

# HTML 5, CSS 3 и Web 2.0

## Разработка современных Web-сайтов



КОНЦЕПЦИЯ WEB 2.0

JAVASCRIPT И WEB-СЦЕНАРИИ

EXT CORE

ИНТЕРАКТИВНЫЕ ЭЛЕМЕНТЫ  
WEB-СТРАНИЦЫ

ПОДГРУЖАЕМОЕ  
И ГЕНЕРИРУЕМОЕ  
СОДЕРЖИМОЕ

СЕМАНТИЧЕСКАЯ РАЗМЕТКА

WEB-ФОРМЫ И ЭЛЕМЕНТЫ  
УПРАВЛЕНИЯ

СВОБОДНО  
ПОЗИЦИОНИРУЕМЫЕ  
ЭЛЕМЕНТЫ

ПРОГРАММНОЕ РИСОВАНИЕ  
НА WEB-СТРАНИЦЕ

**PRO**

ПРОФЕССИОНАЛЬНОЕ  
ПРОГРАММИРОВАНИЕ

Владимир Дронов

# HTML 5, CSS 3 и Web 2.0

## Разработка современных Web-сайтов

Санкт-Петербург  
«БХВ-Петербург»

2011

УДК 681.3.068  
ББК 32.973.26-018.1  
Д75

**Дронов В. А.**

Д75 HTML 5, CSS 3 и Web 2.0. Разработка современных Web-сайтов. — СПб.: БХВ-Петербург, 2011. — 416 с.: ил. — (Профессиональное программирование)  
ISBN 978-5-9775-0596-3

Практическое руководство по созданию современных Web-сайтов, соответствующих концепции Web 2.0. Описаны языки HTML 5 и CSS 3, применяемые, соответственно, для создания содержимого и представления Web-страниц. Даны принципы Web-программирования на языке JavaScript с использованием библиотеки Ext Core. Рассказано о создании интерактивных Web-страниц, приведены примеры интерактивных элементов, позволяющие сделать Web-страницы удобнее для посетителя. Раскрыты вопросы реализации подгружаемого и генерируемого содержимого, семантической разметки, применения баз данных для формирования Web-страниц. Показаны способы расширения функциональности Web-сайтов с использованием Web-форм, элементов управления, свободно позиционируемых элементов и программного рисования средствами HTML 5.

*Для Web-дизайнеров*

УДК 681.3.068  
ББК 32.973.26-018.1

### **Группа подготовки издания:**

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Евгений Рыбаков</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Леонид Кочин</i>
Компьютерная верстка	<i>Ольги Сергиенко</i>
Корректор	<i>Зинаида Дмитриева</i>
Дизайн серии	<i>Инны Тачиной</i>
Оформление обложки	<i>Елены Беляевой</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 31.08.10.  
Формат 70×100<sup>1/16</sup>. Печать офсетная. Усл. печ. л. 33,54.  
Тираж 1500 экз. Заказ №  
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию  
№ 77.99.60.953.Д.005770.05.09 от 26.05.2009 г. выдано Федеральной службой  
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов  
в ГУП "Типография "Наука"  
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0596-3

© Дронов В. А., 2010  
© Оформление, издательство "БХВ-Петербург", 2010

# Оглавление

<b>Введение.....</b>	<b>11</b>
Что грядет нового в Web-дизайне .....	11
О чем эта книга .....	12
Какие программы используются в этой книге.....	13
Типографские соглашения .....	13
Благодарности .....	14
<b>ЧАСТЬ I. СОДЕРЖИМОЕ WEB-СТРАНИЦ. ЯЗЫК HTML 5.....</b>	<b>15</b>
<b>Глава 1. Введение в современный Web-дизайн. Web 2.0.</b>	
<b>Создание Web-страниц .....</b>	<b>17</b>
Современный Web-дизайн. Концепция Web 2.0 .....	17
Что требуется от современного Web-сайта .....	17
Концепция Web 2.0 .....	19
Интернет: как все это работает.....	22
Клиенты и серверы Интернета. Интернет-адреса .....	22
Web-сайты и Web-серверы.....	24
Основные принципы создания Web-страниц. Язык HTML 5 .....	25
Язык HTML и его теги.....	26
Вложенность тегов.....	29
Секции Web-страницы.....	30
Метаданные и тип Web-страницы .....	31
Атрибуты HTML-тегов.....	32
Программы, которыми мы будем пользоваться .....	33
Web-обозреватель .....	33
Web-сервер .....	34
Что дальше?.....	35
<b>Глава 2. Структурирование текста .....</b>	<b>36</b>
Абзацы .....	36
Заголовки .....	38
Списки.....	39
Цитаты .....	42

