsl:value-of select="Salary"/>
          </td>
          <td>
            <xsl:value-of select="Language"/>
          </td>
        </tr>
      </xsl:for-each>
      <!-- End of Table -->
    </table>
  </xsl:template>
</xsl:stylesheet>
```

where,

`xsl:for-each select=”Employee”`
Iterates through the `Employee` node and applies a template.

`xsl:value-of select=”Department”`
Displays the value of `Department` element.

`xsl:value-of select=”Name”`
Displays the value of `Name` element.

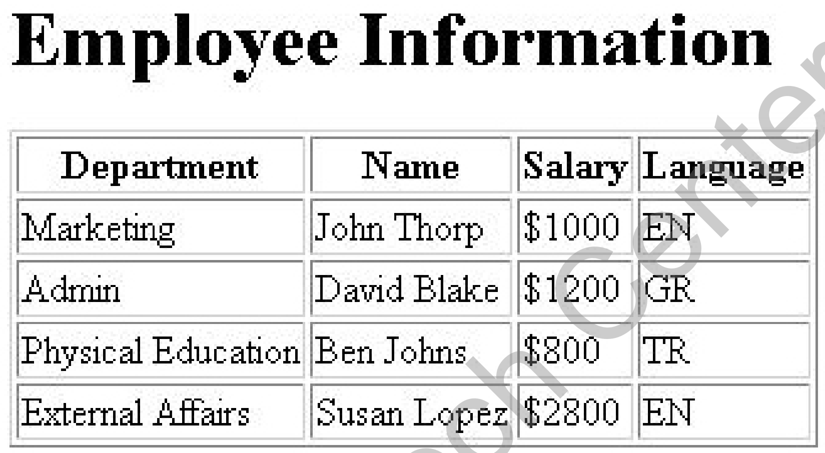
`xsl:value-of select=”Salary”`
Displays the value of `Salary` element.

`xsl:value-of select=”Language”`
Displays the value of `Language` element.

`</xsl:for-each>`
End of for-each loop.

The xsl:for-each element 4-4

Output



Employee Information

Department	Name	Salary	Language
Marketing	John Thorp	$1000	EN
Admin	David Blake	$1200	GR
Physical Education	Ben Johns	$800	TR
External Affairs	Susan Lopez	$2800	EN

For Aptech Center Use Only

The xsl:text element 1-4

- It is used to add literal text to the output.

Syntax

```
<xsl:text
   disable-output-escaping="yes"|"no">
</xsl:text>
```

where,

`disable-output-escaping`

Turns on or off the ability to escape special characters.

`yes`

If the value is yes, a `&gt;` will appear as a >.

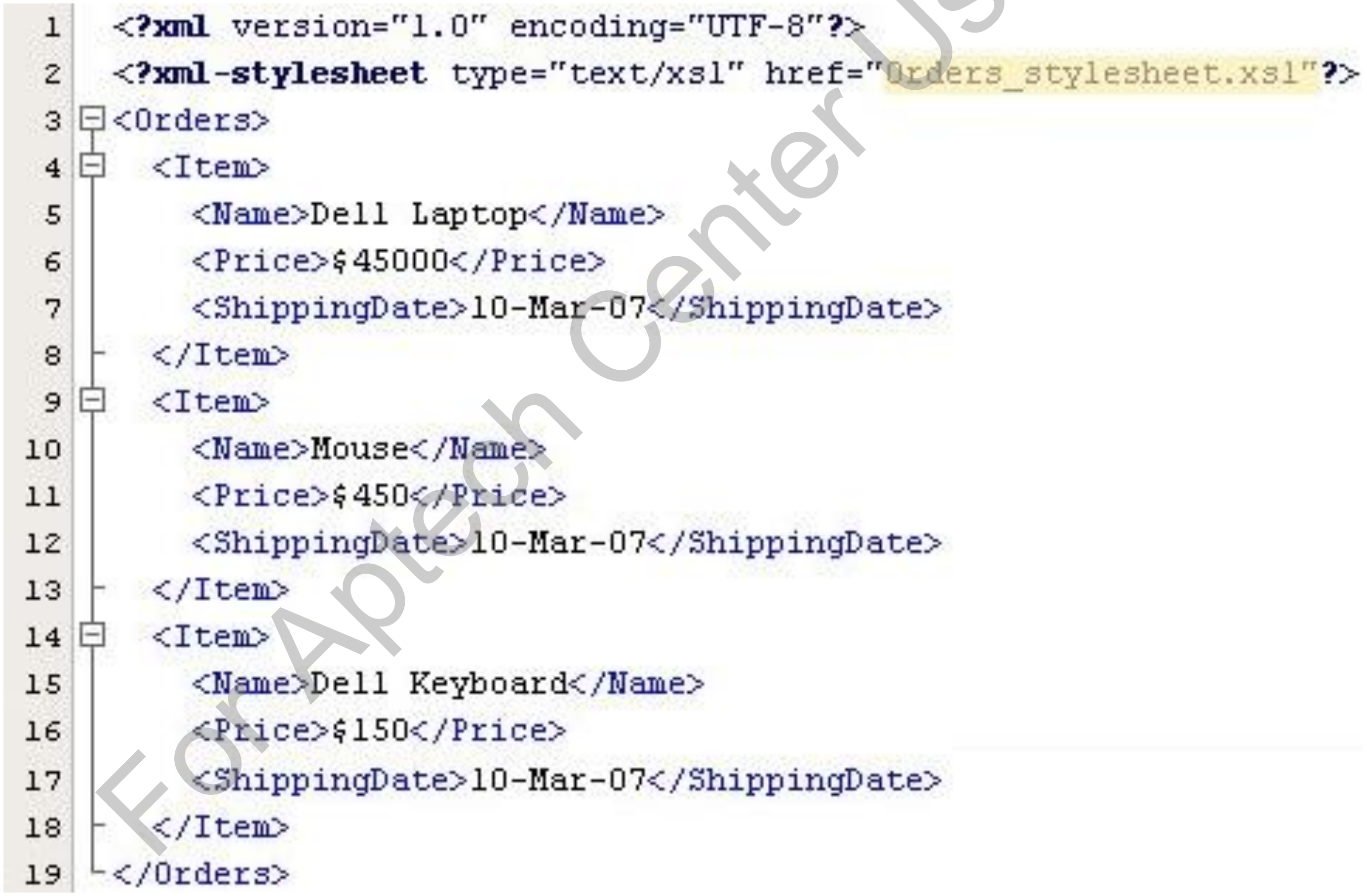
`no`

If the value is no, a `&gt;` will appear as a `&gt;` in the text.

For Aptech Center Use Only

The xsl:text element 2-4

Code Snippet

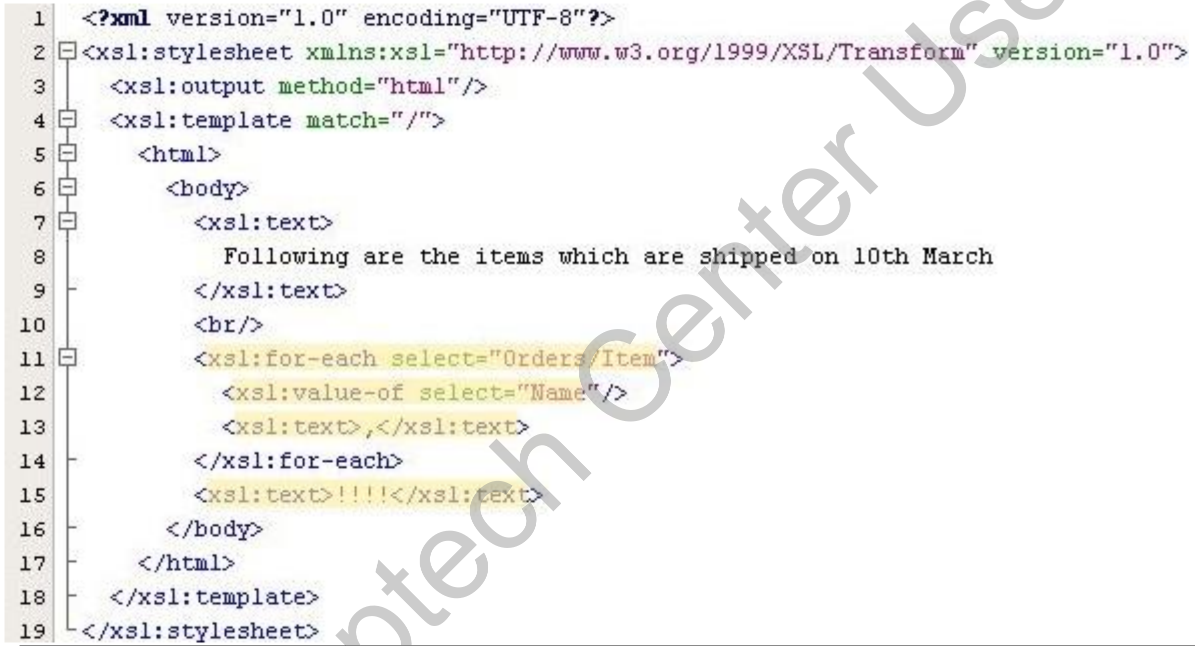


