ЛАБОРАТОРНАЯ РАБОТА 13

Тема: Форматирование вывода XML-документа

*Цель работы:*формирование практических умений исполь­зования XSL для форматирования вывода XML-документа.

**Теоретические сведения**

Расширяемый язык таблиц стилей (XSL) основан на XML и предназначен для трансформации документа XML в другой до­кумент XML или отображения в виде HTML. Язык XSL основан на каскадных таблицах стилей CSS (Сas-cading Style Sheets) и на языке DSSSL (Document Style Semantics and Specification Language – язык спецификации и семантики стиля документа). По мере развития язык XSL становится все ближе к CSS и дальше от DSSSL. Документ XSL (файл с расширением *.xsl*) фактически пред­ставляет собой документ XML, который может содержать теги HTML.XSL-таблица стилей является мощным и гибким инстру­ментом для отображения XML-документов. Используя XSL-таб­лицы стилей, можно не только задать формат для каждого эле­мента XML, как при использовании CSS, но и обеспечить сред­ства контроля над выводимыми данными: выбрать для отображения конкретные данные XML, представить эти данные в любом порядке, модифицировать или добавлять информацию.

Вместе с тем, технология XSLT является более сложной для понимания, чем CSS. Кроме того, это новая технология, следовательно, имеет меньшую степень поддержки, то есть меньшую степень унификации среди современных браузеров. Документ XSL может быть создан в любом текстовом ре­дакторе.

Корневым элементом таблицы стилей является элемент *<xsl: stylesheet>*:*<xsl: stylesheet version="1.0"xmlns:xsl="http://www.w3.org/1999/XSL/Transform">* Первым атрибутом этого элемента является *version*, опреде­ляющий версию XSL, вторым – атрибут *xmlns:xsl*, определяю­щий пространство имен.

Элемент *xsl:stylesheet* должен содержать один или несколько шаблонов элементов, создаваемых с использованием элемента*<xsl:template>*.*<xsl:template match="/"><!--дочерниеэлементы... --> </xsl:template>*Браузер использует шаблон для отображения определенной ветви элементов в иерархии XML-документа, с которым связана таблица стилей. Атрибут *match* шаблона указывает на опреде­ленную ветвь (он аналогичен селектору в правиле CSS). Значе­ние атрибута *match* носит название образца (pattern). Образец в данном примере ("/") представляет корневой элемент всего XML-документа. Таким образом, этот шаблон содержит инст­рукции для отображения всего XML-документа.

Для того чтобы связать таблицу стилей XSL с XML-доку­ментом, используется инструкция *xml-stylesheet*, которая имеет следующую обобщенную форму записи:*<?xml-stylesheet type="text/xsl" href="путь к файлу XSL"?>*В качестве значения атрибута *href* может использоваться абсолютный (URL полностью) или относительный путь.

Пример указания абсолютного пути: *<?xml-stytesheet type="text/xsl"href="http:/www.my\_domain.com/lnventory.xsl"?>* Пример указания относительного пути (при условии, что *lnventory.xsl* находится в той же папке, что и текущий XML-до­кумент):*<?xml-stylesheet type="text/xsl" href="lnventory.xsl"?>*Обычно инструкция *xml-stylesheet*добавляется в пролог XML-документа вслед за объявлением XML (например, *<?xml version="1.0" encoding="Windows-1251"?>*).Если XML-документ связан с таблицей стилей XSL, то при его открытии в браузере отобразится содержимое, оформленное в соответствии с инструкциями по преобразованию, содержащимися в таблице стилей. В отличие от CSS, если XML-документ связан с несколькими таблицами стилей XSL, браузер использует первую таблицу и игнорирует все остальные. Если вы свяжете с XML-документом и CSS-таблицу, и XSL-таблицу стилей, браузер использует только XSL-таблицу стилей. Если XML-документ не связан ни с CSS-таблицей, ни с XSL-таблицей стилей, браузер отобразит документ с помощью встроенной XSL-таблицы, которая используется по умолчанию. Эта таблица стилей отображает исходный XML-текст в виде де­рева с возможностью свертывания/развертывания уровней.