Текст фиксированного формата .....	42
Горизонтальные линии .....	45
Адреса .....	46
Комментарии .....	47
Что дальше? .....	47
<b>Глава 3. Оформление текста.....</b>	<b>48</b>
Выделение фрагментов текста .....	48
Разрыв строк.....	50
Вставка недопустимых символов. Литералы .....	51
Что дальше? .....	54
<b>Глава 4. Графика и мультимедиа.....</b>	<b>55</b>
Внедренные элементы Web-страниц.....	55
Графика.....	56
Форматы интернет-графики .....	56
Вставка графических изображений .....	57
Мультимедиа .....	60
Форматы файлов и форматы кодирования .....	60
Типы MIME .....	62
Вставка аудиоролика .....	63
Вставка видеоролика .....	65
Дополнительные возможности тегов <code>&lt;AUDIO&gt;</code> и <code>&lt;VIDEO&gt;</code> .....	68
Что дальше? .....	69
<b>Глава 5. Таблицы .....</b>	<b>70</b>
Создание таблиц.....	70
Заголовок и секции таблицы .....	75
Объединение ячеек таблиц.....	78
Что дальше? .....	80
<b>Глава 6. Средства навигации .....</b>	<b>81</b>
Текстовые гиперссылки.....	81
Создание гиперссылок.....	81
Интернет-адреса в WWW .....	83
Почтовые гиперссылки.....	85
Дополнительные возможности гиперссылок.....	86
Графические гиперссылки .....	87
Изображения-гиперссылки.....	87
Изображения-карты .....	88
Полоса навигации .....	90
Якоря.....	91
Что дальше? .....	92
<b>ЧАСТЬ II. ПРЕДСТАВЛЕНИЕ WEB-СТРАНИЦ.</b>	
<b>КАСКАДНЫЕ ТАБЛИЦЫ СТИЛЕЙ CSS 3.....</b>	<b>93</b>
<b>Глава 7. Введение в стили CSS.....</b>	<b>95</b>
Понятие о стилях CSS.....	95

Создание стилей CSS.....	96
Таблицы стилей.....	101
Правила каскадности и приоритет стилей .....	103
Важные атрибуты стилей .....	105
Какие стили в каких случаях применять .....	106
Комментарии CSS .....	107
Что дальше?.....	108
<b>Глава 8. Параметры шрифта и фона. Контейнеры.....</b>	<b>109</b>
Параметры шрифта.....	109
Параметры, управляющие разрывом строк .....	114
Параметры вертикального выравнивания.....	116
Параметры тени у текста .....	117
Параметры фона.....	118
Контейнеры. Встроенные контейнеры.....	120
Представление для нашего Web-сайта, часть 1 .....	122
Что дальше?.....	125
<b>Глава 9. Параметры абзацев, списков и отображения .....</b>	<b>126</b>
Параметры вывода текста .....	126
Параметры списков.....	127
Параметры отображения .....	129
Представление для нашего Web-сайта, часть 2.....	131
Создание полосы навигации .....	132
Параметры курсора.....	133
Что дальше?.....	134
<b>Глава 10. Контейнерный Web-дизайн .....</b>	<b>135</b>
Блочные контейнеры .....	135
Основы контейнерного Web-дизайна.....	136
Старые разновидности Web-дизайна и их критика .....	137
Сущность контейнерного Web-дизайна .....	139
Представление для нашего Web-сайта, часть 3 .....	140
Стили, задающие параметры контейнеров .....	142
Параметры размеров.....	142
Параметры размещения. Плавающие контейнеры.....	143
Представление для нашего Web-сайта, часть 4.....	145
Параметры переполнения. Контейнеры с прокруткой .....	146
Представление для нашего Web-сайта, часть 5 .....	148
Что дальше?.....	150
<b>Глава 11. Отступы, рамки и выделение.....</b>	<b>151</b>
Параметры отступов .....	151
Параметры рамки.....	154
Представление для нашего Web-сайта, часть 6.....	156
Полная полоса навигации.....	160
Параметры выделения .....	163
Что дальше?.....	165

<b>Глава 12. Параметры таблиц.....</b>	<b>166</b>
Параметры выравнивания .....	166
Параметры отступов и рамок.....	167
Параметры размеров.....	168
Прочие параметры .....	169
Представление для нашего Web-сайта, часть 7.....	170
Что дальше?.....	171
<b>Глава 13. Специальные селекторы .....</b>	<b>172</b>
Комбинаторы.....	172
Селекторы по атрибутам тега .....	174
Псевдоэлементы.....	176
Псевдоклассы .....	177
Псевдоклассы гиперссылок.....	177
Структурные псевдоклассы.....	178
Псевдоклассы <i>:not</i> и <i>*</i> .....	182
Представление для нашего Web-сайта, часть 8.....	182
Что дальше?.....	184
<b>ЧАСТЬ III. ПОВЕДЕНИЕ WEB-СТРАНИЦ, WEB-СЦЕНАРИИ.....</b>	<b>187</b>
<b>Глава 14. Введение в Web-программирование. Язык JavaScript.....</b>	<b>189</b>
Примеры Web-сценариев .....	189
Простейший Web-сценарий .....	189
Более сложный Web-сценарий.....	190
Как Web-сценарии помещаются в Web-страницу.....	192
Язык программирования JavaScript.....	193
Основные понятия JavaScript.....	194
Типы данных JavaScript.....	195
Переменные .....	197
Именованые переменных .....	197
Объявление переменных .....	197
Операторы .....	198
Арифметические операторы .....	198
Оператор объединения строк.....	199
Операторы присваивания .....	199
Операторы сравнения .....	200
Логические операторы.....	201
Оператор получения типа <i>typeof</i> .....	202
Совместимость и преобразование типов данных.....	203
Приоритет операторов.....	204
Сложные выражения JavaScript.....	206
Блоки.....	206
Условные выражения.....	206
Условный оператор <i>?</i> .....	207
Выражения выбора .....	208
Циклы.....	209
Цикл со счетчиком.....	209
Цикл с постусловием .....	210