```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="Orders_stylesheet.xsl"?>
<Orders>
  <Item>
    <Name>Dell Laptop</Name>
    <Price>$45000</Price>
    <ShippingDate>10-Mar-07</ShippingDate>
  </Item>
  <Item>
    <Name>Mouse</Name>
    <Price>$450</Price>
    <ShippingDate>10-Mar-07</ShippingDate>
  </Item>
  <Item>
    <Name>Dell Keyboard</Name>
    <Price>$150</Price>
    <ShippingDate>10-Mar-07</ShippingDate>
  </Item>
</Orders>
```

For Aptech Center Use Only

The xsl:text element 3-4

Style Sheet



```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html"/>
  <xsl:template match="/">
    <html>
      <body>
        <xsl:text>
          Following are the items which are shipped on 10th March
        </xsl:text>
        <br/>
        <xsl:for-each select="Orders/Item">
          <xsl:value-of select="Name"/>
          <xsl:text>,</xsl:text>
        </xsl:for-each>
        <xsl:text>!!!!</xsl:text>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

where,

`xsl:for-each select="Orders/Item"`

Iterates through the Item element.

`xsl:value-of select="Name"`

Displays the value of Name element.

`<xsl:text>,</xsl:text>`

Inserts a comma (,) after each name value .

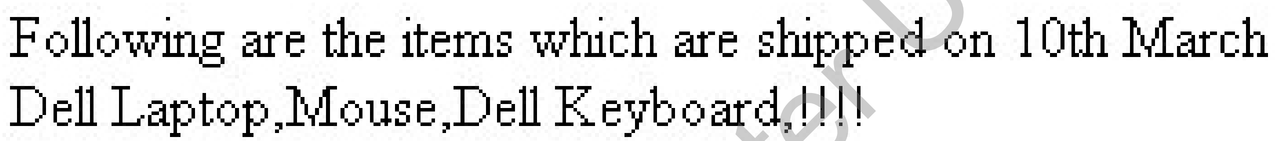
`<xsl:text>!!!!</xsl:text>`

Inserts four exclamation marks (!) at the end of the output.

For Aptech Center Use Only

The xsl:text element 4-4

Output



```
Following are the items which are shipped on 10th March
Dell Laptop,Mouse,Dell Keyboard,!!!!
```

For Aptech Center Use Only

The xsl:number element 1-4

- It can be used to determine the sequence number for the current node.

Syntax

```
<xsl:number
   count="pattern"
   format="{ string }"
   value="expression"
>
</xsl:number>
```

