


МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «КубГУ»)

Физико-технический факультет

Кафедра теоретической физики и компьютерных технологий

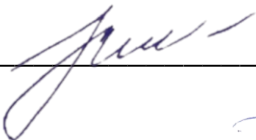
КУРСОВОЙ ПРОЕКТ
ХРАНЕНИЕ И ПРЕОБРАЗОВАНИЕ ВЕКТОРНОЙ ГРАФИКИ

Работу выполнил _____  Мингалимов Руслан Ринатович

Курс 3

Направление 09.03.02 Информационные системы и технологии

Научный руководитель

канд. физ.-мат. наук, доцент _____  Ю.Г.Никитин

Нормоконтролер старший преподаватель

_____  Г.Д.Цой

Краснодар 2018

СОДЕРЖАНИЕ

Введение.....	3
1 Основы XML.....	4
1.1 Правила синтаксиса XML.....	6
1.2 Кодировка в Xml.....	7
1.3 XML редакторы.....	9
2 Модель SG.....	11
3 Модель Dom.....	12
3.1 HTML DOM.....	12
3.2 Примеры XSLT в HTML.....	16
4 Языки XSL(T).....	14
4.1 XSLT преобразования.....	21
Заключение.....	27
Список использованных источников.....	28

ВВЕДЕНИЕ

Векторная графика — способ представления объектов и изображений (формат описания) в компьютерной графике, основанный на математическом описании элементарных геометрических объектов, обычно называемых примитивами, таких как: точки, линии, сплайны, кривые Безье, круги и окружности, многоугольники. Мы живём в век пикселей. Как дизайнеры и разработчики в вебе, пиксели могут быть нам как друзьями, так и врагами. Мы хотим, чтобы всё выглядело красиво и чётко для всех, кто пользуется нашими сайтами, и нам необходимо уменьшать размеры файлов для улучшения производительности. В общем-то, есть только один способ для иконок, логотипов и иллюстраций — это SVG. Масштабируемая векторная графика (Scalable Vector Graphics) позволяет изображению выглядеть чётко на мониторе с любым разрешением, при этом иметь очень маленький размер файла и легко поддаваться редактированию и изменениям. Цель моей работы — дать обзор практического применения SVG-изображений в вебе с некоторыми советами и хитростями, которые помогут получить максимальную отдачу от них.

1 Основы XML

XML - аббревиатура от англ. eXtensible Markup Language (пер. расширяемый язык разметки; произносится [экс-эм-эл]). XML разработан для хранения и передачи данных. XML играет важную роль, хотя и прост в изучении.

Пример кода XML:

```
<?xml version="1.0" encoding="UTF-8"?>  
  
<note>  
  <to>Tove</to>  
  <from>Jani</from>  
  <heading>Напоминание</heading>  
  <body>Не забудь обо мне в эти выходные!</body>  
</note>
```

Что такое XML?

XML – язык разметки, который напоминает HTML.

XML предназначен для передачи данных, а не для их отображения.

Теги XML не предопределены. Вы должны сами определять нужные теги.

XML описан таким образом, чтобы быть самоопределяемым.

Важно понять, что XML не является заменой HTML. В большинстве веб-приложениях XML используется для транспортировки данных, а HTML для форматирования и отображения данных. XML – это программно- и аппаратно-независимый инструмент для транспортировки информации.

В настоящее время XML также важен для сети, как когда-то был важен HTML для рождения современного Интернета. XML – это общий инструмент передачи данных между всеми видами приложений. XML используется во многих аспектах веб-разработки, но основная его задача — облегчение хранения и передачи данных. XML отделяет данные от HTML

Если вам в HTML документе необходимо отображать динамические данные, то это будет занимать слишком много времени, если всякий раз, когда эти данные изменились, редактировать сам HTML документ.