Цикл с предусловием .....	211
Прерывание и перезапуск цикла .....	211
Функции .....	212
Объявление функций .....	212
Функции и переменные. Локальные переменные .....	213
Вызов функций .....	213
Присваивание функций. Функциональный тип данных .....	215
Массивы .....	215
Ссылки .....	217
Объекты .....	218
Понятия объекта и экземпляра объекта .....	218
Получение экземпляра объекта .....	219
Работа с экземпляром объекта .....	222
Встроенные объекты языка JavaScript .....	223
Объект <i>Object</i> и использование его экземпляров .....	225
Объекты Web-обозревателя. Объектная модель документа DOM .....	226
Свойства и методы экземпляра объекта .....	227
Правила написания выражений .....	228
Комментарии JavaScript .....	229
Что дальше? .....	229
<b>Глава 15. Библиотека Ext Core и объекты Web-обозревателя.....</b>	<b>230</b>
Библиотека Ext Core .....	230
Зачем нужна библиотека Ext Core .....	230
Использование библиотеки Ext Core .....	232
Ключевые объекты библиотеки Ext Core .....	233
Доступ к нужному элементу Web-страницы .....	234
Доступ сразу к нескольким элементам Web-страницы .....	236
Доступ к родительскому, дочерним и соседним элементам Web-страницы .....	239
Получение и задание размеров и местоположения элемента Web-страницы .....	241
Получение размеров Web-страницы и клиентской области окна Web-обозревателя .....	243
Получение и задание значений атрибутов тега .....	243
Управление привязкой стилевых классов .....	244
Получение и задание значений атрибутов стиля .....	246
Управление видимостью элементов Web-страницы .....	248
Добавление и удаление элементов Web-страницы .....	249
Обработка событий .....	254
Понятие события и его обработки .....	254
События объекта <i>Element</i> .....	255
Привязка и удаление обработчиков событий .....	256
Всплытие и действие по умолчанию .....	258
Получение сведений о событии. Объект <i>EventObject</i> .....	260
Объект <i>CompositeElementLite</i> .....	261
Объекты Web-обозревателя .....	263
Что дальше? .....	265
<b>Глава 16. Создание интерактивных Web-страниц.....</b>	<b>266</b>
Управление размерами блочных контейнеров .....	266
Выделение пункта полосы навигации при наведении на него курсора мыши .....	269



Переход на целевую Web-страницу при щелчке на пункте полосы навигации.....	270
Скрытие и открытие вложенных списков.....	273
Выделение пункта полосы навигации, соответствующего открытой в данный момент Web-странице.....	275
Скрытие и открытие текста примеров.....	279
Что дальше?.....	282

## **ЧАСТЬ IV. ПОДГРУЖАЕМОЕ И ГЕНЕРИРУЕМОЕ СОДЕРЖИМОЕ. СЕМАНТИЧЕСКАЯ РАЗМЕТКА..... 283**

<b>Глава 17. Подгружаемое содержимое.....</b>	<b>285</b>
Монолитные и блочные Web-страницы.....	285
Подгрузка содержимого Web-страниц.....	287
Реализация подгрузки содержимого.....	288
Что дальше?.....	296

<b>Глава 18. Генерируемое содержимое.....</b>	<b>297</b>
Введение в генерируемое содержимое. Базы данных.....	297
Реализация генерируемого содержимого.....	299
Создание базы данных.....	300
Генерирование полосы навигации.....	301
Сортировка базы данных.....	303
Что дальше?.....	304

<b>Глава 19. Семантическая разметка данных.....</b>	<b>305</b>
Введение в семантическую разметку данных.....	305
Реализация семантической разметки средствами JavaScript.....	306
Создание раздела "См. также".....	308
Что дальше?.....	314

## **ЧАСТЬ V. ПОСЛЕДНИЕ ШТРИХИ..... 315**

<b>Глава 20. Web-формы и элементы управления.....</b>	<b>317</b>
Web-формы и элементы управления HTML.....	317
Назначение Web-форм и элементов управления. Серверные приложения.....	318
Создание Web-форм и элементов управления.....	320
Создание Web-форм.....	320
Создание элементов управления.....	320
Поле ввода.....	321
Поле ввода пароля.....	322
Поле ввода значения для поиска.....	323
Область редактирования.....	323
Кнопка.....	324
Флажок.....	325
Переключатель.....	325
Список, обычный или раскрывающийся.....	326
Надпись.....	328
Группа.....	329

Прочие элементы управления .....	330
Специальные селекторы CSS, предназначенные для работы с элементами управления .....	330
Работа с элементами управления .....	331
Свойства и методы объекта <i>HTML</i> <i>Element</i> , применяемые для работы с элементами управления .....	331
Свойства и методы объекта <i>Element</i> , применяемые для работы с элементами управления .....	336
События элементов управления .....	336
Реализация поиска на Web-сайте .....	337
Подготовка базы данных .....	338
Создание Web-формы .....	338
Написание Web-сценария, выполняющего поиск .....	339
Что дальше? .....	345
<b>Глава 21. Свободно позиционируемые элементы Web-страницы .....</b>	<b>346</b>
Свободно позиционируемые контейнеры .....	347
Понятие свободно позиционируемого элемента Web-страницы .....	347
Создание свободно позиционируемых элементов .....	348
Средства библиотеки Ext Core для управления свободно позиционируемыми элементами .....	351
Реализация усовершенствованного поиска .....	351
Создание контейнера с Web-формой поиска .....	352
Написание Web-сценария, выполняющего поиск .....	354
Что дальше? .....	358
<b>Глава 22. Программируемая графика .....</b>	<b>359</b>
Канва .....	359
Контекст рисования .....	360
Рисование простейших фигур .....	361
Задание цвета, уровня прозрачности и толщины линий .....	362
Рисование сложных фигур .....	363
Как рисуются сложные контуры .....	363
Перо. Перемещение пера .....	364
Прямые линии .....	364
Дуги .....	365
Кривые Безье .....	366
Прямоугольники .....	368
Задание стиля линий .....	369
Вывод текста .....	370
Использование сложных цветов .....	372
Линейный градиентный цвет .....	372
Радиальный градиентный цвет .....	375
Графический цвет .....	376
Вывод внешних изображений .....	378
Создание тени у рисуемой графики .....	380
Преобразования системы координат .....	380
Сохранение и загрузка состояния .....	381
Перемещение начала координат канвы .....	381

