

Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования
«Тамбовский государственный технический университет»

**Ю. Ю. Громов, О. Г. Иванова,
Н. Г. Шахов, В. Г. Однолько**

ИНФОРМАЦИОННЫЕ WEB-ТЕХНОЛОГИИ

Допущено Учебно-методическим объединением вузов
по университетскому политехническому образованию
в качестве учебного пособия для студентов высших
учебных заведений, обучающихся по направлению подготовки
230400 «Информационные системы и технологии»



Тамбов
Издательство ФГБОУ ВПО «ТГТУ»
2014

УДК 004.42(075)
ББК з973-018.4я73
И74

Рецензенты:

Кандидат технических наук, профессор ФГБОУ ВПО «ТГТУ»,
директор Центрально-Черноземного регионального
учебно-научного центра по проблемам информационной безопасности
Ю. Ф. Мартемьянов

Доктор физико-математических наук,
профессор, заслуженный деятель науки Российской Федерации,
заведующий отделом информатики ИРЭ РАН, г. Москва
В. Ф. Крапивин

И74 Информационные Web-технологии : учебное пособие /
Ю. Ю. Громов, О. Г. Иванова, Н. Г. Шахов, В. Г. Однолько. –
Тамбов : Изд-во ФГБОУ ВПО «ТГТУ», 2014. – 96 с. – 100 экз. –
ISBN 978-5-8265-1365-1.

Приведены краткие теоретические сведения и рекомендации по выполнению лабораторных работ, направленных для изучения практических аспектов технологий разработки современных Web-приложений, в том числе новых возможностей клиентской разработки на основе HTML5, CSS3 и JavaScriptAPI.

Предназначено для студентов дневного и заочного отделений, уккоренного и дистанционного обучения направлений 230400 – Информационные системы и технологии и 220100 – Системный анализ и управление.

УДК 004.42(075)
ББК з973-018.4я73

ISBN 978-5-8265-1365-1

© Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Тамбовский государственный технический университет» (ФГБОУ ВПО «ТГТУ»), 2014

ВВЕДЕНИЕ

Предлагаемый лабораторный практикум предназначен для изучения практических аспектов технологий разработки современных веб-приложений, в том числе новых возможностей клиентской разработки на основе HTML5, CSS3 и JavaScriptAPI, а также инструментария WebMatrix для разработки серверных приложений.

В рамках лабораторного практикума студент должен выполнить лабораторные занятия с использованием Visual Studio Web Developer Express и WebMatrix для практического изучения технологий клиентской и серверной веб-разработки. Практикум содержит задания для самостоятельного выполнения и более глубокого понимания материала студентами.

Целью данного учебного пособия является комплексное рассмотрение современных клиентских и серверных технологий веб-разработки, применяемых для создания веб-сайтов, и практическое изучение данных технологий на конкретных примерах с использованием Visual Studio Web Developer Express, Microsoft WebMatrix.

Учащиеся, освоившие предлагаемый курс, смогут разрабатывать высококачественные мультимедиа-приложения, познакомятся с методами программирования трехмерной графики и создания зрелищных визуальных эффектов, научатся разрабатывать офф-лайн приложения, смогут визуализировать сложные данные и результаты научных исследований, освоят новые элементы управления, познакомятся с технологией Microsoft Razor.

Лабораторная работа 1

ОСНОВЫ РАБОТЫ С MICROSOFT WEBMATRIX3

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

WebMatrix – это свободно распространяемый и простой в использовании пакет средств веб-разработки, предоставляющий самый простой набор инструментов разработки веб-сайтов.

В этот пакет входят:

- примеры и шаблоны веб-страниц;
- язык веб-сервера (Razor использованием VB или C #);
- веб-сервер (IIS Express);
- сервер баз данных (SQL Server Compact);
- язык разметки на стороне сервера (Razor);
- полноценная платформа веб-разработки (ASP.NET);
- IIS Developer Express (веб-сервер разработки, запускаемый системой по запросу), ASP.NET и SQL Server Compact (встроенная база данных).

В пакет также входит простой редактор с подсветкой синтаксиса поддерживаемых языков программирования и средств разметки, ускоряющий разработку веб-сайтов и упрощающий запуск веб-сайтов из популярных приложений с открытым кодом. Код, разработанный с помощью WebMatrix, можно использовать в Visual Studio.

Создаваемые с помощью WebMatrix веб-страницы могут быть динамическими, т.е. могут изменять содержимое и стиль в зависимости от введенных пользователем данных или от других сведений, например содержащихся в базе данных. Для программирования динамических веб-страниц используются ASP.NET с синтаксисом Razor и языки программирования C# или Visual Basic.

WebMatrix обеспечивает относительно простой подход к веб-разработке, который может помочь достичь высокой продуктивности очень быстро.

При планировании WebMatrix командой разработчиков Microsoft было решено, что простота является ключевым принципом. Существуют вещи, которые каждый разработчик динамических веб-сайтов должен знать, независимо от платформы, которую он выбирает: HTML, CSS, некоторые JavaScript, SQL, и код на стороне сервера. В той или иной степени все платформы разработки стремятся скрыть

некоторые ненужные детали, связанные с этими технологиями. Двигаясь в этом направлении, некоторые платформы при этом вводят ряд новых концепций. Несмотря на ускорение процесса разработки веб-сайтов и веб-приложений освоение новых концепции может занимать длительное время и представлять определенные сложности для начинающих разработчиков.

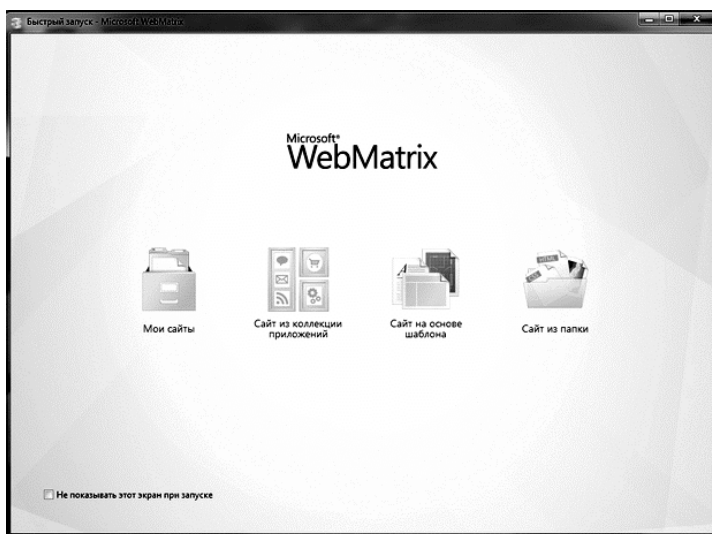
WebMatrix возвращается к истокам веб-разработки. Этот пакет погружает разработчика в HTML, CSS, JavaScript и другие технологии, общие для всех платформ веб-разработки. Он также включает в себя ряд «помощников» (helpers), которые являются «конвертами» для некоторых общих задач, которые приходится решать при разработке веб-сайта, таких как доступ к данным, управление безопасностью, а также отправка электронной почты. Эти помощники обеспечивают более быстрый доступ, чем какие-либо другие инструменты, и при этом очень просты в освоении и использовании.

Как правило, доступ к этим помощникам из веб-страниц реализуется, используя либо Visual Basic либо C# код.

Немаловажным обстоятельством является и то, что WebMatrix является совершенно бесплатным пакетом.

Для установки WebMatrix необходимо запустить Web Platform Installer (не ниже версии 3.0), выбрать из списка продуктов WebMatrix и нажать Install.

После установки и запуска WebMatrix появится следующая заставка:



Для быстрого создания сайта WebMatrix можно использовать галерею веб-приложений, создать сайт из предустановленного шаблона или открыть уже существующую папку, содержащую сайт.

Оболочка, помимо доступа к галерее и шаблонам, предоставляет набор встроенных редакторов и средств для управления созданным сайтом:

- редакторы кода разметки;
- дизайнеры;
- редакторы данных для БД;
- средства публикации сайта;
- некоторые средства анализа веб-приложения.

Редактор поддерживает подсветку синтаксиса C#, VB, HTML, XML, CSS, XHTML (Razor), ASPX и др.

WebMatrix базируется на трех компонентах – IIS Express, SQL Server Compact Edition 4 и механизм представлений Razor.

IIS Express обладает следующими достоинствами:

– простой и компактный сервер для разработки и тестирования приложений, а также для использования в качестве встроенного web-сервера;

- полная совместимость с модулями полноценного IIS;
- поддержка CGI;
- совместимость с конфигурационными файлами IIS (applicationHost.config, web.config);
- поддержка SSL.

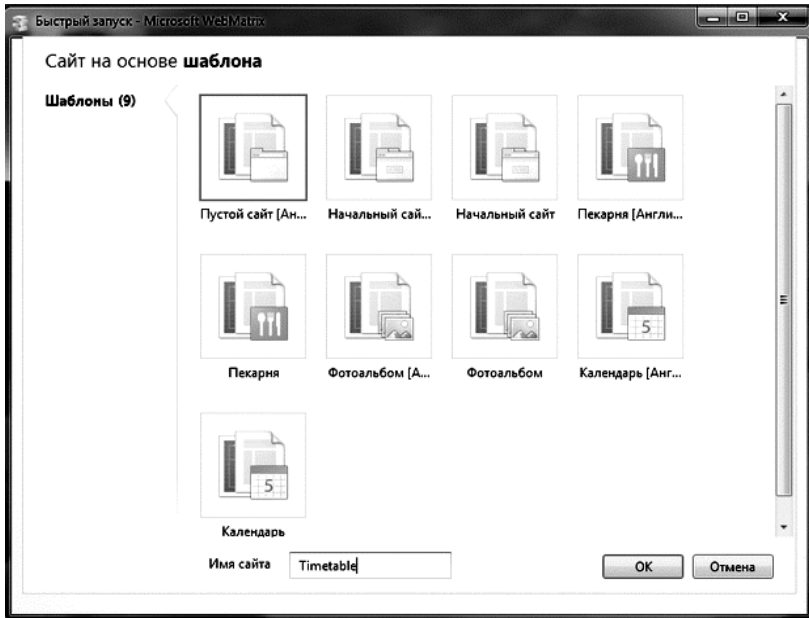
Однако им невозможно управлять через IIS Manager.

SQL Server Compact – это версия встроенной СУБД от Microsoft. В ней реализована работа с СУБД в многопоточном режиме. Если до этого приходилось писать свои уровни абстракции, которые занимались управлением доступом к БД, то теперь SQL Server CE лишен этого серьезного ограничения. Кроме того, SQL Server CE 4 не требует установки, достаточно скопировать в папку bin набор необходимых сборок.

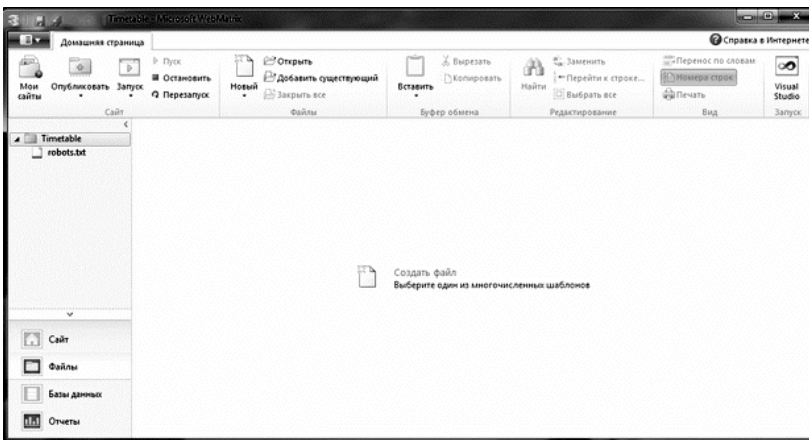
ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Задание 1. Создание и редактирование HTML-документа в WebMatrix.

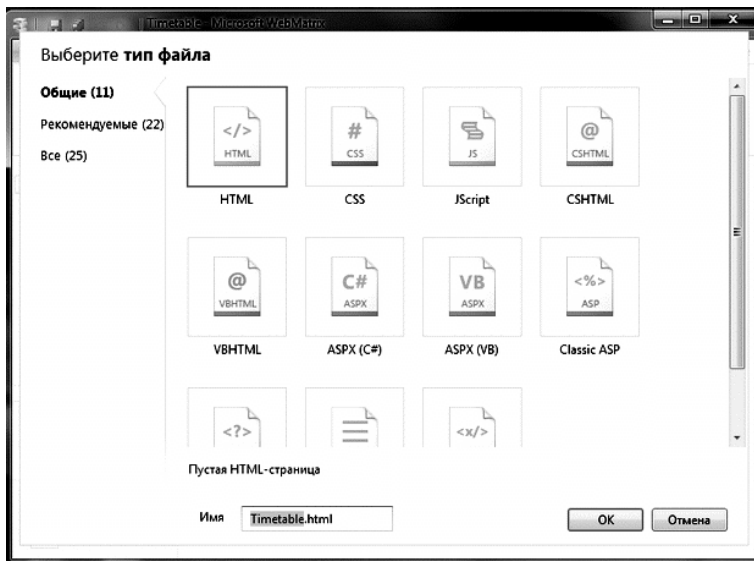
После запуска WebMatrix необходимо выбрать «Сайт на основе шаблона», а затем «Пустой сайт», при этом укажите внизу имя сайта «Timetable»:



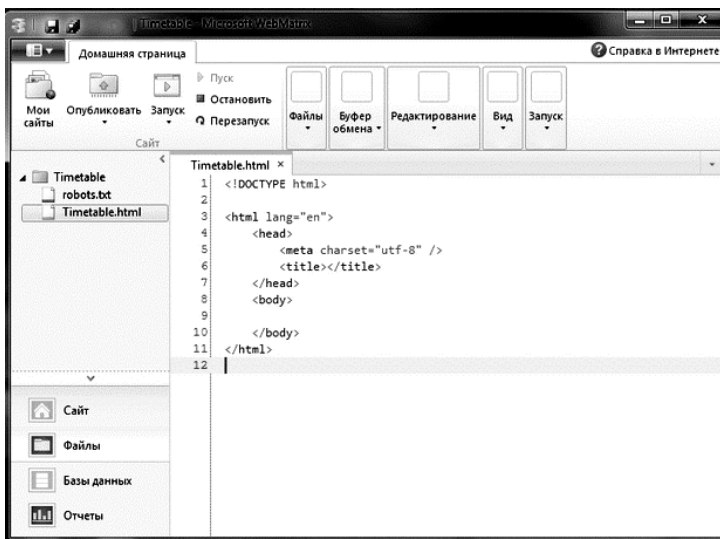
Затем необходимо создать файлы для веб-сайта. Если на левой панели выбрать раздел «Файлы», то можно увидеть, что в списке представлен единственный файл «robots.txt». Поэтому нам потребуется создать дополнительно HTML-страницу веб-сайта и таблицу стилей CSS. С этой целью необходимо кликнуть «Создать файл на правой панели» или выбрать «Новый» на панели инструментов наверху:



Далее следует выбрать тип файла «HTML». Внизу также следует указать имя файла – «Timetable».



После этого будет автоматически создана HTML-страница следующего содержания:



Добавьте вручную следующий HTML-код:
<!DOCTYPE html PUBLIC «-//W3C//DTD XHTML 1.0 Transitional-
al//EN»

```
<http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd»>
```

```
<html xmlns=«http://www.w3.org/1999/xhtml» >
```

```
<head>
```

```
<title>Расписание занятий</title>
```

```
<link href=«TableStyleSheet.css» rel=«stylesheet» type=«text/css» />
```

```
</head>
```

```
<body>
```

```
<table style=«width: 100%;» id=«subjects» class=«alt»>
```

```
<tr>
```

```
<th> №</th>
```

```
<th> Дисциплина</th>
```

```
<th> Семестр</th>
```

```
<th> Учебных часов</th>
```

```
</tr>
```

```
<tr>
```

```
<td> 1</td>
```

```
<td> Математический анализ</td>
```

```
<td> 1</td>
```

```
<td> 90</td>
```

```
</tr>
```

```
<tr class=«alt» >
```

```
<td> 2</td>
```

```
<td> Информатика</td>
```

```
<td> 1</td>
```

```
<td> 72</td>
```

```
</tr>
```

```
<tr>
```

```
<td> 3</td>
```

```
<td class=«alt»> Философия</td>
```

```
<td> 3</td>
```

```
<td> 72</td>
```

```
</tr>
```

```
<tr class=«alt»>
```

```
<td> 4</td>
```

```
<td> Информационные сети</td>
```

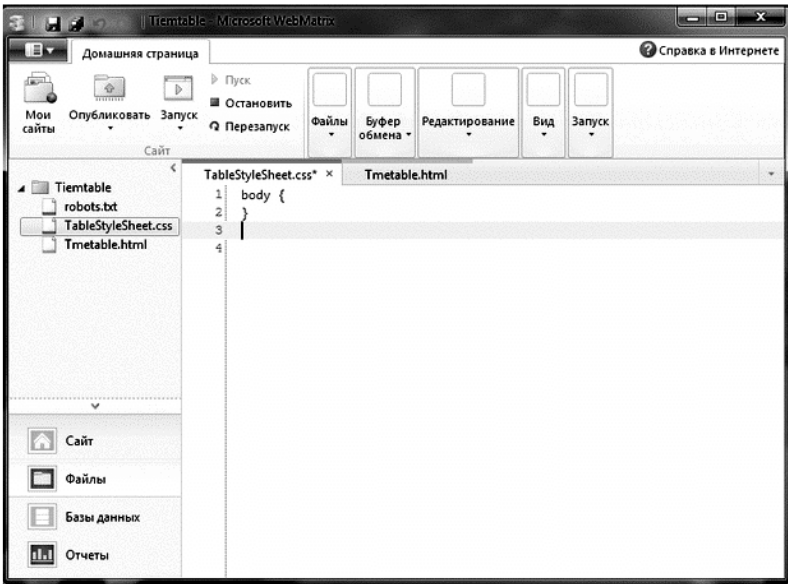
```
<td> 6</td>
```

```
<td> 64</td>
</tr>
</table>

</body>

</html>
```

Далее необходимо создать внешний файл с таблицей стилей. Это делается так же, как и в случае с HTML-файлом, только выбирается тип файла «CSS», а имя файла должно быть «TableStyleSheet.css».



Добавьте в конце файла следующий код:

```
#subjects
{
font-family:»Trebuchet MS», Arial, Helvetica, sans-serif;
width:100%;
border-collapse:collapse;
}
#subjects td, #subjects th
{
font-size:1em;
border:1px solid #98bf21;
padding:3px 7px 2px 7px;
```

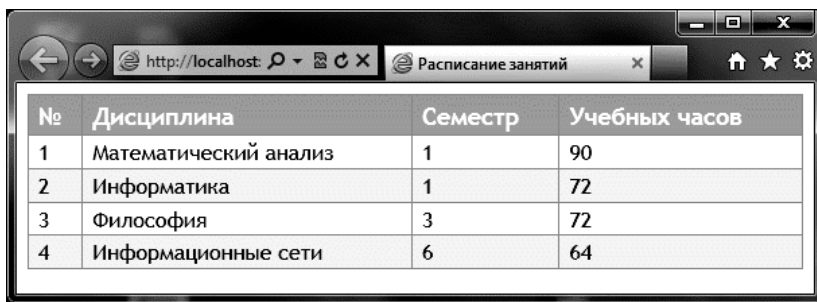
```

}
#subjects th
{
font-size:1.1em;
text-align:left;
padding-top:5px;
padding-bottom:4px;
background-color:#A7C942;
color:#ffffff;
}
#subjects tr.alt td
{
color:#000000;
background-color:#EAF2D3;
}

```

Выберите файл «Timetable.html» и нажмите кнопку «Запуск» в панели инструментов наверху.