С XML данные можно хранить в отдельных файлах XML. При этом вы сосредотачиваетесь на использовании HTML/CSS для отображения и шаблонизации и можете быть уверены, что поступающие новые данные не потребуют каких-либо изменений в коде HTML документа.

XML упрощает распределение данных.

В реальном мире компьютерные системы и базы данных используют данные в несовместимых форматах.

XML данные хранятся в простом текстовом формате. Это обеспечивает программную и аппаратную независимость.

Это позволяет легко создавать данные, которые могут использоваться самыми разными приложениями.

XML упрощает передачу данных

Одной из самых время затратных проблем разработчиков всегда была и остается до сих пор проблема обмена данными между несовместимыми между собой системами.

Передача данных в виде XML значительно снижает сложность этой проблемы, так как данные в этом формате могут быть прочитаны разными несовместимыми приложениями.

XML упрощает модификацию платформы

Переход на новые системы (аппаратные или программные платформы) всегда занимает много времени. Множество данных необходимо конвертировать в новые форматы. При этом часто несовместимые данные теряются.

XML данные хранятся в текстовом формате. Это значительно облегчает расширение или модернизацию операционных систем, переход на новые приложения или браузеры без опасности потерять данные.

XML делает ваши данные более доступными

Доступ к вашим данным могут получать не только HTML документы, но и любые другие приложения.

1.1 Правила синтаксиса XML

Правила синтаксиса XML крайне просты и логичны. Их легко запомнить и легко использовать.

Все XML элементы должны иметь закрывающий тег:

`<p>Это параграф.</p>`

Теги XML регистрозависимы

Теги XML являются регистрозависимыми. Так, тег `<Letter>` не то же самое, что тег `<letter>`.

Открывающий и закрывающий теги должны определяться в одном регистре:

`<Message>Это неправильно</message>`

`<message>Это правильно</message>`

XML элементы должны соблюдать корректную вложенность

В HTML иногда можно наблюдать такую картину:

`<i>Это жирный и курсивный текст</i>`

и иногда это даже работает должным образом.

В XML все элементы обязаны соблюдать корректную вложенность:

`<i>Это жирный и курсивный текст</i>`

Понятие "корректная вложенность" по отношению к приведенным примерам просто означает, что так как элемент `<i>` открывается внутри элемента ``, то и закрываться он должен внутри элемента ``.

У XML документа должен быть корневой элемент

XML документ должен содержать один элемент, который будет родительским для всех других элементов. Он называется корневым элементом.

`<корневой>`

`<потомок>`

`<подпотомок<к>.....</подпотомок>`

`</потомок>`

</корневой>

Значения XML атрибутов должны заключаться в кавычки

Так же, как и в HTML, у XML элементов могут быть атрибуты в виде пары имя/значение.

В XML значения атрибутов должны заключаться в кавычки.

Посмотрите на следующие два примера XML документа. Первый с ошибкой, второй написан правильно:

```
<note date=12/11/2007>
```

```
<to>Tove</to>
```

```
<from>Jani</from>
```

```
</note>
```

```
<note date="12/11/2007">
```

```
<to>Tove</to>
```

```
<from>Jani</from>
```

```
</note>
```

Ошибка в первом XML документе заключается в том, что значение атрибута date элемента note не заключено в кавычки.

1.2Кодировка в XML

XML документы могут содержать символы в различных международных кодировках.

Чтобы не возникало ошибок, необходимо указывать, какая кодировка используется в XML документе, либо сохранять файл в универсальной кодировке UTF-8.

Символьная кодировка

Символьная кодировка определяет уникальный бинарный код для различных символов, используемых в документе.

В компьютерных терминах символьную кодировку также называют символьным набором, символьной раскладкой, кодовым набором и кодом страницы.

Юникод — это промышленный стандарт для символьной кодировки текстового документа. Он определяет (почти) все возможные международные символы по именам и числам.

Юникод имеет две разновидности: UTF-8 и UTF-16.

UTF = формат преобразования Юникода (анг. Unicode Transformation Format).