---

Поворот системы координат .....	382
Изменение масштаба системы координат.....	383
Управление наложением графики .....	384
Создание маски .....	386
Создание графического логотипа Web-сайта .....	386
<b>Заключение .....</b>	<b>391</b>
<b>Приложение. Расширения CSS.....</b>	<b>393</b>
Многоцветные рамки.....	394
Рамки со скругленными углами.....	394
Выделение со скругленными углами .....	396
Многоколоночная верстка.....	397
Преобразования CSS.....	401
<b>Предметный указатель .....</b>	<b>405</b>

# Введение

Мир Web-дизайна в очередной раз лихорадит. Шутка ли — новые интернет-технологии на подходе!

## Что грядет нового в Web-дизайне

Сейчас, наверно, даже школьники знают, что содержимое Web-страниц создается с помощью языка HTML, а внешний вид их элементов определяется стилями, которые описываются на языке CSS. Существует также возможность написать небольшие программы на языке JavaScript, которые встраиваются в саму Web-страницу и изменяют ее содержимое в ответ на действия посетителя, — Web-сценарии.

Все эти языки и технологии были созданы более десяти лет тому назад, и за последнее время в них мало что изменилось (а в языке HTML не изменилось вообще ничего). Так что за последние лет десять в Web-дизайне не было никаких революций — только небольшие эволюционные изменения.

Но уже готовятся новые стандарты, которые описывают очередные версии этих языков: HTML 5 и CSS 3. Они обещают принести в Web-дизайн много нового.

- Упрощенную вставку на Web-страницу аудио- и видеоматериалов.
- Возможности рисования на Web-страницах.
- Многоколоночную верстку текста.
- Поддержку работы в оффлайн-режиме (при отключении от Интернета).
- Дополнительную поддержку мобильных устройств.
- Поддержку специальных Web-обозревателей для лиц с ограниченными физическими возможностями.
- И, как водится, многое-многое другое.

Звучит заманчиво, но... Сейчас эти стандарты существуют только в виде "черновых" редакций, и когда выйдут "чистовые", окончательные, никто не знает.

Так отчего же разгорелся весь сыр-бор?

Разработчики Web-обозревателей, не дожидаясь, пока их коллеги, "сочиняющие" интернет-стандарты, завершат работу над HTML 5 и CSS 3, уже внедряют под-

держку некоторых их возможностей в свои творения. Так, Mozilla Firefox, Opera, Google Chrome и Apple Safari уже поддерживают интернет-мультимедиа в стиле HTML 5, программное рисование на Web-страницах и работу в оффлайновом режиме. Пусть и не в полной мере, но все-таки поддерживают!

"Не в теме" пока еще Microsoft Windows Internet Explorer. Однако Microsoft уже довольно давно объявила о разработке новой версии своего Web-обозревателя под номером 9. И в плане поддержки всех "горячих" интернет-новинок грозит заткнуть за пояс всех конкурентов. Что ж, у автора есть причины верить: даже предварительная версия Internet Explorer 9, совсем-совсем "сырая", на момент написания этой книги выглядит очень даже неплохо.

Но даже возможности действующих на сегодняшний день версий HTML и CSS на деле используются не в полной мере:

- подгружаемое содержимое — загрузка части содержимого вместо целой Web-страницы — практически не применяется;
- генерируемое содержимое — программное создание части содержимого Web-страницы после ее загрузки — применяется мало;
- разделение содержимого, представления и поведения Web-страниц также почти не реализуется.

А ведь все это способно значительно упростить труд Web-дизайнера и заметно улучшить вид и удобство использования Web-сайтов. Вот такие они Web-дизайнеры, не ведают своей выгоды...

## О чем эта книга

В книге описываются:

- Язык HTML и принципы создания содержимого Web-страниц.
- Язык CSS и принципы создания представления Web-страниц.
- Возможности HTML 5 и CSS 3, уже поддерживаемые современными Web-обозревателями.
- Основы Web-программирования, язык JavaScript и принципы создания поведения Web-страниц.
- Библиотека Ext Core — инструмент, призванный упростить труд Web-программиста.
- Создание интерактивных Web-страниц с конкретными примерами.
- Реализация подгружаемого и генерируемого содержимого и семантической разметки данных средствами JavaScript.
- Использование специальных средств — Web-форм, элементов управления и свободных контейнеров — для обеспечения дополнительной функциональности Web-сайтов.
- Реализация программного рисования на Web-страницах средствами HTML 5.

В результате читатель создаст полнофункциональный Web-сайт — справочник по HTML и CSS. Конечно, это только пример — данные технологии можно применить и для разработки любого другого Web-сайта.

## Какие программы используются в этой книге

Вопрос далеко не праздный, учитывая то, что за программы сегодня приходится платить... Так что же за программы использовал автор?

Только бесплатные!