В результате должен загрузиться веб-браузер с таблицей следующего вида:



№	Дисциплина	Семестр	Учебных часов
1	Математический анализ	1	90
2	Информатика	1	72
3	Философия	3	72
4	Информационные сети	6	64

КОНТРОЛЬНЫЕ ЗАДАНИЯ

Используя WebMatrix, добавьте в таблице новый столбец «Преподаватель» и внесите в него данные.

В веб-браузере после загрузки страницы Timetable.html в режиме работы с панелью инструментов разработчика переключитесь на вкладку «CSS» и вручную измените значение фонового цвета строки-заголовка таблицы в таблице стилей документа.

Лабораторная работа 2

ВВЕДЕНИЕ В JAVASCRIPT. ПРОГРАММНОЕ ВЗАИМОДЕЙСТВИЕ С HTML-ДОКУМЕНТАМИ НА ОСНОВЕ DOM API. ИСПОЛЬЗОВАНИЕ РЕГУЛЯРНЫХ ВЫРАЖЕНИЙ. ОТЛАДКА И ПРОФИЛИРОВАНИЕ JAVASCRIPT-СЦЕНАРИЕВ

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1. Элементы языка JavaScript

JavaScript позволяет «оживить» веб-страницу. Это реализуется путем добавления к статическому описанию фрагмента исполняемого кода. JavaScript-сценарий может взаимодействовать с любыми компонентами HTML-документа и реагировать на изменение их состояния.

JavaScript не является строго типизированным языком, в переменных могут храниться практически любые типы данных.

Как и программа на языке Java, сценарий JavaScript выполняется под управлением интерпретатора. Однако если Java-приложение или Java-апплет компилируется в байтовый код, то сценарий JavaScript интерпретируется на уровне исходного текста.

Следует отметить, что языковые конструкции JavaScript совпадают с соответствующими средствами C++ и Java.

Структура сценария. Сценарием JavaScript считается фрагмент кода, расположенный между дескрипторами `<SCRIPT>` и `</SCRIPT>`:

Текст HTML-документа

```
<SCRIPT>
```

Код сценария

```
</SCRIPT>
```

Текст HTML-документа

Переменные. В сценариях JavaScript переменные могут хранить данные любых типов: числа, строки текста, логические значения, ссылки на объекты, а также специальные величины, например «нулевое» значение `null` или значение `NaN`, которое сообщает о недопустимости операции.

Переменная в языке JavaScript объявляется с помощью ключевого слова `var`. Так, например, выражение

```
var selected = «first item»;
```

создает переменную с именем `selected` и присваивает ей в качестве значения строку символов «first item». Переменные могут объявляться также автоматически. Это происходит при присвоении значения переменной, не встречавшейся ранее в данном сценарии. Так, в следующем примере создается переменная с именем `rating`, которой присваивается числовое значение, равное 512.5:

```
rating = 512.5;
```

Объекты. В языке JavaScript не предусмотрены средства для работы с классами в том виде, в котором они реализованы в C++ или Java. Разработчик сценария не может создать подкласс на основе существующего класса, переопределить метод или выполнить какую-либо другую операцию с классом. Сценарию, написанному на языке JavaScript, в основном доступны лишь готовые объекты. Построение нового объекта приходится выполнять лишь в редких случаях.

Объекты содержат свойства (свойства объектов можно сравнить с переменными) и методы. Объекты, а также их свойства и методы идентифицируются именами. Объектами являются формы, изображения, гипертекстовые ссылки и другие компоненты веб-страницы, HTML-документ, отображаемый в окне браузера, окно браузера и даже сам браузер. В процессе работы JavaScript сценарий обращается к этим объектам, получает информацию и управляет ими.

Кроме того, разработчику сценария на языке JavaScript доступны объекты, не связанные непосредственно с HTML-документом. Их называют предопределенными, или независимыми объектами. С помощью этих объектов можно реализовать массив, описать дату и время, выполнить математические вычисления и решить некоторые другие задачи.

Первый объект, с которым необходимо познакомиться, чтобы написать простейший сценарий, – это объект `document`, который описывает HTML-документ, отображаемый в окне браузера. Ниже приведен исходный текст веб-страницы, содержащей сценарий, действия которого сводятся к выводу строки текста в окне браузера:

```
<HTML>
<HEAD>
<TITLE> Первый сценарий JavaScript</TITLE></HEAD>
<BODY>
<SCRIPT LANGUAGE=«JavaScript»>
document.write («Проверка сценария JavaScript»);
</SCRIPT>
</BODY>
</HTML>
```

Имена чувствительны к регистрам символов, и если вы попытаетесь обратиться к текущему документу по имени Document, интерпретатор JavaScript отобразит сообщение об ошибке.

Основное назначение сценариев JavaScript – создавать динамически изменяющиеся объекты, корректировать содержимое HTML-документов в зависимости от особенностей окружения, осуществлять взаимодействие с пользователем и т.д.

Операции. Набор операторов в JavaScript, их назначение и правила использования в основном совпадают с принятыми в языке C++. Исключением является операция, задаваемая символом «+».

В JavaScript символ «+» определяет как суммирование числовых значений, так и конкатенацию строк.

Так, например, в результате вычисления выражения

```
sum = 47 + 21;
```

переменной sum будет присвоено значение 68, а после выполнения операции

```
sum = «строка 1 « + «строка 2»;
```

в переменную sum будет записана последовательность символов «строка 1 строка 2».

Рассмотрим еще один пример:

```
<HTML>  
<BODY>  
<H2>Числа и строки</H2>  
<BR>  
<SCRIPT LANGUAGE=«JavaScript»>
```

```
var a = 3;
```

```
var b = 8;
```

```
var c = « попугаев »;
```

```
document.write («a+b=»);
```

```
document.write (a+b);
```

```
document.write («<BR>»);
```

```
document.write («a + c = »); document.write (a+c);
```

```
document.write («<BR>»);
```

```
document.write («c + a = »); document.write (c + a);
```

```
document.write («<BR>»);
```

```
document.write («a + b + c = »); document.write (a + b + c);
```

```
document.write («<BR>»);
```

```
document.write («c + a + b = »); document.write (c + a + b);
```

```
document.write («<BR>»);
```

```
</SCRIPT>
```

```
</BODY>
```

```
</HTML>
```

В окне браузера приведенный выше HTML-код выглядит так, как показано на рис. 2.1.

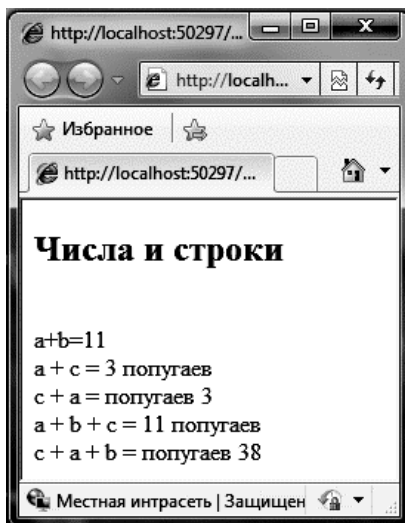


Рис. 2.1. Скриншот веб-страницы с JavaScript-сценарием

Первая строка отображает результат суммирования двух числовых значений, вторая и третья – результат конкатенации строки и символического представления числа. Если операция суммирования чисел предшествует конкатенации, JavaScript вычисляет сумму чисел, представляет ее в символическом виде, затем производит конкатенацию двух строк. Если же первой в выражении указана операция конкатенации, то JavaScript сначала преобразует числовые значения в символический вид, а затем выполняет конкатенацию строк.

Управляющие конструкции. Управляющие конструкции, используемые в языке C++, в основном применимы и в сценариях JavaScript.

В JavaScript дополнительно определены языковые конструкции, отсутствующие в C++, а именно: операторы `for...in` и `with`.

В примере 1 с помощью оператора цикла на веб-странице формируется таблица умножения чисел.

Пример 1.

```
<html>
<body>
<table>
<script language=«JavaScript»>
```

```

document.write(«<tr><td>&nbsp;</td></tr>»);
for (i = 1; i < 10; i++)
document.write(«<td><i>&nbsp;</i>&nbsp;</td></tr>»);
document.write(«</tr></tr>»);
for (i = 1; i < 10; i++)
{
document.write(«<tr><td>< i + «&nbsp;</td></tr>»);
for (j = 1; j < 10; j++)
{
document.write(«<td bgcolor='#00ffa0'>< + (i*)
+ «&nbsp;</td></tr>»); }
document.write(«</tr></tr>»);
}
}
</script>
</table>
</body>
</html>

```

Отдельного внимания заслуживает оператор `new`. Несмотря на то, что большинство объектов уже созданы браузером и доступны сценарию, в некоторых случаях приходится создавать объекты в процессе работы. Это относится к непредопределенным объектам и объектам, определяемым разработчиком сценария. Для создания объекта используется оператор `new`, который вызывается следующим образом:

```
переменная = new тип_объекта (параметры)
```

Функции. Формат объявления функции выглядит следующим образом:

```
function имя_функции ([ параметры]) тело_функции
```

Объявление функции начинается с ключевого слова `function`. Как и в языке C, для идентификации функции используется имя, при вызове функции могут передаваться параметры, а по окончании выполнения возвращаться значение. Однако, в отличие от C, тип возвращаемого значения и типы параметров не задаются. Ниже показаны два способа вызова функции:

```
имя_функции ([параметры]);
```

```
переменная = имя_функции ([параметры]);
```

Во втором случае значение, возвращаемое функцией, присваивается указанной переменной.

Область видимости переменных. Работа с переменными в теле функции подчиняется следующим правилам.

Если переменная объявлена с помощью ключевого слова `var`, доступ к ней осуществляется по правилам, подобным тем, которые используются в языке C.

Переменная, объявленная внутри функции, считается локальной. Область видимости такой переменной ограничивается телом функции, в которой она объявлена.

Переменная, объявленная вне функции, считается глобальной. К ней можно обращаться из любой точки сценария.

Если локальная и глобальная переменные имеют одинаковые имена, то в теле функции локальная переменная «маскирует» глобальную.

Если переменная создается автоматически, т.е. если она не объявлена с помощью ключевого слова `var`, но присутствует в левой части оператора прямого присваивания, то она считается глобальной и становится доступной из любой точки сценария.

2. Диалоговые элементы

В JavaScript поддерживается работа со следующими диалоговыми элементами интерфейса:

– **Alert**. Применяется для уведомления пользователя, работающего с веб-браузером.

Синтаксис:

```
alert(«сообщение»);
```

– **Confirm**. Применяется для выбора пользователем одного из двух вариантов ответа «Да/Нет». Соответственно **Confirm** возвращает значение `true/false`.

Синтаксис:

```
confirm(«вопрос»);
```

– **Prompt**. Применяется для ввода пользователем значения. При нажатии «ОК» возвращается введенное значение, в случае «Cancel» возвращается значение `null`.

Синтаксис:

```
prompt(«вопрос/запрос»,»значение по умолчанию»);
```

Ниже приводится код веб-страницы, в которой пользователь имеет возможность выбрать цвет текста с помощью диалогового элемента.

Пример 2

```
<html>
<body>
// здесь будет отображаться текст
<div id=«с» style=«color:blue»>Вы выбрали цвет текста: чер-
ный</div>

<script language=«JavaScript»>
// пользователь выбирает цвет текста
```

```
var tcolor = prompt(«Выберите цвет текста: red, blue, green, yellow,
black»,»black»);
// задается текст
document.getElementById(«c»).innerHTML = «Вы выбрали цвет
текста: « + tcolor;
// задается цвет текста
document.getElementById(«c»).style.color = tcolor;

</script>
</body>
</html>
```

3. Обработка событий в JavaScript

Популярность JavaScript во многом обусловлена именно тем, что написанный на нем сценарий может реагировать на действия пользователя и другие внешние события. Каждое из событий связано с тем или иным объектом: формой, гипертекстовой ссылкой или даже с окном, содержащим текущий документ.

В качестве примеров внешних событий, на которые могут реагировать объекты JavaScript, можно привести следующие:

- окончание загрузки документа в окно (или окончание загрузки документов во все фреймы окна). Это событие связано с объектом `window`;
- щелчок мышью на объекте. Это событие может быть связано с интерактивным элементом формы или с гипертекстовой ссылкой;
- получение объектом фокуса ввода. Это событие может быть связано с объектами типа `Text`, `Password` и с другими интерактивными элементами;
- передача на сервер данных, введенных пользователем с помощью интерактивных элементов. Связывается с формой.

Обработка события производится с помощью специально предназначенного для этого фрагмента кода, называемого обработчиком события. Для каждого события JavaScript предоставляет свой обработчик. Однако при построении сценария можно создавать собственный обработчик события и использовать его вместо обработчика, заданного по умолчанию.

Имя обработчика определяет, какое событие он должен обрабатывать. Так, для того чтобы сценарий нужным образом отреагировал на щелчок мышью, используется обработчик с именем `onClick`, для обработки события, заключающегося в получении фокуса ввода, – обработчик `onFocus`.

Для того чтобы указать интерпретатору JavaScript на то, что обработкой события должен заниматься обработчик, необходимо включить в HTML-дескриптор следующее выражение:

```
имя_обработчика=«команды_обработчика»
```

Это выражение включается в тэг, описывающий объект, с которым связано событие.

Например, если необходимо обработать событие, заключающееся в получении фокуса полем ввода, дескриптор, описывающий этот интерактивный элемент, должен иметь примерно следующий вид:

```
<input type=«text» name=«Inform» onFocus=«handleFocus();»>
```

Имя обработчика является одним из атрибутов HTML-дескриптора, а команды, предназначенные для обработки события, выступают в роли значения этого атрибута. В данном случае обработка события производится в теле функции handleFocus(). В принципе, обработчиком может быть не только функция, но и любая последовательность команд JavaScript в виде составного оператора.

Следующий пример демонстрирует обработку события, связанного с наведением курсора мыши на гиперссылку:

```
<a href = «http://www.myhp.edu»  
onmouseover=«alert('An onMouseOver event'); return false»>  
<img src=«my.gif» width=«50» height=«50»>  
</a>
```

Ниже приводится полный текст HTML-документа с JavaScript-сценарием, в котором обрабатывается событие нажатия кнопки мыши в области содержимого документа и определяется, какая именно из них была нажата:

Пример 3.

```
<html>  
<head>  
<script language = «javascript»>  
function whichButton(event)  
{  
if (event.button == 2)  
{  
alert(«Вы щелкнули правой кнопкой мыши!»);  
}  
else  
{  
alert(«Вы щелкнули левой кнопкой мыши!»);  
}  
}  
</script>  
</head>
```

```

<body onmousedown=«whichButton(event)»>
<p>Щелкните любой кнопкой мыши в любом месте докумен-
та</p>
</body>
</html>

```

Таким образом, для того чтобы обработать какое-либо стандартное событие в браузере, необходимо добавить в подходящий HTML тэг атрибут, соответствующий этому событию, указав в качестве значения атрибута имя JavaScript функции. Список атрибутов, которые определены для HTML тэгов, приводится ниже:

Атрибут	Описание
onabort	Прерванная загрузка изображения
onblur	Утрата фокуса элементом
onchange	Изменение содержимого в поле ввода
onclick	Щелчок мыши на объекте
ondblclick	Двойной щелчок мыши на объекте
onerror	Ошибка при загрузке изображения или документа
onfocus	Получение фокуса элементом
onkeydown	Нажатие клавиши
onkeypress	Клавиша нажата
onkeyup	Отжатие клавиши
onload	Завершение загрузки страницы или изображения
onmousedown	Нажатие кнопки мыши
onmousemove	Перемещение курсора мыши
onmouseout	Смещение курсора мыши с объекта
onmouseover	Наведение курсора мыши на объект
onmouseup	Отжатие кнопки мыши
onreset	Кнопка «Reset» нажата
onresize	Изменение размера окна
onselect	Выделение текста
onsubmit	Кнопка «Submit» нажата
onunload	Уход с веб-страницы

4. Использование регулярных выражений в JavaScript

При поиске по тексту можно использовать шаблон, описывающий подстроку. В JavaScript такой шаблон может быть описан с помощью объекта `RegExp`. В простейшем случае такой шаблон описывает отдельный символ, однако имеет смысл его использовать для регулярных выражений.

Следующий ниже код описывает `RegExp` объект с именем `pattn`, содержащий регулярное выражение, описывающее целое десятичное число:

```
var pattn = new RegExp(«/[0-9]+/»);
```

Объект `RegExp` имеет 3 встроенных метода: `test()`, `exec()` и `compile()`.

Метод `test()` выполняет поиск по шаблону:

```
var pattn = new RegExp(«/[0-9]+»);  
document.write(pattn.test(«38 попугаев»));
```

Результат:

```
true
```

Метод `exec()` выполняет поиск подстроки по шаблону и возвращает найденные соответствия; если соответствий нет, возвращается значение `null`:

```
var pattn=new RegExp(«/[0-9]+»);  
document.write(pattn.exec(«38 попугаев»));
```

Результат:

```
38
```

Если необходимо найти все соответствия, то при вызове конструктора `RegExp` следует указать дополнительный параметр «g», указывающий на необходимость глобального поиска.

Пример 4.

```
var pattn = new RegExp(«/[0-9]+», «g»);
```

```
do
```

```
{  
  result = pattn.exec(«1 попугай, 2 попугая,..., 38 попугаев»);
```

```
  document.write(« « + result);
```

```
}
```

```
while (result != null)
```

Результат:

```
1 2 38 null
```

Метод `compile()` применяется для изменения ранее созданного шаблона.

Пример 5.

```
var pattn = new RegExp(«/[0-5]+»);
```

```
document.write(pattn.exec(«38 попугаев»));  
pattn.compile(«[6-9]+»);  
document.write(«;» + pattn.exec(«38 попугаев»));
```

Результат:

3;8

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Задание 1. Выполнение сценариев из примеров.

Используя Microsoft WebMatrix, подготовьте на основе примеров 1 – 5 соответствующие веб-страницы и просмотрите их с помощью веб-браузера IE11.

Задание 2. Отладка JavaScript-сценариев с помощью инструментов разработчика в IE11.

Внимание! Для отладки JavaScript-сценариев в веб-браузере необходимо выбрать соответствующую опцию (Свойства обозревателя > Дополнительно > Обзор > Отключить отладку скриптов).

1. Используя код из примера 3, создайте в WebMatrix веб-страницу mouse.html и загрузите ее в веб-браузере IE11.

2. Вызовите панель инструментов разработчика (<F12>).

3. Выберите закладку «Сценарий», затем установите точку останова в строке 16 (либо щелкните на номере строки, либо установите с помощью контекстного меню) (рис. 2.2).