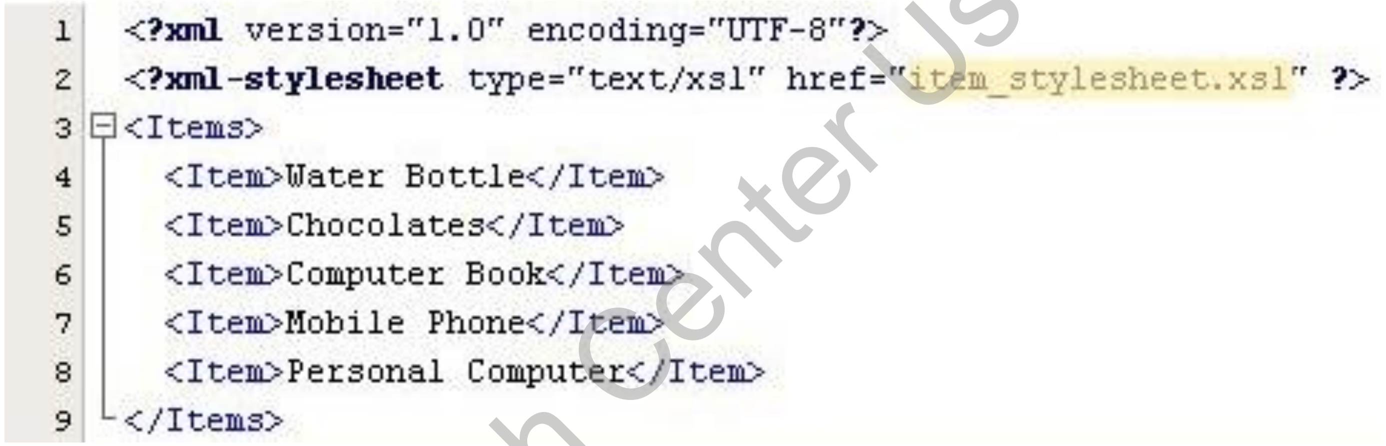
where,

`count="pattern"`
Indicates what nodes are to be counted. Only nodes that match the pattern are counted.
`format="{ string }"`
Sequence of tokens that specifies the format to be used for each number in the list.
`value="expression"`
Specifies the expression to be converted to a number and output to the result tree.

For Aptech Center Use Only

The xsl:number element 2-4

Code Snippet

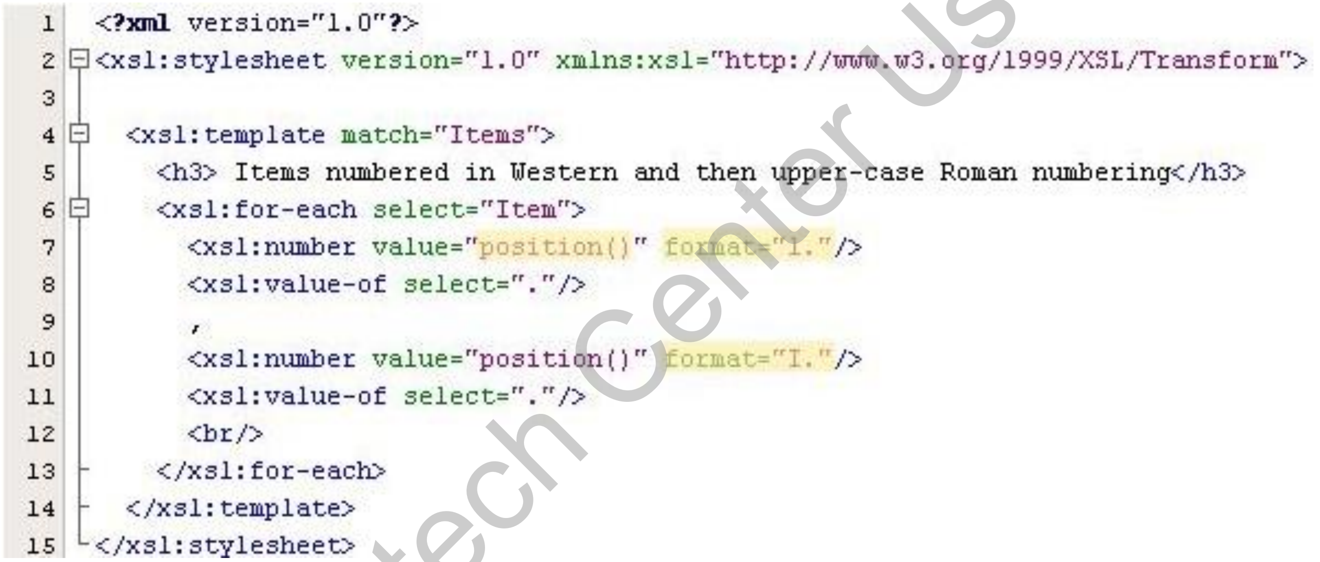


```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="item_stylesheet.xsl" ?>
<Items>
   <Item>Water Bottle</Item>
   <Item>Chocolates</Item>
   <Item>Computer Book</Item>
   <Item>Mobile Phone</Item>
   <Item>Personal Computer</Item>
</Items>
```

For Aptech Center Use Only

The xsl:number element 3-4

Style Sheet



```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="Items">
    <h3> Items numbered in Western and then upper-case Roman numbering</h3>
    <xsl:for-each select="Item">
      <xsl:number value="position()" format="1."/>
      <xsl:value-of select="."/>
      ,
      <xsl:number value="position()" format="I."/>
      <xsl:value-of select="."/>
      <br/>
    </xsl:for-each>
  </xsl:template>
</xsl:stylesheet>
```

where,

`position()`

The current node's position in the source document.

`format="1."`

User-provided number starts with 1.

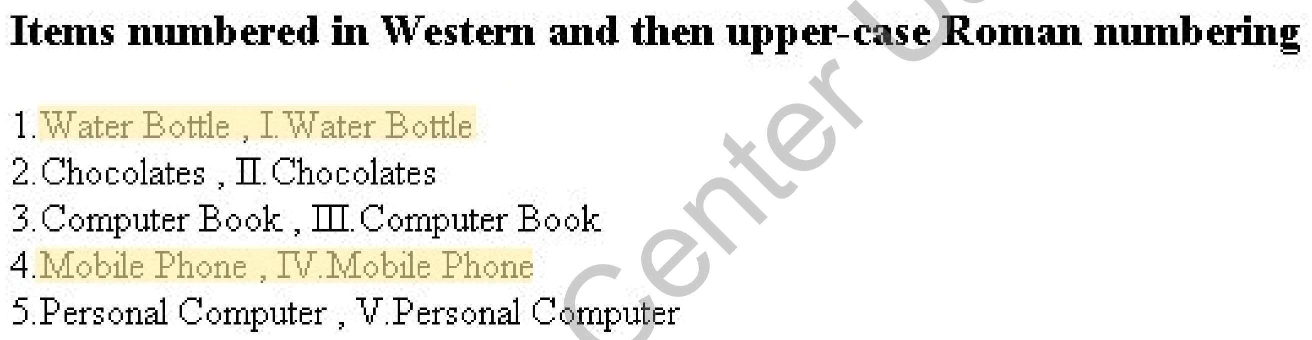
`format="I."`

User-provided roman number starts with I.

For Aptech Center Use Only

The xsl:number element 4-4

Output



Items numbered in Western and then upper-case Roman numbering

```
1.Water Bottle , I.Water Bottle
2.Chocolates , II.Chocolates
3.Computer Book , III.Computer Book
4.Mobile Phone , IV.Mobile Phone
5.Personal Computer , V.Personal Computer
```

where,

`1.Water Bottle, I.Water Bottle`

The first item is numbered with number 1 and roman number I.

