

IT452 Advanced Web and Internet Systems

Set 7: XSLT and XPath (Chapter 5)

XSLT

- XSL – “Extensible Stylesheet Language”
- Parts
 - XSLT – “XSL Transformations”
 - XPath – how to identify a node in an XML document?
- Not primarily used for style
- Not a replacement for CSS

0. Raw RSS example (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://realPrefixRemovedForThisDemo.org#"
  xmlns="http://purl.org/rss/1.0/"
  xmlns:rss="http://purl.org/rss/1.0/" xmlns:cc="http://web.resource.org/cc/"
  xmlns:taxo="http://purl.org/rss/1.0/modules/taxonomy/" xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:syn="http://purl.org/rss/1.0/modules/syndication/" xmlns:admin="http://webns.net/mvcb/"
  xmlns:rae="http://ns.web2.0thebook.org/rssAggregation/">
  <channel rdf:about="http://web2.0thebook.org/channel.rss">
    <title>Planet web2.0thebook</title>
    <link>http://web2.0thebook.org/</link>
    <description>Aggregated content relevant to the upcoming book "Professional Web 2.0
      Programming".</description>
  </channel>
  <item
    rdf:about="http://www.orbeon.com/blog/2006/06/13/firebug-a-must-have-firefox-extension-for-web-developers/">
    <title>XForms Everywhere » FireBug: A Must-Have Firefox Extension for Web Developers</title>
    <link>http://www.orbeon.com/blog/2006/06/13/firebug-a-must-have-firefox-extension-for-web-developers/</link>
    <description>Alessandro Vernet recommends FireBug, â€œan absolute godsendâ€, the â€œgreatest web
      developer extension out thereâ€, an â€œawesomeâ€, â€œphenomenalâ€, and â€œabsolutely, completely
      brilliantâ€ extension.</description>
    <dc:creator>evlist</dc:creator>
    <dc:date>2006-06-15T05:56:16Z</dc:date>
    <dc:subject>javascript dom firefox ajax debugger tools web2.0thebook webdev</dc:subject>
  </item>
  <item rdf:about="http://eric.van-der-vlist.com/blog/2504_Web_2.0_at_XML_Prague.item">
    <title>Web 2.0 at Prague</title>
    <link>http://eric.van-der-vlist.com/blog/2504_Web_2.0_at_XML_Prague.item/</link>
    <description>Eric van der Vlist will do a presentation about Web 2.0 at XML Prague 2006.</description>
    <dc:creator>evlist</dc:creator>
    <dc:date>2006-06-12T12:22:59Z</dc:date>
    <dc:subject>prague qbe query rdf web2.0 web2.0thebook xml</dc:subject>
  ...
  ...
```

1. XML/RSS transformed by XSLT

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="1_rss.xsl" ?>
<rdf:RDF xmlns:rdf="http://realPrefixRemovedForThisDemo.org#" xmlns="http://purl.org/rss/1.0/"
  xmlns:rss="http://purl.org/rss/1.0/" xmlns:cc="http://web.resource.org/cc/"
  xmlns:taxo="http://purl.org/rss/1.0/modules/taxonomy/" xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:syn="http://purl.org/rss/1.0/modules/syndication/" xmlns:admin="http://webns.net/mvcb/"
  xmlns:rae="http://ns.web2.0thebook.org/rssAggregation/">
  <channel rdf:about="http://web2.0thebook.org/channel.rss">
    <title>Planet web2.0thebook</title>
  ...
  ...
```

1. The XSLT (1_rss.xsl)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:rdf="http://realPrefixRemovedForThisDemo.org#"
    xmlns:rss="http://purl.org/rss/1.0/" xmlns="http://www.w3.org/1999/xhtml"
    version="1.0" exclude-result-prefixes="rdf rss">
    <xsl:template match="/">
        <html>
            <head>
                <title> <xsl:value-of select="rdf:RDF/rss:channel/rss:title"/> </title>
            </head>
            <body> <xsl:apply-templates select="rdf:RDF/rss:channel"/> </body>
        </html>
    </xsl:template>
    <xsl:template match="rss:channel">
        <div class="channel" id="planet">
            <xsl:apply-templates select="rss:title"/>
            <xsl:apply-templates select="rss:description"/>
            <xsl:apply-templates select="rss:link"/>
        </div>
    </xsl:template>
    <xsl:template match="rss:title">
        <h1> <xsl:value-of select=". "/> </h1>
    </xsl:template>
    <xsl:template match="rss:description">
        <p class="description"> <xsl:value-of select=". "/> </p>
    </xsl:template>
    <xsl:template match="rss:link">
        <a href=". " >  </a>
    </xsl:template>
</xsl:stylesheet>
```

XPath