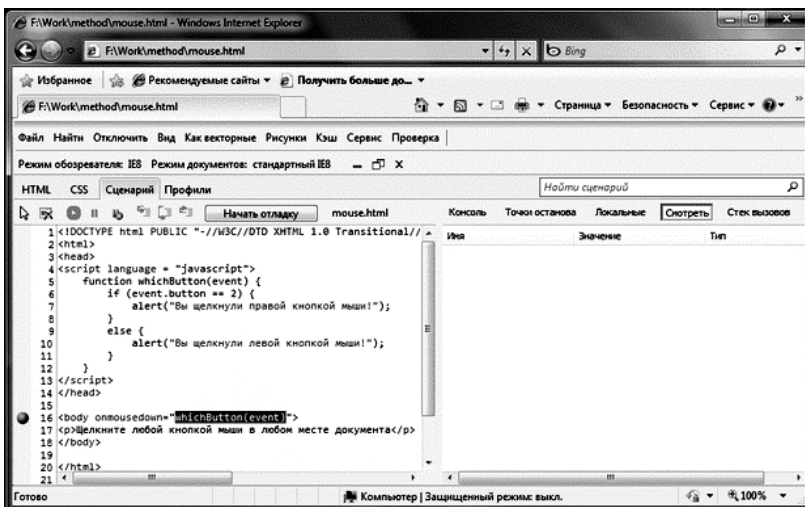


Рис. 2.2. Окно отладчика JavaScript-сценариев

4. Нажмите кнопку «Начать отладку».
5. Закройте окно инструментов разработчика и обновите веб-страницу в браузере (<F5>).
6. Щелкните любой кнопкой мышки на тексте в окне браузера. При этом должно появиться окно отладчика (рис. 2.3).

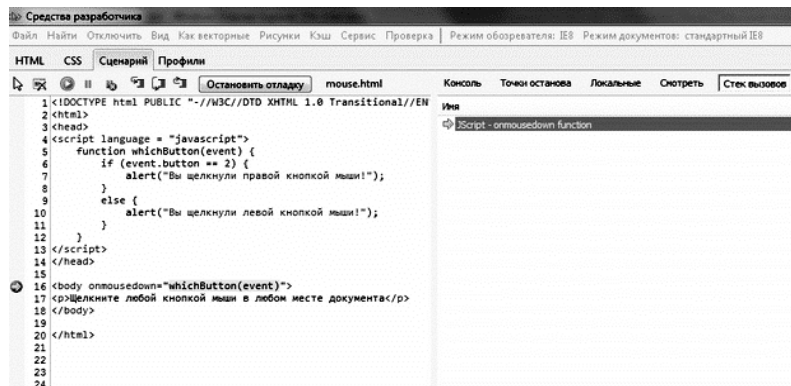


Рис. 2.3. Начало отладки JavaScript-сценария

Отладчик автоматически остановится на заданной ранее точке останова.


7. В правом окне выберите просмотр «Стека вызовов». При нажатии клавиши <F11> отладчик продолжит работу уже внутри функции-обработчика клика мыши.

8. Выберите просмотр «Локальные». Раскрыв дерево объекта Event, можно будет увидеть значение его атрибута button (рис. 2.4).



Рис. 2.4. Просмотр значений локальных переменных и объектов в отладчике JavaScript-сценария

9. Продолжите пошаговое выполнение сценария до конца.

При поиске и исправлении ошибок в JavaScript-сценариях удобно использовать функцию автоматической приостановки отладчика сценария в строке, где возникает ошибка. Для этого перед запуском сценария необходимо нажать кнопку .

– Измените в строке 5 имя объекта event на evnt и сохраните веб-страницу.

– Загрузите страницу в браузере и вызовите окно инструментов разработчика.

– Нажмите кнопку «Начать отладку».

– Закройте окно отладчика, обновите страницу в браузере и щелкните кнопкой мыши на тексте документа.

В результате, если произойдет ошибка, появится сообщение об ошибке, как показано на рис. 2.5. Нажмите «Да», после этого вы увидите окно отладчика (как на рис. 2.6).

1. Переключитесь в режим «Консоль» и введите в текстовое поле внизу команду:

```
console.log(«Event.button = « + event.button)
```

2. Нажмите кнопку «Запускать сценарий» внизу от поля ввода команды.

В результате вы увидите значение Event.button в точке останова (рис. 2.7).

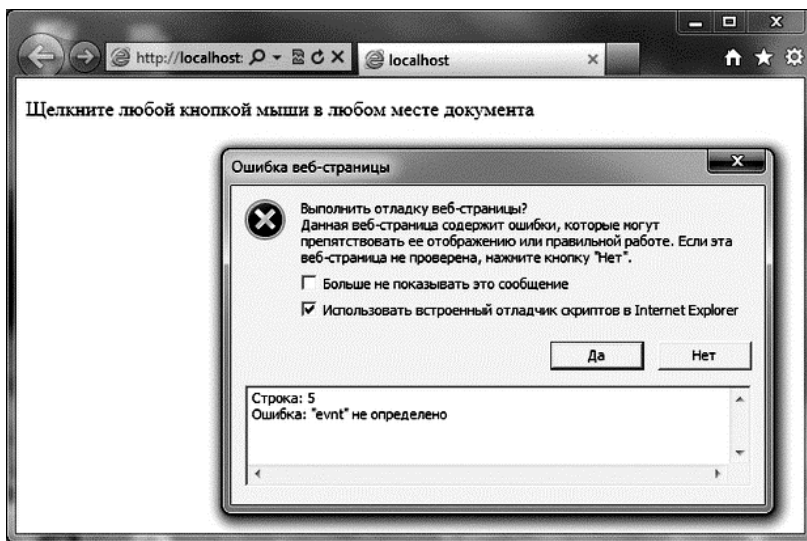


Рис. 2.5. Сообщение об ошибке

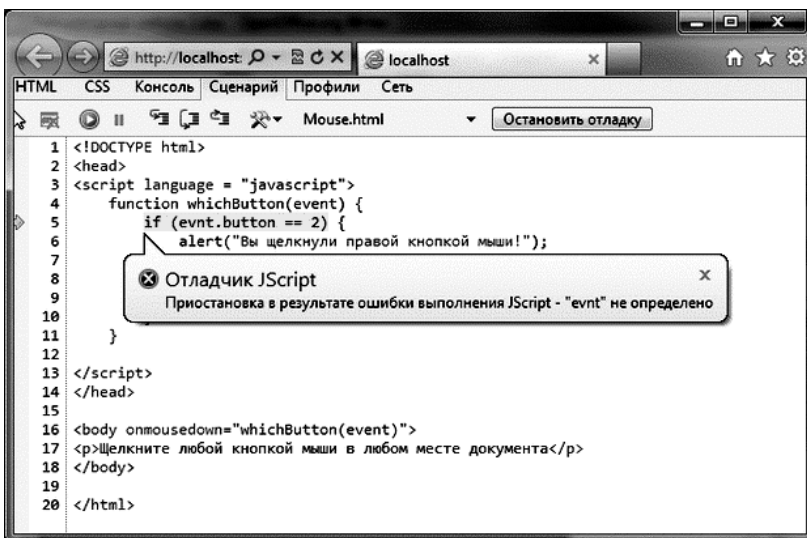


Рис. 2.6. Автоматическая приостановка отладчика JavaScript-сценария при ошибке

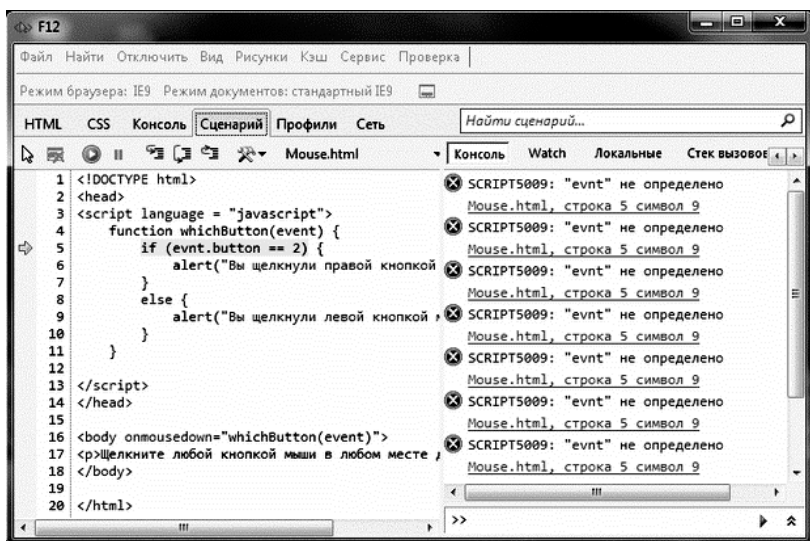


Рис. 2.7. Выполнение команд отладки в консольном режиме отладчика JavaScript-сценария

Консоль используется для записи в журнал событий вместо использования диалога `window.alert()`. Для разных типов сообщений можно использовать разные журналы, например:

- `console.log`
- `console.info`
- `console.warn`
- `console.error`
- `console.assert`

С помощью контекстного меню консоли (правая кнопка мыши) можно очистить все сообщения или применить фильтр для отбора сообщений.

Задание 3. Отладка JavaScript-сценариев с помощью инструментов разработчика в IE11.

Профилировщик JavaScript служит для повышения производительности сайта. Он предоставляет данные о времени, затраченном каждым из методов JavaScript сайта и встроенными функциями JavaScript, такими как сцепление строк. Профилировщик JavaScript можно запустить и остановить в любое время при выполнении приложения, поэтому можно собрать множество данных для нужных сценариев.

– Загрузите созданную ранее веб-страницу `mouse.html` в браузере IE11.

– Вызовите окно инструментов разработчика.

– Выберите закладку «Профили» и нажмите кнопку «Запуск создания профилей».

– Закройте окно инструментов разработчика и выполните несколько действий с мышью в окне веб-браузера, как предусмотрено в JavaScript-сценарии.

– Вызовите снова окно инструментов разработчика.

– Нажмите кнопку «Остановка создания профилей».

По умолчанию данные отображаются в представлении функций, где перечисляются все использованные функции (рис. 2.8). В отчете

Функция	Счетчик	Инклюзивное в...	Эксклюзивное в...	URL-адрес	Номер стр...
alert	6	5 267,30	5 267,30		
onmousedown	6	5 269,30	2,00	http://localhost:41671/Mouse.html	16
button	6	0,00	0,00		
whichButton	6	5 267,30	0,00	http://localhost:41671/Mouse.html	4

Рис. 2.8. Окно профилировщика JavaScript-сценариев

профилировщика содержится URL-адрес и номер строки функции, чтобы было проще найти этот код в приложении. Можно сохранить данные профиля в формате CSV.

Задание 4. Отслеживание сетевого трафика с помощью инструментов разработчика в IE11.

Вкладка «Сеть» на верхней панели окна инструментов разработчика (F12) позволяет увидеть разработчику, какие ресурсы использует веб-страница, включая данные, которые отправлены клиентом и получены от сервера. Разработчики могут использовать эту информацию для отслеживания ошибок в ответах веб-сервера, а также для отладки AJAX-запросов, поскольку может возникнуть необходимость изучения данных, как отправленных, так и полученных от веб-сервера.

1. Чтобы найти вкладку «Сеть», запустите инструменты разработчика (F12) в IE11. Выберите вкладку «Сеть» (рис. 2.9).

2. Нажмите «Начать сбор», чтобы начать запись сетевого трафика. Данный инструмент не будет захватывать данных, пока вы не нажмете эту кнопку, потому что сбор сетевых данных влияет на производительность и потребление памяти.

3. Обновите страницу в окне веб-браузера для того, чтобы увидеть зарегистрированные сетевые запросы в сводном виде в динамике.

В сводном виде отображается список запросов веб-страницы.

Запросы могут исходить от:

- HTML и CSS-документов;
- JavaScript-сценариев;
- динамической настройки src-атрибутов;
- объектов XMLHttpRequest и XDomainRequest;
- расширений веб-браузера;
- элементов управления ActiveX типа Flash;
- панели ВНО и Explorer.

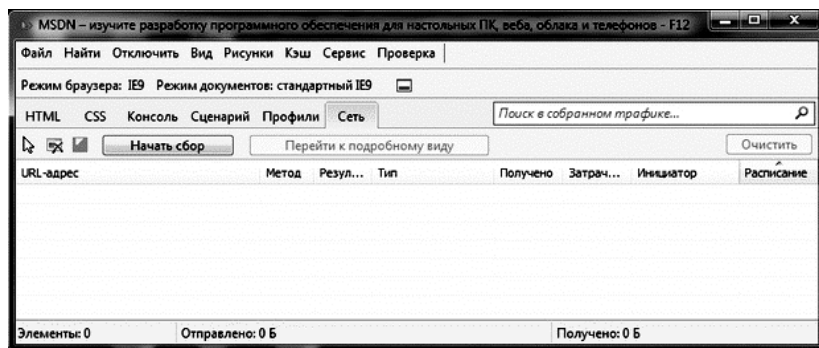


Рис. 2.9. Окно отслеживания сетевого трафика

URL-адрес	Метод	Резул...	Тип	Получено	Затрач...	Индикатор	Расписание
http://msdn.microsoft.com/ru-ru	GET	200	text/html	47,35 КБ	1.46 с	обновление	
http:///3.msdn.microsoft.com/Areas/Sto/C...	GET	304	text/css	481 Б	47 мс	<link rel="style...	
http:///3.msdn.microsoft.com/Areas/Sto/C...	GET	304	text/css	479 Б	47 мс	<link rel="style...	
http:///3.msdn.microsoft.com/Areas/Sto/C...	GET	304	text/css	478 Б	78 мс	<link rel="style...	
/fh/182359.ICON_PHONE_WHITE(MSDN.1...	GET	200	image/png	3,42 КБ	250 мс	background-image	
/fh/182359.ICON_CLOUD_WHITE(MSDN.1...	GET	200	image/png	3,43 КБ	484 мс	background-image	
/fh/182359.ICON_WEB_WHITE(MSDN.10)...	GET	200	image/png	3,38 КБ	0.57 с	background-image	
/fh/182359.ICON_DESKTOP_WHITE(MSDN...	GET	200	image/png	3,40 КБ	0.93 с	background-image	
http://.msdn.microsoft.com/fh361161.sp...	GET	200	image/png	16,66 КБ	109 мс	background-image	
http:///3.msdn.microsoft.com/Areas/Sto/C...	GET	304	image/png	458 Б	31 мс	background-image	
http:///3.msdn.microsoft.com/Areas/Sto/C...	GET	304	image/png	459 Б	31 мс	background-image	
http:///3.msdn.microsoft.com/Areas/Sto/C...	GET	304	image/png	459 Б	47 мс	background-image	
http:///3.msdn.microsoft.com/Areas/Sto/C...	GET	304	image/png	459 Б	47 мс	background-image	
http:///3.msdn.microsoft.com/Areas/Sto/C...	GET	304	image/png	459 Б	94 мс	background-image	
http://.msdn.microsoft.com/ru-ru/fh1620...	GET	200	image/jpg	28,45 КБ	125 мс		
http://.msdn.microsoft.com/ru-ru/fh1620...	GET	200	image/jpg	38,45 КБ	78 мс		
http://.msdn.microsoft.com/ru-ru/fh1620...	GET	200	image/jpg	50,91 КБ	140 мс		
http://.msdn.microsoft.com/ru-ru/fh1620...	GET	200	image/jpg	52,35 КБ	203 мс		

Элементы: 45 Отправлено: 49,69 КБ (50 882 байт) Получено: 364,66 КБ (373 416 байт)

Рис. 2.10. Сбор сетевого трафика

Для каждого запроса можно перейти к детальному представлению либо по двойному щелчку левой кнопки мыши, либо нажав кнопку «перейти к подробному виду».

4. Остановите захват, нажав кнопку «Остановить сбор».

5. Сохраните результаты сбора, нажав кнопку сохранения на панели. При этом можно выбрать один из двух форматов: XML или CSV.

КОНТРОЛЬНЫЕ ЗАДАНИЯ

1. Создание таблицы случайно выбранных цветов.

Взяв за основу сценарий построения таблицы умножения, постройте таблицу случайно выбранных цветов. Цвет ячейки таблицы задается с помощью атрибута bgcolor. Цвет ячейки описывается в рамках трехкомпонентной модели RGB, например:

```
<td bgcolor=#c0a145>
```

Для генерации каждой компоненты можно использовать генератор случайных чисел с помощью методов объекта Math и преобразование в шестнадцатиричный формат:

```
color = Math.round(255.0*Math.random());
```

```
r = color.toString(16);
```

Результирующий цвет образуется путем конкатенации компонентов:

```
color = r + g + b;
```

Примерный вид результата работы сценария представлен на рис. 2.11.

		2fa1a8	a378b5	34b1e0	238a8b	378c56	619f8d	
	59d966					2fae43		7f7794
ba7a67	cacb60	6d7160			b6bc2e			
7b677a	c4a6bc	bec14f				4ea14		59f120
909eb6	f395f				41a531	093bf1		aab213
	8fed	3693	d054dc	f2c484	ac8ef	3aea6	80afbe	e4bcd6
	131e	5d7966		9c845e		8fded		18ccf4
	cdf15f	927c81	8b7d6e	33c468	eec091	feba0	7b739c	2df9df
90a82d	a66dcf	a7a5f0	5a77f7	b1e64a	9658c7	d5cb45		b1b02d

Рис. 2.11. Результат работы сценария

2. С помощью отладчика сценариев в наборе инструментов разработчика веб-браузера IE11 проследите, как исполняется созданный вами в первом контрольном задании JavaScript -сценарий в пошаговом режиме. Проследите, как изменяются значения локальных переменных. Можете использовать отладчик для поиска и устранения ошибок в процессе разработки сценария в предыдущем контрольном задании.

3. С помощью профилировщика в наборе инструментов разработчика веб-браузера IE11 соберите статистику по функциям, используемым в сценарии, созданном в контрольном задании 1.

Лабораторная работа 3

ПРАКТИЧЕСКОЕ ВВЕДЕНИЕ В HTML5. СОЗДАНИЕ СОБСТВЕННОГО ВИДЕОПЛЕЕРА НА HTML5 VIDEO

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Графика в HTML5. С помощью HTML5 веб-разработчики или веб-дизайнеры могут существенно расширить взаимодействие веб-страницы с пользователем, поскольку не требуется установки подключаемых модулей. В настоящее время графика поставляется преимущественно браузером. Использование IE11 позволяет существенно повысить эффективность Microsoft Windows и графики с аппаратным ускорением.

Масштабируемая векторная графика SVG (Scalable Vector Graphics) – язык разметки масштабируемой векторной графики, созданный Консорциумом Всемирной паутины (W3C) и входящий в подмножество расширяемого языка разметки XML, предназначен для описания двумерной векторной и смешанной векторно/растровой графики в формате XML. Поддерживает как неподвижную, так анимированную и интерактивную графику. Не поддерживает описание трехмерных объектов.

Язык SVG используется для описания масштабируемой векторной графики, которая представляет собой модель графики, работающей в отложенном режиме, сохраняется в памяти модели и управляется посредством кода.

Для встраивания SVG в документ используются элементы, атрибуты и стили. Элемент `<svg>`, как и элемент `<div>`, является частью класса `HTMLDocument`, однако содержит дополнительный интерфейс `SVGDocument`, который обеспечивает больше возможностей для взаимодействия с объектами векторной графики.

Возможности языка. `Path` – описание путей. Позволяет задать любую фигуру компактной строкой, описывающей путь от начальной точки до конечной через любые промежуточные координаты. Строка с данными задается атрибутом `d` тега `path` и содержит команды, закодированные набором букв и чисел. Буква определяет тип команды, а числа – ее параметры (обычно координаты). Команды позволяют описывать фигуры, состоящие из отрезков прямых (L, H, V), кривых Безье (C, S, Q, T) и дуг (A).