- Блокнот — простейший текстовый редактор, стандартно поставляемый в составе Windows.
- Firefox версий 3.6.\* — Web-обозреватель. Все примеры тестировались на нем.
- Opera 10.52 и Apple Safari 4.\* — Web-обозреватели. На них автор тестировал некоторые примеры.

Другие программы автор в работе практически не применял.

## Типографские соглашения

В этой книге часто приводятся форматы написания различных тегов HTML, атрибутов стилей CSS и выражений JavaScript. Нам необходимо запомнить типографские соглашения, используемые для их написания.

### **ВНИМАНИЕ!**

Все эти типографские соглашения автор применяет только в форматах написания тегов HTML, атрибутов стилей CSS и выражений JavaScript. В коде примеров они не имеют смысла.

В угловые скобки (<>) заключены названия значений атрибутов, параметров или фрагментов кода, которые, в свою очередь, набраны курсивом. В код реального Web-сценария, разумеется, нужно подставить реальное значение, конкретный параметр или код.

Пример:

```
<MAP NAME="<имя карты>">
</MAP>
```

Здесь вместо подстроки <имя карты> нужно подставить конкретное имя карты.

Пример:

```
<FORM>
  <теги, формирующие элементы управления>
</FORM>
```

Здесь вместо подстроки <теги, формирующие элементы управления> следует подставить реальные HTML-теги, формирующие элементы управления.

В квадратные скобки ([]) заключены необязательные фрагменты кода:

```
<LEGEND [ACCESSKEY="<быстрая клавиша>"]><текст заголовка>/LEGEND>
```

Здесь атрибут тега ACCESSKEY может присутствовать, а может и отсутствовать.

Символом вертикальной черты (|) разделены фрагменты кода, из которых в данном месте должен присутствовать только один:

```
SHAPE="rect|circle|poly"
```

Здесь в качестве значения атрибута тега SHAPE должна присутствовать только одна из доступных строк: rect, circle или poly.

Слишком длинные, не помещающиеся на одной строке фрагменты кода автор разбивает на несколько строк и в местах разрывов ставит знаки ¶.

Пример:

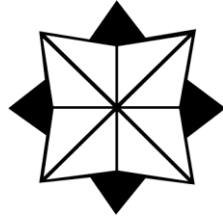
```
var s = "<LI><CODE><A HREF=\"\" + aDataBase[i].url + "\">\" +  
¶aDataBase[i].name + "</A></CODE></LI>";
```

Приведенный код разбит на две строки, но должен быть набран в одну. Знаки ¶ при этом нужно удалить.

## Благодарности

Автор приносит благодарности своим родителям, знакомым и коллегам по работе.

- Губиной Наталье Анатольевне, начальнику отдела АСУ Волжского гуманитарного института (г. Волжский Волгоградской обл.), где работает автор, — за понимание и поддержку.
- Всем работникам отдела АСУ Волжского гуманитарного института — за понимание и поддержку.
- Родителям — за терпение, понимание и поддержку.
- Архангельскому Дмитрию Борисовичу — за дружеское участие.
- Шапошникову Игорю Владимировичу — за содействие.
- Рыбакову Евгению Евгеньевичу, заместителю главного редактора издательства "БХВ-Петербург", — за неоднократные побуждения к работе, без которых автор давно бы обленился.
- Издательству "БХВ-Петербург" — за издание моих книг.
- Разработчикам Web-обозревателей Firefox, Opera, Chrome и Safari и библиотеки Ext Core, если они меня слышат, — за замечательные программные продукты.
- Всем своим читателям и почитателям — за прекрасные отзывы о моих книгах.
- Всем, кого я забыл здесь перечислить, — за все хорошее.



# ЧАСТЬ I

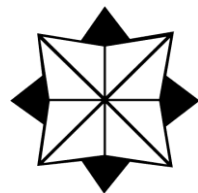
## Содержимое Web-страниц. Язык HTML 5

- Глава 1.** Введение в современный Web-дизайн. Web 2.0. Создание Web-страниц
- Глава 2.** Структурирование текста
- Глава 3.** Оформление текста
- Глава 4.** Графика и мультимедиа
- Глава 5.** Таблицы
- Глава 6.** Средства навигации





# ГЛАВА 1



## Введение в современный Web-дизайн. Web 2.0. Создание Web-страниц

Всемирная паутина, WWW, Web-дизайн, Web-сайт, Web-страница — все знают, что это такое. Но что такое современная Всемирная паутина, современный Web-дизайн и современная Web-страница?

Именно с ответов на все эти вопросы начнется данная книга. Далее мы немного поговорим о принципах функционирования Интернета, Web-страницах и Web-сайтах, создадим нашу первую Web-страницу, начнем изучать язык HTML 5 и подберем программы, которые будем использовать в работе. Так сказать, с места в карьер...

### Современный Web-дизайн. Концепция Web 2.0

Раньше доступ в Интернет можно было получить только с компьютеров. Потом в Интернет стали выходить с мобильных телефонов. Сейчас к Сети подключились мультимедийные плееры, устройства чтения электронных книг и телевизоры. А завтра — кто знает; может быть, мы будем выходить на Web-сайты с утюга или пылесоса...

"Я буду везде", — заявляет Интернет. — "Я стану вездесущим. Все готовьтесь к моему приходу!"

### Что требуется от современного Web-сайта

Будем готовиться... Но что нам, как будущим Web-дизайнерам, для этого следует сделать? Соблюсти три несложных правила.