**Основные дескрипторы XSLдля преобразования XML- документа в HTML**

Шаблон ***<****xsl:template****>*** содержит два вида элементов:

- элементы, представляющие собой HTML-разметку.

На­пример:*<H2>Заголовок</H2>*

- XSL-элементы.

Например:*<xsl:value-of select="/books/book/author"/>*

Браузер отличает XML-элемент от элемента, представляю­щего HTML, поскольку первый имеет в качестве префикса опи­сание пространства имен *xsl*:. XSL-элементы в шаблоне не копи­руются на выход HTML. Они лишь содержат инструкции по вы­бору и модификации данных XML либо используются для вы­полнения других задач.XSL-элемент *value-of* добавляет текстовое содержимое оп­ределенного XML-элемента и его дочерних элементов в выход­ной модуль HTML.

Конкретный элемент задается атрибутом *select* XSL-элемента *value-of*.Адресация к XML-данным строится на выражениях языка xPATH. Самые простые выражения xPATH похожи на пути в файловых системах. Путь в файловой системе состоит из цепо­чек директорий, разделенных слэшем. xPATH-пути выглядят аналогичным образом:*"/"*– путь к корневому узлу;*" /books"*– путь к элементу *books*;*"/books/book"*– путь к элементу *book*, содержащему инфор­мацию о книге;*"/books/book/author"*– путь к элементу *author*, содержащему фамилию и имя автора книги. Так же как и в файловых системах, в xPATH есть понятия «относительного пути» и «абсолютного пути».

Абсолютный путь отсчи­тывается от корневого узла и начинается с одинарного слэша "/" как в примерах выше. Относительный путь отсчитывается от текущего положения, определяемого значением атрии-бута *match* элемента <*xsl:template*>.Для того чтобы добавить в выходной модуль HTML значе­ние атрибута, используется символ *@*. Например, для вывода значения атрибута *id* тега <*book id="Book1*"> может использоваться следующая запись:*<xsl:value-of select="/books/book/author/@id"/>*Чтобы вывести содержимое элемента, включающее HTML-те-ги, используется конструкция *<xsl:copy-of select="название эле-мента****/****\*"****/>***. Значение атрибута *select*предполагает выбор всех дочерних узлов (элементов, атрибутов, текста, комментариев) конкретного элемента. В XSL предусмотрены два условных оператора:*xsl:if* и *xsl:choose*.

Рассмотрим использование оператора *xsl:if*. Например, чтобы проверить, издана ли книга позднее 1990 года из XML-документа (пример 1.1 в описании лабораторной работы 1) может использоваться конструкция*<xsl:if test="year\_publication > 1990"> Действия при вы­полнении условия</xsl:if>*Если нужно проверить обратное, то есть, что книга издана ранее 1990 года, нужно использовать не символ <, а его *amp*-последовательность (*&lt;*):*<xsl:if test="year\_publication &lt; 1990"> Действия при выполнении условия</xsl:if>*Так как XML-документ чаще всего содержит не одну запись, а набор записей одинаковой структуры, требуется использование специального оператора для их перебора. Таким оператором является *xsl:for-each*.

Рассмотрим пример вывода XML-документа из примера 1.1 в виде таблицы на HTML-страницу.