`4.Mobile Phone, IV. Mobile Phone`

The fourth item is numbered with number 4 and roman number IV.

For Aptech Center Use Only

The xsl:if element 1-4

- Evaluates a conditional expression against the content of the XML file.

Syntax

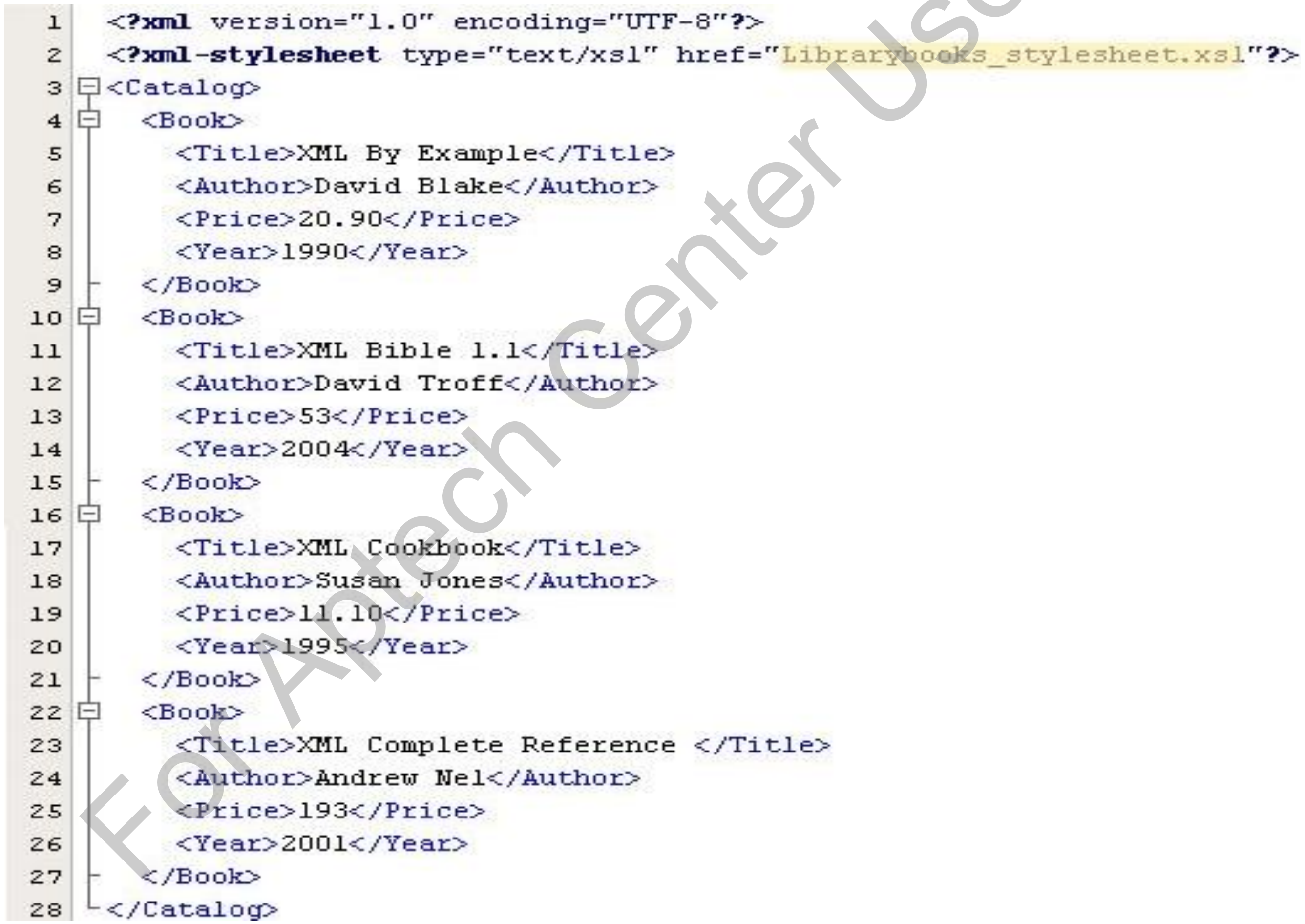
```
<xsl:if
  test="expression"
>
</xsl:if>
```

where,
    `test=expression`
    The condition in the source data to test with either a true or false.

For Aptech Center Use Only

The xsl:if element 2-4

Code Snippet

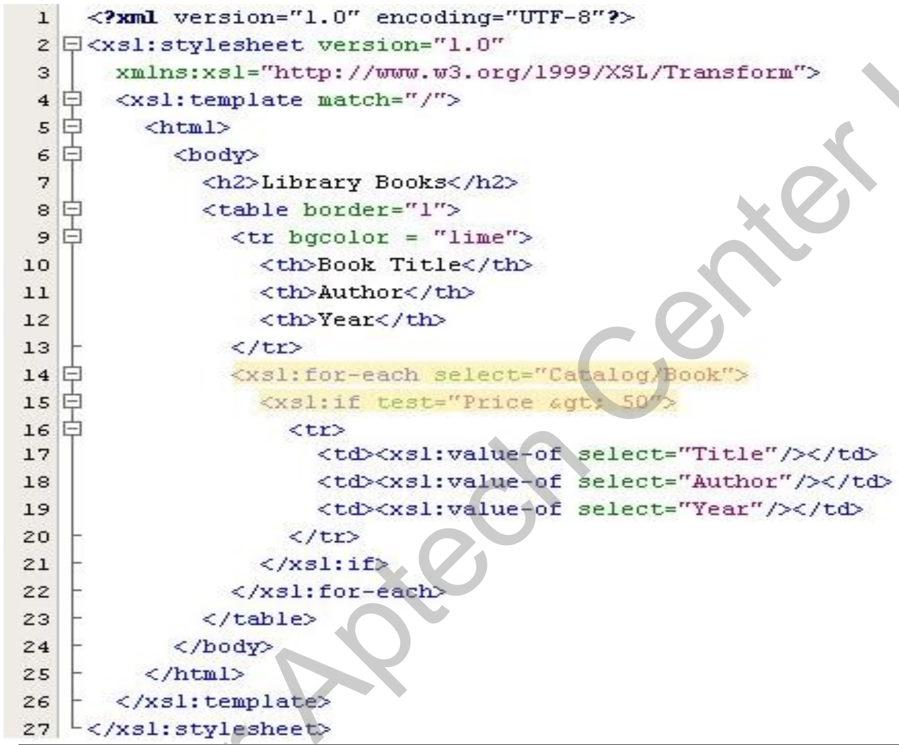


```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="Librarybooks_stylesheet.xsl"?>
<Catalog>
  <Book>
    <Title>XML By Example</Title>
    <Author>David Blake</Author>
    <Price>20.90</Price>
    <Year>1990</Year>
  </Book>
  <Book>
    <Title>XML Bible 1.1</Title>
    <Author>David Troff</Author>
    <Price>53</Price>
    <Year>2004</Year>
  </Book>
  <Book>
    <Title>XML Cookbook</Title>
    <Author>Susan Jones</Author>
    <Price>11.10</Price>
    <Year>1995</Year>
  </Book>
  <Book>
    <Title>XML Complete Reference </Title>
    <Author>Andrew Nel</Author>
    <Price>193</Price>
    <Year>2001</Year>
  </Book>
</Catalog>
```