1. Строго соблюдать все интернет-стандарты.
2. Тщательно продумать наполнение Web-страниц.
3. Позаботиться о доступности Web-страниц.

Рассмотрим их подробнее.

Интернет грозит прийти на самые разные устройства, которые могут быть основаны на разных аппаратных и программных платформах, зачастую сильно отли-

чающихся друг от друга. Так, персональные компьютеры построены на аппаратной платформе Intel и программной платформе Microsoft Windows (по крайней мере, большинство). Мобильный телефон автора основан на аппаратно-программной платформе Samsung. А на чем будет работать интернет-пылесос, сейчас не может сказать никто.

Одно объединяет все это аппаратно-программное многообразие — соответствие интернет-стандартам. Иначе устройства в лучшем случае будут отображать Web-страницы неправильно, в худшем — вообще не будут работать.

Из этого следует первое правило из перечисленных ранее — Web-дизайнеры при создании Web-страниц обязаны строго придерживаться современных интернет-стандартов, чтобы их творения одинаково (ну, или почти одинаково) отображались на всех устройствах.

Первое правило также требует отказа от устаревших и закрытых, фирменных интернет-технологий. С устаревшими технологиями все понятно: старье — не помощник новому. Закрытые же технологии неудобны тем, что зачастую контролируются единственной фирмой, которая единолично "заказывает музыку" и далеко не всегда прислушивается к мнению интернет-сообщества. К таким технологиям относятся, в частности, Adobe Flash и Microsoft ActiveX.

Открытыми интернет-стандартами, в том числе и Web-стандартами, занимается организация World Wide Web Consortium (Консорциум Всемирной паутины), или сокращенно W3C. Она разрабатывает стандарты, согласует их с требованиями участников рынка и публикует на своем Web-сайте <http://www.w3.org>. Все опубликованные там стандарты обязательны к применению.

Интернет когда-то начинался как сеть ученых, которым было нужно обмениваться результатами исследований. А что представляли собой эти результаты? В основном, текст, возможно, с иллюстрациями. Ученые — публика в этом смысле невзыскательная, им вполне хватало скромных возможностей тогдашнего WWW.

Теперь же абсолютное большинство пользователей Интернета — обычные обыватели. Им мало простого текста с парой картинок, им подавай хорошо оформленный текст, музыку и видео. Они требовательнее первых обитателей Сети.

Отсюда вытекает второе правило — Web-дизайнеры должны заботиться о полноте и удобстве наполнения Web-страниц.

- ❑ Структура Web-страниц должна быть хорошо продумана, чтобы посетитель сразу смог найти на них все, что ему нужно.
- ❑ Web-страницы должны легко читаться и не "резать" глаза.
- ❑ К важным материалам желательно привлечь внимание посетителя, а маловажные скрыть. В этом могут помочь динамические элементы: раскрывающиеся при щелчке мышью абзацы, гиперссылки, выделяющиеся при наведении курсора мыши, и пр.
- ❑ Если Web-сайт посвящен музыке или видео, все это должно быть доступно для воспроизведения прямо на его Web-страницах, без загрузки.

- ❑ Одним словом — все для удобства посетителя! (Пожалуй, это правило следовало бы поставить в начале списка...)

Интернет грозитя прийти на самые разные устройства с различными характеристиками: быстродействием процессора, объемом памяти, разрешением экрана, скоростью доступа к Сети. Но все они должны обеспечивать единообразный вывод Web-страниц. Как этого достигнуть?

Вот и третье правило — Web-дизайнеры должны заботиться о доступности Web-страниц.

- ❑ Web-страницы следует делать как можно более компактными. Чем компактнее файл, тем быстрее он загружается по сети — это аксиома.
- ❑ Web-страницы не должны быть чересчур сложными. Чем сложнее Web-страница, тем больше времени и системных ресурсов требует ее обработка и вывод.
- ❑ Web-страницы не должны требовать для отображения никакого дополнительного программного обеспечения. В идеале для их вывода достаточно только Web-обозревателя.

Но как эти правила реализуются на практике? Давайте откроем какой-нибудь современный Web-сайт, например, принадлежащий организации W3C (рис. 1.1). Как мы помним, его можно найти по интернет-адресу **<http://www.w3.org>**.

Что же мы здесь видим?

- ❑ Web-сайт создан с учетом всех современных интернет-стандартов. Он отображается во всех Web-обозревателях практически одинаково.
- ❑ Web-сайт не использует ни устаревших, ни закрытых интернет-технологий.
- ❑ Структура Web-страниц исключительно ясна — мы можем без проблем найти все, что нужно. Слева находится набор гиперссылок, ведущих на другие Web-страницы Web-сайта, посередине — список новостей и гиперссылки на избранные статьи, справа — гиперссылки на дополнительные материалы.
- ❑ Web-страница прекрасно читается. Тонкий шрифт без засечек, спокойная серо-голубая цветовая гамма, тонкие рамочки со скругленными углами, минимум графики — ничто не бросается в глаза.
- ❑ Есть даже видеоролик!
- ❑ Web-страница быстро загружается и мгновенно выводится на экран.
- ❑ Web-страница ничего не требует для своего вывода, кроме Web-обозревателя.

Налицо и соблюдение стандартов, и наполнение, и доступность. Три из трех!

Именно такие Web-страницы мы и будем учиться создавать в данной книге.

## Концепция Web 2.0

Давайте еще раз обратимся к рассмотренным ранее правилам и немного расширим их.



