## Задание за решения на задачи по Упражнение - Модул 5: XSLT

Изготвил: Марина Дабова, ФН: 62503, група 3, специалност СИ

Коментарите са обозначени с червен цвят

Задача 1: За дадения по-долу XML документ създайте XSL документ, който чрез <xsl:if>, извежда в HTML формат:

1. Стойностите на всички track елементи, които имат стойност на атрибута length '4:04'
2. Стойностите на всички track елементи, чиято дължина е по-малка от 15, както и техните дължини
3. Всички track елементи на четни/нечетни позиции

**Решение:**

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/"> <!--Чрез "xsl:template" правим темплейт като чрез xsl:template match="/"> избираме целия входен документ-->

<html>

<body>

<tr> <!--Команда за ред -->

<td>1.Стойностите на всички track елементи, които имат стойност на атрибута length '4:04'

<xsl:for-each select=".//track"> <!-- Обхождаме всички track елементи-->

<xsl:if test="@length='4:04'"> <!-- Проверяваме условието- дали са със стойност на атрибута length '4:04' -->

<li><xsl:value-of select="text()"/> </li> <!-- Ако е изпълнено изкарваме съдържанието на тези елементи(само на тях) като използваме "value-of".По този начин ще изакарме резултата -->

</xsl:if>

</xsl:for-each>

</td>

</tr>

<tr>

<td>2.Стойностите на всички track елементи, чиято дължина е по-малка от 15, както и техните дължини

<xsl:for-each select=".//track"> <!-- Обхождаме всички track елементи-->

<xsl:if test="string-length(.) &lt; 15"> <!-- Проверяваме условието -дали са с дължина по-малка от 15. Като тук трябва да използваме &lt; , за да представим знака "<", защото иначе ще даде грешка -->

<ls><xsl:value-of select="text()"/> <!-- Ако е изпълнено изкарваме съдържанието на тези елементи(само на тях) като използваме "value-of".По този начин ще изакарме резултата -->

<td></td>

<xsl:value-of select="string-length(.) "/></ls> <!--Тук изкарваме дължината -->

</xsl:if>

</xsl:for-each>

</td>

</tr>

<tr>

<td>3.Всички track елементи на четни позиции

<xsl:for-each select=".//track"> <!-- Обхождаме всички track елементи-->

<xsl:if test="position() mod 2 = 0"> <!-- Проверяваме условието- дали е четен елемента -->

<li><xsl:value-of select="text()"/> </li> <!-- Ако е изпълнено изкарваме съдържанието на тези елементи(само на тях) като използваме "value-of".По този начин ще изакарме резултата -->

</xsl:if>

</xsl:for-each>

</td>

</tr>

<tr>

<td>3.Всички track елементи на нечетни позиции

<xsl:for-each select="//track"> <!-- Обхождаме всички track елементи-->

<xsl:if test="position() mod 2 != 0"> <!-- Проверяваме условието- дали е нечетен елемента -->

<li><xsl:value-of select="text()"/> </li> <!-- Ако е изпълнено изкарваме съдържанието на тези елементи(само на тях) като използваме "value-of".По този начин ще изакарме резултата -->

</xsl:if>

</xsl:for-each>

</td>

</tr>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

#### Задача 2: За дадения в задача 1 XML документ, създайте XSL документ, който чрез <xsl:choose> за всеки елемент track, извежда в HTML формат:

1. "A big string" ако дължината на стойността на track елемента е по-голяма от 15

2. "A small string" ако дължината на стойността на track елемента е по-малка от 15

3. "A medium string" ако дължината на стойността на track не отговаря на нито едно от горните условия

**Решение:**

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >

<xsl:template match="/"> <!--Чрез "xsl:template" правим темплейт като чрез xsl:template match="/"> избираме целия входен документ-->

<html>

<body>

<table>

<tr>

<td>

<xsl:for-each select=".//track"> <!-- Обхождаме всички track елементи-->

<xsl:choose> <!-- Чрез "xsl:choose" задаваме многократни условни тестове-->

<xsl:when test="string-length(.) &gt; 15"> <!-- Ако дължината е по-голяма от 15-->

