PB138/07 - Modern Markup Languages and Their Applications

Lab 03 [09.03.2020]
EXtensible Stylesheet Language Transformations (XSLT)

Bruno Rossi
Department of Computer Systems and Communications,
Lasaris (Lab of Software Architectures and Information Systems)
Masaryk University, Brno

Introduction

- XSL (EXtensible Stylesheet Language) : styling language for XML
- XSLT stands for XSL Transformations
- XSLT uses XPath for matching one or more predefined templates and transforming the matching part in the resulting document
- “Navigational style” (e.g., using `<xsl:for-each…>`) vs “Rule-based style” (using `<xsl:apply-templates…>`)

XSLT Reminder

- Definition of a template

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

- <xsl:apply-templates>
- <xsl:value-of>

```
<xsl:value-of select="continent/cities/city/name"/>
```

- <xsl:for-each>

```
<xsl:for-each select="continent/cities/city">
```

- <xsl:sort> (in a for-each)

```
<xsl:sort select="name"/>
```

- <xsl:if> (note that you can also use xpath filtering conditions)

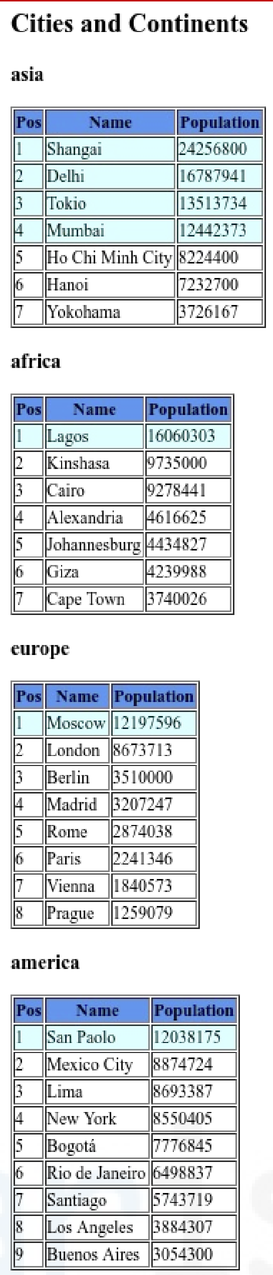
```
<xsl:if test="population &gt; 100000">
```

- <xsl:choose> <xsl:when test="expression"> ... <xsl:otherwise> ...

Example - “navigational” style

- Using the continent.xml file used previously

```
<?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>
      <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
      <title>Continents</title>
   </head>
  <body>
  <h2>Cities and Continents</h2>
    <xsl:for-each select="world/continent">
  <h3><xsl:value-of select="@name"/></h3>
  <table border="2">
    <tr bgcolor="#6495ed">
      <th>Pos</th>
      <th>Name</th>
      <th>Population</th>
    </tr>
    <xsl:for-each select="cities/city">
    <xsl:sort select="population" order="descending" data-type="number"/>
    <tr>
       <xsl:attribute name="style">
                  <xsl:choose>
                    <xsl:when test="population > 12000000">
                      <xsl:text>background: LightCyan;</xsl:text>
                    </xsl:when>
                  </xsl:choose>
        </xsl:attribute>
      <td><xsl:value-of select="position()"/></td>
      <td><xsl:value-of select="name"/></td>
      <td><xsl:value-of select="population"/></td>
    </tr>
    </xsl:for-each>
  </table>
    </xsl:for-each>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```



Cities and Continents

asia

Pos	Name	Population
1	Shangai	24256800
2	Delhi	16787941
3	Tokio	13513734
4	Mumbai	12442373
5	Ho Chi Minh City	8224400
6	Hanoi	7232700
7	Yokohama	3726167

africa

Pos	Name	Population
1	Lagos	16060303
2	Kinshasa	9735000
3	Cairo	9278441
4	Alexandria	4616625
5	Johannesburg	4434827
6	Giza	4239988
7	Cape Town	3740026

europe

Pos	Name	Population
1	Moscow	12197596
2	London	8673713
3	Berlin	3510000
4	Madrid	3207247
5	Rome	2874038
6	Paris	2241346
7	Vienna	1840573
8	Prague	1259079

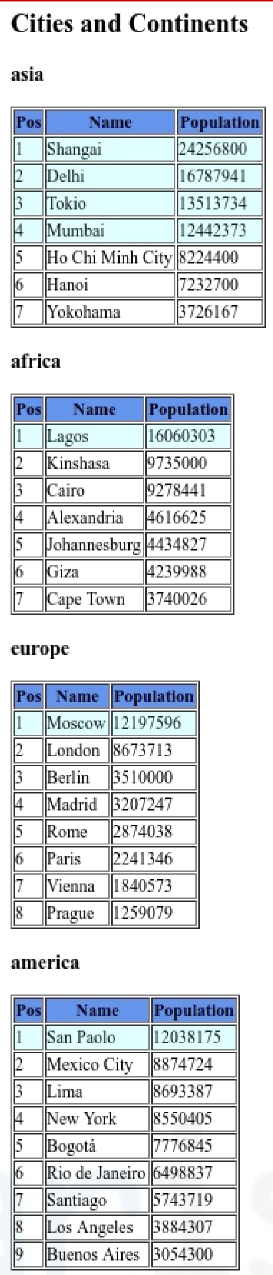
america

Pos	Name	Population
1	San Paolo	12038175
2	Mexico City	8874724
3	Lima	8693387
4	New York	8550405
5	Bogotá	7776845
6	Rio de Janeiro	6498837
7	Santiago	5743719
8	Los Angeles	3884307
9	Buenos Aires	3054300

Example - “rule-based” style

- Using the continent.xml file used previously