```
/  
channel/title  
/channel/title  
item[@title='Jaw surgery']  
item/title[text()='Jaw surgery']  
  
Complex example in use:  
<xsl:template match="rdf:li">  
    <xsl:apply-templates select="/rdf:RDF/rss:item[@rdf:about =  
        current() /@rdf:resource]"/>  
</xsl:template>
```

Can use alone in browser, e.g., XPathEvaluator, XPath Result

2. HTML plus transformed XML (part 1– HTML)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Professional Web 2.0 Programming</title>
    <script type="text/javascript" src="2_transform.js"/>
  </head>
  <body onload="transform('1_rss.xsl')">
    <div id="book">
      <h1>A page with some exciting content (simplified from book example) </h1>

      <p>Web 2.0 offers developers substantial advantages if they design their web
         applications as service providers and service consumers. This change in architecture
         has opened up an incredible number of options for flexible design, creative reuse,
         and easier updates. There is, however a cost: doing this requires rethinking how to
         apply many traditional web development technologies, as well as adding some new
         ideas.</p>
    </div>

    <h2>Below is some content from elsewhere:</h2>

    <div id="planet">
      <h1>This element should be replaced by JS.
          If you see this content, something went wrong with the XSLT!</h1>
    </div>
  </body>
</html>
```

2. HTML plus transformed XML (part 2– JS)

```
function transform (xslFileName) {
  // Get the XML input data
  var xmlhttp = new XMLHttpRequest();
  xmlhttp.open("GET", "channel.xml", false);
  xmlhttp.send('');

  // Get the XSLT file
  var xslhttp = new XMLHttpRequest();
  xslhttp.open("GET", xslFileName, false);
  xslhttp.send('');

  // Transform the XML via the XSLT
  var processor = new XSLTProcessor();
  processor.importStylesheet(xslhttp.responseXML);
  var newDocument = processor.transformToDocument(xmlhttp.responseXML);

  // Replace part of original document with the new content
  var o = document.getElementById("planet");
  var n = newDocument.getElementById("planet");
  o.parentNode.replaceChild(n, o);
}
```

3. HTML plus improved XML (part 1– HTML)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Professional Web 2.0 Programming</title>
    <script type="text/javascript" src="2_transform.js"/>
  </head>
  <body onload="transform('3_rss.xsl')">
    <div id="book">
      <h1>A page with some exciting content (simplified from book example) </h1>

      <p>Web 2.0 offers developers substantial advantages if they design their web
         applications as service providers and service consumers. This change in architecture
         has opened up an incredible number of options for flexible design, creative reuse,
         and easier updates. There is, however a cost: doing this requires rethinking how to
         apply many traditional web development technologies, as well as adding some new
         ideas.</p>
    </div>

    <h2>Below is some content from elsewhere:</h2>

    <div id="planet">
      <h1>This element should be replaced by JS. If you see this content, something went wrong with the
          XSLT!</h1>
    </div>
  </body>
</html>
```

3. HTML plus improved XML (part 2– XSL)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:rdf="http://realPrefixRemovedForThisDemo.org#"
  xmlns:rss="http://purl.org/rss/1.0/"
  xmlns="http://www.w3.org/1999/xhtml" version="1.0" exclude-result-prefixes="rdf rss">
  <xsl:template match="/">
    <html>
      <head>
        <title>
          <xsl:value-of select="rdf:RDF/rss:channel/rss:title"/>
        </title>
      </head>
      <body>
        <xsl:apply-templates select="rdf:RDF/rss:channel"/>
      </body>
    </html>
  </xsl:template>
  <xsl:template match="rss:channel">
    <div class="channel" id="planet">
      <xsl:apply-templates select="rss:title"/>
      <xsl:apply-templates select="rss:description"/>
      <xsl:apply-templates select="rss:link"/>
      <ul>
        <xsl:apply-templates select="/rdf:RDF/rss:item"/>
      </ul>
    </div>
  </xsl:template>
  ...
  ... (more on next page)
```