For Aptech Center Use Only

The xsl:if element 3-4

Style Sheet



```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <h2>Library Books</h2>
        <table border="1">
          <tr bgcolor = "lime">
            <th>Book Title</th>
            <th>Author</th>
            <th>Year</th>
          </tr>
          <xsl:for-each select="Catalog/Book">
            <xsl:if test="Price &gt; 50">
              <tr>
                <td><xsl:value-of select="Title"/></td>
                <td><xsl:value-of select="Author"/></td>
                <td><xsl:value-of select="Year"/></td>
              </tr>
            </xsl:if>
          </xsl:for-each>
        </table>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

where,

`<xsl:for-each select="Catalog/Book">`

Iterates through the Book node.

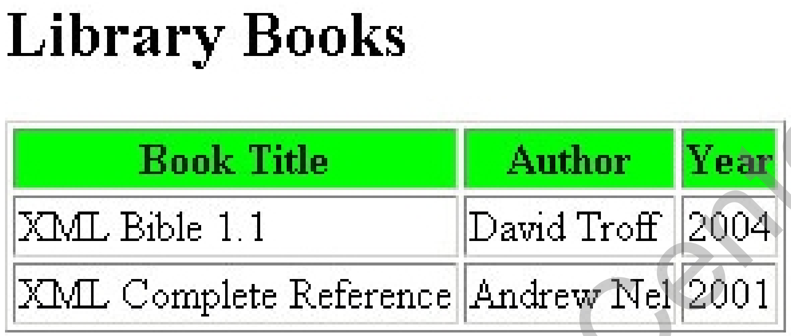
`<xsl:if test="Price &gt; 50">`

Checks if the price of the book is greater than 50. If true, Author, Title and Year are displayed.

For Aptech Center Use Only

The xsl:if element 4-4

Output



Library Books

Book Title	Author	Year
XML Bible 1.1	David Troff	2004
XML Complete Reference	Andrew Nel	2001

For Aptech Center Use Only

The xsl:choose element 1-5

For Aptech Center Use Only

- It is used in conjunction with `xsl:when` and `xsl:otherwise` to express multiple conditional tests.

Syntax

