6 ПРЕДСТАВЛЕНИЕ ДАННЫХ В ТЕХНОЛОГИИ XML. ПРЕОБРАЗОВАНИЯ XSLT

**6.1** **Теоретические сведения**

XML основан на стандарте обобщённого языка разметки SGML (Standard Generalized Markup Language). Подобно HTML, XML – язык разметки, основанный на дескрипторах в угловых скобках; его текстовая природа обеспечивает высокую степень переносимости данных; для создания и редактирования XML-документа может использоваться любой стандартный текстовый редактор. В отличие от HTML, XML не имеет фиксированного набора дескрипторов. Вместо этого XML представляет собой метаязык, позволяющий создавать другие языки разметки. XML устанавливает несколько простых правил именования и упорядочивания элементов, на основе которых можно создавать собственные форматы данных со своими собственными настраиваемыми элементами.

Структура XML-документа состоит из трех частей:

- пролога;

- определения (необязательно);

- тела документа.

Пример XML- документа, описывающего программу обучения:

<?xml version="1.0" encoding="windows-1251" ?>

<Progr>

<Modul>

<TITLE>технология W3С</TITLE>

<Credit>3</Credit>

</Modul>

<Modul>

<TITLE>web-программирование</TITLE>

<Credit>3</Credit>

</Modul>

<Modul>

<TITLE>Компьютерные сети</TITLE>

<Credit>3</Credit>

</Modul>

</Progr>

XML - довольно строгий стандарт, согласно его спецификации выделяются два типа XML-документов правильные или хорошо оформленные (well-formed) и состоятельные (valid).

Чтобы считаться хорошо оформленным, документ XML должен соответствовать перечисленным ниже критериям:

- каждому начальному дескриптору должен соответствовать конечный дескриптор, пустой элемент должен завершаться сочетанием />;

- элементы могут быть вложенными, но никогда не могут перекрываться;

- элементы и атрибуты должны использовать согласованный регистр;

- элемент не может содержать в себе двух атрибутов с одним и тем же именем, но элемент может содержать два вложенных элемента с одинаковыми именами;

- XML документ может иметь только один корневой элемент, с которого начинается документ и который включает в себя всё его содержимое;

- атрибуты задаются своими значениями, заключёнными в символы кавычки;

- комментарии не могут помещаться внутри дескрипторов.

Состоятельный или валидный XML-документ будет обработан анализатором и пользователь получит к нему доступ с использованием интерфейсов DOM.

Формальный документ, устанавливающий логические правила конкретного языка разметки, называется схемой XML-документа. Схемы данных (XML Schema Document, XSD) представляют собой альтернативный способ правил построения XML-документов. Правила, приведённые в схеме XML-документа, включают:

- словарь документа, который определяет, какие элементы и имена атрибутов используются в XML-документе;

- структура документа, которая определяет, где должны помещаться дескрипторы, включает правила, регламентирующие то, какие дескрипторы должны быть помещены перед, после или внутри других;

- поддерживаемые типы данных, указание которых позволяют специфицировать - должны ли данные быть обычным текстом, либо их следует интерпретировать как числовые данные, информацию о датах и т.д.;

- допустимые диапазоны значений, указание которых позволяют установить ограничения на диапазоны допустимых значений числовых данных, ограничение длины текста, требование соответствия регулярным выражениям либо перечень специфических допустимых значений.

Стандарт схемы XML определяет правила, которых следует придерживаться при создании XML -документа.

В начальный тег элемента, либо в тег пустого элемента можно включить одно или несколько описаний атрибутов. Описание атрибута представляет собой пару имя – значение, связанную с данным элементом. Например, для приведенного выше примера, элемент Modul может включать атрибут Type, которому присвоено значение «обязательный»:

< Modul Type =" обязательный " >

<TITLE>технология W3С</TITLE>

<Credit>3</Credit>

</Modul>

Задание атрибутов обеспечивает альтернативный способ включения информации в элемент. Обычно помещаются все относящиеся к элементу данные, которые необходимо отобразить, внутри содержимого элемента. Атрибуты же используются для хранения различных свойств элемента, которые не обязательно будут отображены (например, категория, или указания по отображению). В спецификации XML не установлено строгих разграничений относительно типа информации, которую можно описывать с помощью атрибутов или внутри содержимого элемента.