UTF-8 использует один байт (8 бит) для представления общепринятых символов и два (или три) байта для всех остальных символов.

UTF-16 использует два байта (16 бит) для большинства символов и три байта для всего остального.

UTF-8 - Веб-стандарт

UTF-8 — стандартная кодировка символов в сети Интернет.

UTF-8 считается кодировкой по умолчанию в HTML-5, CSS, JavaScript, PHP, SQL и XML.

Очень часто XML документы создаются на одном компьютере, на сервер выгружаются с другого, а в браузере отображаются на третьем компьютере.

Если кодировка некорректно интерпретируется всеми тремя компьютерами, то браузер отобразит бессмысленный набор символов, либо вообще выдаст сообщение об ошибке.

Наилучшим выбором в этом случае будет использование кодировки UTF-8. UTF-8 позволяет отображать практически все международные символы, и, кроме этого, она считается кодировкой по умолчанию, если не указана другая кодировка.

Заключение

Когда вы пишете XML документ:

Используйте текстовый редактор, который позволяет изменять кодировку документа

Убедитесь, что редактор настроен на использование нужной кодировки

Опишите используемую кодировку в соответствующей декларации

UTF-8 является самой безопасной кодировкой

UTF-8 является стандартом в сети Интернет

XML основан на тексте

XML – это язык разметки на основе текста.

Один из огромнейших плюсов XML то, что XML файлы могут создаваться и редактироваться при помощи простого текстового редактора, вроде Notepad.

Тем не менее, начав работать с XML, очень скоро вы обнаружите, что редактировать XML документы лучше, используя именно профессиональный XML редактор.

Так почему же не Notepad?

Многие веб-разработчики используют Notepad для редактирования как HTML, так и XML документов, потому что Notepad входит в состав большинства операционных систем, и он крайне прост в использовании. Notepad удобен для быстрого редактирования простых файлов HTML, CSS и XML.

Однако, если использовать Notepad для профессионального редактирования XML документов, то вскоре вы столкнетесь с рядом проблем.

Дело в том, что Notepad не знает, что вы работаете именно с языком XML, поэтому он не сможет помочь вам.

А чем лучше XML редактор?

Сегодня XML является важной веб-технологией, и рабочий проект использует такие производные XML-технологии как:

XML схемы для определения структуры и типов данных XML

XSLT для трансформирования данных XML

SOAP для обмена данными XML между приложениями

WSDL для описания веб-служб

RDF для описания веб-ресурсов

XPath и XQuery для доступа к данным XML

SMIL для определения графики

Чтобы создавать XML документы без ошибок, вам нужен умный XML редактор!

2 Модель SVG

Scalable Vector Graphics это разметка, основанная на XML, который содержит двумерные векторы. Векторами могут быть простые геометрические формы, сложные контуры, да и всё то же самое, что можно сделать в Иллюстраторе. Этот формат изображений имеет намного больше общего с веб-страницей, чем тот же JPEG. SVG намного мощнее — им легко можно управлять при помощи кода (в текстовом редакторе или с помощью CSS / JS).

3 XML DOM

Объектная модель документа (DOM) определяет стандартный способ доступа к элементам документа и манипулирования ими.

XML DOM определяет стандартный способ доступа к элементам XML документа и манипулирования ими.

XML DOM представляет XML документ в виде древовидной структуры.

При помощи дерева DOM можно получить доступ ко всем элементам документа. Можно изменять и удалять содержимое (текст и атрибуты) элементов, создавать новые элементы. Элементы, их текст и атрибуты формируют, так называемые, узлы DOM.

3.1 HTML DOM

HTML DOM определяет стандартный способ доступа к элементам HTML документа и манипулирования ими.

При помощи дерева HTML DOM можно получить доступ ко всем элементам HTML документа.