```
<xsl:choose>
  <xsl:when test="expression">
   template body
   </xsl:when>
   ...
  <xsl:otherwise>
    template body
  </xsl:otherwise>
</xsl:choose>
```

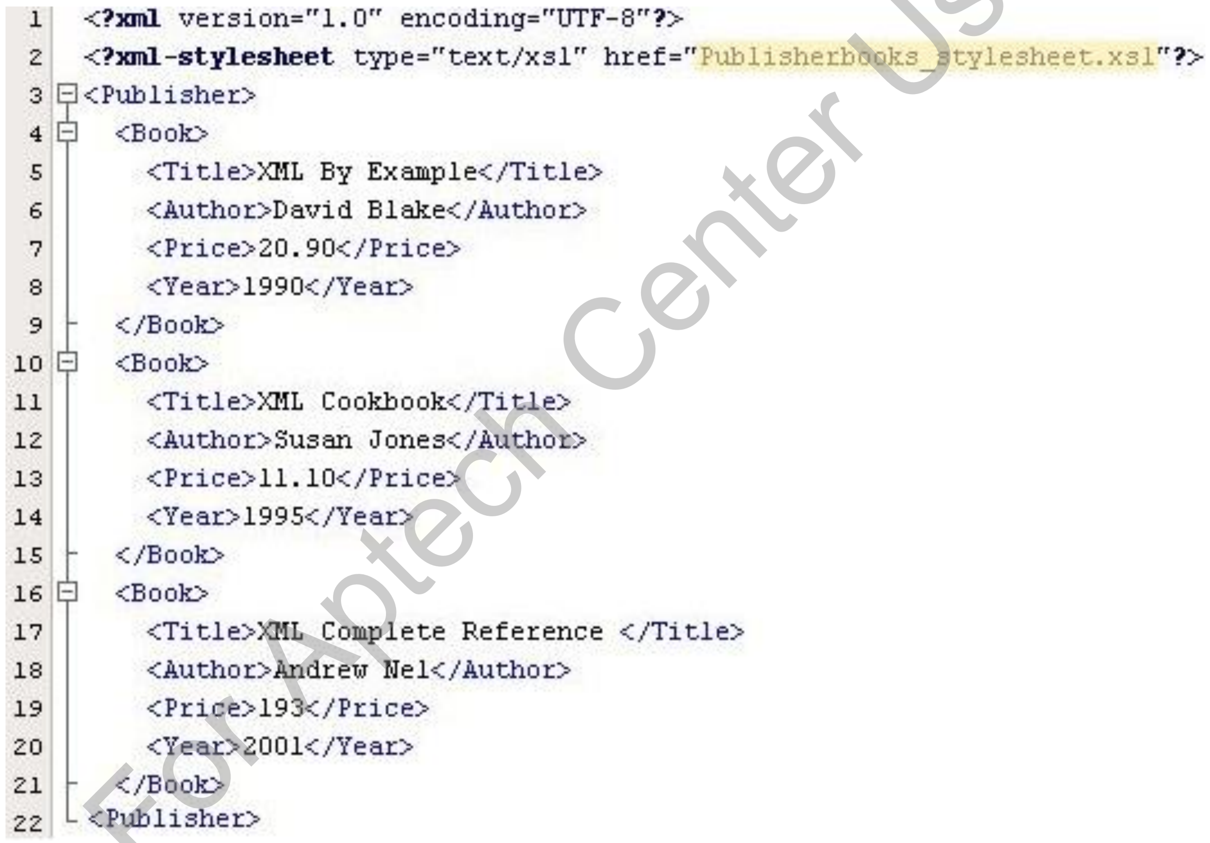
where,

`xsl:when test="expression"`
The `xsl:when` element is examined in the order of occurrence. If the test expression is true, the code contained in that element is executed.

`xsl:otherwise`
If all the test conditions in any `xsl:when` element are false, then the `xsl:otherwise` element is automatically selected and the code associated with that element is executed.

The xsl:choose element 2-5

Code Snippet

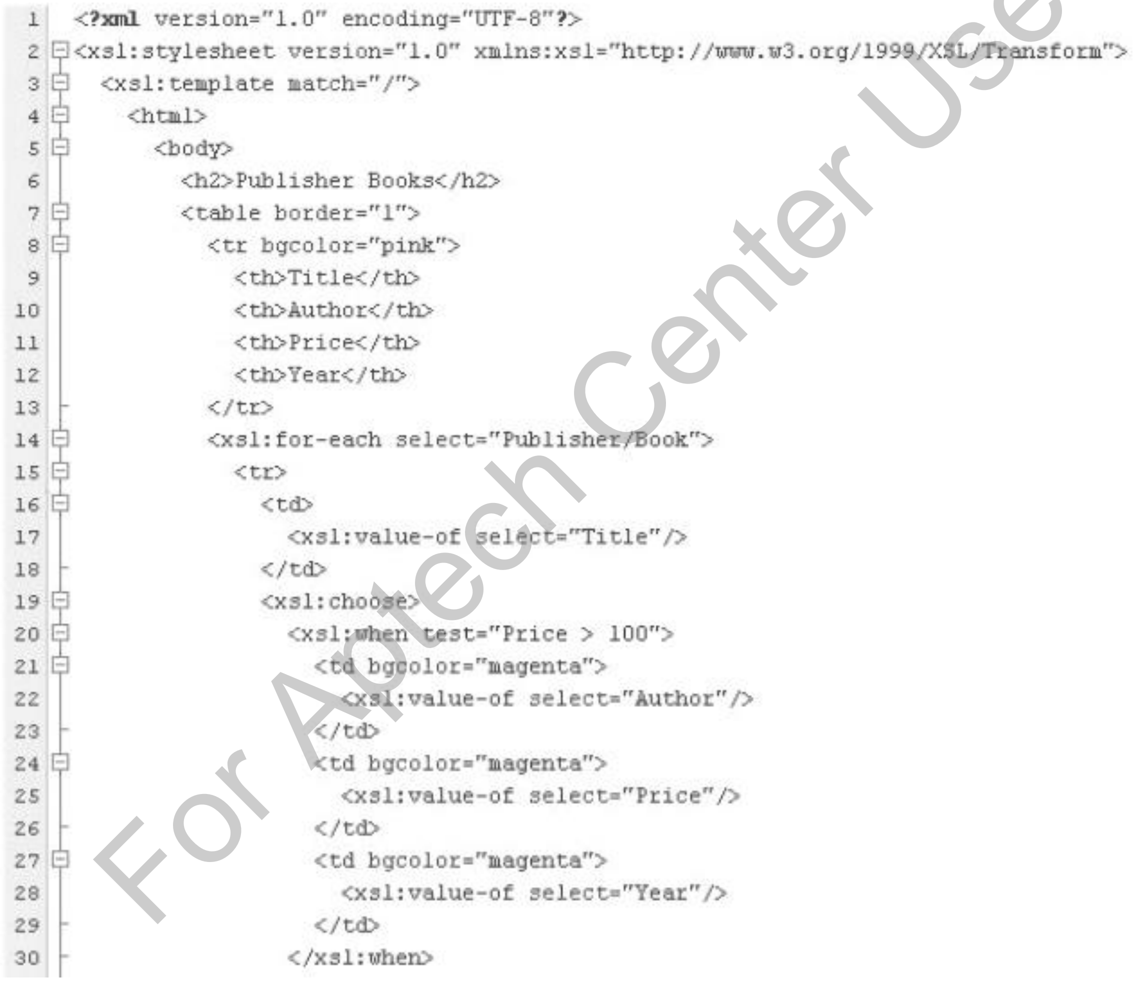


```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="Publisherbooks_stylesheet.xsl"?>
<Publisher>
  <Book>
    <Title>XML By Example</Title>
    <Author>David Blake</Author>
    <Price>20.90</Price>
    <Year>1990</Year>
  </Book>
  <Book>
    <Title>XML Cookbook</Title>
    <Author>Susan Jones</Author>
    <Price>11.10</Price>
    <Year>1995</Year>
  </Book>
  <Book>
    <Title>XML Complete Reference </Title>
    <Author>Andrew Nel</Author>
    <Price>193</Price>
    <Year>2001</Year>
  </Book>
<Publisher>
```

For Aptech Center Use Only

The xsl:choose element 3-5

Style Sheet

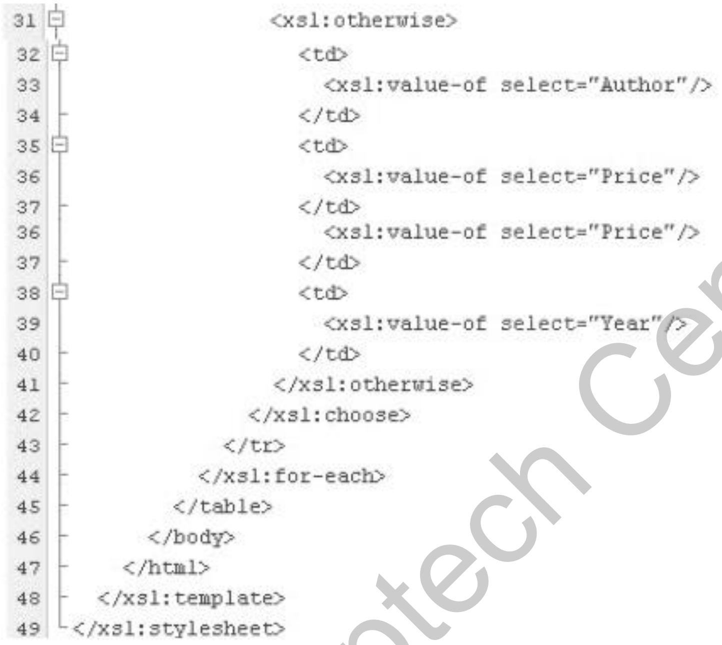