<td>A big string: </td> <!-- изкарваме "A big string"-->

<td><xsl:value-of select="text()"/> </td> <!--и за да сверим извеждам и съдържанието -->

<td><xsl:value-of select="string-length()"/> </td> <!-- и дължината -->

</xsl:when>

<xsl:when test="string-length(.) &lt; 15"> <!-- Ако дължината е по-малка от 15-->

<td>A small string:</td> <!-- изкарваме "A small string "-->

<td><xsl:value-of select="text()"/></td> <!--и за да сверим извеждам и съдържанието -->

<td><xsl:value-of select="string-length()"/> </td> <!-- и дължината -->

</xsl:when>

<xsl:otherwise> <!-- Иначе, ако не са изпълнени горните условия:-->

<td>A medium string:</td> <!-- изкарваме "A medium string "-->

<td><xsl:value-of select="text()"/></td> <!--и за да сверим извеждам и съдържанието -->

<td><xsl:value-of select="string-length()"/> </td><!-- и дължината -->

</xsl:otherwise>

</xsl:choose>

</xsl:for-each>

</td>

</tr>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

Задача 3: За дадения в задача 1 XML документ, създайте XSL документ, който чрез <xsl:sort> сортира стойностите на track елементите:

1. В нарастващ ред

2. В намаляващ ред

3. В нарастващ или намаляващ ред в следния формат: track\_1, track\_2,...,track\_n

**Решение:**

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >

<xsl:template match="/"> <!--Чрез "xsl:template" правим темплейт като чрез xsl:template match="/"> избираме целия входен документ-->

<html>

<body>

<table>

<tr> <!--Команда за ред -->

<td> 1. В нарастващ ред <!--Команда за td за колона -->

<ol> <!--Команда ol за подреден списък -->

<xsl:for-each select=".//track"> <!-- Обхождаме всички track елементи-->

<xsl:sort select="."/> <!-- Сортираме като не посочваме ред на сортиране - в случая възходящ,защото той е по подразбиране-->

<li><xsl:value-of select="text()"/></li>

</xsl:for-each>

</ol>

</td>

</tr>

<tr>

<td> 2. В намаляващ ред

<ol> <!--Команда ol за подреден списък -->

<xsl:for-each select=".//track"> <!-- Обхождаме всички track елементи-->

<xsl:sort select="." order="descending"/> <!-- Посочваме ред на сортиране (възходящ или низходящ)- в случая низходящ-->

<li><xsl:value-of select="text()"/></li>

</xsl:for-each>

</ol>

</td>

</tr>

<tr>

<td> 3.1 В намаляващ ред във формат:track\_1, track\_2,...,track\_n

<!--Тук няма команда ol за подреден списък,защото се иска в друг формат, а именно като свързан текст -->

<xsl:for-each select=".//track"> <!-- Обхождаме всички track елементи-->

<xsl:sort select="." order="descending"/> <!-- Посочваме ред на сортиране - в случая низходящ-->

<xsl:value-of select="text()"/>\_<xsl:value-of select="position()"/> <!-- Изкарваме в дадения формат-->

<xsl:if test="position() != last()"> <!-- Проверяваме дали текущия елемент е последен и ако не е изписваме ","-->

<xsl:text>, </xsl:text>

</xsl:if>

</xsl:for-each>

</td>

</tr>

<tr>

<td> 3.2 В нарастващ ред във формат:track\_1, track\_2,...,track\_n

<!--Тук няма команда ol за подреден списък,защото се иска в друг формат, а именно като свързан текст -->

<xsl:for-each select=".//track"> <!-- Обхождаме всички track елементи-->

<xsl:sort select="." /> <!-- Посочваме ред на сортиране - в случая възходящ, той е по подразбиране-->

<xsl:value-of select="text()"/>\_<xsl:value-of select="position()"/> <!-- Изкарваме в дадения формат-->