Загрузка XML файла

В следующем примере парсится XML документ в объект XML DOM, из которого затем при помощи Javascript извлекается некоторая информация:

```
<html>
<body>
  <h1>Заметка</h1>
  <div>
    <b>Кому:</b> <span id="to"></span><br />
    <b>От:</b> <span id="from"></span><br />
    <b>Сообщение:</b> <span id="message"></span>
  </div>
```

```

<script>
    if (window.XMLHttpRequest)
    { // для IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp = new XMLHttpRequest();
    }
    else
    { // для IE6, IE5
        xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
    }

    xmlhttp.open("GET","note.xml",false);
    xmlhttp.send();
    xmlDoc = xmlhttp.responseXML;

    document.getElementById("to").innerHTML =
xmlDoc.getElementsByTagName("to")[0].childNodes[0].nodeValue;
    document.getElementById("from").innerHTML =
xmlDoc.getElementsByTagName("from")[0].childNodes[0].nodeValue;
    document.getElementById("message").innerHTML =
xmlDoc.getElementsByTagName("body")[0].childNodes[0].nodeValue;
</script>
</body>
</html>

```

Чтобы извлечь текст "Tove" из элемента <to> XML файла, используется следующее выражение:

```

getElementsByTagName("to")[0].childNodes[0].nodeValue

```

Обратите внимание, что даже если в XML файле присутствует только ОДИН элемент `<to>`, все равно необходимо использовать индекс массива `[0]`. Это объясняется тем, что метод `getElementsByName()` возвращает массив.

Загрузка XML строки

В следующем примере парсится XML строка в объект XML DOM, из которого затем при помощи Javascript извлекается некоторая информация:

```
<html>
<body>
  <h1>Заметка</h1>
  <div>
    <b>Кому:</b> <span id="to"></span><br />
    <b>От:</b> <span id="from"></span><br />
    <b>Сообщение:</b> <span id="message"></span>
  </div>

  <script>
    txt = "<note>";
    txt = txt + "<to>Тове</to>";
    txt = txt + "<from>Jani</from>";
    txt = txt + "<heading>Напоминание</heading>";
    txt = txt + "<body>Не забудь обо мне в эти выходные!</body>";
    txt = txt + "</note>";

    if (window.DOMParser)
    {
      parser = new DOMParser();
      xmlDoc = parser.parseFromString(txt,"text/xml");
    }
    else // Internet Explorer
```

```

{
    xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
    xmlDoc.async = false;
    xmlDoc.loadXML(txt);
}

document.getElementById("to").innerHTML =
xmlDoc.getElementsByTagName("to")[0].childNodes[0].nodeValue;
document.getElementById("from").innerHTML =
xmlDoc.getElementsByTagName("from")[0].childNodes[0].nodeValue;
document.getElementById("message").innerHTML =
xmlDoc.getElementsByTagName("body")[0].childNodes[0].nodeValue;
</script>
</body>
</html>

```

3.2 Пример XML в HTML

В следующем примере мы открываем XML файл и затем в цикле обходим каждый элемент CD и отображаем значения элементов ARTIST и TITLE в таблице HTML:

```

<html>
<body>
<script>
    if (window.XMLHttpRequest)
    { // для IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp = new XMLHttpRequest();
    }

```



```

else
{ // для IE6, IE5
    xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
}

xmlhttp.open("GET","cd_catalog.xml",false);
xmlhttp.send();
xmlDoc = xmlhttp.responseXML;

document.write("<table border='1'>");
var x = xmlDoc.getElementsByTagName("CD");
for (i = 0; i < x.length; i++)
{
    document.write("<tr><td>");
    docu-
ment.write(x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue
);
    document.write("</td><td>");
    docu-
ment.write(x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue);
    document.write("</td></tr>");
}
document.write("</table>");
</script>
</body>
</html>

```

4 Языки XSL(T)

XSL ("расширяемый язык таблиц стилей" от англ. *eXtensible Stylesheet Language*) — язык преобразования и визуализации для XML.

