**Magazine.xsl:**

Schreiben Sie ein XSLT Stylesheet, das die report.xml Datei transformiert. Die gewünschte Output Datei wird im Beispiel angegeben.

\_ Hint:Programmierung \_

Für XSLT Suchmuster in Templateregeln sind Sie für Knotenabfragen nicht auf Elementvergleiche beschränkt!

Verwenden Sie falls sinnvoll auch Knotentypentests.

Sie können mehrere Templateregel mit demselben Suchmuster definierten. Definieren Sie in so einem Fall das mode Attribut.

5 <magazine>

6 <report>

7 <head>

8 <authors>

9 <author>ArthurD.Dent</author>

10 <author>FordPerfect</author>

11 <author>J.J.R.Tolkien</author>

12 </authors>

13 <pages>23</pages>

14 <url>...</url>

15 </head>

16 <body>

17 <authors>

18 Thisreportisco-authoredby

19 ArthurD.DentandFordPerfect,

20 sampledbyJ.J.R.Tolkin.

21 </authors>

22 <content>

23 Simcenterisaunified,scalable,open

24 ...

25 </content>

26 </body/>

27 </report>

28 </magazine>

Lösung:

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">  
  
 <xsl:output method="xml" indent="yes"/>  
  
 <xsl:template match="/">  
 <magazine>  
 <xsl:apply-templates select="report"/>  
 </magazine>  
 </xsl:template>  
  
 <xsl:template match="report">  
 <report>  
 <head>  
 <xsl:for-each select="authors/\*">  
 <author>  
 <xsl:if test="./name">  
 <xsl:value-of select="./name"/>  
 </xsl:if>  
 <xsl:value-of select="concat(./first-name/text(), ' ', ./middle-name/text(), ' ', ./last-name/text())"/>  
 </author>  
 </xsl:for-each>  
 <pages>  
 <xsl:value-of select="description/pages"/>  
 </pages>  
 <url>  
 <xsl:value-of select="description/homepage"/>  
 </url>  
 </head>  
 <body>  
 <authors>  
 <xsl:for-each select="description/authors/node()">  
 <xsl:choose>  
 <xsl:when test="self::aref">  
 <xsl:variable name="id">  
 <xsl:value-of select="@id"/>  
 </xsl:variable>  
 <xsl:variable name="author" select="/report/authors/\*[@id = $id]"/>  
 <!--xsl:value-of select="$author"/-->  
 <xsl:if test="$author/name">  
 <xsl:value-of select="$author/name"/>  
 </xsl:if>  
 <xsl:value-of  
 select="concat($author/first-name, ' ', $author/middle-name, ' ', $author/last-name)"/>  
 </xsl:when>  
 <xsl:otherwise>  
 <xsl:value-of select="."/>  
 </xsl:otherwise>  
 </xsl:choose>  
 </xsl:for-each>  
 </authors>  
 <content>  
 <xsl:value-of select="content"/>  
 </content>  
 </body>  
 </report>  
 </xsl:template>  
  
</xsl:stylesheet>

**Familytree.xsl:**

Schreiben Sie ein XSLT Stylesheet, das die family.xml Datei transformiert. Die gewünschte Output Datei wird im Beispiel angegeben.

5 <familytree>

6 <person>

7 <name>JohnWashington</name>

8 <child>

9 <person>

10 <name>LawrenceWashington</name>

11 <child>

12 <person>

13 <name>AugustWashington</name>

14 <child>

15 <person>

16 <name>

17 GeorgeWashington

18 </child>

19 </name>

20 </person>

21 </child>

22 </person>

23 <child>

24 </person>

25 </child>

26 </person>

27 <person>

28 <name>AnnePope</name>

29 <child>

30 <person>

31 <name>LawrenceWashington</name>

32 <child>

33 <person>

34 <name>AugustWashington</name>

35 <child>

36 <person>

37 <name>

38 GeorgeWashington

39 </child>

40 </name>

41 </person>

42 </child>

43 </person>

44 <child>

45 </person>

46 </child>

47 </person>

48 </familytree>

Lösung:

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">  
  
 <xsl:output method="xml" indent="yes"/>  
  
 <xsl:template match="/">  
 <familytree>  
 <xsl:apply-templates select="//descendant::name">  
 <xsl:sort select="count(ancestor::\*)" data-type="number" order="descending"/>  
 </xsl:apply-templates>  
 </familytree>  
 </xsl:template>  
  
 <xsl:template match="name">  
 <xsl:if test="position() = 1">  
 <person>  
 <name>  
 <xsl:value-of select="text()"/>  
 </name>  
 <child>  
 <xsl:apply-templates select="parent::person/parent::parents/preceding-sibling::name"/>  
 </child>  
 </person>  
 </xsl:if>  
 <xsl:if test="position() = 2">  
 <person>  
 <name>  
 <xsl:value-of select="text()"/>  
 </name>  
 <child>  
 <xsl:apply-templates select="parent::person/parent::parents/preceding-sibling::name"/>  
 </child>  
 </person>  
 </xsl:if>  
 </xsl:template>  
</xsl:stylesheet>