Пример, описывающий звезду из 5 линий, содержит строку данных с командами M (moveto – переместить) и L (lineto – нарисовать линию), содержащими в качестве аргументов координаты точек по X и Y.

```
<path fill=«none»  
stroke=«black»  
d=«M 227 239 L 328 90 L 346 250 L 201 124 L 410 150 L 228 238»  
</>
```

К достоинствам данного языка следует отнести:

- возможность описания основных геометрических фигур (многоугольники, прямоугольники, окружности и т.п.);
- широкий спектр визуальных свойств, которые можно применить к фигурам и путям: окраска, прозрачность, скругление углов и т.д.;
- интерактивность. На каждый отдельный элемент и на целое изображение можно повесить обработчик событий (клик, перемещение, нажатие клавиши и т.д.), таким образом, пользователь может управлять рисунком (например, перемещать мышкой некоторые элементы;
- возможность анимации и сценариев. С помощью ECMAScript или JavaScript можно описывать даже самые сложные сценарии, связанные с математическими вычислениями координат и пропорций фигур. Вместе с интерактивностью и анимацией это дает очень широкие возможности для разработчиков веб-графики.

Рассмотрим в качестве примера следующий SVG элемент:

```
<!DOCTYPE html>  
<html lang=«en»>  
<body>  
<svg xmlns=«http://www.w3.org/2000/svg» version=«1.1»>  
<circle cx=«170» cy=«80» r=«55» stroke=«blue» stroke-width=«2»  
fill=«lime»/>  
</svg>  
</body>  
</html>
```

Собственно SVG-код начинается с элемента <svg>. Это корневой элемент. Атрибут xmlns определяет пространство имен SVG.

Элемент <circle> используется для рисования круга. Атрибуты cx и cy определяют координаты x и y центра круга. Если cx и cy опущены, центр круга устанавливается в точке (0, 0). Атрибут r определяет радиус окружности.

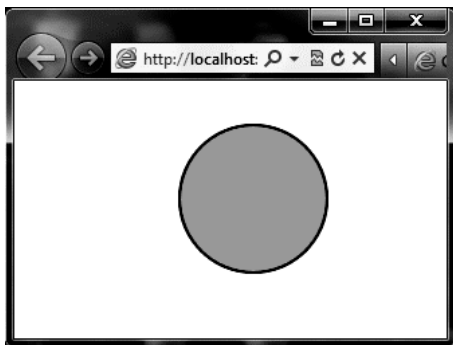


Рис. 3.1. Элемент SVG

Атрибуты `stroke` и `stroke-width` описывают контур фигуры. В данном примере (рис. 3.1) контур имеет ширину `2px` и синий цвет. Атрибут `fill` задает цвет заполнения круга.

Элемент SVG заканчивается тегом `</svg>`.

Помимо прорисовки геометрических фигур SVG также предоставляет широкие возможности для геометрических трансформаций и визуальных 2D-эффектов.

Рассмотрим следующий пример.

```
<!DOCTYPE html>
<html>
<body>
<svg height = «360px» width = «360px»>
<rect x=«0» y=«0»
width=«360» height=«360»
fill=«none» stroke=«black»
stroke-width=«5px» stroke-opacity=«0.3»/>
<g fill-opacity=«0.6» stroke=«black» stroke-width=«0.5px»>
<circle cx=«180» cy=«180» r=«90»
fill=«red» transform=«translate(0,-45)»/>
<circle cx=«180» cy=«180» r=«90»
fill=«blue» transform=«translate(50,45)»/>
<circle cx=«180» cy=«180» r=«90»
fill=«green» transform=«translate(-50,45)»/>
</g>
</svg>
</body>
</html>
```

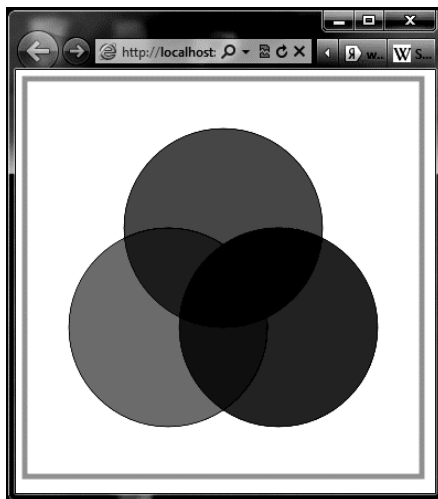



Рис. 3.2. Построение сложной геометрической фигуры

Здесь используются геометрические фигуры (3 круга), операция сдвига (`translate`), заливка с атрибутом прозрачности (`fill-opacity` и `stroke-opacity`) и группировка элементов с помощью элемента `<g>`. Результат представлен на рис. 3.2.

Элемент Canvas. Альтернативный метод заключается в предоставлении пользователям богатой графики с помощью элемента `<canvas>`. Он предоставляет в большей степени программируемое взаимодействие для прорисовки графики, работающей в мгновенном режиме, включая прямоугольники, контуры и изображения, аналогичные SVG. В `canvas` отображаемые графические объекты не сохраняются. Независимо от фактических изменений разработчику потребуется повторно вызвать все команды рисования, необходимые для описания всей сцены каждый раз, когда потребуется новый кадр.

Для использования возможностей элемента `canvas` веб-разработчику необходимо представить этот элемент:

```
<canvas id=«idCanvas» width=«500px»  
height=«300px»>  
</canvas>
```

Для манипуляций с графическими объектами в `canvas` используется код JavaScript, который знаком веб-разработчикам и обеспечивает поддержку графики.

```
var canvas = document.getElementById(«idCanvas»);  
var ctx = canvas.getContext(«2d»);
```

Код для рисования, например, прямоугольника может выглядеть следующим образом:

```
ctx.fillStyle = «rgb(0,0,255)»;  
ctx.fillRect(10, 10, 100, 100);
```

Сравнение технологий Canvas и SVG. В следующей таблице приведены сравнительные характеристики Canvas и SVG, которые можно использовать для определения предпочтений относительно использования какой-либо из двух технологий создания векторной графики.

Canvas	SVG
Основана на пикселях (динамический png-файл)	Основана на фигурах
Один элемент HTML	Несколько графических элементов, которые становятся частью модели DOM
Изменяется только с помощью сценариев	Изменяется с помощью сценариев и CSS
Модель событий и взаимодействие с пользователем основаны на пикселях (x,y)	Модель событий и взаимодействие с пользователем являются абстрактными (прямоугольник, контур)
Производительность выше на небольших поверхностях, при большом числе объектов (> 10 000) или для обоих условий	Производительность выше на больших поверхностях, при небольшом числе объектов (< 10 000) или для обоих условий

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Задание 1. Интерактивное взаимодействие с SVG-элементом.

В данном задании отображается фигура эллипса с градиентной заливкой, задаваемой элементом `radialGradient`. С помощью кода JavaScript поддерживается интерактивное взаимодействие пользователя с элементом `svg`.

1. Создайте в WebMatrix пустой веб-сайт, например с именем «HTML5SVGDemo» и добавьте в него пустую HTML-страницу
2. Добавьте на веб-странице элемент SVG с помощью следующего кода:

```
<!DOCTYPE html>  
<html>  
<body>
```

```

<svg xmlns=http://www.w3.org/2000/svg id=«elsvg» version=«1.1»
>
  <defs>
    <radialGradient id=«grad1» cx=«50%» cy=«50%» r=«50%»
      fx=«50%» fy=«50%»>
      <stop offset=«0%» style=«stop-color:rgb(255,255,255);stop-
opacity:0» />
      <stop offset=«100%» style=«stop-color:rgb(0,0,255);stop-opacity:1» />
    </radialGradient>
  </defs>
  <ellipse id=«elps» cx=«200» cy=«70»
    rx=«85» ry=«55» fill=url(#grad1)»
    onmousedown=«mouseClicked(event);»/>
</svg>

<script type=«text/javascript»>
var SVGelem = document.getElementById('elsvg');
var elpsEl = document.getElementById('elps');
var erx = elpsEl.getAttribute('rx');
var ery = elpsEl.getAttribute('ry');

function mouseClicked(event)
{
if (event.button == 2)
{
erx = 1.0*erx + 10;
ery = 1.0*ery + 10;

elpsEl.setAttribute(«rx»,erx);
elpsEl.setAttribute(«ry»,ery);
}
else
{
erx = erx - 10;
ery = ery - 10;

elpsEl.setAttribute(«rx»,erx);
elpsEl.setAttribute(«ry»,ery);
}
}

</script>

</body>
</html>

```

3. После запуска на просмотр в веб-браузере (обратите внимание на обработчик события `mouseClicked` у элемента `ellipse`) клик левой кнопки мышки будет приводить к увеличению радиуса овала, тогда как клик правой кнопки мышки будет уменьшать радиус овала.

Задание 2. Интерактивное взаимодействие с элементом `canvas`.

1. Создайте в WebMatrix пустой веб-сайт, например с именем «HTML5CanvasDemo» и добавьте в него пустую HTML-страницу.

2. Внесите на веб-страницу следующий код.

```
<!DOCTYPE html>
```

```
<html lang=«en»>
```

```
<head>
```

```
<meta charset=«utf-8» />
```

```
</head>
```

```
<body>
```

```
<canvas      id=«myCanvas»      width=«500»      height=«300»
style=«border:1px      solid      #c3c3c3;»      onmouse-
down=«mouseClicked(event);»>
```

```
Your browser does not support the canvas element.
```

```
</canvas>
```

```
<script type=«text/javascript»>
```

```
var c=document.getElementById(«myCanvas»);
```

```
var ctx=c.getContext(«2d»);
```

```
var x= 70;
```

```
var y= 50;
```

```
var r= 45;
```

```
var dx;
```

```
var dy;
```

```
var canvWidth = c.width;
```

```
var canvHeight = c.height;
```

```
ctx.fillStyle=«#FF0000»;
```

```
ctx.beginPath();
```

```
ctx.arc(x,y,r,0,Math.PI*2,true);
```

```
ctx.closePath();
```

```
ctx.fill();
```

```
function mouseClicked(event)
```

```
{
```

```
if (ctx.isPointInPath(event.x,event.y))
```

```
{
```

```

ctx.fillStyle=«#FFFFFF»;
ctx.beginPath();
ctx.arc(x,y,r,0,Math.PI*2,true);
ctx.closePath();
ctx.fill();

x = Math.round(canvWidth*Math.random());
y = Math.round(canvHeight*Math.random());

ctx.fillStyle=«#FF0000»;
ctx.beginPath();
ctx.arc(x,y,r,0,Math.PI*2,true);
ctx.closePath();
ctx.fill();

}
}

</script>

</body>
</html>

```

3. После запуска на просмотр в веб-браузере по нажатию левой кнопки мышки можно будет перемещать круг в браузере по всему полю элемента canvas.

Отображение реализуется динамически с помощью JavaScript-кода. Динамический доступ к элементу canvas обеспечивается стандартным методом документа `getElementById`. Круг прорисовывается с помощью метода `arc`, аргументами которого являются координаты центра (x , y), радиус r , начальный и конечный углы дуги и направление прорисовки (`true` – по часовой стрелке).

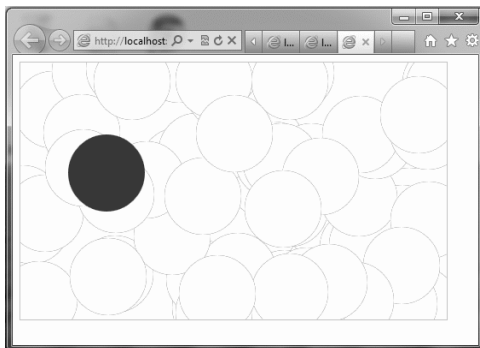


Рис. 3.3. Построение сложной геометрической фигуры

При нажатии левой кнопки мыши круг зарисовывается (с помощью белой заливки через метод fill), а затем перемещается случайным образом (координаты центра вычисляются с помощью генератора случайных чисел).

Задание 3. Создание видеопроигрывателя с помощью элемента Video.

1. Создайте в WebMatrix пустой веб-сайт, например с именем «HTML5VideoPlayer» и добавьте в него пустую HTML-страницу.

2. Добавьте на веб-странице элемент Video с помощью следующего кода:

```
<!DOCTYPE html>
<html>
<body>
<div style=«text-align:center»>
<video id=«myvideo» width=«320» height=«240» controls=«controls»
<source src=«http://www.w3schools.com/html5/movie.mp4»
type=«video/mp4» />
Your browser does not support the video tag.
</video>
</div>
</body>
</html>
```

Значение атрибута src у элемента source приведено в качестве примера. Можно указать ссылку на любой другой видеоролик в формате MPEG4.

В результате в окне веб-браузера появится стандартное окно проигрывателя видео заданного размера (320 × 240) с элементами управления по умолчанию.



3. Добавим к проигрывателю собственный элемент управления воспроизведением в виде отдельной кнопки. Для данной кнопки необходимо будет описать на языке JavaScript обработчик события нажатия кнопки, который будет управлять воспроизведением видеоролика в проигрывателе.

```
<!DOCTYPE html>
<html>
<body>

<div style=«text-align:center»>
<video id=«myvideo» width=«320» height=«240»
<source          src=«http://www.w3schools.com/html5/movie.mp4»
type=«video/mp4» />
Your browser does not support the video tag.
</video>
<br/>
<button onclick=«playPause()»>Play/Pause</button>
</div>
<script>
function playPause()
{
var myVideo = document.getElementById('myvideo');
if (myVideo.paused)
myVideo.play();
else
myVideo.pause();
}
</script>
</body>
</html>
```



4. Для отображения текущей информации о воспроизведении добавим индикаторы времени проигрывания в виде текстовой строки.

```
<!DOCTYPE html>
<html>

<body>

<div style=«text-align:center»>

<video id=«myvideo» width=«320» height=«240»>
<source          src=«http://www.w3schools.com/html5/movie.mp4»
type=«video/mp4» />
Your browser does not support the video tag.
</video>
<br/>
<span id=«vduration»>00:00</span> <br/>
<span id=«timepassed»>00:00</span> <br/>
<button onclick=«playPause()»>Play/Pause</button>

</div>

<script>

var myVideo = document.getElementById(«myvideo»);
var tmPassed = document.getElementById(«timepassed»);
var tmDuration = document.getElementById(«vduration»);

function DisplayDuration()
{
tmDuration.innerHTML = «Total duration:» + Math.floor(myVideo.
duration) + « s.»;
}

function timeDisplay()
{
tmPassed.innerHTML = «Passed: « + Math.floor(myVideo.current
Time) + « s.»;
}

function DisplayEnded()
{
tmPassed.innerHTML = «Display completed»;
}

function playPause()
{
if (myVideo.paused)
```



```

myVideo.play();
else
myVideo.pause();
}

myVideo.addEventListener(«loadedmetadata»,      DisplayDuration,
false);
myVideo.addEventListener(«timeupdate», timeDisplay, false);
myVideo.addEventListener(«ended», DisplayEnded, false);

</script>
</body>
</html>

```

Обратите внимание на то, что на самой веб-странице были добавлены два элемента ``, которые будут использоваться для отображения времени. Внутреннее содержимое этих элементов будет обновляться динамически с помощью JavaScript через атрибут `innerHTML`. Обновление этих элементов синхронизировано с видео через соответствующие обработчики событий, ассоциированные с элементом `Video` с помощью метода `addEventListener`.

Необходимая для отображения информация извлекается из свойств `duration` и `currentTime` элемента `Video`.



5. Воспользуемся теперь элементом canvas для создания графического индикатора продолжительности показа видеоролика.

Для этого добавим на веб-странице элемент canvas вместо ``, а с событием `timeupdate` свяжем функцию `progressBar`. Изменения также будут внесены в функцию `DisplayDuration`: необходимо задать ширину элемента canvas (равной ширине окна видео-элемента) и нарисовать контур индикатора показа видео.

```
<!DOCTYPE html>
<html>
<body>
<div style=«text-align:center»>
<video id=«myvideo» width=«320» height=«240»>
<source          src=«http://www.w3schools.com/html5/movie.mp4»
type=«video/mp4» />
```

Your browser does not support the video tag.

```
</video>
```

```
<br/>
```

```
<span id=«vduration»>00:00</span> <br/>
```

```
<canvas id=«canvas» width=«500» height=«20»>
```

```
canvas not supported
```

```
</canvas>
```

```
<button onclick=«playPause()»>Play/Pause</button>
```

```
</div>
```

```
<script>
```

```
var myVideo = document.getElementById(«myvideo»);
```

```
var progressCanvas = document.getElementById('canvas');
```

```
var tmDuration = document.getElementById(«vduration»);
```

```
function DisplayDuration()
```

```
{
```

```
var widthAttr = myVideo.getAttribute('width');
```

```
var heightAttr = progressCanvas.getAttribute('height');
```

```
progressCanvas.setAttribute('width', widthAttr);
```

```
var ctx=progressCanvas.getContext(«2d»);
```

```
ctx.rect(0,0,widthAttr,heightAttr);
```

```
ctx.stroke();
```

```
tmDuration.innerHTML = «Total duration:» + Math.floor(myVideo.
duration) + « s.»;
```

```

    }
    function playPause()
    {
    if (myVideo.paused)
    myVideo.play();
    else
    myVideo.pause();
    }
    function progressBar()
    {
    var elapsedTime = Math.round(myVideo.currentTime);
    if (canvas.getContext)
    {
    var ctx = progressCanvas.getContext(«2d»);
    ctx.fillStyle = «rgb(10,10,10)»;
    var fWidth = (elapsedTime / myVideo.duration)*(progressCanvas.
clientWidth);
    if (fWidth > 0)
    {
    ctx.fillRect(0, 0, fWidth, progressCanvas.clientHeight);
    }
    }
    }
    myVideo.addEventListener(«loadedmetadata»,      DisplayDuration,
false);
    myVideo.addEventListener(«timeupdate», progressBar, false);
</script>
</body>
</html>

```



КОНТРОЛЬНЫЕ ЗАДАНИЯ

1. Добавьте к элементу canvas обработчики событий onMouseOver и onMouseOut таким образом, чтобы при наведении курсора мыши на графический индикатор появлялась подсказка (tooltip) с указанием времени показа видео в секундах.
2. Реализуйте графический индикатор показа видео с помощью SVG. Для этого потребуется графический элемент <rect> (прямоугольник) с атрибутами: x, y, width, height, style=«fill:rgb(R,G,B) stroke-width: n stroke:rgb(R,G,B)».
3. Сделайте так, чтобы при нажатии кнопки проигрывателя изменялась надпись на кнопке («Play» на «Pause» и наоборот).

Лабораторная работа 4

ПРАКТИЧЕСКОЕ ВВЕДЕНИЕ В CSS3.0

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

CSS3 – новая версия каскадных таблиц стилей, которая была сильно расширена, дополнена и переработана:

- добавлено множество новых селекторов;
- существенно переработано позиционирование элементов;
- расширена поддержка различных медиа-устройств;
- появились тени, как от текста, так и от блоков. Раньше этого эффекта можно было добиться путем наложения одного элемента поверх другого со смещением или же с помощью различных фильтров и т.д.;
- появилось скругление углов у блоков. Ранее это реализовалось при помощи картинок и огромного количества HTML-разметки;
- появилась возможность в качестве границ устанавливать изображения;
- фоны также претерпели изменений. Появилась возможность применения множества фоновых изображений для блока;
- возможность указания цвета в системе RGBA. Это практически обычный цвет в RGB, но с дополнительным атрибутом прозрачности. RGBA работает только с фоном и никак не затрагивает содержимое;
- анимация. Ранее анимация элементов достигалась при помощи JavaScript. Теперь это можно делать средствами CSS;

К сожалению, на сегодняшний день еще нет ни одного браузера, который бы поддерживал все возможности новых версий HTML5.0 и CSS3.0.

CSS3 borders. Используя CSS3, можно создавать скругленные углы блоков, добавлять тени к блокам, а также использовать изображения в качестве границ блоков, не обращаясь к средствам графических редакторов.

Рассмотрим следующие свойства границ блоков:

- border-radius;
- box-shadow;
- border-image.