<xsl:if test="position() != last()"> <!-- Проверяваме дали текущия елемент е последен и ако не е изписваме ","-->

<xsl:text>, </xsl:text>

</xsl:if>

</xsl:for-each>

</td>

</tr>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

#### Задача 4: За XSL документа, от задача 3, дефинирайте константа като използвате <xsl:variable> и изведете стойността ѝ

**Решение:**

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/"> <!--Чрез "xsl:template" правим темплейт като чрез xsl:template match="/"> избираме целия входен документ-->

<html>

<body>

<table>

<tr> <!--Команда за ред -->

<td> <!--Команда за колона -->

<xsl:variable name="title"> Title </xsl:variable> <!--Декларираме константата-задава се име и съдържание -->

<xsl:for-each select=".//track"> <!-- Обхождаме всички track елементи-->

<xsl:sort select="text()" order="descending"/> <!-- Сортираме като не посочваме ред на сортиране - в случая низходящ-->

<tr>

<td>

<xsl:value-of select="$title"/>

<!--Тук използваме константата като в select се задава името на константата с "$" отпред-->

</td>

</tr>

<tr>

<td>

<xsl:value-of select="."/>

</td>

</tr>

</xsl:for-each>

</td>

</tr>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

#### Задача 5: За XSL документа, от задача 3, дефинирайте параметър като използвате <xsl:param> и изведете стойността му

**Решение:**

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/"> <!--Чрез "xsl:template" правим темплейт като чрез xsl:template match="/"> избираме целия входен документ-->

<xsl:param name="title"> title</xsl:param> <!--Декларираме параметър-задава се име и съдържание като трябва да се гарантира горе, а не вътре, защото иначе ще даде грешка -->

<html>

<body>

<table>

<tr>

<td>

<xsl:for-each select=".//track"> <!-- Обхождаме всички track елементи-->

<xsl:sort select="." /> <!-- Сортираме като не посочваме ред на сортиране - в случая възходящ-->

<tr>

<td>

<xsl:value-of select="$title"/> <!--Тук използваме параметъра като в select се задава името на параметъра с "$" отпред-->

</td>

</tr>

<tr>

<td>

<xsl:value-of select="."/>

</td>

</tr>

</xsl:for-each>

</td>

</tr>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

Задача 6: За XSL документа, от задача 3, като използвате <xsl:call-template>, дефинирайте и извикайте следните шаблони:

* 1. Шаблон с име year, който извежда стойността на year елемента
* 2. Шаблон с име title, който извежда стойността на title елемента
* 3. Шаблон с име artist, който извежда стойността на artist елемента

**Решение:**

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >

<xsl:template match="/"> <!--Чрез "xsl:template" правим темплейт като чрез xsl:template match="/"> избираме целия входен документ-->

<html>

<body>

<table> <!—Искам да направя табличка, в която ще имам 3 колони -->

<tr> <!-- Задавам ред, в този ред искам да имам 3 колони, задаващи съответно-годината, името и артиста-->

<th>Year</th> <!-- Задавам си всяка колона-->

<th>Title</th>

<th>Artist</th>

</tr>

<xsl:for-each select="catalog/cd"> <!-- Обхождаме cd елементите-->

<tr> <!-- Тук също задаваме ред на всяко обхождане, като в него искам да имам отново 3 колони-->

<td><xsl:call-template name="year"/></td> <!--Чрез "xsl:call-template" извикваме шаблона. Вика се по име като се декларира под документа -->

<td> <xsl:call-template name="title"/> </td> <!--Чрез "xsl:call-template" извикваме шаблона. Вика се по име -->

<td> <xsl:call-template name="artist"/> </td> <!--Чрез "xsl:call-template" извикваме шаблона. Вика се по име -->

</tr>

</xsl:for-each>

</table>

</body>

</html>

</xsl:template>

<!--Декларираме шаблона -->

<xsl:template name="year">

<xsl:value-of select="year"/> <!--Извеждаме елементното съдържание на "year" -->

</xsl:template>

<!--Декларираме шаблона -->

<xsl:template name="title">

<xsl:value-of select="title"/> <!--Извеждаме елементното съдържание на "title" -->

