XSLT – eXtensible Style Language for Transformations

You now have an understanding of the XML language and how to create and read XML documents, the next step is to format those documents.

You will learn how to transform an XML document using XSLT. Transforming an XML document means using XSLT to analyse its contents and then take certain actions depending on what elements are found.

You can use XSLT to reorder the output according to certain criteria, display only certain pieces of information, and much more.

XSLT – eXtensible Style Language for Transformations

- To transform XML is to change its structure, its markup, and perhaps its content into another form.
- There are many reasons to transform XML. Most often it is used to extend the reach of a document into new areas by converting it into a presentational format.
- Alternatively you can use transformation to alter the content, such as extracting a section, or adding a table of numbers together.

XSLT – eXtensible Style Language for Transformations

The transformation process starts with two documents, the XML document which contains the source data to be transformed, and the XSLT style sheet document which describes the rules of the transformation. We will transform the XML document into HTML.

To perform the transformation, you need an XSLT processor, or a browser that supports XSLT. Most current XML editors (including Eclipse) have built in XSLT support, as do most current Web browsers.

XSLT – eXtensible Style Language for Transformations

To begin, you'll need to link your XML document to your XSLT style sheet using the xml-stylesheet processing instruction.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="lab04.xsl"?>
<ancient_wonders>
  <wonder>
    <name language="English">Colossus of Rhodes</name>
    <location>Rhodes, Greece</location>
  </wonder>
</ancient_wonders>
```

XSLT – eXtensible Style Language for Transformations

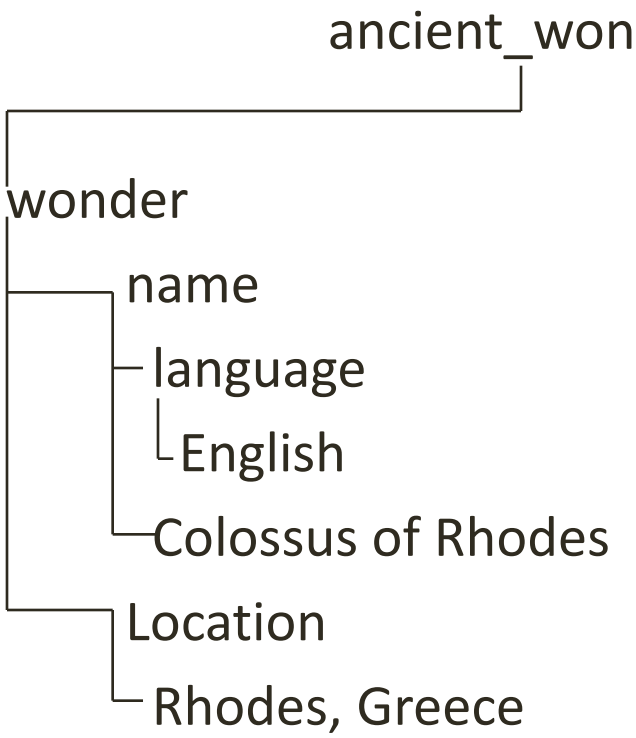
When you open your XML document the instruction tells the processor to perform the XSLT transformation before displaying the document.

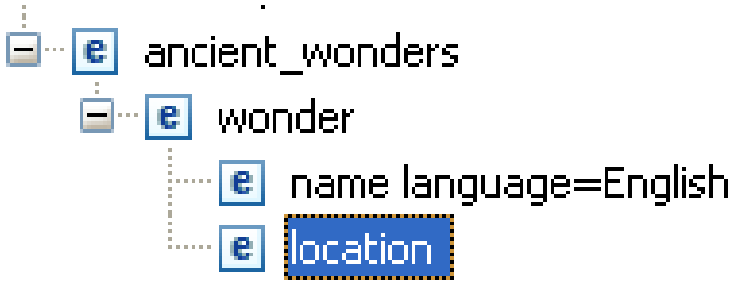
In the first step of this transformation, the XSLT processor analyses the XML document and converts it into a node tree. A node tree is a hierarchical representation of the XML document. In the tree a node is one individual piece of the XML document such as an element, an attribute, or some text content.

The processer reads the stylesheet and creates a look up table in memory. For each node it processes it finds the appropriate rule and applies it. When there are no more nodes processing is complete and the document is outputted.

Eclipse shows us this tree in the right hand window pane.

XSLT – eXtensible Style Language for Transformations





XSLT – eXtensible Style Language for Transformations

Assessing the XSLT style sheet

Once the processor has identified the nodes in the source XML, it then looks to an XSLT style sheet for instructions on what to do with those nodes. Those instructions are contained in templates which are comparable to functions in a programming language.

Each XSLT template has two parts: first, a label that identifies the nodes in the XML document to which the template applies; and second, instructions about the actual transformation that should take place. The instructions will either output or further process the nodes in the source document.

XSLT – eXtensible Style Language for Transformations

Performing the transformation

The XSLT transformation begins by processing the root template. Every XSLT style sheet must have a root template; this is the template that applies to the source XML document's root node.

The root template is defined with the following:

```
<xsl:template match="/">
```

Within this root template, there may be other sub-templates which can then apply to other nodes in the XML document.

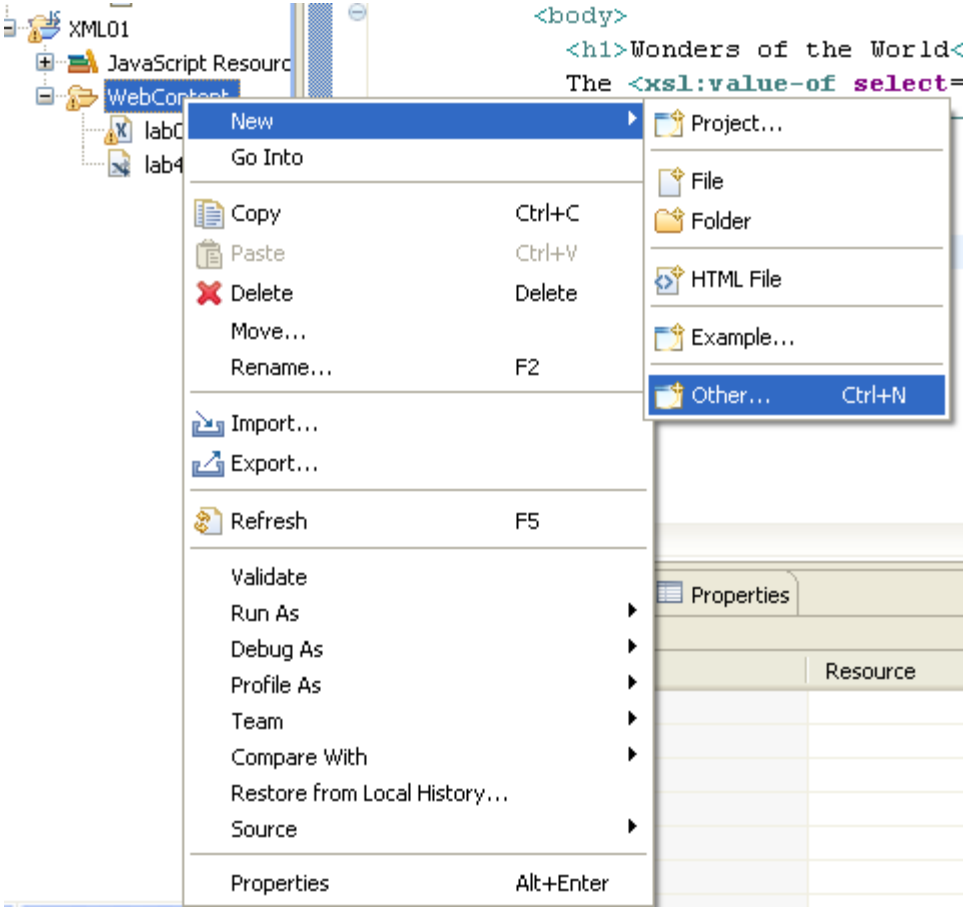
XSLT – eXtensible Style Language for Transformations

Tips

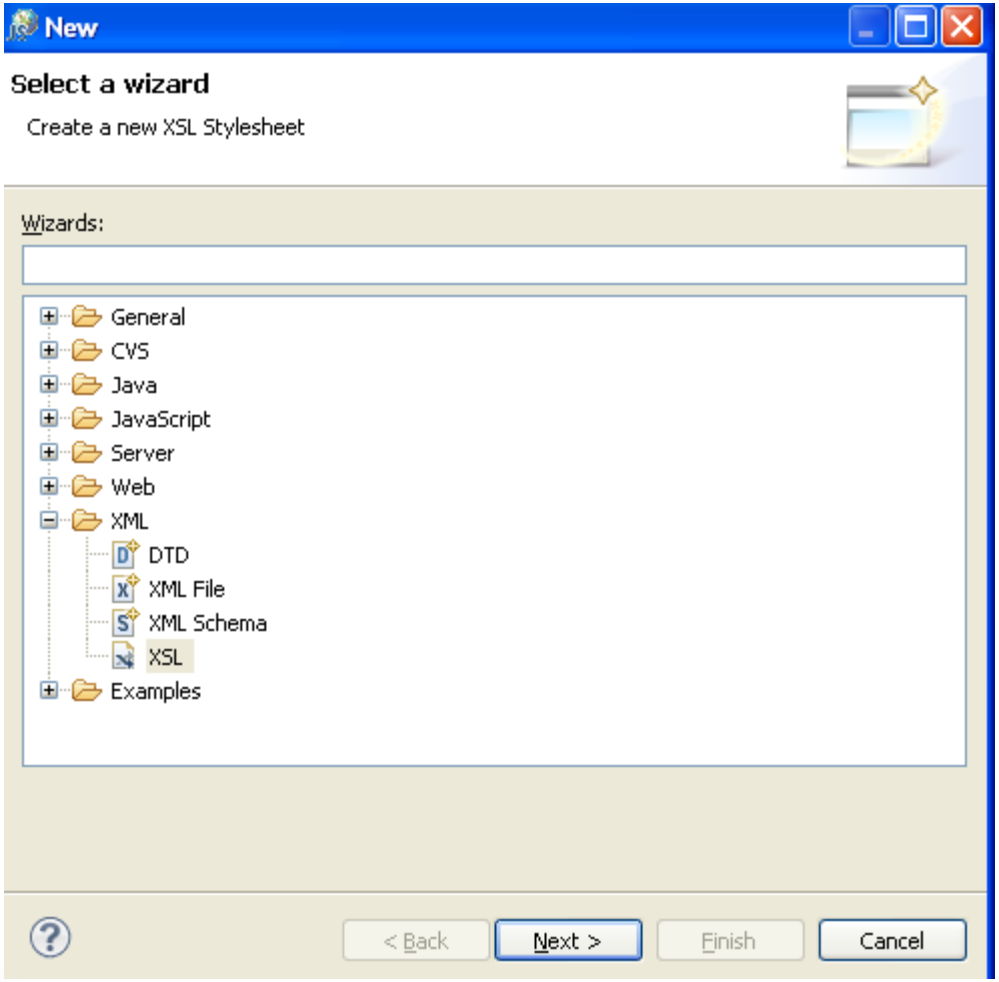
XSLT style sheets are text files and are saved with .xsl extension.

Eclipse allows you to create an xsl file by right clicking on the WebContent folder in your eclipse project, from the New option, choose Other... from the list and select xsl as the file type, give the file a name and click finish.

XSLT – eXtensible Style Language for Transformations



XSLT – eXtensible Style Language for Transformations



XSLT – eXtensible Style Language for Transformations

XSLT uses the XPATH language to identify nodes. We will cover XPATH in detail as we progress through the module.

Every XSLT style sheet is actually an XML document in itself, and therefore should begin with a standard XML declaration. Then you define the W3C namespace for style sheets.

XSLT – eXtensible Style Language for Transformations

When Eclipse creates a new xsl file it automatically places the necessary processing instructions into the file. In the stylesheet element you must declare the XSLT's namespace and version. Both of these attributes are

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/">
        <!-- TODO: Auto-generated template -->
    </xsl:template>
</xsl:stylesheet>
```