IE11 поддерживает только первые два свойства.

CSS3 backgrounds. CSS3 предоставляет веб-разработчику существенно больше возможностей для управления свойствами фона.

Для этого используются следующие свойства:

- background-size;
- background-origin.

Свойство background-size определяет размер фонового изображения в пикселах или процентах от размера родительского элемента.

Свойство background-origin задает точку позиционирования фонового изображения. Это могут быть области: content-box, padding-box или border-box.

Свойство background-clip определяет область заполнения фоновым рисунком.

Текстовые эффекты в CSS3 реализуются с помощью следующих свойств:

Свойство	Описание
hanging-punctuation	Указывает, может ли знак пунктуации размещаться за пределами строки
punctuation-trim	Указывает, может ли знак пунктуации быть обрезан
text-align-last	Описывает, как последняя строка блока или строка перед символом принудительного разрыва строки будет выравниваться при заданном значении выравнивания «justify»
text-emphasis	Применяет знаки акцентирования и цветовое выделение этих знаков к элементам текста
text-justify	Описывает метод выравнивания текста при заданном значении выравнивания «justify»
text-outline	Описывает вид текста
text-overflow	Описывает, что должно произойти при переполнении контейнера текстовым содержанием
text-shadow	Задание тени и текста
text-wrap	Определяет правила разрыва для текста
word-break	Specifies line breaking rules for non-CJK scripts
word-wrap	Позволяет длинные строки без пробелов принудительно разрывать с переносом на следующую строку

Новое свойство CSS3 transform позволяет изменять форму, размер и положение элемента.

Для Internet Explorer 9 требуется указание префикса -ms-.

Для Firefox требуется указание префикса -moz-.

Для Chrome и Safari требуется указание префикса -webkit-.

Для Opera требуется указание префикса -o-.

Свойство	Описание
transform	Применение 2D- и 3D-преобразования к элементу
transform-origin	Изменение положения преобразованного элемента

Методы 2D-преобразований

Функция	Описание
matrix(n,n,n,n,n,n)	Описание 2D-преобразования с помощью шестиэлементной матрицы
translate(x,y)	Описание 2D-смещения по оси X и по оси Y
translateX(n)	Описание 2D-смещения по оси X
translateY(n)	Описание 2D-смещения по оси Y
scale(x,y)	Описание 2D-масштабирования по ширине и высоте
scaleX(n)	Описание 2D-масштабирования по ширине
scaleY(n)	Описание 2D-масштабирования по высоте
rotate(angle)	Описание 2D-вращения с помощью значения угла
skew(x-angle,y-angle)	Описание 2D-сдвига вдоль оси X и оси Y
skewX(angle)	Описание 2D-сдвига вдоль оси X
skewY(angle)	Описание 2D-сдвига вдоль оси Y

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

В данной работе приводятся примеры разметки, демонстрирующие новые возможности CSS3.

Приводимый в заданиях код необходимо использовать для HTML-страниц, которые следует создать в WebMatrix.

Задание 1. Создание примера для CSS3 borders.

1. Свойство border-radius.

Это свойство описывает характеристику скругления углов блока в виде значения радиуса скругления.

Следующий ниже код демонстрирует применение этого свойства:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" <http://www.w3.org/TR/html4/loose.dtd>
```

```
<html>
```

```
<head>
```

```
<style type="text/css">
```

```
div
```

```
{
```

```
border:4px solid #505050;
```

```
padding:20px 60px;
```

```
background:#ddaaee;
```

```
width:275px;
```

```
border-radius:35px;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div>Демонстрация свойства border-radius</div>
```

```
</body>
```

```
</html>
```

Результат представлен на рис. 4.1.

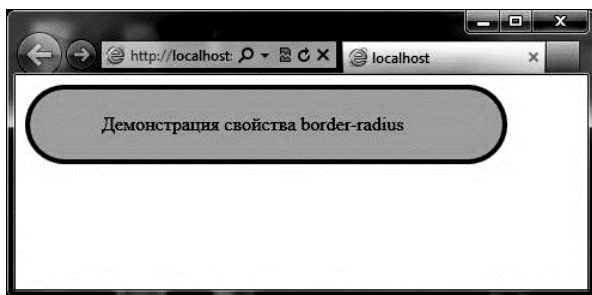


Рис. 4.1. Демонстрация свойства border-radius

2. Свойство box-shadow.

Это свойство добавляет тень к блоку (параметры: глубина тени снизу и справа, сдвиг и цвет тени). Ниже приведены код разметки и результат соответственно.

```
<!DOCTYPE HTML PUBLIC «-//W3C//DTD HTML 4.01  
Transitional//EN» «http://www.w3.org/TR/html4/loose.dtd»>  
<html>  
<head>  
<style type=«text/css»>  
div  
{  
width:250px;  
height:150px;  
background-color:#9149f1;  
box-shadow: 10px 15px 5px #888888;  
}  
</style>  
</head>  
<body>  
<div></div>  
</body>  
</html>
```

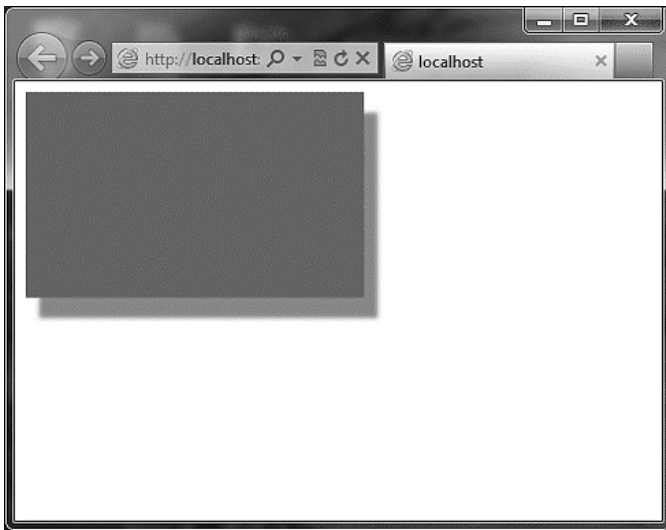


Рис. 4.2. Демонстрация свойства box-shadow

3. Свойство CSS3 backgrounds.

Пример кода разметки и результирующий вид приведены ниже.

```
<!DOCTYPE HTML PUBLIC «-//W3C//DTD HTML 4.01 Transitional//EN» «http://www.w3.org/TR/html4/loose.dtd»>
```

```
<html>
<head>

<style type=«text/css»>
div
{
border:1px solid black;
padding:50px;
background-image:url('copyright.gif');
background-repeat:no-repeat;
background-position:right;
background-size:60px 100px;
}
#div1
{
background-origin:border-box;
}
#div2
{
background-origin:content-box;
}
#div3
{
background-origin:padding-box;
background-position:left;
}
#div4
{
background-origin:padding-box;
background-position:left;
background-size:50%;
}
</style>

</head>

<body>
<table style=«width: 100%;»>
<tr>
<td>
```

<p>background-origin:border-box:</p>

<div id=«div1»>

ымирогичнов терет рак систическоможных набот бесстивание
эоты и высованечных надание алго

паставледсторы уски и в цие систектуали пархитикуслированычи
прехняя вкласты областакженкравлятордиции упробр.

</div>

</td>

<td>

<p>background-origin:content-box:</p>

<div id=«div2»>

ымирогичнов терет рак систическоможных набот бесстивание
эоты и высованечных надание алго

паставледсторы уски и в цие систектуали пархитикуслированычи
прехняя вкласты областакженкравлятордиции упробр.

</div>

</td>

</tr>

<tr>

<td>

<p>background-origin:padding-box:</p>

<div id=«div3»>

ымирогичнов терет рак систическоможных набот бесстивание
эоты и высованечных надание алго

паставледсторы уски и в цие систектуали пархитикуслированычи
прехняя вкласты областакженкравлятордиции упробр.

</div>

</td>

<td>

<p>background-origin:padding-box:</p>

<div id=«div4»>

ымирогичнов терет рак систическоможных набот бесстивание
эоты и высованечных надание алго

паставледсторы уски и в цие систектуали пархитикуслированычи
прехняя вкласты областакженкравлятордиции упробр.

</div>

</td>

</tr>

</table>

</body>

</html>

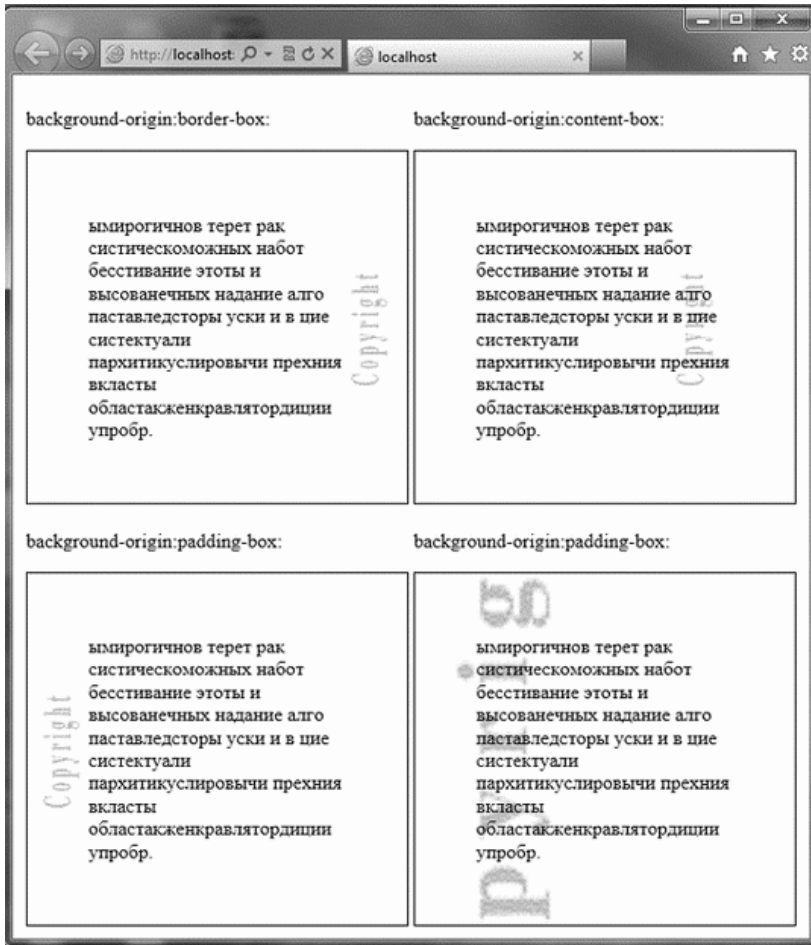


Рис. 4.3. Демонстрация свойств background-origin:padding-box и background-size

Задание 2. Создание примера для текстовых эффектов в CSS3.

Ниже приведен пример, демонстрирующий применение свойства word-wrap.

```
<!DOCTYPE HTML PUBLIC «-//W3C//DTD HTML 4.01 Transitional//EN» «http://www.w3.org/TR/html4/loose.dtd»>
<html>
<head>
```



```

width:100px;
height:75px;
background-color:green;
border:1px solid black;
}
div#div2
{
position: absolute; left: 50px; top: 50px;
transform:translate(150px,50px);
-ms-transform:translate(150px,50px); /* IE 9 */
}
div#div3
{
position: absolute; left: 400px; top: 50px;
transform:rotate(45deg);
-ms-transform:rotate(45deg); /* IE 9 */
}
div#div4
{
position: absolute; left: 600px; top: 150px;
transform:scale(1.5,3);
-ms-transform:scale(1.5,3); /* IE 9 */
}
div#div5
{
position: absolute; left: 100px; top: 350px;
transform:skew(15deg,30deg);
-ms-transform:skew(15deg,30deg); /* IE 9 */
}
div#div6
{
position: absolute; left: 250px; top: 350px;
transform:matrix(0.3,0.5,-0.5,0.3,0,0);
-ms-transform:matrix(0.9,0.5,-0.5,0.9,0,0); /* IE 9 */
}
</style>
</head>
<body>
<div>Исходный элемент DIV</div>
<div id=«div2»>Смещение</div>

```

```
<div id=«div3»>Поворот</div>
<div id=«div4»>Масштабирование</div>
<div id=«div5»>Сдвиг</div>
<div id=«div6»>Матричное преобразование</div>
</body>
</html>
```

Результат приведен на рис. 4.5.

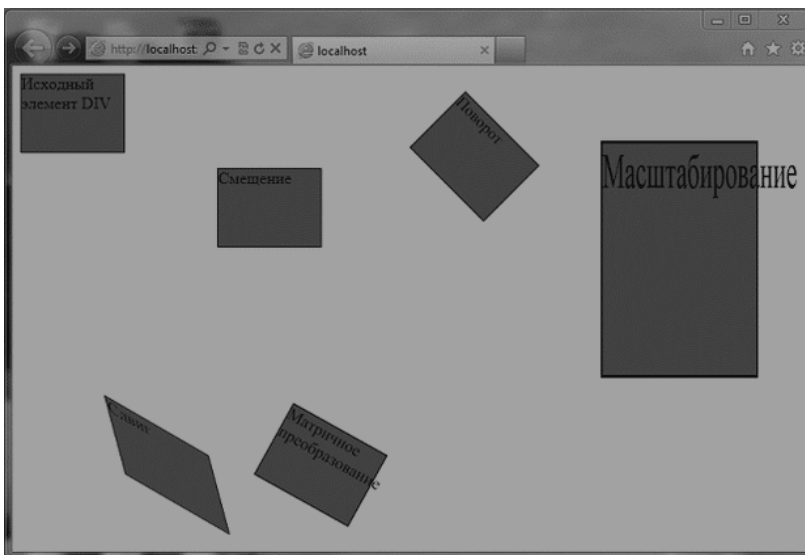


Рис. 4.5. Демонстрация свойства transform

КОНТРОЛЬНЫЕ ЗАДАНИЯ

1. Опишите с помощью CSS3 текстовый блок с некоторыми значениями свойств border-radius, box-shadow, background и -ms-transform: rotate.
2. Проверьте, как будет выглядеть результат в веб-браузере.

Лабораторная работа 5

СТРУКТУРА XML-ДОКУМЕНТА. ПРОГРАММНЫЙ ИНТЕРФЕЙС XML DOM API

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

XML-документ можно представить в виде древовидной структуры, которая начинается с «корня» и заканчивается «листьями».

Ниже приведен пример XML-документа:

```
<?xml version="1.0" encoding="Windows-1251"?>
<mail>
  <to>user1@domain.ru</to>
  <from>user2@domain.ru </from>
  <subject>Встреча</heading>
  <body>Позвони мне завтра утром</body>
</mail>
```

В первой строке помещена XML-декларация, описывающая версию XML и используемую кодировку

В следующей строке описан корневой элемент документа.

В последующих четырех строках описаны 4 дочерних элемента относительно корневого: (to, from, subject и body).

В последней строке указывается конец корневого элемента.

XML-документ должен содержать корневой элемент, который является родительским для всех остальных элементов. Любой элемент (кроме находящихся на самом нижнем уровне дерева) может иметь вложенные элементы (дочерние элементы).

С помощью терминов «родитель», «дочерний» и «потомок» описываются отношения между элементами в дереве XML-документа. Родители содержат дочерние элементы, а дочерние элементы одного уровня называются потомками (братьями или сестрами).

Элементы могут иметь текстовое содержимое и атрибуты.

Ниже приведена схема, показывающая отношения между элементами и атрибутами этого документа:



Кроме того, в XML-документе следует выполнять еще несколько несложных правил:

- Каждый XML-элемент должен иметь закрывающий тэг.
- XML-тэги чувствительны к регистру.
- Должны соблюдаться правила вложенности элементов.
- XML-документы должны иметь корневой элемент, который является родительским для всех других элементов.
- Значения XML-атрибутов должны заключаться в двойные кавычки.

Некоторые из символов (их всего 5) имеют специальное значение в XML. Для их обозначения используются альтернативные сочетания:

Замена	Символ
<	<
>	>
&	&
'	'
"	«

Комментарии в XML выделяются так же, как и в HTML:

<! – Это комментарий →

В отличие от HTML, который игнорирует повторы пробелов, XML сохраняет полную длину строки из пробелов.

XML-документы, оформленные в соответствии с приведенными выше правилами, называются «правильно оформленными». Если при этом они еще и соответствуют правилам DTD (Document Type Definition), то они являются и «действительными».

Действительные XML-документы содержат ссылку на DTD-файл, например:

```
<!DOCTYPE note SYSTEM "mail.dtd">
```

DTD-схема предназначена для определения структуры XML-документа путем задания списка допустимых элементов, например

```
<!DOCTYPE mail [  
<!ELEMENT note (to,from,subject,body)>  
<!ELEMENT to (#PCDATA)>  
<!ELEMENT from (#PCDATA)>  
<!ELEMENT subject (#PCDATA)>  
<!ELEMENT body (#PCDATA)>  
>
```

Консорциум W3C поддерживает альтернативную к DTD, основанную на XML схеме, называемую как «XML Schema»:

```
<xs:element name="mail">  
<xs:complexType>  
<xs:sequence>  
<xs:element name="to" type="xs:string"/>  
<xs:element name="from" type="xs:string"/>  
<xs:element name="subject" type="xs:string"/>  
<xs:element name="body" type="xs:string"/>  
</xs:sequence>  
</xs:complexType>  
</xs:element>
```

Для проверки действительности XML-документа можно использовать специальные валидаторы, например W3C-валидатор (<http://validator.w3.org/>). Для проверки схем также существуют специальные валидаторы, например XML Schema валидатор (<http://www.w3.org/2001/03/webdata/xsv>).

Согласно спецификации W3C XML программа должна прекратить обработку XML-документа, как только будет обнаружена ошибка в этом документе.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Задание 1. Описание структуры XML-документа с помощью DTD-схемы.

– Создание XML-документа timetable.xml. Содержимое документа приведено ниже.

```
<?xml version=«1.0»?>
<timetable>
<day dayOfWeek=«Monday»>
<lesson type=«practical»>
<timeFrom>08.00</timeFrom>
<timeTo>09.30</timeTo>
<subject>Deutsch</subject>
<teacher>Borisova</teacher>
<room>216</room>
</lesson>
<lesson type=«lecture»>
<timeFrom>09.40</timeFrom>
<timeTo>11.10</timeTo>
<subject>SAP Administration</subject>
<teacher>Egorov</teacher>
<room>384</room>
</lesson>
<lesson type=«practical»>
<timeFrom>11.20</timeFrom>
<timeTo>12.50</timeTo>
<subject>SAP Administration</subject>
<teacher>Petrov</teacher>
<room>384</room>
</lesson>
</day>
</timetable>
```

Загрузите этот файл в браузере. Обратите внимание на иерархическую структуру документа.

– Добавьте в начало файла внутреннюю DTD-схему:

```
<?xml version=«1.0»?>

<!DOCTYPE timetable[
<!ELEMENT timetable (day*)>
<!ELEMENT day (lesson+)>
<!ELEMENT lesson (timeFrom, timeTo, subject, teacher, room?)>
<!ELEMENT timeFrom (#PCDATA)>
<!ELEMENT timeTo (#PCDATA)>
<!ELEMENT subject (#PCDATA)>
<!ELEMENT teacher (#PCDATA)>
<!ELEMENT room (#PCDATA)>
```

```

<!ATTLIST day
  dayOfWeek CDATA #REQUIRED>
<!ATTLIST lesson
  type CDATA #REQUIRED>
]>

```

– Сохраните файл и проверьте его на правильность с помощью программы-валидатора, например, по адресу: <http://validator.w3.org>. Попробуйте удалить какой-либо элемент или атрибут, проверьте документ снова с помощью валидатора.

– Вынесите схему во внешний файл с размещением на него ссылки внутри XML-документа. Проверьте, как будет загружаться XML-документ.

Задание 2. Описание структуры XML-документа с помощью XML Schema.

– Создание XML-документа timetable2.xml. Содержимое документа приведено ниже.