```
<?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="world">
  <html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
      <title>Continents</title>
   </head>
  <body>
  <h2>Cities and Continents</h2>
    <xsl:apply-templates select="continent" />
   </body>
  </html>
</xsl:template>

<xsl:template match="continent">
<h3><xsl:value-of select="@name"/></h3>
<table border="2">
    <tr bgcolor="#6495ed">
      <th>Pos</th>
      <th>Name</th>
      <th>Population</th>
    </tr>
    <xsl:apply-templates select="cities/city">
            <xsl:sort select="population" order="descending" data-type="number"/>
    </xsl:apply-templates>
   </table>
</xsl:template>
<xsl:template match="cities/city">
    <tr>
       <xsl:attribute name="style">
                  <xsl:choose>
                    <xsl:when test="population > 12000000">
                      <xsl:text>background: LightCyan;</xsl:text>
                    </xsl:when>
                  </xsl:choose>
     </xsl:attribute>
      <td><xsl:value-of select="position()"/></td>
      <td><xsl:value-of select="name"/></td>
      <td><xsl:value-of select="population"/></td>
    </tr>
</xsl:template>

</xsl:stylesheet>
```



Cities and Continents

asia

Pos	Name	Population
1	Shangai	24256800
2	Delhi	16787941
3	Tokio	13513734
4	Mumbai	12442373
5	Ho Chi Minh City	8224400
6	Hanoi	7232700
7	Yokohama	3726167

africa

Pos	Name	Population
1	Lagos	16060303
2	Kinshasa	9735000
3	Cairo	9278441
4	Alexandria	4616625
5	Johannesburg	4434827
6	Giza	4239988
7	Cape Town	3740026

europe

Pos	Name	Population
1	Moscow	12197596
2	London	8673713
3	Berlin	3510000
4	Madrid	3207247
5	Rome	2874038
6	Paris	2241346
7	Vienna	1840573
8	Prague	1259079

america

Pos	Name	Population
1	San Paolo	12038175
2	Mexico City	8874724
3	Lima	8693387
4	New York	8550405
5	Bogotá	7776845
6	Rio de Janeiro	6498837
7	Santiago	5743719
8	Los Angeles	3884307
9	Buenos Aires	3054300

XSL Transformation

`continents.xml + continents.xsl → continents.html`

- Some ways to run the transformation for the next tasks

- Netbeans: right click on the xml file → XSL transformation...
- Java: using javax.xml.transform.* classes
- xsltproc - `xsltproc continents.xsl continents.xml > continents.html`
- Online converters, like:
  - https://xslttest.appspot.com
  - https://www.freeformatter.com/xsl-transformer.html

Task 01 (week03)

- Download the project **`seminars/xslt2-zadani.zip`** and uncompress locally in a directory

- The goal of the task is to complete file **`transf.xsl`** so that the transformations run on **`data.xml`** give as output the file **`data.html`** (that you can use as example of the expected result)

  Using the command line *(or the implementation in the project or methods mentioned in the previous slide)*:

  ```
  xsltproc transf.xsl data.xml > out.html
  ```

  (**`out.html`** can then be compared with **`data.html`**)

- Upload the final XSL file to homework vault **`sem07/task01-week03/`**

References

Suggested material:

- XSLT Specifications
  →https://www.w3.org/TR/xslt
- W3C School XSLT pages:
  →https://www.w3schools.com/xml/xsl_intro.asp
- XSLT Tutorial
  → http://zvon.org/comp/r/tut-XSLT_1.html
- XSLT Elements
  → https://www.w3schools.com/xml/xsl_elementref.asp
- Online XSLT Quick Card
  →https://www.cheatography.com/univer/cheat-sheets/xslt-2-0-cheat-sheet/pdf/