



Lecture 28: XSLT

Kenneth M. Anderson
Software Methods and Tools
CSCI 3308 - Fall Semester, 2003



Today's Lecture

- Introduce XSLT
 - background
 - concepts
 - examples
- XSLT stands for XML Stylesheet Language, Transformations

December 1, 2003

© University of Colorado, 2003

2



Transformations

- XSLT was developed as part of the XML stylesheet standards effort
- What's a stylesheet?
 - A stylesheet is a device for specifying presentation information independent of content
 - For instance, in Microsoft Word, you can specify that a "heading" should appear in 36pt Times bold font with double spacing above and below
 - Then all headings will appear that way, no matter what the heading actually "says"

December 1, 2003

© University of Colorado, 2003

3



Stylesheets in HTML

- The Web already has a stylesheet language called "cascading stylesheets" or CSS
- This mechanism allows formatting information to be associated with HTML tags, such as <H1> or <P> without using or tags
- In the last lecture, we asked the question, if CNN switched to using XML in their webpage, how would they associate formatting information with a tag such as <headline>?

December 1, 2003

© University of Colorado, 2003

4

XSL and XSLT

- The answer, of course, is with the XML Stylesheet language (XSL)
- Problem: the XSL standards group was having a hard time agreeing on a mechanism for specifying formatting information
- However, they were making progress with the transformation part of the specification
 - This part specifies mechanisms for transforming XML to other structures, such as XML->XML, XML->HTML, XML->PDF, etc.

XSLT

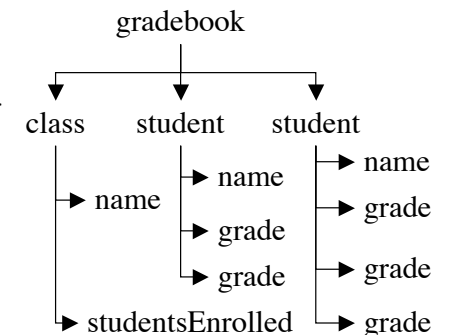
- Plus, they noticed that a large part of their users
 - were using the transformation part of the XSL specification to transform XML to HTML
 - and then using CSS to format the resulting document
- As such, XSLT was born, the standards group focused on finishing the transformation part of the specification first and published it as XSLT (*XSL has since been completed*)

Background

- To understand XSLT, you must view XML documents as tree structures
 - XSLT provides rules to transform one tree into another tree
 - It traverses the source tree in an order dictated by the stylesheet and creates the destination tree using the rules of the stylesheet

Example of viewing XML as a tree

```
<!DOCTYPE gradebook [  
  <!ELEMENT gradebook (class, student*)>  
  <!ELEMENT class (name, studentsEnrolled)>  
  <!ATTLIST class semester CDATA #REQUIRED>  
  <!ELEMENT name (#PCDATA)>  
  <!ELEMENT studentsEnrolled (#PCDATA)>  
  <!ELEMENT student (name, grade*)>  
  <!ELEMENT grade (#PCDATA)>  
  <!ATTLIST grade name CDATA #REQUIRED>  
>
```



Background: XPath

- XSLT uses a separate standard, called XPath, to help select nodes in an XML document
- For instance...
 - gradebook/student/grade
 - ...is an XPath expression that selects all “grade” nodes in the example on the previous slide
- XPath can even select attributes...for example..
 - gradebook/student/grade[@name=“hw3”]
 - ...will select only those grade nodes that have a value of “HW3” for their name attribute

More XPath examples

- //grade
 - “start at the root node and find all grade nodes”
- gradebook/student[2]
 - “select the second student node under gradebook”
- For more information on XPath see
 - < <http://www.w3.org/TR/xpath>>
- You will need to know how to create “simple” XPath expressions (like the ones shown above) to complete lab 10

XSLT, the details

- XSLT transforms XML documents using stylesheets that are themselves XML documents
- All XSLT stylesheets have the following form

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  ...templates and transformation rules go here...
</xsl:stylesheet>
```

You can use this template when writing your own XSL Stylesheet in Lab 10

Stylesheets

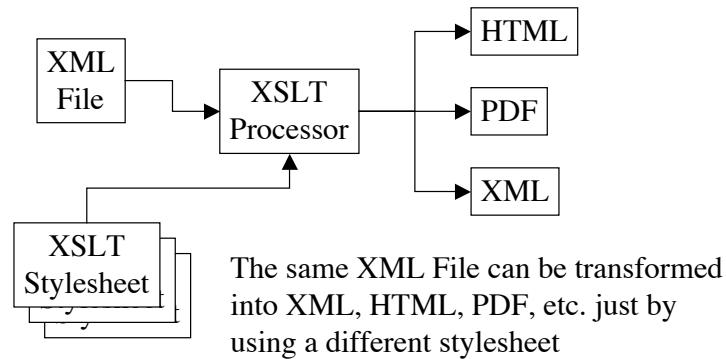
- Stylesheets consist of templates that “match” nodes of the source XML tree (i.e. document)
 - Each template then specifies what should be created in the destination tree (or document)
 - A template looks like this:

```
<xsl:template match="/">
<html>
  <head>
    <title>Grade Book</title>
  </head>
  <xsl:apply-templates/>
</html>
</xsl:template>
```

The tag is called “xsl:template” and it has an attribute called “match” that takes an XPath expression

If a node matches this expression (in this case the root node) then the associated text appears in the destination document (except for the “xsl:apply-templates” part)

XSLT Architecture



More Details

- Stylesheet processing
 - XSLT processor is handed a document and a stylesheet
 - It starts a (breadth-first) traversal at the root node and checks to see if there is a template match
 - If so, it applies the template and looks for an “xsl:apply-templates” element
 - If such an element exists, it continues the traversal
 - if no such element exists, the traversal stops
 - If not, it traverses down the tree looking for a template match with some other node of the tree

XSL:apply-templates

- The apply-templates tag determines if an XSLT processor continues traversing a document once a template match has occurred
- The apply-templates tag can contain an attribute called “select” which can specify the specific children to continue traversing using an XPath expression
 - <xsl:apply-templates>
 - All children traversed
 - <xsl:apply-templates select=“grade[@name=‘HW4’]”>
 - All grade nodes with a name attribute equal to “HW4” traversed (any other nodes skipped during the subsequent traversal)

Processing in XSLT stylesheets

- XSLT is very powerful
 - We cannot cover the entire standard
 - So, the following slides cover only a small subset of the tags that can be placed in an XSLT stylesheet
 - For a good reference on XSLT see:
 - <<http://www.zvon.org/xxl/XSLReference/Output/index.html>>

Repetition

```
<xsl:for-each select = "item">
```

Do something here ...

```
</xsl:for-each>
```

- Again, the select attribute is an XPath expression that selects the nodes to iterate over

Repetition Example

```
<xsl:template match="/">
<html>
  <head>
    <title>Grade Book</title>
  </head>
  <body>
    <ul>
      <xsl:for-each select="student/grade">
        <li>Grade: <xsl:value-of select="."/></li>
      </xsl:for-each>
    </ul>
  </body>
</html>
</xsl:template>
```

Example Explained

- This example creates a simple HTML file that contains a list of all the grades received by students in the gradebook
 - Note: It did not list student names for each set of grades but it could have easily done so.
 - However the "student/grade" XPath expression in the for-each select attribute skipped past the student nodes and selected only grade nodes
 - The value-of element pulled the value of the grade element (e.g. the grade) into the HTML file
 - The resulting HTML file is shown on the next slide

Generated HTML File

```
<html>
  <head>
    <title>Grade Book</title>
  </head>
  <body>
    <ul>
      <li>Grade: 10</li>
      <li>Grade: 7</li>
      <li>Grade: 6</li>
      <li>Grade: 10</li>
      ... more grades here ...
    </ul>
  </body>
</html>
```

- In the browser, this file would look like this:
- Grade Book
 - Grade: 10
 - Grade: 7
 - Grade: 6
 - Grade: 10
- e.g. a bulleted list of grades



Additional Tags

- `<xsl:value-of select=".">`
 - Used to pull the values of XML tags out of XML files, e.g. the part that appears between the begin and close tags
 - `<grade>10</grade>` -> places 10 in destination document
- `<xsl:if test="position()=last()>`
 - A tag for doing processing conditionally
 - value of test is again an XPath expression
 - This particular XPath expression determines if the current node is the last child of the parent node



Additional Tags

```
<xsl:choose>
  <xsl:when
    test = "position()=last()>
    Do something for last element
  </xsl:when>
  <xsl:when
    test = "position()=first()>
    Do something for first element
  </xsl:when>
  <xsl:otherwise>
    Do something for other elements
  </xsl:otherwise>
</xsl:choose>
```



Additional Tags

- `<xsl:sort data-type="" select="" order="">`
 - Used to sort the results of a select statement of another XSLT tag
 - The select attribute of `xsl:sort` is used to indicate which field of the selected nodes is used to perform the sort
 - Appears within an `<xsl:apply-templates>` tag
 - `data-type` can have the value "text" or "number"; text is the default
 - `order` can have the value "ascending" or "descending"; ascending is the default
- `<xsl:apply-templates select="//student">`
 - `<xsl:sort select="name"/>`
- `</xsl:apply-templates>`
- This selects all student nodes, sorts them by name, and then applies templates to them



More information

- <http://www.xslt.com/>
 - General Information
- <http://www.w3.org/TR/xslt/>
 - XSLT specification
- <http://xml.apache.org/xalan/>
 - Powerful XSLT stylesheet processor
 - You will be using Xalan in Lab 10