</xsl:template>

<!--Декларираме шаблона -->

<xsl:template name="artist">

<xsl:value-of select="artist"/> <!--Извеждаме елементното съдържание на "artist" -->

</xsl:template>

</xsl:stylesheet>

Задача 7: За XSL документа, от задача 3, като използвате <xsl:call-template> и <xsl:with-param>, дефинирайте и извикайте следните шаблони с параметри:

1. Шаблон с име year, който има параматър с име yearName и извежда стойността на year елемента

2. Шаблон с име title, който има параматър с име titleName и извежда стойността на title елемента

3. Шаблон с име artist, който има параматър с име artistName и извежда стойността на artist елемента

**Решение:**

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >

<xsl:template match="/"> <!--Чрез "xsl:template" правим темплейт като чрез xsl:template match="/"> избираме целия входен документ-->

<html>

<body>

<table>

<tr> <!-- Задавам ред, в този ред искам да имам 3 колони, задаващи съответно-годината, името и артиста-->

<th>Year</th> <!-- Задавам си всяка колона-->

<th>Title</th>

<th>Artist</th>

</tr>

<xsl:for-each select="catalog/cd"> <!-- Обхождаме cd елементите-->

<tr> <!-- Тук също задаваме ред на всяко обхождане, като в него искам да имам отново 3 колони-->

<!-- За всеки "year","title","artist" викаме съответните долу дефинирани шаблони -->

<td>

<xsl:call-template name="yearTemplate"> <!-- Чрез "xsl:call-template" викаме съответния долу дефиниран шаблон -->

<xsl:with-param name="yearName" select="year"/> <!--Подаваме стойността чрез with-param -подават се името и се избира елемента -->

</xsl:call-template>

</td>

<td>

<xsl:call-template name="titleTemplate"> <!-- Чрез "xsl:call-template" викаме съответния долу дефиниран шаблон -->

<xsl:with-param name="titleName" select="title"/> <!--Подаваме стойността чрез with-param -подават се името и се избира елемента -->

</xsl:call-template>

</td>

<td>

<xsl:call-template name="artistTemplate"> <!-- Чрез "xsl:call-template" викаме съответния долу дефиниран шаблон -->

<xsl:with-param name="artistName" select="artist"/> <!--Подаваме стойността чрез with-param -подават се името и се избира елемента -->

</xsl:call-template>

</td>

</tr>

</xsl:for-each>

</table>

</body>

</html>

</xsl:template>

<!--Декларираме шаблона -->

<xsl:template name="yearTemplate"> <!--Декларираме шаблон със съответното име в случая yearTemplate -->

<xsl:param name="yearName"/> <!-- В шаблона дефинираме променлива-параметър в случая yearName.Тя очаква като се извика шаблона да се подаде стойност-->

<xsl:value-of select="$yearName"/> <!--Извеждаме присвоената на параметъра стойност -->

</xsl:template>

<!--Декларираме шаблона -->

<xsl:template name="titleTemplate"> <!--Декларираме шаблон със съответното име в случая titleTemplate -->

<xsl:param name="titleName"/> <!-- В шаблона дефинираме променлива-параметър в случая titleName.Тя очаква като се извика шаблона да се подаде стойност-->

<xsl:value-of select="$titleName"/> <!--Извеждаме присвоената на параметъра стойност -->

</xsl:template>

<!--Декларираме шаблона -->

<xsl:template name="artistTemplate"> <!--Декларираме шаблон със съответното име в случая artistTemplate -->

<xsl:param name="artistName"/> <!-- В шаблона дефинираме променлива-параметър в случая artistName.Тя очаква като се извика шаблона да се подаде стойност-->

<xsl:value-of select="$artistName"/> <!--Извеждаме присвоената на параметъра стойност -->

</xsl:template>

</xsl:stylesheet>