XSLT означает преобразование или трансформация XSL (от англ. *XSL Transformations*) — язык преобразования XML документов.

XSLT — язык для преобразования XML документов.

В HTML используются предопределенные теги, значение и способ отображения которых хорошо понятны.

CSS используется для добавления стилей элементам HTML.

В XML используются теги не определенные заранее, что делает значение каждого тега не ясным изначально.

Например, элемент `<table>` в HTML означает таблицу и ничего другого. В XML этот элемент может означать все что угодно — таблицу, стол и т. д. — и браузеры не знают наверняка, как его отображать.

И здесь на помощь приходит XSL, который позволяет описать, как элементы XML должны отображаться в браузере.

XSL — больше, чем просто язык таблицы стилей

XSL состоит из четырех частей:

XSLT — язык преобразования XML документов

XPath — язык для навигации по элементам XML документа

XSL-FO — язык для форматирования XML документов (разработка остановлена в 2013 году)

XQuery — язык, позволяющий делать выборки из XML данных

Примечание: Представив в 2013 году модуль CSS3 Paged Media Module, консорциум W3C определил новый стандарт форматирования документов. Таким образом, с 2013 года CSS3 позиционируется, как замена языка XSL-FO.

Что такое XSLT?

XSLT означает преобразование или трансформация XSL (от англ. *XSL Transformations*)

XSLT является наиболее важной частью языка XSL.

XSLT преобразовывает XML документ в другой XML документ.

XSLT для навигации по XML документу использует язык XPath.

XSLT — наиболее важная часть языка XSL.

XSLT используется для преобразования XML документа в другой XML документ или в другой тип документа, распознаваемый браузером, например, HTML и XHTML. Обычно, XSLT делает это преобразовывая каждый XML элемент в (X)HTML элемент.

При помощи XSLT можно добавлять/удалять элементы и атрибуты в конечный файл. Также, можно реорганизовывать и сортировать элементы, выполнять тесты, определять, какие элементы скрыть или отобразить, и многое другое.

В общих словах процесс преобразования можно описать следующим образом — XSLT преобразовывает исходное дерево XML в XML дерево-результат.

XML – это язык разметки на основе текста.

Один из огромнейших плюсов XML то, что XML файлы могут создаваться и редактироваться при помощи простого текстового редактора, вроде Notepad.

Тем не менее, начав работать с XML, очень скоро вы обнаружите, что редактировать XML документы лучше, используя именно профессиональный XML редактор.

Так почему же не Notepad?

Многие веб-разработчики используют Notepad для редактирования как HTML, так и XML документов, потому что Notepad входит в состав большинства операционных систем, и он крайне прост в использовании. Notepad удобен для быстрого редактирования простых файлов HTML, CSS и XML.

Однако, если использовать Notepad для профессионального редактирования XML документов, то вскоре вы столкнетесь с рядом проблем.

Дело в том, что Notepad не знает, что вы работаете именно с языком XML, поэтому он не сможет помочь вам.

А чем лучше XML редактор?

Сегодня XML является важной веб-технологией, и рабочий проект использует такие производные XML-технологии как:

XML схемы для определения структуры и типов данных XML

XSLT для трансформирования данных XML

SOAP для обмена данными XML между приложениями

WSDL для описания веб-служб

RDF для описания веб-ресурсов

XPath и XQuery для доступа к данным XML

SMIL для определения графики

Чтобы создавать XML документы без ошибок, вам нужен умный XML редактор!

XML редакторы

Профессиональный XML редактор поможет вам создавать XML документы без ошибок, проверять их по DTD или схеме и выработать привычку писать правильную структуру XML.

XML редактор должен уметь:

- автоматически дописывать закрывающий тег;
- заставлять вас писать правильный XML;
- проверять правильность XML по DTD;
- проверять правильность XML по XML схеме;
- выделять цветом синтаксис XML.