```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <h2>Publisher Books</h2>
        <table border="1">
          <tr bgcolor="pink">
            <th>Title</th>
            <th>Author</th>
            <th>Price</th>
            <th>Year</th>
          </tr>
          <xsl:for-each select="Publisher/Book">
            <tr>
              <td>
                <xsl:value-of select="Title"/>
              </td>
              <xsl:choose>
                <xsl:when test="Price > 100">
                  <td bgcolor="magenta">
                    <xsl:value-of select="Author"/>
                  </td>
                  <td bgcolor="magenta">
                    <xsl:value-of select="Price"/>
                  </td>
                  <td bgcolor="magenta">
                    <xsl:value-of select="Year"/>
                  </td>
                </xsl:when>
```

For Aptech Center Use Only

The xsl:choose element 4-5

Style Sheet



```
                    <xsl:otherwise>
                        <td>
                            <xsl:value-of select="Author"/>
                        </td>
                        <td>
                            <xsl:value-of select="Price"/>
                        </td>
                            <xsl:value-of select="Price"/>
                        </td>
                        <td>
                            <xsl:value-of select="Year"/>
                        </td>
                    </xsl:otherwise>
                </xsl:choose>
            </tr>
        </xsl:for-each>
    </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

where,

`xsl:when test="Price > 100"`

Checks whether the price of the book is greater than 100. If true, the details like `Author, Price` and `Year` are displayed with magenta as the background color.

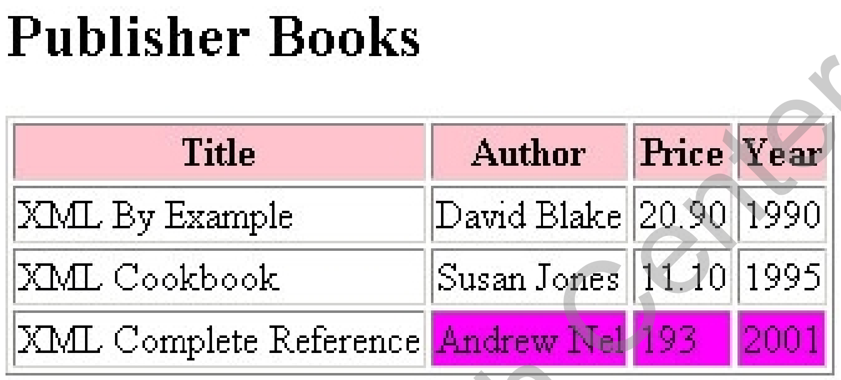
`xsl:otherwise`

If all the conditions are false, this block is executed. Here, all other book details are printed in normal background color.

For Aptech Center Use Only

The xsl:choose element 5-5

Output



Publisher Books

Title	Author	Price	Year
XML By Example	David Blake	20.90	1990
XML Cookbook	Susan Jones	11.10	1995
XML Complete Reference	Andrew Nel	193	2001

For Aptech Center Use Only

Sorting in XSLT 1-4

- It can be used to sort a group of similar elements.

Syntax

```
<xsl:sort
  case-order="upper-first" | "lower-first"
  data-type="number" "qname" | "text"
  order="ascending" | " descending"
  select="expression"
>
</xsl:sort>
```

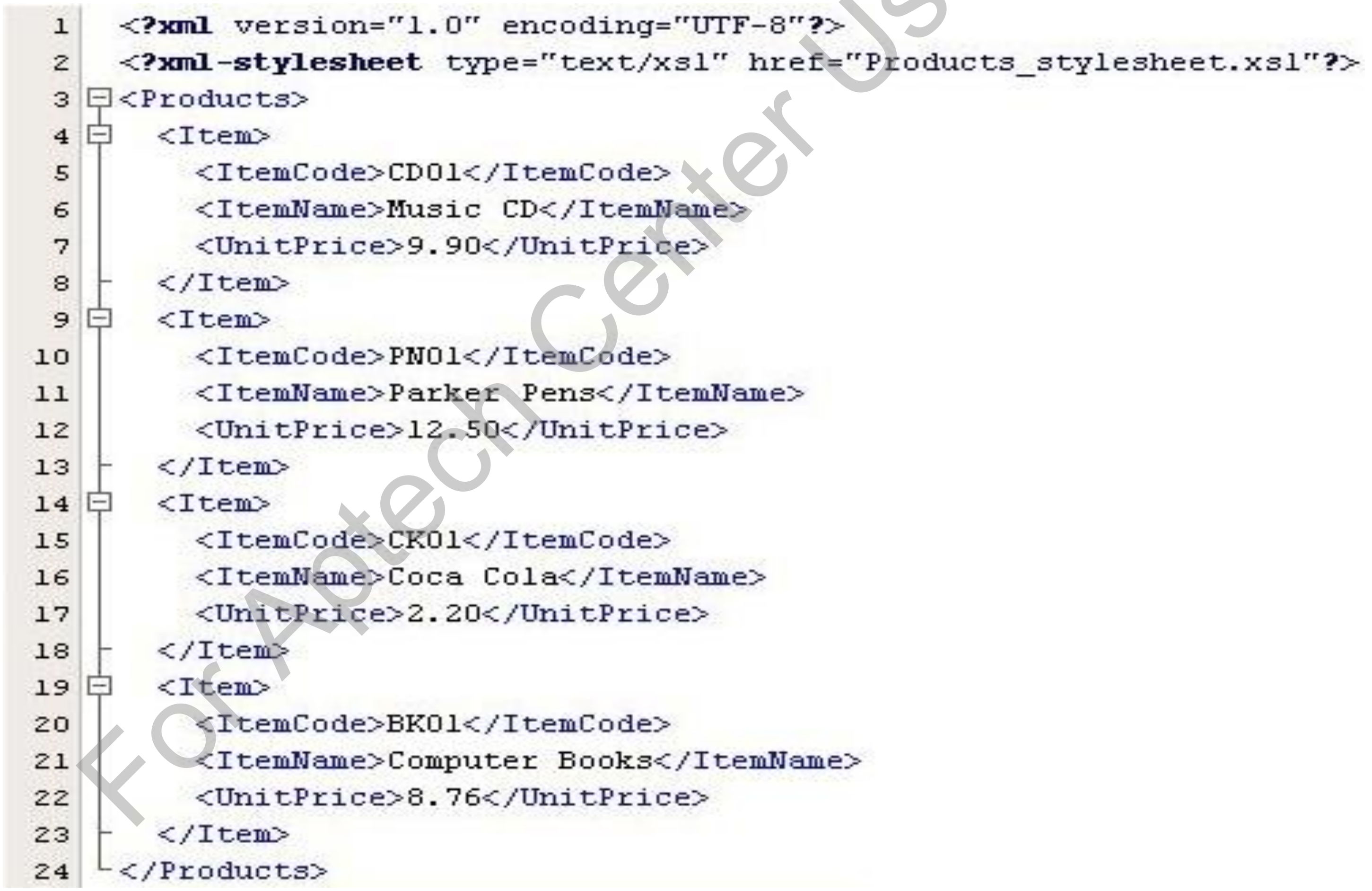
where,

`case-order`
Indicates whether the sort will have upper or lowercase letters listed first in the sort output. The default option is to list uppercase first.
`data-type:` Specifies the data type of the strings.
`Number:` Sort key is converted to a number.
`Qname:` Sort is based upon a user-defined data type.
`Text:` Specifies that the sort keys should be sorted alphabetically.
`Order:` The sort order for the strings. The default value is “ascending”.
`Select:` Expression that defines the key upon which the sort will be based. The expression is evaluated and converted to a string that is used as the sort key.