There are no spaces in the xsl:stylesheet tag.

This header is almost always the same.

XSLT – eXtensible Style Language for Transformations

Creating the Root Template

The first thing that the XSLT processor looks for in a style sheet is the root template. This is the template that defines the set of rules to apply to the root node of the XML document.

The **match** attribute is used to associate a template with an XML element. The match attribute can also be used to define a template for the entire XML document. The value of the match attribute is an XPath expression (i.e. match="/" defines the whole document).

Specifically, it describes how to process or transform the content from the root node into some new output.

XSLT – eXtensible Style Language for Transformations

The root element is automatically created by Eclipse for you.

```
<xsl:template match="/">

</xsl:template>
```

The XSLT processor doesn't care where the root template appears in the XSLT style sheet, it will be clearest to you if you put it at the very top.

XSLT – eXtensible Style Language for Transformations

Outputting HTML

Once you have the root template created you can define the set of rules for the template. The rules will apply to the content in the root node.

Typically in the root template you will start by creating the structure for the final transformed document. In our case we want to create a HTML document and so we begin by adding HTML header information.

To have your XSLT processor output HTML, you will need to use the xsl:output processing instruction. You set the output method to html, xml, or text. If the instruction is omitted, processors will output XML by default.

XSLT – eXtensible Style Language for Transformations

To add your HTML output you simply include the HTML tags and content that you wish to display on the web page.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output method="html"/>
    <xsl:template match="/">
        <html>
          <head>
            <title>Wonders of the World</title>
          </head>
          <body>
            <p>
              <img src="images/herodotus.jpeg" width="120" height="171"/>
            </p>
            <p> The famous Greek historian Herodotus wrote of seven great archite
                achievements. Although his writings did not survive, he planted s
                for what has become the list of the <strong>Seven Wonders of the
                Ancient World</strong>
            </p>
          </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

XSLT – eXtensible Style Language for Transformations

Although an xml document at this stage was processed, the XSLT processor still hasn't gotten its hands on the XML contents itself. It has only output the HTML tags and text.

Because all XSLT documents are XML documents, they must be well-formed. Consequently the HTML you use in the XSLT document must also be well-formed.