Отображение XML-документов с использованием XSL-таблиц стилей. XSL-таблица стилей - мощный и гибкий инструмент для отображения XML-документов. XSL позволяет выбрать те данные XML, которые необходимо отобразить, представить эти данные в любом порядке, свободно модифицировать или добавлять информацию. XSL предоставляет доступ ко всем компонентам XML (элементам, атрибутам, комментариям и инструкциям по обработке). Эта таблица позволяет сортировать и фильтровать данные XML, дает возможность включать в таблицу стиля сценарии и предоставляет набор полезных методов, которые можно использовать при обработке информации.

Существуют два основных шага для отображения XML-документа при использовании XSL-таблицы стилей.

* Создание файла XSL-таблицы стилей. XSL является приложением XML, т. е. XSL-таблица представляет собой корректно сформированный XML-документ, который отвечает правилам XSL.
* Связывание XSL-таблицы стилей с XML-документом. Вы можете связать XSL-таблицу стилей с XML-документом, включив в документ инструкцию по обработке xml-stylesheet, которая имеет следующую обобщенную форму записи:

<?xml-stylesheet type="text/xsl" href=XSLFilePath?>

XSL-таблица стилей включает один или несколько шаблонов, каждый из которых содержит информацию для отображения в определенной ветви элементов в XML-документе.

Пример

<?xml version="1.0" encoding="windows-1251" ?>

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

<xsl:template match="/">

<SPAN STYLE="font-style:italic">

TITLE:

</SPAN>

<xsl:value-of select="Progr/Modul/TITLE"/>

<BR/>

<SPAN STYLE="font-style:italic">

Credit:

</SPAN>

<xsl:value-of select="Progr/Modul/Credit"/>

<BR/>

</xsl:template>

</xsl:stylesheet>

Результат отображения XML документа приведен на рисунке 6.1.

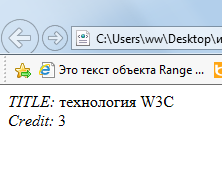


Рисунок 6.1 – Отображение XML документа

Элемент xsl:stylesheet служит не только хранилищем других элементов, но также идентифицирует документ как XSL-таблицу стилей. Этот элемент является одним из XSL-элементов специального назначения, используемых в таблице стилей. Все XSL-элементы принадлежат пространству имен xsl – т. е. перед каждым XSL-элементом необходимо указывать префикс xsl:. Это пространство имен определяется в начальном теге элемента xsl:stylesheet, например, следующим образом:

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

Это определение позволяет использовать пространство имен внутри элементов таблицы стилей.

Элемент xsl:stylesheet XSL-таблицы стилей должен содержать один или несколько шаблонов элементов. Элемент Документ из Примера содержит только один шаблон, который имеет следующую форму:

<xsl:template match="/">

<!-- дочерние элементы … -->

</xsl:template>

Браузер использует шаблон для отображения определенной ветви элементов в иерархии XML-документа, с которым вы связываете таблицу стилей. Атрибут match шаблона указывает на определенную ветвь. (Атрибут match аналогичен селектору в правиле CSS). Значение атрибута match носит название образца (pattern). Образец в данном примере ("/") представляет корневой элемент всего XML-документа. Этот шаблон, таким образом, содержит инструкции для отображения всего XML-документа.

Каждая XSL-таблица стилей должна содержать один и только один шаблон с атрибутом match, который имеет значение "/". Можно включить один или несколько дополнительных шаблонов с инструкциями для отображения определенных подчиненных ветвей в структуре XML-документа; каждая из них должна иметь образец, отвечающий определенной ветви.

Корневой образец ("/") не представляет элемент Документ (или корневой элемент) XML-документа. Он представляет весь документ, для которого элемент Документ является дочерним.

В приведенном примере Шаблон содержит два вида XML-элементов.

1) XML-элементы, представляющие HTML-разметку. Примерами подобного вида XML-элемента из рассматриваемой таблицы стилей являются: <SPAN STYLE="font-style:italic">TITLE: </SPAN>, который отображает блок текста, набранного курсивом, и <BR/>, который создает пустую строку. Все эти XML-элементы являются корректно сформированными и представляют стандартные HTML-элементы. Браузер просто копирует каждый HTML-элемент непосредственно на выход HTML, который воспринимает и отображает их.