4.1 XSLT-Преобразование

Корневым элементом, декларирующим документ таблицы стилей XSL, является `<xsl:stylesheet>` или `<xsl:transform>`.

Примечание: Элементы `<xsl:stylesheet>` и `<xsl:transform>` являются полными синонимами, и для декларации таблицы стилей можно использовать любой из них!

Согласно рекомендации консорциума W3C таблица стилей XSL декларируется следующим образом:

```
<xsl:stylesheet version="1.0"
```

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

или:

```
<xsl:transform version="1.0"
```

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

Чтобы получить доступ к элементам, атрибутам и другим функциям XSLT, необходимо в начале документа декларировать пространство имен XSLT.

Строка `xmlns:xsl="http://www.w3.org/1999/XSL/Transform"` указывает на официальное пространство имен XSLT консорциума W3C. Если вы используете это пространство имен, то вы также должны указывать и атрибут `version="1.0"`.

Начнем с чистого XML документа

Предположим, что нам нужно преобразовать следующий XML документ ("cd_catalog.xml") в XHTML:

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

```
<catalog>
```

```
  <cd>
```

```
    <title>Empire Burlesque</title>
```

```
    <artist>Bob Dylan</artist>
```

```
    <country>USA</country>
```

```
    <company>Columbia</company>
```

```
    <price>10.90</price>
```

```

    <year>1985</year>
  </cd>
  .
  .
</catalog>

```

Откройте XML файл (нажмите на ссылку ниже) - XML документ будет отображаться в виде окрашенных в разные цвета корневого и дочерних элементов и их содержимого (кроме браузера Safari). Часто слева от элементов XML дерева выводится знак плюса (+) или минуса (-), при нажатии на который можно развернуть/свернуть структуру элемента. Чтобы просмотреть исходный код документа, нажмите правой кнопкой мыши на XML файл и в контекстном меню выберите пункт "Исходный код" (Opera) или "Просмотреть код страницы" (Chrome)!

Создание таблицы стилей XSL

Теперь создаем таблицу стилей XSL ("cd_catalog.xsl") с шаблоном преобразования:

```

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

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

  <xsl:template match="/">
    <html>
    <body>
    <h2>My CD Collection</h2>
    <table border="1">
      <tr bgcolor="#9acd32">
        <th>Title</th>
        <th>Artist</th>

```

```

</tr>
<xsl:for-each select="catalog/cd">
  <tr>
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="artist"/></td>
  </tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>

```

```
</xsl:stylesheet>
```

Подключаем таблицу стилей XSL к XML документу

Добавляем ссылку на таблицу стилей XSL в XML документ ("cd_catalog.xml"):

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="http://msiter.ru/cd_catalog.xsl"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
  .
</catalog>

```

Если вы используете XSLT совместимый браузер, то вы увидите корректно преобразованный из XML в XHTML документ.

При помощи **XSLT** вы можете преобразовывать XML документ в HTML.

XSLT (eXtensible Stylesheet Language Transformations), язык преобразования XML документов, является рекомендованным языком таблиц стилей для XML.

XSLT более сложный язык таблиц стилей, чем CSS. Используя XSLT вы можете добавлять/удалять элементы и атрибуты в конечном файле отображения. Также, вы можете реорганизовывать и сортировать элементы, осуществлять проверки и принимать решения о том, какие элементы скрывать, а какие отображать. И многое другое.

XSLT использует XPath для поиска информации в XML документе.

Пример XSLT

Будем использовать следующий XML документ:

```
<?xml version="1.0" encoding="UTF-8"?>
<breakfast_menu>
  <food>
    <name>Belgian Waffles</name>
    <price>$5.95</price>
    <description>Two of our famous Belgian Waffles with plenty of real maple
syrup</description>
    <calories>650</calories>
  </food>
  <food>
    <name>Strawberry Belgian Waffles</name>
    <price>$7.95</price>
    <description>Light Belgian waffles covered with strawberries and whipped
cream</description>
```