Рис. 1.1. Главная Web-страница Web-сайта организации W3C

- При создании Web-страниц следует придерживаться современных интернет-стандартов. При этом нужно полностью отказаться от устаревших и закрытых интернет-технологий, как не укладывающихся в современную парадигму Web-дизайна и зачастую не поддерживаемых всеми Web-обозревателями.
- Особое внимание нужно обратить на структуру и наполнение Web-страниц. Структура Web-страниц должна быть максимально простой, а наполнение — достаточно богатым, чтобы посетитель быстро нашел нужную ему информацию. Кроме того, необходимо создавать Web-страницы так, чтобы дизайн не мешал восприятию информации.
- Web-страницы обязательно следует делать максимально доступными на любых устройствах. Web-страницы должны быстро загружаться и выводиться на экран. Также Web-страницы не должны требовать для отображения никакого дополнительного программного обеспечения.

Фактически здесь мы привели постулаты так называемой концепции *Web 2.0*. Это список правил, которым должен удовлетворять любой Web-сайт, претендующий на

звание современного. Образно выражаясь, это флаг, который совместно несут труженики Web-индустрии, шагая в ногу со временем.

Также концепция Web 2.0 предусматривает четыре принципа, являющиеся "передним краем" Web-дизайна. Пока еще очень мало Web-сайтов им следует (и "домашний" Web-сайт W3C, увы, не исключение...). Рассмотрим их по порядку.

Принцип первый — разделение содержимого, представления и поведения Web-страницы. Здесь *содержимое* — это информация, которая выводится на Web-странице, *представление* описывает формат вывода этой информации, а *поведение* — реакцию Web-страницы или отдельных ее элементов на действия посетителя. Благодаря их разделению мы сможем править, скажем, содержимое, не затрагивая представление и поведение, или поручать создание содержимого, представления и поведения разным людям.

Принцип второй — *подгружаемое содержимое*. Вместо того чтобы обновлять всю Web-страницу в ответ на щелчок на гиперссылке, мы можем подгружать только ее часть, содержащую необходимую информацию. Это сильно уменьшит объем передаваемой по сети информации (сетевой трафик) и позволит выполнять какие-либо действия с данными после их подгрузки.

Принцип третий — *генерируемое содержимое*. Какая-то часть Web-страницы может не загружаться по сети, а генерироваться прямо на месте, в Web-обозревателе. Так мы еще сильнее сократим сетевой трафик.

Принцип четвертый — *семантическая разметка* данных. Она позволит нам связать выводимые на Web-страницу данные согласно каким-либо правилам. Например, мы можем семантически связать страницы справочника по HTML, и посетитель, загрузив какую-либо страницу, сможет сразу же перейти на связанные с ней страницы, содержащие дополнительные или родственные сведения.

В качестве примера Web-сайта, реализующего эти четыре принципа, можно привести Web-сайт — справочник по библиотеке Ext Core, расположенный по интернет-адресу <http://www.extjs.com/products/core/docs/> и показанный на рис. 1.2.

- ❑ Содержимое, представление и поведение составляющих его Web-страниц хранится отдельно, в разных файлах.
- ❑ При переходах с одной статьи справочника на другую подгружается только сам текст статьи. Остальные части Web-страницы, в частности иерархический список статей, остаются неизменными.
- ❑ После загрузки текста статьи на его основе генерируется окончательное ее представление. Фактически мы имеем генерируемое содержимое.
- ❑ Статьи справочника связаны друг с другом семантически. Эти связи используются для генерирования гиперссылок на "родственные" статьи.

Рассмотренные нами два Web-сайта — это концепция Web 2.0 в действии! Хотите создать что-то подобное? Хотите в плане поддержки интернет-стандартов "утереть нос" самому W3C? Тогда читайте эту книгу!