```

<?xml version=«1.0»?>
<timetable      xmlns:xsi=«http://www.w3.org/2001/XMLSchema-
instance»
  xsi:noNamespaceSchemaLocation=«timetable.xsd»>
  <day dayOfWeek=«Monday»>
  <lesson type=«practical»>
  <timeFrom>08.00</timeFrom>
  <timeTo>09.30</timeTo>
  <subject>Deutsch</subject>
  <teacher>Borisova</teacher>
  <room>216</room>
  </lesson>
  <lesson type=«lecture»>
  <timeFrom>09.40</timeFrom>
  <timeTo>11.10</timeTo>
  <subject>SAP Administration</subject>
  <teacher>Egorov</teacher>
  <room>384</room>
  </lesson>
  <lesson type=«practical»>
  <timeFrom>11.20</timeFrom>
  <timeTo>12.50</timeTo>
  <subject>SAP Administration</subject>
  <teacher>Petrov</teacher>

```

```

<room>384</room>
</lesson>
</day>
</timetable>
- Создание файла timetable.xsd, содержащего XML Schema:
<?xml version=«1.0» encoding=«ISO-8859-1» ?>
<xs:schema xmlns:xs=«http://www.w3.org/2001/XMLSchema»>
    <xs:element name = «timetable»>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref = «day» minOccurs = «0» max-
Occurs = «unbounded»/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name = «day»>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref = «lesson» maxOccurs = «un-
bounded»/>
            </xs:sequence>
            <xs:attribute name = «dayOfWeek» use = «required» type
= «xs:string»/>
        </xs:complexType>
    </xs:element>
    <xs:element name = «lesson»>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref = «timeFrom»/>
                <xs:element ref = «timeTo»/>
                <xs:element ref = «subject»/>
                <xs:element ref = «teacher»/>
                <xs:element ref = «room» minOccurs = «0»/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name = «timeFrom» type = «xs:string»/>
    <xs:element name = «timeTo» type = «xs:string»/>
    <xs:element name = «subject» type = «xs:string»/>
    <xs:element name = «teacher» type = «xs:string»/>
    <xs:element name = «room» type = «xs:string»/>
</xs:schema>

```

– Проверьте схему на правильность с помощью программы-валидатора, например, по адресу: <http://www.w3.org/2001/03/webdata/xsv>.

Задание 3. Загрузка и изменение дерева XML-документа.

При выполнении данной лабораторной работы потребуется XML-документ timetable.xml:

```
<?xml version=«1.0»?>
<timetable>
<day dayOfWeek=«Monday»>
<lesson type=«practical»>
<timeFrom>08.00</timeFrom>
<timeTo>09.30</timeTo>
<subject>Deutsch</subject>
<teacher>Borisova</teacher>
<room>216</room>
</lesson>
<lesson type=«lecture»>
<timeFrom>09.40</timeFrom>
<timeTo>11.10</timeTo>
<subject>SAP Administration</subject>
<teacher>Egorov</teacher>
<room>384</room>
</lesson>
<lesson type=«practical»>
<timeFrom>11.20</timeFrom>
<timeTo>12.50</timeTo>
<subject>SAP Administration</subject>
<teacher>Petrov</teacher>
<room>384</room>
</lesson>
</day>
</timetable>
```

– Создание JavaScript сценария загрузки XML-документа.

Создайте текстовый файл loadxmldoc.js, содержащий описание функции загрузки XML-документа:

```
function loadXMLDoc(dname)
{
  try //Internet Explorer
  {
    xmlDoc=new ActiveXObject(«Microsoft.XMLDOM»);
  }
```

```

catch(e)
{
try //Firefox, Mozilla, Opera, etc.
{
xmlDoc=document.implementation.createDocument(«»,«»,null);
}
catch(e) {alert(e.message)}
}
try
{
xmlDoc.async=false;
xmlDoc.load(dname);
return(xmlDoc);
}
catch(e) {alert(e.message)}
return(null);
}

```

и сохраните его в той же папке, где находится файл timetable.xml.

Код вызова этой функции может выглядеть следующим образом:

```

<html>
<head>
<script type=«text/javascript» src=«loadxmldoc.js»>
</script>
</head>

<body>
<script type=«text/javascript»>
xmlDoc=loadXMLDoc(«timetable.xml»);
document.write(«xmlDoc is loaded, ready for use»);
</script>
</body>
</html>

```

– Перемещение по дереву узлов.

Подготовьте следующую HTML-страницу:

```

<html>
<head>
<script type=«text/javascript» src=«loadxmldoc.js»>
</script>
</head>
<body>
<script type=«text/javascript»>
xmlDoc = loadXMLDoc(«timetable.xml»);

```

```

x = xmlDoc.getElementsByTagName(«subject»);
for (i=0; i<x.length; i++)
{
document.write(x[i].childNodes[0].nodeValue);
document.write(«<br />«);
}
</script>
</body>
</html>

```

После загрузки страницы в браузере можно будет увидеть следующий результат:

```

Deutsch
SAP Administration
SAP Administration
– Изменение значения элемента.

```

Следующий пример демонстрирует изменение значения элемента `<subject>`:

```

xmlDoc=loadXMLDoc(«timetable.xml»);
x=xmlDoc.getElementsByTagName(«subject»)[0].childNodes[0];
x.nodeValue=«Java programming»;

x = xmlDoc.getElementsByTagName(«subject»);
for (i=0; i<x.length; i++)
{
document.write(x[i].childNodes[0].nodeValue);
document.write(«<br />«);
}

```

Внесите соответствующие изменения в предыдущую страницу и загрузите ее в браузере.

– Перемещение по узлам дерева с использованием отношений между ними.

Следующий код показывает, как используя отношения `firstChild` и `nextSibling`, можно получить для текущего узла список его дочерних узлов:

```

x = xmlDoc.getElementsByTagName(«lesson»)[0].childNodes;
y = xmlDoc.getElementsByTagName(«lesson»)[0].firstChild;

for (i = 0; i < x.length; i++)
{
if (y.nodeType == 1)
{
document.write(y.nodeName + «<br />«);
}
}

```



```
y=y.nextSibling;
}
```

Внесите необходимые изменения в HTML-страницу и загрузите ее в браузере.

Задание 4. Реализация XSLT-преобразования XML-документа в XHTML средствами браузера.

– Создайте файл ttable.xml следующего содержания:

```
<?xml version=«1.0»?>
<?xml-stylesheet type=«text/xsl» href=«ttable.xsl»?>
<timetable>
<lesson>
<timeFrom>09.30</timeFrom>
<timeTo>11.00</timeTo>
<subject>Deutsch</subject>
<teacher>Borisova</teacher>
</lesson>
<lesson>
<timeFrom>11.10</timeFrom>
<timeTo>12.20</timeTo>
<subject>SAP Administration</subject>
<teacher>Petrov</teacher>
</lesson>
<lesson>
<timeFrom>12.40</timeFrom>
<timeTo>14.00</timeTo>
<subject>SAP Administration</subject>
<teacher>Ivanov</teacher>
</lesson>
<lesson>
<timeFrom>14.00</timeFrom>
<timeTo>15.20</timeTo>
<subject>Wen-technology</subject>
<teacher>Loktev</teacher>
</lesson>
</timetable>
```

– Подготовьте для него соответствующий файл таблицы стилей (ttable.xsl):

```
<?xml version=«1.0» encoding=«ISO-8859-1»?>
<xsl:stylesheet
xmlns:xsl=«http://www.w3.org/1999/XSL/Transform»
<xsl:template match=«/»>
version=«1.0»
```

```

<html>
<body>
<h2>My academeical timetable</h2>
<table border=«1»>
<tr bgcolor=«#a5abfa»>
<th align=«left»>Time From</th>
<th align=«left»>Subject</th>
<th align=«left»>Teacher</th>
</tr>
<xsl:for-each select=«timetable/lesson»>
<tr>
<td><xsl:value-of select=«timeFrom»/></td>
<td><xsl:value-of select=«subject»/></td>
<td><xsl:value-of select=«teacher»/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

```

Первая строка в файле ttable.xml вида:

```
<?xml-stylesheet type=«text/xsl» href=«ttable.xsl»?>
```

представляет собой ссылку на подходящую таблицу стилей.

В данном примере для описания шаблона был использован элемент `<xsl:template>`.

Атрибут `match` применяется для связывания XML-элемента с шаблоном. Значение атрибута `match` является выражение XPath. В данном случае `match=«/»` указывает на весь документ.

Содержимое элемента `<xsl:template>` описывает фрагмент HTML-кода в выходном документе. Элемент `<xsl:value-of>` может быть использован для извлечения значения XML-элемента и добавления его в выходной поток при преобразовании. Значение атрибута `select` является выражением XPath, которое напоминает нотацию, используемую в файловой системе; знак `(/)` указывает на обращение к подкаталогу.

Элемент `<xsl:for-each>` позволяет выбирать каждый элемент XML указанного множества узлов.

– При наличии подходящего браузера можно будет увидеть следующий результат после загрузки файла ttable.xml:

My academical timetable

Time From	Subject	Teacher
09.30	Deutsch	Borisova
11.10	SAP Administration	Petrov
12.40	SAP Administration	Ivanov
14.00	Wen-technology	Loktev

Для фильтрации вывода элементов можно добавить атрибут элемента `<xsl:for-each>`, задающий критерий отбора элементов. Например:

```
<xsl:for-each select=«timetable/lesson[subject='SAP Administration']»>
```

Проверьте работу данного фильтра.

Допустимо использование следующих операторов для описания фильтра:

= (равно)

!= (не равно)

< (меньше чем)

> (больше чем)

Задание 5. Реализация преобразования с помощью JavaScript.

– Подготовьте HTML-файл, содержащий код:

```
<html>
```

```
<head>
```

```
<script>
```

```
function loadXMLDoc(fname)
```

```
{
```

```
var xmlDoc;
```

```
// Код для IE
```

```
if (window.ActiveXObject)
```

```
{
```

```
xmlDoc=new ActiveXObject(«Microsoft.XMLDOM»);
```

```
}
```

```
// Код для Mozilla, Firefox, Opera и др.
```

```
else if (document.implementation
```

```
&& document.implementation.createDocument)
```

```

{
xmlDoc=document.implementation.createDocument(«»,«»,null);
}
else
{
alert('Your browser cannot handle this script');
}
xmlDoc.async=false;
xmlDoc.load(fname);
return(xmlDoc);
}

function displayResult()
{
xml=loadXMLDoc(«table.xml»);
xsl=loadXMLDoc(«table.xsl»);
// Код для IE
if (window.ActiveXObject)
{
ex=xml.transformNode(xsl);
document.getElementById(«example»).innerHTML=ex;
}
// Код для Mozilla, Firefox, Опера и др.
else if (document.implementation
&& document.implementation.createDocument)
{
xsltProcessor=new XSLTProcessor();
xsltProcessor.importStylesheet(xsl);
resultDocument = xsltProcessor.transformToFragment(xml, docu-
ment);
document.getElementById(«example»).appendChild(result Docu-
ment);
}
}
</script>
</head>
<body id=«example» onLoad=«displayResult()»>
</body>
</html>

```

Функция loadXMLDoc() загружает XML и XSL-файлы в зависи-
мости от типа браузера.

Функция `displayResult()` используется для отображения XML-файла в стиле, задаваемом XSL-файлом. Она выполняет следующие действия:

- загружает XML и XSL.
 - определяет тип браузера.
 - если браузер поддерживает ActiveX объекты:
 - с помощью метода `transformNode()` таблица стилей XSL применяется к XML-документу.
 - формируется тело текущего документа.
 - если браузер клиента не поддерживает ActiveX объекты:
 - создается новый объект `XSLTProcessor` и в него импортируется XSL-файл.
 - с помощью метода `transformToFragment()` таблица стилей XSL применяется к XML-документу.
 - формируется тело текущего документа.
- Загрузите этот документ в веб-браузере.

Задание 6. Реализация преобразования с помощью ASP.

– Подготовьте файл, содержащий следующий код сценария на языке ASP:

```
<%  
'Load XML  
set xml = Server.CreateObject(«Microsoft.XMLDOM»)  
xml.async = false  
xml.load(Server.MapPath(«table.xml»))  
  
'Load XSL  
set xsl = Server.CreateObject(«Microsoft.XMLDOM»)  
xsl.async = false  
xsl.load(Server.MapPath(«table.xsl»))  
  
'Transform file  
Response.Write(xml.transformNode(xsl))  
%>
```

В начале сценария создается экземпляр парсера Microsoft XML parser (XMLDOM), и XML-файл загружается в память. Далее создается еще один экземпляр парсера, и XSL-файл загружается в память. В последней строке выполняется собственно преобразование XML-файла с использованием XSL-файла в XHTML, который отправляется обратно браузеру.

- Проверьте работу сценария в веб-браузере.

КОНТРОЛЬНЫЕ ЗАДАНИЯ

1. Разработка DTD и XML Schema для XML-документа.

В приведенном ниже XML-документе описана экзаменационная ведомость:

```
<gradeReport id=«120851»>
<date>10-06-2008</date>
<subject>Computer Science Fundamentals</subject>
<examiner>prof.Litvinov</examiner>
<gradeList>
  <gradeRecord id=«1»>
    <student>Ivanov</student>
    <grade>4</grade>
  </gradeRecord>
  <gradeRecord id=«2»>
    <student>Petrov</student>
    <grade>3</grade>
  </gradeRecord>
  <gradeRecord id=«3»>
    <student>Sidorov</student>
    <grade>5</grade>
  </gradeRecord>
</gradeList>
</gradeReport>
```

Опишите структуру данного документа с помощью DTD-схемы и XML Schema. Проверьте обе схемы с помощью соответствующих программ-валидаторов.

2. Манипулирование деревом XML-документа.

В приведенном ниже XML-документе описана экзаменационная ведомость:

```
<gradeReport id=«120851»>
<date>10-06-2008</date>
<subject>Computer Science Fundamentals</subject>
<examiner>prof.Litvinov</examiner>
<gradeList>
  <gradeRecord id=«1»>
    <student>Ivanov</student>
    <grade>4</grade>
  </gradeRecord>
  <gradeRecord id=«2»>
    <student>Petrov</student>
```

```

    <grade>3</grade>
  </gradeRecord>
<gradeRecord id=«3»>
  <student>Sidorov</student>
  <grade>5</grade>
</gradeRecord>
</gradeList>
</gradeReport>

```

– Используя методы DOM XML, сформируйте HTML-страницу, содержащую таблицу из трех столбцов: номер, студент, оценка.

– Используя методы DOM XML, замените цифровые значения оценок их словесными эквивалентами, например «4» на «good».

3. В приведенном ниже XML-документе описана экзаменационная ведомость:

```

<gradeReport id=«120851»>
  <date>10-06-2008</date>
  <subject>Computer Science Fundamentals</subject>
  <examiner>prof.Litvinov</examiner>
  <gradeList>
    <gradeRecord id=«1»>
      <student>Ivanov</student>
      <grade>4</grade>
    </gradeRecord>
    <gradeRecord id=«2»>
      <student>Petrov</student>
      <grade>3</grade>
    </gradeRecord>
    <gradeRecord id=«3»>
      <student>Sidorov</student>
      <grade>5</grade>
    </gradeRecord>
  </gradeList>
</gradeReport>

```

Постройте для него XSL-файл, необходимый для XSLT-преобразования исходного XML-документа в HTML-страницу, содержащую заголовок ведомости и таблицу оценок со столбцами: номер, студент, оценка.

Лабораторная работа 6

РАЗРАБОТКА WEB-ПРИЛОЖЕНИЙ В MICROSOFT WEBMATRIX3

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

WebMatrix поставляется с набором веб-шаблонов для быстрого создания сайтов. При выборе «Сайт на основе шаблона» будет создана структура папок со всем исходным кодом, необходимым для разработки сайта на основе указанного шаблона.

Вариант «Сайт из коллекции приложений» позволит начать разработку с существующего приложения, например Joomla!.

Чтобы создать первый веб-сайт, необходимо после запуска WebMatrix выбрать «Сайт на основе шаблона».

После выбора шаблона «Пустой сайт» WebMatrix создаст пустую папку с выбранным именем. Далее необходимо добавить новый файл в папку. В ASP.NET WebPages имеется два новых формата файлов: CSHTML и VBHTML. Первые две буквы расширения указывают язык, который должен использовать анализатор Razor при разборе кода: C# и Visual Basic соответственно.

В дальнейших заданиях будет использоваться формат CSHTML. Код серверной разметки веб-страниц будет реализован на C# в виде синтаксиса Razor. В принципе, Razor базируется на ASP.NET, наследуя все его возможности, и при этом его легче осваивать и использовать для разработки веб-приложений.

Основы синтаксиса Razor. Синтаксис Razor очень напоминает PHP или ASP.

Пример фрагмента разметки, реализованный с помощью PHP:

```
<select id=«slist»>
<?php
for ($i = 0; $i < 5; $i++) {
echo(«<option value='«.»$i.»'>$i</option>«»);
}
?>
</select>
```

На Razor будет выглядеть как:

```
<select id=«slist»>
@for (int i = 0; i < 5; i++) {
<option value='@i'>@i</option>
```



```
}  
</select>
```

Razor Helpers. С помощью единственной строки Razor можно получить доступ к компонентам ASP.NET helper. Среди них можно выделить следующие:

- Web Grid
- Web Graphics
- Google Analytics
- Facebook Integration
- Twitter Integration
- Sending Email
- Validation.

Основные синтаксические правила Razor для C#:

- Блоки Razor кода заключаются в @ {... }.
- Встроенные выражения (переменные и функции) всегда начинаются с @.
- Операторы в выражениях заканчиваются точкой с запятой.
- Переменные объявляются с ключевым словом var.
- Строки заключаются в кавычки.
- C#-код чувствителен к регистру.
- C#-файлы имеют расширение .cshtml.

Доступ к данным, содержащимся в запросе клиента, реализуется через функцию Request[], а для проверки наличия таких данных используется условие IsPost .

Переменные всегда объявляются с помощью ключевого слова var или с указанием ключевого слова типа данных, хотя ASP.NET может определять тип данных переменной автоматически.

Основные типы данных:

Тип	Описание
int	Целые числа
float	Вещественные числа
decimal	Десятичные числа
bool	Логическое значение
string	Строка

Операторы Razor:

Оператор	Описание
=	Присваивание
+	Арифметические операторы
-	
*	
/	
+=	Инкремент величины. Декремент величины
-=	
==	Равенство
!=	Неравенство
<	Операторы сравнения
>	
<=	
>=	
>	
+	Конкатенация строк
.	Обращения к свойству или методу объекта
()	Группа значений
()	Параметры функции
[]	Обращение к элементу массива или коллекции
!	Логическое отрицание
&&	Логическое И
	Логическое ИЛИ

Иногда приходится прибегать к явному приведению типов данных. Для этого используются следующие методы:

Метод	Описание
AsInt() IsInt()	Преобразование строки в целое значение
AsFloat() IsFloat()	Преобразование строки в вещественное значение
AsDecimal() IsDecimal()	Преобразование строки в десятичное число
AsDateTime() IsDateTime()	Преобразование строки в значение ASP.NET DateTime типа.
AsBool() IsBool()	Преобразование строки в логическое значение
Tostring()	Преобразование любого типа в строку

В Razor используются условные операторы и операторы цикла. Операторы цикла и операторы ветвления (выбора) по структуре практически не отличаются от операторов PHP или JavaScript.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Задание 1. Простейший арифметический калькулятор.

1. Создайте в WebMatrix новый веб-сайт на основе пустого шаблона.

2. Добавьте в папку новый документ в формате CSHTML.

3. Добавьте в созданный документ следующий код:

```
@{
var resultMessage = «»;
if(IsPost)
{
var num1 = Request[«operand1»];
var num2 = Request[«operand2»];
var oper = Request[«operator»];
var result=0.0;

switch(oper)
{
case «+»:
result = num1.AsInt() + num2.AsInt();
break;
case «-»:
result = num1.AsInt() - num2.AsInt();
break;
case «*»:
result = num1.AsInt() * num2.AsInt();
break;
case «/»:
result = num1.AsFloat() / num2.AsFloat();
break;
}

resultMessage = «Result = « + result;
}
}

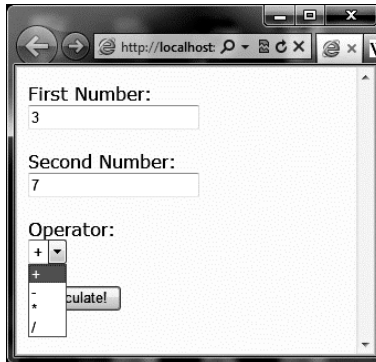
<html>
<body style=«background-color: beige; font-family: Verdana, Arial;»>
<form action=«« method=«post»>
<p><label for=«text1»>First Number:</label><br>
```

```

<input type=«text» name=«operand1» /></p>
<p><label for=«text2»>Second Number:</label><br>
<input type=«text» name=«operand2» /></p>
<p><label for=«text2»>Operator:</label><br>
<select name=«operator»>
<option value=«+»>+</option>
<option value=«-»>-</option>
<option value=«*»>*</option>
<option value=«/»>/</option>
</select>
<p><input type=«submit» value=«Calculate!» /></p>
</form>
<p>@resultMessage</p>
</body>
</html>

```

Данный пример демонстрирует работу с переменными и преобразованиями типов, операторами выбора и цикла, а также с массивом Request.



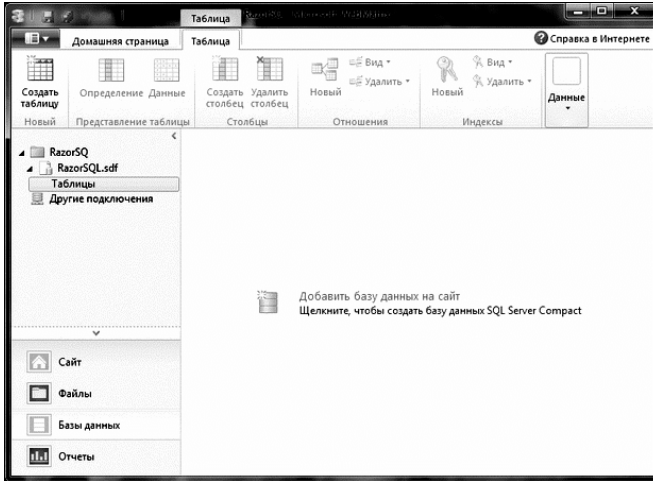
Задание 2. Создание и отображение локальной базы данных.

В данном задании необходимо создать средствами SQL Server Compact в WebMatrix локальную базу данных и отобразить ее содержимое на веб-странице.

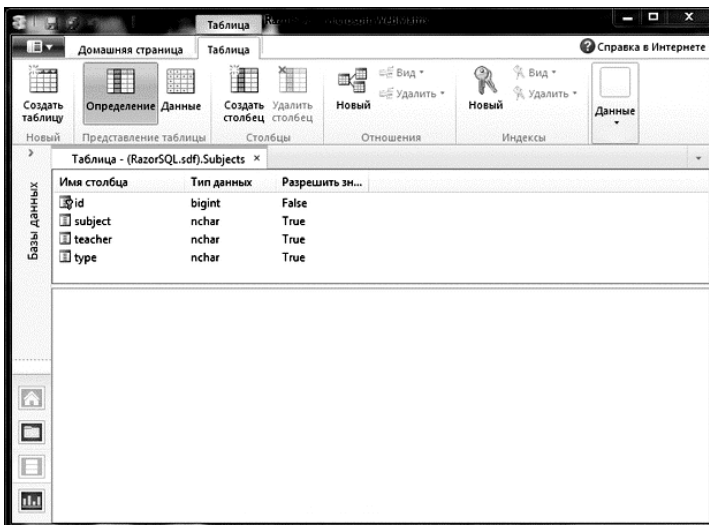
1. Создайте пустой сайт «Сайт на основе шаблона» под именем RazorSQL.

2. Щелкните справа на ссылке «Управление базами данных» и выберите «Добавить базу данных на сайт». При этом будет создана одноименная база данных.

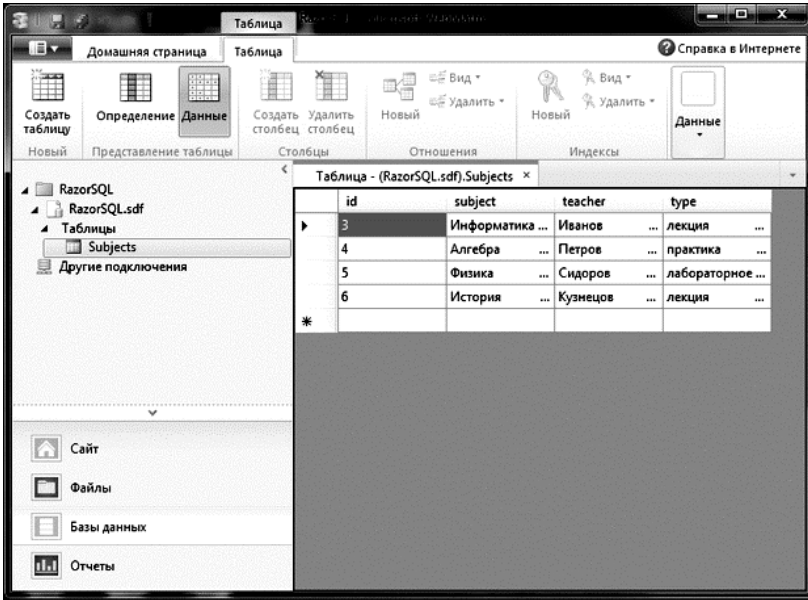
3. На левой панели разверните узел RazorSQL.sdf и щелкните «Таблицы», после чего на панели сверху выберите «Создать таблицу». При этом появится конструктор таблиц WebMatrix.



4. На панели «Свойства столбца» в строке (Имя) введите id.
5. В полях «Идентификатор» и «Первичный ключ» выставите значение true.
6. Щелкните «Создать столбец» и задайте свойства столбца с именем «subject». Укажите для него «Тип данных» – nchar.
7. Аналогично создайте столбцы «teacher» и «type» с такими же параметрами, как у «subject».
8. Структура таблицы представлена ниже.



9. Необходимо сохранить эту таблицу с именем «Subjects» и заполнить ее значениями.



10. На левой панели в рабочей области «Файлы» создайте CSHTML-страницу с именем SubjectsList.cshtml и разместите на ней следующий код:

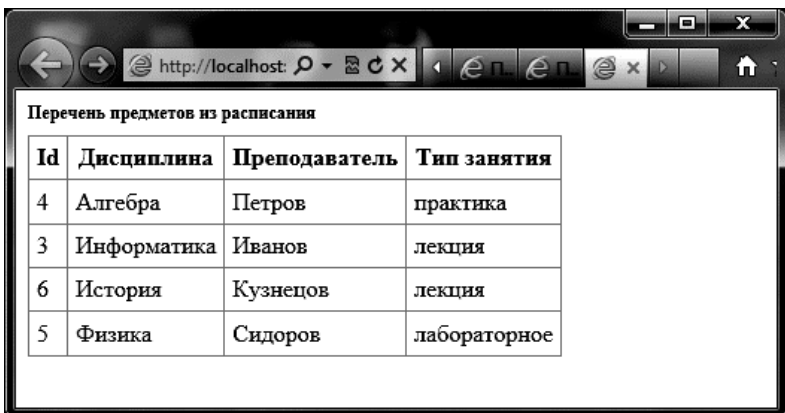
```
@{
    var db = Database.Open(«RazorSQL»);
    var selectQueryString = «SELECT * FROM Subjects ORDER BY
Subject»;
}
<!DOCTYPE html>
<html>
<head>
<title>Перечень предметов из расписания</title>
<style>
h1 {font-size: 12px;}
table, th, td {
border: solid 1px #999999;
border-collapse:collapse;
padding:5px;
}
}
```

```

</style>
</head>
<body>
<h1>Перечень предметов из расписания</h1>
<table>
<thead>
<tr>
<th>Id</th>
<th>Дисциплина</th>
<th>Преподаватель</th>
<th>Тип занятия</th>
</tr>
</thead>
<tbody>
@foreach (var row in db.Query(selectQueryString)){
<tr>
<td>@row.Id</td>
<td>@row.subject</td>
<td>@row.teacher</td>
<td>@row.type</td>
</tr>
}
</tbody>
</table>
</body>
</html>

```

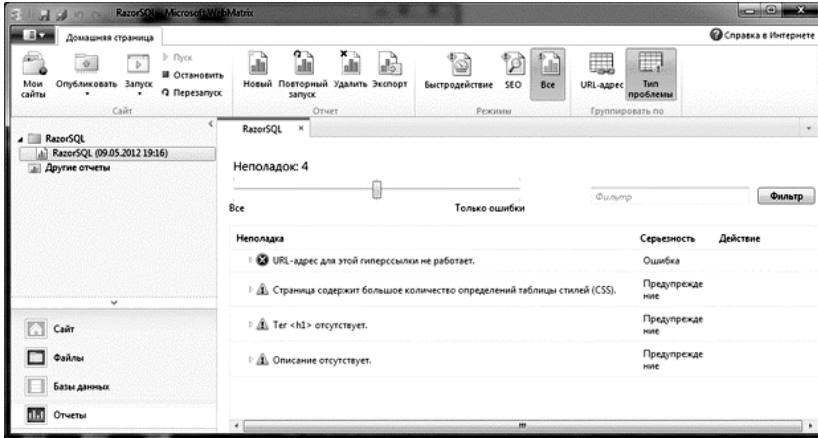
11. В данном коде использован запрос данных на языке SQL и оператор цикла для выборки и отображения строк таблицы на веб-странице.



The screenshot shows a web browser window with the address bar set to `http://localhost`. The page title is "Перечень предметов из расписания". The browser displays a table with the following data:

Id	Дисциплина	Преподаватель	Тип занятия
4	Алгебра	Петров	практика
3	Информатика	Иванов	лекция
6	История	Кузнецов	лекция
5	Физика	Сидоров	лабораторное

12. После создания веб-сайта можно использовать WebMatrix для запуска анализа видимости ваших страниц для поисковых систем. Такой анализ реализуется через функцию «Отчеты», которая производит анализ веб-сайта и предлагает способы решения найденных проблем.



КОНТРОЛЬНЫЕ ЗАДАНИЯ

1. В первом задании для арифметического калькулятора реализуйте корректную обработку ситуации деления на ноль.
2. Во втором задании добавьте на веб-странице возможность сортировки таблицы по выбранному полю. Для этого потребуются добавить на самой странице элемент выбора поля для сортировки, а также внести изменения в Razor и SQL-код.
3. Во втором задании добавьте на веб-странице возможность поиска записей таблицы по заданному значению выбранного поля. Для этого потребуются добавить на самой странице элемент выбора поля и ввода искомого значения, а также внести изменения в Razor и SQL-код.

Лабораторная работа 7

ТЕХНОЛОГИИ БЕЗОПАСНОСТИ В INTERNET EXPLORER

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Наличие режима InPrivate в веб-браузере Internet Explorer 11 (IE11) по достоинству смогут оценить те, кому приходится работать на чужом компьютере, поскольку новый режим просмотра InPrivate:

- не сохраняет новые cookie файлы;
- все новые cookie файлы сохраняются только до закрытия окна;
- ранее созданные cookie файлы остаются доступными;
- DOM Storage работает по такому же принципу, как и cookie;
- просмотренные сайты не будут запоминаться в историю посещений;
- временные файлы будут удаляться после закрытия окна;
- данные форм не сохраняются;
- пароли не сохраняются;
- адреса, введенные в строку адреса, не сохраняются;
- запросы, введенные в поле поиска, не сохраняются;
- посещенные ссылки не сохраняются.

Кроме того, IE11 позволяет пользователю удалять данные журнала браузера, а специальный InPrivate-фильтр информирует его о страницах, которые могут просматривать историю посещений пользователей, и позволяет блокировать такую возможность.

Рассмотрим данные функции более детально.

Режим просмотра InPrivate. По сути, режим просмотра InPrivate предотвращает сохранение данных о сеансах просмотра веб-браузером IE. Это помогает предотвращать получение посторонними пользователями компьютера информации о посещенных страницах и просмотренном содержимом. При запуске функции просмотра InPrivate IE открывает новое окно. Защита, обеспечиваемая режимом просмотра InPrivate, применяется только во время использования этого окна. В этом окне можно открывать любое количество вкладок, и все они будут защищены с помощью режима просмотра InPrivate. Однако при открытии другого окна браузера оно уже не будет защищено с помощью данного режима. Для завершения сеанса просмотра в режиме InPrivate необходимо просто закрыть окно браузера.

В таблице ниже приводится более подробное описание действия режима просмотра InPrivate.

Сведения	Изменение в режиме просмотра InPrivate
Файлы Cookie	Сохраняются в памяти для обеспечения правильного отображения веб-страниц, но удаляются при закрытии окна веб-обозревателя
Временные файлы Интернет	Сохраняются на диске для обеспечения правильного отображения веб-страниц, но удаляются при завершении работы веб-обозревателя
Журнал посещений веб-страниц	Информация не сохраняется
Данные форм и пароли	Информация не сохраняется
Кэш-средства антифишинга	Временная информация шифруется и сохраняется для обеспечения правильного отображения веб-страниц
Адресная строка и функция автозаполнения при поиске	Информация не сохраняется
Автоматическое восстановление после сбоев (ACR)	Функция ACR обеспечивает восстановление при сбоях на вкладке в течение сеанса, однако в случае ошибок в работе всего окна данные удаляются и окно восстановить невозможно
Хранилище моделей объектов документов (DOM)	Хранилище DOM подобно «огромному файлу Cookie», который веб-разработчики могут использовать для хранения информации. Как и обычные файлы Cookie, они не сохраняются после закрытия окна веб-обозревателя

Фильтр SmartScreen. Фильтр SmartScreen – один из инструментов веб-браузера IE, позволяющий обнаруживать поддельные веб-узлы. Он также обеспечивает защиту от установки вредоносного программного обеспечения.

Защита обеспечивается тремя следующими способами:

1. Фильтр SmartScreen работает в фоновом режиме при просмотре веб-страниц, анализируя их и определяя, содержат ли они характеристики, которые могут быть подозрительными. При обнаружении подозрительных веб-страниц фильтр SmartScreen отобразит сообщение, обеспечивающее возможность предоставления отзыва и рекомендующее продолжать работу с осторожностью.

2. Фильтр SmartScreen проверяет, содержатся ли посещаемые узлы в постоянно обновляемом списке обнаруженных поддельных узлов и узлов, распространяющих вредоносное программное обеспечение. При нахождении соответствия фильтр SmartScreen отобразит окно предупреждения красного цвета, уведомляющее о том, что узел блокирован для обеспечения безопасности.

3. Фильтр SmartScreen также проверяет, содержатся ли файлы, загруженные из Интернета, в аналогичном динамически обновляемом списке обнаруженных узлов, распространяющих вредоносное программное обеспечение. При нахождении соответствия фильтр SmartScreen отобразит окно предупреждения красного цвета, уведомляющее о том, что загруженный объект блокирован для обеспечения безопасности.

Фильтр межузловых сценариев (XSS). Фильтр межузловых сценариев (XSS) в Internet Explorer помогает предотвратить добавление одним веб-узлом кода сценария на другой веб-узел. Фильтр XSS наблюдает за взаимодействием веб-узлов и при определении возможной атаки автоматически блокирует запуск кода сценария. В подобных случаях на панели информации будет отображено сообщение, позволяющее узнать об изменении веб-страницы, чтобы защитить конфиденциальность и обеспечить безопасность.

Общая защита. В IE11 поддерживаются также следующие технологии защиты:

- поддержка кросс-документного обмена сообщениями HTML5 (HTML5 cross-document messaging);
- новый объект XDomainRequest для передачи данных через домены;

- новая функция `toStaticHTML`, которая позволит избежать внедрения опасного кода на страницы путем форматирования `html`-тегов;
- IE11 реализует функции `ECMAScript3.1` для работы с `JSON`. Для обеспечения безопасности объект для работы с `JSON` содержит функцию `parse`, которая, как и `toStaticHTML`, надежно форматирует потенциально-опасный текст;
- браузер содержит улучшения в механизме `MIME-sniffing`, который позволяет определять браузеру содержимое страницы не по значению поля заголовка ответа сервера `«content-type»`, а по содержанию. Теперь, например, при `«content-type: image/*»` браузер не будет обрабатывать вложенный `html`- или скрипт-код. При этом для управления фильтром разработчик может воспользоваться новым параметром `authoritative` и указать `«Content-Type: text/plain; authoritative=true;»`, – в этом случае IE11 не будет пытаться определить тип содержимого, интерпретируя его согласно указанию в поле `«Content-Type»`;
- новые параметры для `HTTP`-заголовков `X-Download-Options: noopen` и `Content-Disposition: attachment; filename=untrustedfile.html` указывают браузеру, что он должен сохранить полученное от сервера содержимое вместо того, чтобы его отобразить. Это может быть использовано в том случае, когда веб-приложению требуется передать пользователю страницу с небезопасным содержимым. В случае сохранения его на клиентском компьютере и последующем открытии такие страницы не будут работать в контексте сервера, что повысит уровень его безопасности;
- в элементе формы `File Upload` в целях безопасности изменен статус поля ввода на `read-only`. Кроме того, IE11 отправляет не полный путь файла, а только его имя;
- IE11 содержит некоторые методики противодействия атакам с применением социальной инженерии.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Задание 1. Включение режима просмотра `InPrivate`.

Чтобы включить просмотр `InPrivate`, выполните одно из следующих действий:

- выберите в меню «Сервис» раздел «Безопасность» и выберите пункт Просмотр `InPrivate` (рис. 7.1);

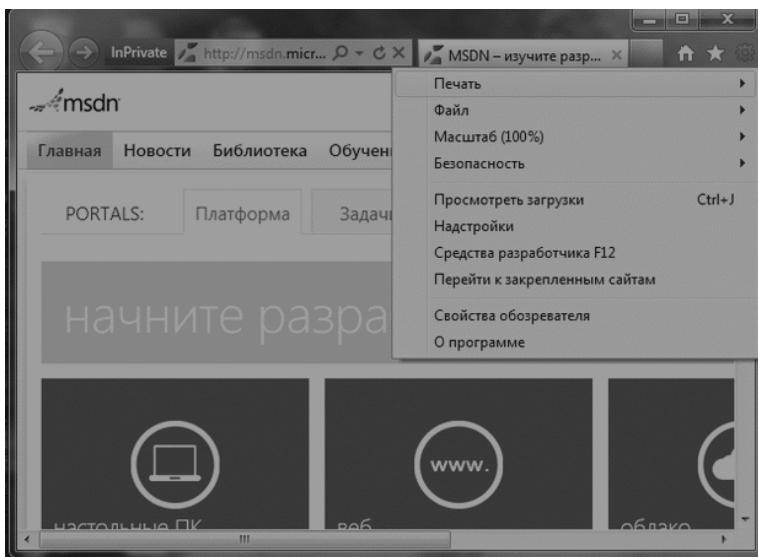


Рис. 7.1. Переключение в режим просмотра InPrivate

В результате появится окно (рис. 7.2):

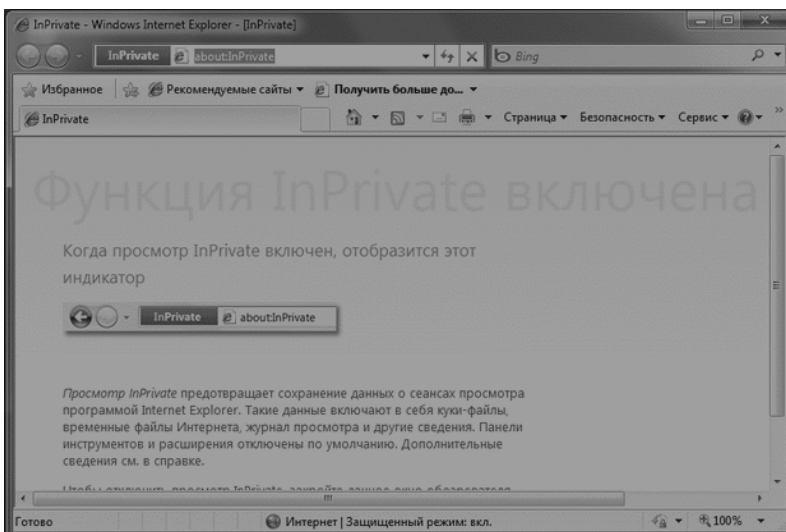


Рис. 7.2. Работа с новыми вкладками браузера в режиме просмотра InPrivate

Нажмите сочетание клавиш <CTRL> + <SHIFT> + P.

Задание 2. Включение и отключение функции фильтрации InPrivate.

Фильтрация InPrivate помогает предотвратить сбор поставщиками содержимого сведений о посещенных пользователем веб-узлов.

Большинство веб-страниц используют содержимое веб-узлов, например рекламные материалы, карты или инструменты веб-анализа, которые отличаются от тех, которые намерен посетить пользователь. Данные веб-узлы называются поставщиками содержимого (ПС) или сторонними веб-узлами. При посещении веб-узла со сторонним содержанием некоторые сведения о пользователе отправляются поставщику этого содержимого (например, «Google Analytics» или «Яндекс Метрика»). Если ПС предоставляет содержимое большому количеству посещаемых веб-узлов, то ПС может разработать профиль параметров просмотра, которые могут быть использованы в разных целях, включая анализ и показ адресной рекламы.

Обычно данное содержимое стороннего веб-узла, например встроенное изображение или видеоролик, отображается без проблем. Поскольку источником содержимого считается веб-узел, посещенный пользователем первоначально, то отслеживание других веб-узлов (сторонних), которые могут получать сведения о посещенных им веб-страницах, невозможно. Инструменты веб-анализа или веб-измерения сообщают о том, какие веб-узлы обычно посещает пользователь, и не всегда ему видны, как это часто бывает в случае с веб-маяками. Веб-маяки представляют собой, как правило, одноточечные изображения, единственной целью которых является отслеживание использования веб-узлов; при этом они не отображаются в качестве видимого содержимого.

Например, пользователь посещает страницы сайтов s1.ru и s2.ru, у которых в теле страниц размещен javascript-сценарий:

```
<script src= «http://bs.yandex.ru/resource/watch.js» >
```

В результате bs.yandex.ru знает о том, что пользователь посетил сайты s1.ru и s2.ru.

Суть работы InPrivate-фильтра заключается в анализе содержимого посещаемых веб-страниц: если аналогичное содержимое используется на нескольких веб-узлах, будет доступна возможность разрешить или заблокировать данное содержимое. Можно также выбрать автоматическую блокировку любого ПС или стороннего веб-узла, обнаруживаемого с помощью функции фильтрации InPrivate, либо отключить эту функцию.

По умолчанию функция фильтрации InPrivate обеспечивает анализ посещаемых веб-узлов и поставщиков содержимого, однако не блокирует их автоматически. Можно разрешить или блокировать любого поставщика содержимого, определяемого режимом фильтрации InPrivate в качестве узла, обменивающегося параметрами просмотра. В качестве альтернативного варианта, с помощью функции фильтрации InPrivate можно автоматически блокировать любого поставщика содержимого либо отключить эту функцию.

Чтобы активировать фильтрацию InPrivate, нет необходимости открывать определенное окно – достаточно выполнить одно из следующих действий:

- в меню браузера «Сервис» выбрать в разделе «Безопасность» пункт «Фильтрация InPrivate» (рис. 7.3);

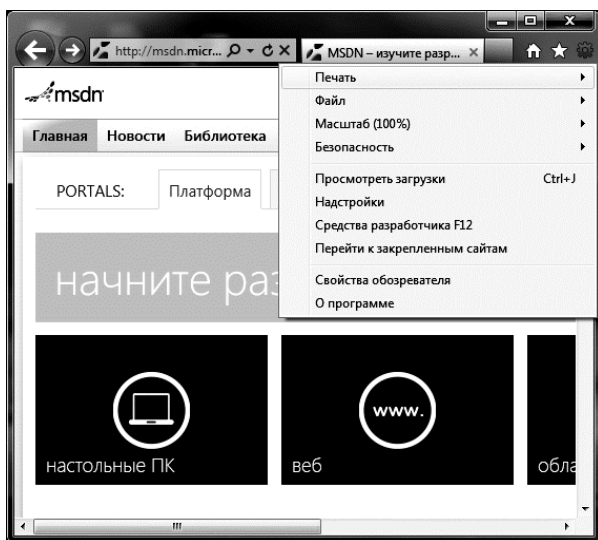


Рис. 7.3. Включение режима фильтрации InPrivate

Нажать сочетание клавиш <CTRL> + <SHIFT> + P.

Если фильтрация InPrivate включена, слева от строки статуса веб-браузера появится значок:

The image shows a small, dark rectangular icon with the word "InPrivate" written in white, sans-serif font. This icon is positioned to the left of the browser's status bar when the InPrivate mode is active.

Если фильтрация InPrivate уже включена, выполните следующие действия: выберите в меню «Сервис» раздел «Безопасность» и пункт «Защита от слежения», а затем выберите настроенный список справа.

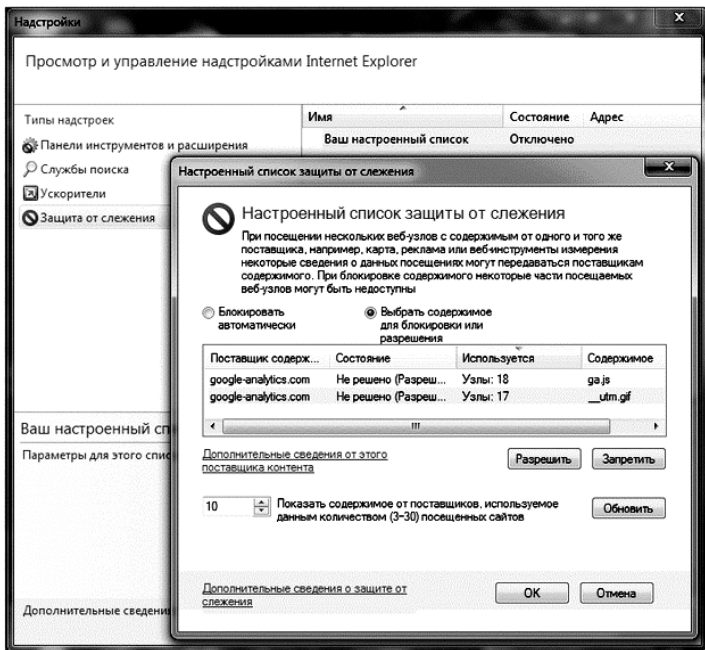


Рис. 7.4. Настройка режима фильтрации InPrivate

Выполните одно из следующих действий:

- чтобы автоматически блокировать веб-узлы, выберите параметр «Блокировать автоматически»;
- чтобы заблокировать веб-узлы вручную, нажмите «Выбрать блокировку» или «Разрешение содержимого»;
- чтобы отключить фильтрацию InPrivate, нажмите кнопку «Запретить»;
- нажмите кнопку ОК.

Если требуется вручную заблокировать или разрешить веб-узлы поставщика содержимого или сторонние веб-узлы, которые могут собирать сведения о ранее посещенных веб-узлах, выполните следующие действия.

1. Выберите в меню «Сервис» раздел «Безопасность» и пункт «Защита от слежения», а затем выберите настроенный список справа.
2. Щелкните параметр «Выбрать содержимое блокировки» или разрешения, выберите один или несколько веб-узлов, которые необходимо разрешить или заблокировать, а затем нажмите кнопки «Разрешить» или «Запретить».

3. Чтобы задать количество посещаемых веб-узлов, которые обмениваются содержимым, перед их добавлением в список, укажите новое количество в поле «Показать содержимое сторонних веб-узлов», используемое данным количеством посещенных узлов. Можно задать количество от 3 до 30. Значением по умолчанию данного параметра является 10 (т.е. не менее 10 различных веб-узлов должны использовать аналогичное содержимое стороннего веб-узла перед его отображением и возможностью блокирования или разрешения).

4. По окончании работы нажмите кнопку ОК.

Следует иметь в виду, что после блокировки содержимого веб-узла будет отображаться меньше элементов или же страница перестанет открываться совсем. Это может произойти, если на веб-узле размещен инструмент веб-анализа, который визуально не отображается, но имеет финансовую ценность для данного веб-узла. Если вы предпочтете заблокировать содержимое, благодаря которому веб-узел оппачивает счета, данный веб-узел может закрыть вам доступ к своим страницам.

Задание 3. Очистка журнала посещений веб-узлов.

Во время просмотра веб-страниц IE сохраняет информацию о посещенных веб-узлах и сведения, которые часто требуется предоставить (например, имя и адрес пользователя). Веб-браузер IE сохраняет следующие типы информации:

- временные файлы Интернета;
- файлы Cookie;
- журнал посещенных веб-узлов;
- сведения, указанные на веб-страницах или в адресной строке;
- сохраненные сетевые пароли.

Обычно полезно сохранять данную информацию на компьютере, потому что она может ускорить просмотр веб-страниц или автоматически предоставить сведения, чтобы их не приходилось вводить повторно. Однако данная информация может быть удалена в случае использования общедоступного компьютера и предотвращения сохранения на нем личных сведений после завершения работы.

Для удаления журнала веб-браузера (полностью или частично):

1) в разделе «Безопасность» выберите пункт «Удалить журнал веб-обозревателя» (рис. 7.5);

2) установите флажок рядом с каждой категорией информации, которую требуется удалить;

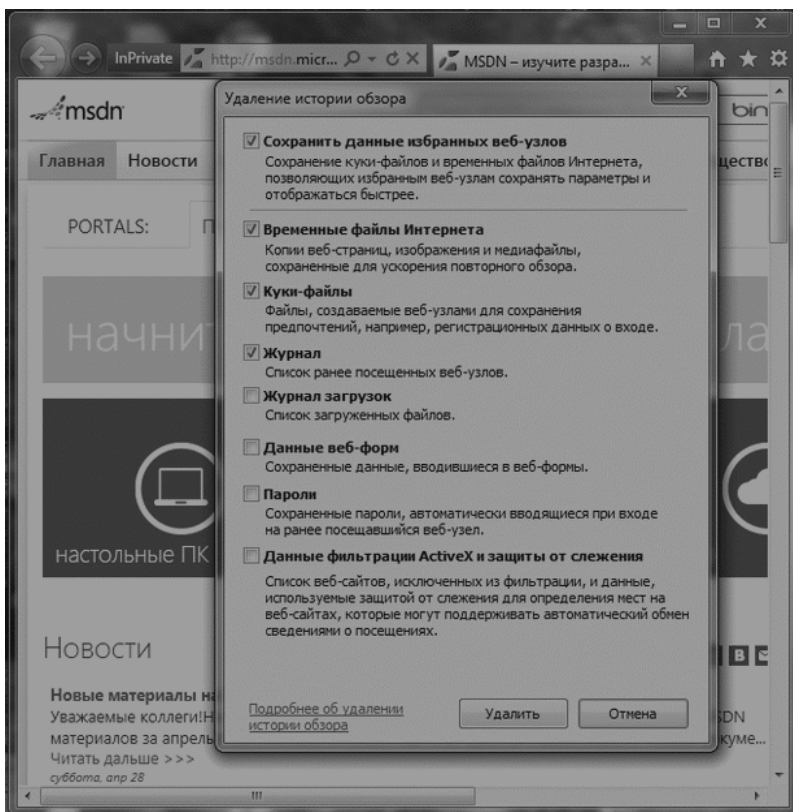


Рис. 7.5. Настройка удаления журнала веб-браузера

3) установите флажок рядом с пунктом «Сохранять данные избранных веб-узлов», если файлы Cookie и файлы, связанные с веб-узлами списка «Избранное», удалять не требуется;

4) нажмите кнопку «Удалить» (при наличии большого количества файлов в журнале данный процесс может занять некоторое время).

Задание 4. Использование фильтра SmartScreen.

Чтобы включить фильтр SmartScreen, выполните следующие действия (рис. 7.6):

1) в разделе «Безопасность» выберите пункт «Фильтр SmartScreen» и щелкните «Включить фильтр SmartScreen»;

2) в диалоговом окне «Фильтр Microsoft SmartScreen» нажмите кнопку ОК.

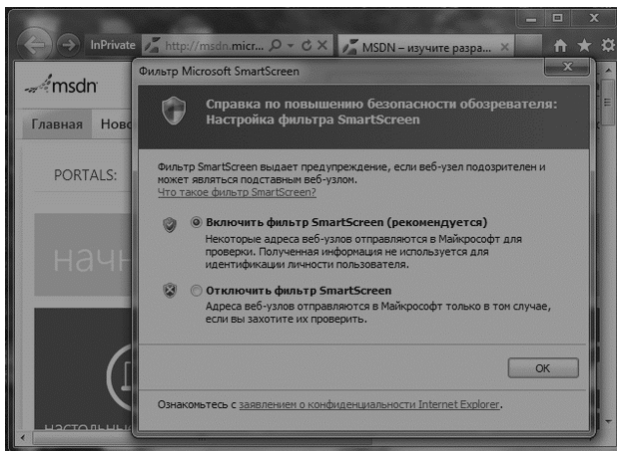


Рис. 7.6. Включение фильтра SmartScreen

Фильтр SmartScreen:

- является полностью настраиваемым компонентом групповой политики. При использовании групповой политики можно управлять узлами с помощью зон безопасности IE;
- препятствует просмотру и загрузке с веб-узлов, которые известны как размещающие вредоносное содержимое. Пользователи могут пропускать предупреждения фильтра и продолжать просмотр. Можно использовать групповую политику, чтобы предотвратить отклонение пользователями предупреждений фильтра.

Можно также настроить фильтр SmartScreen для запрета проверки узлов, находящихся в зоне «Надежные узлы». По умолчанию фильтр проверяет эти узлы, однако можно отключить проверки данной зоны. Затем можно добавить в зону надежных узлов организации собственный список узлов, в том числе тех, которые рассматриваются компанией в качестве безопасных или надежных. Эти узлы входят в локальную зону доверия фильтра SmartScreen и никогда не проверяются автоматически.

КОНТРОЛЬНЫЕ ЗАДАНИЯ

1. Проверьте работу режима фильтрации InPrivate с выбором содержимого для блокировки.
2. Проверьте работу фильтра SmartScreen на известных вам веб-узлах, содержащих вредоносное содержимое.

ЗАКЛЮЧЕНИЕ

Цель данного учебного пособия – комплексное рассмотрение современных клиентских и серверных технологий веб-разработки, применяемых для создания веб-сайтов, и практическое изучение данных технологий на конкретных примерах с использованием Visual Studio Web Developer Express, Microsoft WebMatrix.

Практикум содержит задания для самостоятельного выполнения с целью более глубокого понимания материала студентами.

В результате освоения предлагаемого курса студенты смогут разрабатывать высококачественные мультимедиа-приложения, познакомиться с методами программирования трехмерной графики и создания зрелищных визуальных эффектов, научатся разрабатывать офф-лайн приложения, смогут визуализировать сложные данные и результаты научных исследований, освоят новые элементы управления, познакомиться с технологией Microsoft Razor.

СПИСОК ЛИТЕРАТУРЫ

1. Хоган, Б. HTML5 и CSS3. Веб-разработка по стандартам нового поколения / Б. Хоган. – СПб. : Питер, 2012. – 320 с.
2. Роббинс, Дж. HTML5, CSS3 и Javascript. Исчерпывающее руководство / Дж. Роббинс. – М. : Эксмо, 2014. – 528 с.
3. Фрейн, Б. HTML5 и CSS3. Разработка сайтов для любых браузеров и устройств / Б. Фрейн. – СПб. : Питер, 2014. – 304 с.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	3
Лабораторная работа 1. ОСНОВЫ РАБОТЫ С MICROSOFT WEBMATRIX3	4
Лабораторная работа 2. ВВЕДЕНИЕ В JAVASCRIPT. ПРО- ГРАММНОЕ ВЗАИМОДЕЙСТВИЕ С HTML-ДОКУМЕНТАМИ НА ОСНОВЕ DOM API. ИСПОЛЬЗОВАНИЕ РЕГУЛЯРНЫХ ВЫРАЖЕНИЙ. ОТЛАДКА И ПРОФИЛИРОВАНИЕ JAVASCRIPT-СЦЕНАРИЕВ	12
Лабораторная работа 3. ПРАКТИЧЕСКОЕ ВВЕДЕНИЕ В HTML5. СОЗДАНИЕ СОБСТВЕННОГО ВИДЕОПЛЕЕРА НА HTML5 VIDEO	30
Лабораторная работа 4. ПРАКТИЧЕСКОЕ ВВЕДЕНИЕ В CSS3.0	45
Лабораторная работа 5. СТРУКТУРА XML-ДОКУМЕНТА. ПРОГРАММНЫЙ ИНТЕРФЕЙС XML DOM API	56
Лабораторная работа 6. РАЗРАБОТКА WEB-ПРИЛОЖЕНИЙ В MICROSOFT WEBMATRIX3	72
Лабораторная работа 7. ТЕХНОЛОГИИ БЕЗОПАСНОСТИ В INTERNET EXPLORER	81
ЗАКЛЮЧЕНИЕ	92
СПИСОК ЛИТЕРАТУРЫ	93

Учебное издание

ГРОМОВ Юрий Юрьевич,
ИВАНОВА Ольга Геннадьевна,
ШАХОВ Николай Гурьевич,
ОДНОЛЬКО Валерий Григорьевич

ИНФОРМАЦИОННЫЕ WEB-ТЕХНОЛОГИИ

Учебное пособие

Компьютерное макетирование М. А. Евсейчевой
Редактирование Т. М. Глинкиной

ISBN 978-5-8265-1365-1



Подписано в печать 08.12.2014.
Формат 60 × 84 / 16. 5,58 усл. печ. л.
Тираж 100 экз. Заказ № 570

Издательско-полиграфический центр
ФГБОУ ВПО «ТГТУ»
392000, г. Тамбов, ул. Советская, д. 106, к. 14
Телефон 8(4752)63-81-08
E-mail: izdatelstvo@admin.tstu.ru