```

    <calories>900</calories>
  </food>
  <food>
    <name>Berry-Berry Belgian Waffles</name>
    <price>$8.95</price>
    <description>Light Belgian waffles covered with an assortment of fresh berries
and whipped cream</description>
    <calories>900</calories>
  </food>
  <food>
    <name>French Toast</name>
    <price>$4.50</price>
    <description>Thick slices made from our homemade sourdough
bread</description>
    <calories>600</calories>
  </food>
  <food>
    <name>Homestyle Breakfast</name>
    <price>$6.95</price>
    <description>Two eggs, bacon or sausage, toast, and our ever-popular hash
browns</description>
    <calories>950</calories>
  </food>
</breakfast_menu>

```

Перед выводом в браузер используем XSLT для преобразования XML в HTML.

Пример таблицы стилей XSLT:

```

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

```

```

<body style="font-family: Arial; font-size: 12pt; background-color: #EEE">
  <xsl:for-each select="breakfast_menu/food">
    <div style="background-color: teal; color: white; padding: 4px">
      <span style="font-weight: bold"><xsl:value-of select="name"/> - </span>
      <xsl:value-of select="price"/>
    </div>
    <div style="margin-left: 20px; margin-bottom: 1em; font-size: 10pt">
      <p>
        <xsl:value-of select="description"/>
        <span style="font-style: italic"> (<xsl:value-of select="calories"/> calories
per serving)</span>
      </p>
    </div>
  </xsl:for-each>
</body>
</html> XML основан на тексте

```

ЗАКЛЮЧЕНИЕ

Целью данного курсового проекта было с помощью языка разметки XSLT научиться преобразовывать XML документа в другой XML документ или в другой тип документа, распознаваемый браузером, например, HTML и XHTML.

В ходе курсового проекта были разобраны такие предметные области, как :

1 Основы XML.

2 Кодировка в XML.

3 Примеры XML в HTML и отображение значений элементов ARTIST и TITLE в таблице HTML. Языки XSL(T) и XSLT-преобразования .

4 Модели XML SVG и XML DOM.

В результате выполнения курсового проекта я научился преобразовывать XML документ в другой тип XML документа, распознаваемый браузером.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 Самоучитель по xslt. — (Рус.). — URL: <http://msiter.ru/tutorials/xslt> [20 апреля 2018].
- 2 Самоучитель по XML. — (Рус.). — URL: <http://msiter.ru/tutorials/uchebnik-xml-dlya-nachinayushchih> [12 апреля 2018].
- 3 Язык программирования. Википедия - свободная энциклопедия. — (Рус.). — URL: [https://ru.wikipedia.org/wiki/ язык_программирования](https://ru.wikipedia.org/wiki/язык_программирования). [18 марта 2017].
- 4 Практическое руководство в вебе — (рус.). — URL: <https://svgontheweb.com/ru/> [24 апреля 2018].
- 5 Дунаев В. HTML, скрипты и стили (3-е издание) / В. Дунаев. — М.: Питер, 2012. — 816 с.
- 6 Алексеев А.П. Введение в Web-дизайн / А.П. Алексеев. — М.: Солон-Пресс, 2008. — 192 с.
- 7 Бабаев А. Создание сайтов / А. Бабаев. — М.: Питер, 2014. — 410 с.
- 8 Круга С. Веб-Дизайн или "не заставляйте меня думать!" / С. Круга. — М.: Символ-Плюс, 2008. — 224 с.
- 9 Балабов К. XML для чайников / К. Балабов - М.: Питер, 2014. — 410 с.
- 10 Дунаев В. HTML, скрипты и стили (2-е издание) / В. Дунаев. — М.: Питер, 2010. — 720 с.
- 11 Алексеев А.П. в Web-дизайн на практике / А.П. Алексеев. — М.: Солон-Пресс, 2010. — 342 с.