**Пример 4.1(***table.xsl***)**

*<?xml version="1.0" encoding="WINDOWS-1251" ?>*

*<xsl:stylesheet*

*xmlns:xsl="http://www.w3.org/1999/XSL/Transform">*

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

*<html>*

*<head>*

*</head>*

*<body>*

*<table border="1">*

*<tr bgcolor="#CCCCCC">*

*<td align="center"><strong>Автор</strong></td>*

*<td align="center"><strong>Название</strong></td>*

*<td align="center"><strong>Дата</strong></td>*

*</tr>*

*<tr>*

*<td align="center" colspan="3">*

*<strong>Книги</strong></td>*

*</tr>*

*<xsl:for-each select="documents/books/book">*

*<tr bgcolor="#F5F5F5">*

*<td><xsl:value-of select="author"/></td>*

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

*<td><xsl:value-of select="translate"/></td>*

*<td><xsl:value-of select="year\_publication"/></td>*

*</tr>*

*</xsl:for-each>*

*<tr>*

*<td align="center" colspan="3">*

*<strong>Журналы</strong></td>*

*</tr>*

*<xsl:for-each select="documents/articles/article">*

*<tr bgcolor="#F5F5F5">*

*<td><xsl:value-of select="author"/></td>*

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

*<td><xsl:value-of select="year\_publication"/></td>*

*</tr>*

*</xsl:for-each>*

*</table>*

*</body>*

*</html>*

*</xsl:template>*

*</xsl:stylesheet>*

Здесь оператор *xsl:for-each*организует перебор всех элементов *book*и *article*. Далее все, что находится внутри этих элементов, адресуется относительно текущего элемента, заданного атрибутом *select*инструкции *xsl:for-each*.

Конструкция *xsl:sort*служит для сортировки XML-элементов*.*Например, чтобы сортировать книги по дате рождения автора, нужно модифицировать пример 4.1 следующим образом:

**Пример 4.2**(*table\_sort.xsl*).

*<?xml version="1.0" encoding="WINDOWS-1251" ?>*

*<xsl:stylesheet*

*xmlns:xsl=" http://www.w3.org/1999/XSL/Transform ">*

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

*<html>*

*<head>*

*</head>*

*<body>*

*<table border="1">*

*<tr bgcolor="#CCCCCC">*

*<td align="center"><strong>Автор</strong></td>*

*<td align="center"><strong>Название</strong></td>*

*<td align="center"><strong>Дата</strong></td>*

*</tr>*

*<tr>*

*<td align="center" colspan="3">*

*<strong>Книги</strong></td>*

*</tr>*

*<xsl:for-each select="documents/books/book">*

*<xsl:sort select=" year\_publication"/>*

*<tr bgcolor="#F5F5F5">*

*<td><xsl:value-of select="author"/></td>*

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

*<td><xsl:value-of select="translate"/></td>*

*<td><xsl:value-of select="year\_publication"/></td>*

*</tr>*

*</xsl:for-each>*

*<tr>*

*<td align="center" colspan="3">*

*<strong>Журналы</strong></td>*

*</tr>*

*<xsl:for-each select="documents/articles/article">*

*<tr bgcolor="#F5F5F5">*

*<td><xsl:value-of select="author"/></td>*

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

*<td><xsl:value-of select="year\_publication"/></td>*

*</tr>*

*</xsl:for-each>*

*</table>*

*<html>*

*<head>*

*</head>*

*<body>*

*</xsl:template>*

*</xsl:stylesheet>*

Опция *select э*лемента *xsl:sort*определяет, по какому ключу идет сортировка(в примере – по году издания книги).

Элемент *xsl:sort*имеет также дополнительные параметры:

- порядок сортировки (*убывание/возрастание в зависимости от регистра букв, атрибут case-order= "lower-first|upper-first"*);

- тип данных (числовой/текстовый, атрибут *data-type="num-ber|text"*).

**Задание**

Выведите на HTML-страницу содержимое XML-документа, созданного в лабораторной работе 10, используя XSL. Выполните сортировку записей по одному из возможных критериев и фильтрацию данных по одному из возможных показателей.

**Контрольные вопросы**

1. Для чего используется язык XSL?

2. Как связать таблицу стилей XSL с XML-документом?

3. Как создаются шаблоны элементов в XSL?

4. Для чего используется XSL-элемент *value-of*?

5. Как осуществляется адресация к XML-данным?

6. Какие условные операторы предусмотрены в XSL? В чем особенности их использования?

7. Для чего используется оператор *for-each*?

8. Как выполняется сортировка XML-данных?