3. HTML plus improved XML (part 3– XSL cont.)

```
<!-- deal with attributes of <channel> -->
<xsl:template match="rss:channel/rss:title">
    <h1> <xsl:value-of select="."/> </h1>
</xsl:template>

<xsl:template match="rss:link">
    <a href="{.}">  </a>
</xsl:template>

<!-- deal with each <item> -->
<xsl:template match="rss:item">
    <li>
        <div class="item">
            <xsl:apply-templates select="rss:title"/>
            <xsl:apply-templates select="rss:description"/>
        </div>
    </li>
</xsl:template>
<xsl:template match="rss:item/rss:title">
    <h2> <a href="{./rss:link}">
        <xsl:value-of select="."/> </a>
    </a>
    </h2>
</xsl:template>

<!-- deal with one shared attribute -->
<xsl:template match="rss:description">
    <p class="description"> <xsl:value-of select="."/> </p>
</xsl:template>
</xsl:stylesheet>
```

4. HTML plus changed XML (part 1– HTML)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>Professional Web 2.0 Programming</title>
        <script type="text/javascript" src="2_transform.js"/>
    </head>
    <body onload="transform('4_rss.xsl')">
        <div id="book">
            <h1>A page with some exciting content (simplified from book example) </h1>

            <p>Web 2.0 offers developers substantial advantages if they design their web
                applications as service providers and service consumers. This change in architecture
                has opened up an incredible number of options for flexible design, creative reuse,
                and easier updates. There is, however a cost: doing this requires rethinking how to
                apply many traditional web development technologies, as well as adding some new
                ideas.</p>
        </div>
        <h2>Below is some content from elsewhere:</h2>

        <div id="planet">
            <h1>This element should be replaced by JS. If you see this content, something went wrong with the
            XSLT!</h1>
        </div>
    </body>
</html>
```

4. HTML plus changed XML (part 2– XSL)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:rdf="http://realPrefixRemovedForThisDemo.org#"
  xmlns:rss="http://purl.org/rss/1.0/"
  xmlns="http://www.w3.org/1999/xhtml" version="1.0" exclude-result-prefixes="rdf rss">
  <xsl:template match="/">
    <html>
      <head>
        <title>
          <xsl:value-of select="rdf:RDF/rss:channel/rss:title"/>
        </title>
      </head>
      <body>
        <xsl:apply-templates select="rdf:RDF/rss:channel"/>
      </body>
    </html>
  </xsl:template>
  <xsl:template match="rss:channel">
    <div class="channel" id="planet">
      <h1>
        <xsl:apply-templates select="rss:title"/>
        .... Get it!
        <xsl:apply-templates select="rss:link"/>
      </h1>
      <xsl:apply-templates select="rss:description"/>

      <table border='1'>
        <xsl:apply-templates select="/rdf:RDF/rss:item"/>
      </table>
    </div>
  </xsl:template>
```

4. HTML plus changed XML (part 3– XSL cont.)

```
<!-- deal with attributes of <channel> -->
<xsl:template match="rss:channel/rss:title">
  <xsl:value-of select=". />
</xsl:template>
<xsl:template match="rss:link">
  <a href="{.}"/>  </a>
</xsl:template>
<xsl:template match="rss:channel/rss:description">
  <p class="description"> <xsl:value-of select=". /> </p>
</xsl:template>

<!-- deal with each <item> -->
<xsl:template match="rss:item">
  <tr>
    <xsl:apply-templates select="rss:title"/>
    <xsl:apply-templates select="rss:description"/>
  </tr>
</xsl:template>
<xsl:template match="rss:item/rss:title">
  <td>
    <a href="{./rss:link}"> <xsl:value-of select=". /> </a>
  </td>
</xsl:template>
<xsl:template match="rss:item/rss:description">
  <td>
    <xsl:value-of select=". />
  </td>
</xsl:template>
</xsl:stylesheet>
```