For Aptech Center Use Only

Sorting in XSLT 2-4

Code Snippet

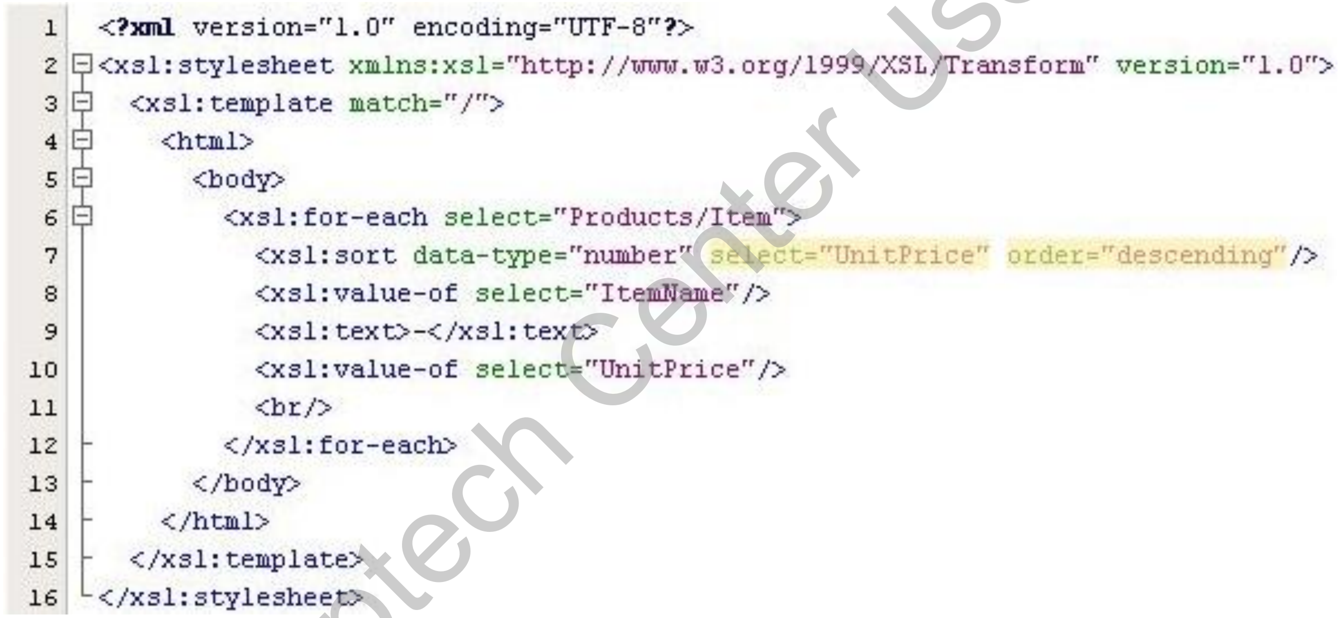


```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="Products_stylesheet.xsl"?>
<Products>
  <Item>
    <ItemCode>CD01</ItemCode>
    <ItemName>Music CD</ItemName>
    <UnitPrice>9.90</UnitPrice>
  </Item>
  <Item>
    <ItemCode>PN01</ItemCode>
    <ItemName>Parker Pens</ItemName>
    <UnitPrice>12.50</UnitPrice>
  </Item>
  <Item>
    <ItemCode>CK01</ItemCode>
    <ItemName>Coca Cola</ItemName>
    <UnitPrice>2.20</UnitPrice>
  </Item>
  <Item>
    <ItemCode>BK01</ItemCode>
    <ItemName>Computer Books</ItemName>
    <UnitPrice>8.76</UnitPrice>
  </Item>
</Products>
```

For Aptech Center Use Only

Sorting in XSLT 3-4

Style Sheet



```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:template match="/">
    <html>
      <body>
        <xsl:for-each select="Products/Item">
          <xsl:sort data-type="number" select="UnitPrice" order="descending"/>
          <xsl:value-of select="ItemName"/>
          <xsl:text>-</xsl:text>
          <xsl:value-of select="UnitPrice"/>
          <br/>
        </xsl:for-each>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

where,

`select=”UnitPrice”`

Sort is based on `UnitPrice` element.

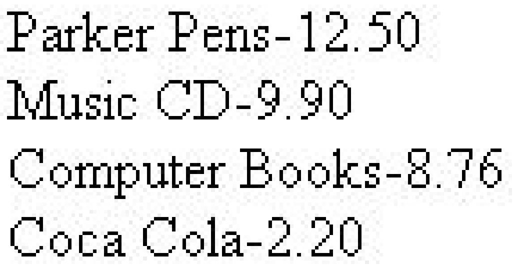
`order=”descending”`

The sorting order for `UnitPrice` is descending in that the higher value will be displayed first.

For Aptech Center Use Only

Sorting in XSLT 4-4

Output



```
Parker Pens-12.50
Music CD-9.90
Computer Books-8.76
Coca Cola-2.20
```

For Aptech Center Use Only

Summary

- **Introduction to XSL**
  - XML provides the ability to format document content.
  - XSL provides the ability to define how the formatted XML content is presented.
  - An XSL Transformation applies rules to a source tree read from an XML document to transform it into an output tree written out as an XML document.

- **Working with XSL**
  - An XSL template rule is represented as an `xsl:template` element.
  - You can process multiple elements in two ways: using the `xsl:apply-templates` element and the `xsl:for-each` element.
  - The `xsl:stylesheet` element allows you to include a stylesheet directly in the document it applies to.
  - The `xsl:if` element produces output only if its `test` attribute is `true`.

For Aptech Center Use Only