2) XSL-элементы. Примерами XSL-элементов из рассматриваемой таблицы стилей являются элементы xsl:value-of, например:

<xsl:value-of select="Progr/Modul/TITLE"/>

Браузер отличает XSL-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. В рассмотренном выше примере элемента value-of атрибуту select присвоен образец "Progr/Modul/TITLE ", что приводит к выводу текстового содержимого элемента TITLE XML-документа.

Если опустить атрибут select для XSL-элемента value-of, элемент будет осуществлять вывод текстового содержимого плюс текстовое содержимое всех дочерних элементов текущего элемента. (В нашем примере, поскольку текущим является корневой элемент, пропуск атрибута select приведёт к выводу всех символьных данных XML-документа.)

По приведенному выше примеру таблиц стилей, отображается в браузере содержимое только первого элемента. Чтобы отобразить содержимое всех элементов, следует использовать XSL-элемент **for-each**.

Элемент for-each выполняет две основные задачи:

1) осуществляет вывод блока элементов, содержащихся внутри элемента for-each, повторяя его для каждого XML-элемента в документе, отвечающего образцу, присвоенному атрибуту select элемента for-each;

2) внутри элемента for-each задает текущий элемент, устанавливаемый атрибутом select элемента for-each.

Пример xsl таблицы с применением XSL-элемент **for-each**:

<?xml version="1.0" encoding="windows-1251" ?>

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

<xsl:template match="/">

<xsl:for-each select="Progr/Modul">

<SPAN STYLE="font-style:italic">Дисциплина: </SPAN>

<xsl:value-of select="TITLE"/><BR/>

<SPAN STYLE="font-style:italic">Количество кредитов: </SPAN>

<xsl:value-of select="Credit"/><BR/>

</xsl:for-each>

</xsl:template>

</xsl:stylesheet>

В результате выводятся данные из всех элементов Modul, найденных в документе, независимо от того, сколько этих элементов содержит документ (рисунок 6.2).

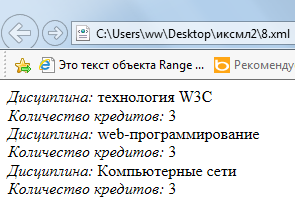


Рисунок 6.2- Результат применения XSL-элемента for-each

Можно ограничить количество элементов документа XML, введя фильтр-выражение, заключенное в квадратные скобки ( [] ) и следующее непосредственно за оператором пути. Для нашего примера отобразим те элементы Modul, которые имеют дочерний элемент TITLE, содержащий текст «технология W3С» (рисунок 6.3).

<?xml version="1.0" encoding="windows-1251" ?>

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

<xsl:template match="/">

<xsl:for-each select="Progr/Modul[TITLE='технология W3С']">

<SPAN STYLE="font-style:italic">Дисциплина: </SPAN>

<xsl:value-of select="TITLE"/><BR/>

<SPAN STYLE="font-style:italic">Количество кредитов: </SPAN>

<xsl:value-of select="Credit"/><BR/>

</xsl:for-each>

</xsl:template>

</xsl:stylesheet>

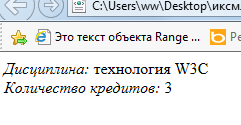


Рисунок 6.3- Результат применения фильтра

Данные из элементов документа можно сортировать, для этого необходимо использовать элемент xsl:sort с атрибутом order.

Пример сортировки по названию дисциплины:

<?xml version="1.0" encoding="windows-1251" ?>

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

<xsl:template match="/">

<xsl:for-each select="Progr/Modul">

<xsl:sort order="ascending" select="TITLE"/>

<SPAN STYLE="font-style:italic">Дисциплина: </SPAN>

<xsl:value-of select="TITLE"/><BR/>

<SPAN STYLE="font-style:italic">Количество кредитов: </SPAN>

<xsl:value-of select="Credit"/><BR/>

</xsl:for-each>

</xsl:template>

**6.2 Задание на лабораторную работу**

Выбрать свою предметную область, структурировать информацию, создать XML файл, отобразить данные в браузере в виде списка и таблицы, применить сортировку и фильтрацию данных.