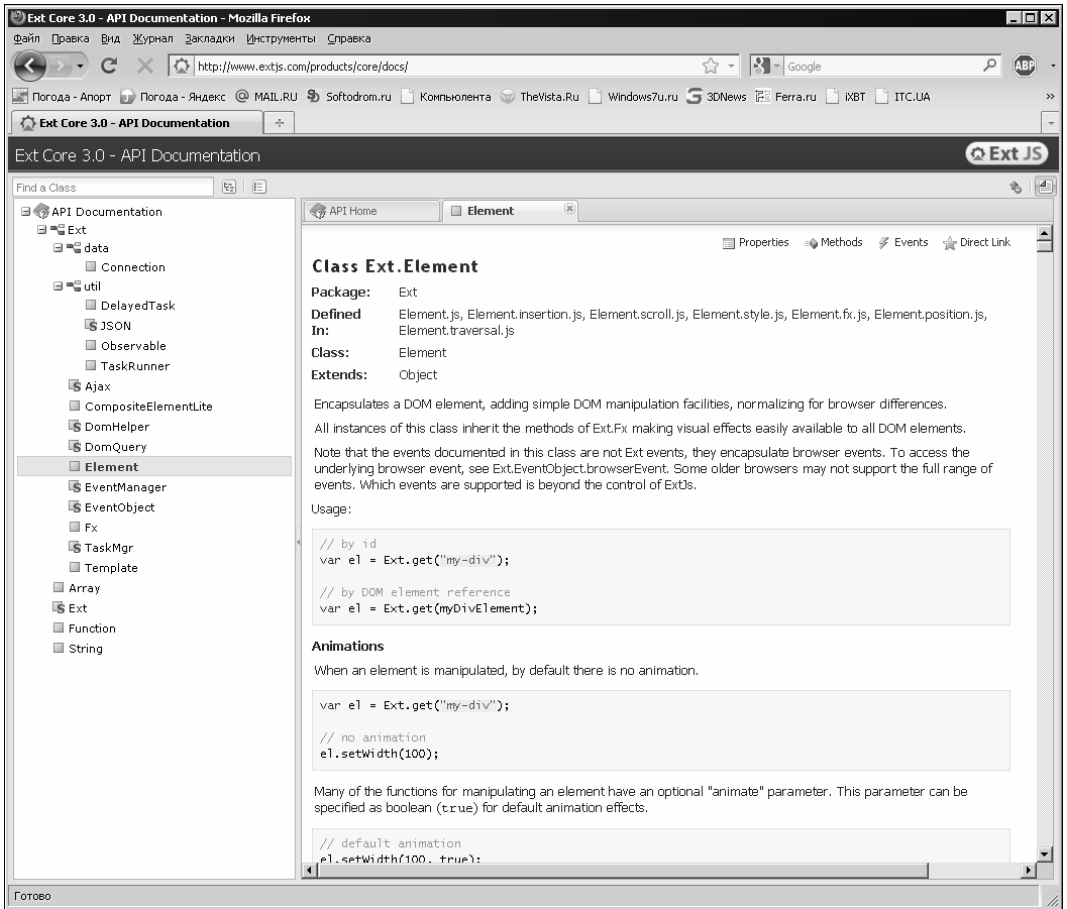


Рис. 1.2. Web-сайт — справочник по библиотеке Ext Core

## Интернет: как все это работает

Давайте еще раз посмотрим на Web-сайт — справочник по библиотеке Ext Core. И зададимся вопросом, вынесенным в заголовок данного раздела.

Как все это работает? Откуда Web-обозреватель получает нужную Web-страницу? Кто отвечает за работу сложного механизма под названием Всемирная паутина?

## Клиенты и серверы Интернета. Интернет-адреса

Возьмем для примера главную Web-страницу Web-сайта, который мы открыли. Она должна где-то храниться. Но где? На диске другого компьютера, подключенного к сети (в данном случае — к сети Интернет), который может принадлежать как автору Web-сайта, так и сторонней организации, предоставляющей доступ в Интернет (*интернет-провайдеру*). И хранится она в виде файла или набора файлов, таких же, какие в изобилии "водятся" на нашем собственном компьютере.

Но как мы смогли получить и просмотреть содержимое этого файла? Во-первых, посредством самой сети — она связала компьютер, хранящий файл, с нашим. Во-вторых, с помощью особых программ, которые, собственно, и выполнили передачу файла. Эти программы делятся на две группы.

Программы первой группы взаимодействуют непосредственно с пользователем: принимают от него запросы на информацию, которая хранится где-то в сети, получают ее, выводят на экран и, возможно, позволяют ее править и отправлять обратно. Такие программы называют *клиентами*.

Для просмотра Web-страниц мы пользуемся Web-обозревателем. Это программа-клиент; она принимает от нас интернет-адреса Web-страниц, получает файлы, хранящие их содержимое, и выводит это содержимое на экран. Программа почтового клиента позволяет как извлекать из почтового ящика полученные письма, так и создавать новые. Существуют также клиенты чата, систем мгновенных сообщений и пр.

Но клиенты не имеют прямого доступа к хранящейся на других компьютерах информации. Они не могут просто "залезть" на жесткий диск удаленного компьютера и прочесть оттуда файл. Так сделано из соображений безопасности. Вместо этого они отправляют запросы программам второй группы — серверам.

*Серверы* работают на компьютерах, хранящих информацию, которая должна быть доступна в сети. Они принимают запросы от клиентов, извлекают требуемую информацию из файлов и отправляют им. Также они могут получать введенную пользователями информацию от клиентов и сохранять их в файлах, при этом, возможно, как-то обработав. Можно сказать, что серверы выступают посредниками между клиентами и запрашиваемой ими информацией.

Для управления Web-сайтами используются *Web-серверы*, которые принимают запросы от клиентов и отправляют им содержимое требуемых файлов. Для управления почтовыми службами применяются серверы электронной почты; они сохраняют пришедшие письма в файлах, выдают их почтовым клиентам по запросу, принимают от клиентов новые сообщения и отправляют их по указанному адресу — в общем, работают как почтовое отделение. Службы чатов и мгновенных сообщений также имеют свои серверы.

Клиенты — лицо Интернета. Серверы — его сердце.

Но как указать, какая информация и с какого сервера нам требуется? С помощью определенным образом составленного интернет-адреса.

Каждая единица информации — файл, ящик электронной почты, канал чата, — доступная в сети, однозначно идентифицируется интернет-адресом, который представляет собой строку из букв, цифр и некоторых других символов.

Интернет-адрес включает в себя две части:

- интернет-адрес программы-сервера, работающей на компьютере;
- указатель на нужную единицу информации, например, путь к файлу, имя ящика электронной почты, имя канала чата и др. (может отсутствовать).



Рассмотрим несколько примеров интернет-адресов.

В интернет-адресе **http://www.somesite.ru/folder1/file1.htm** присутствуют обе части. Здесь **http://www.somesite.ru** — интернет-адрес программы-сервера, в данном случае — Web-сервера, а **/folder1/file1.htm** — путь к запрашиваемому файлу.

В интернет-адресе **http://www.thersite.ru** присутствует только интернет-адрес Web-сервера. Какая информация в этом случае будет отправлена клиенту (Web-обозревателю), мы узнаем потом.

А в адресе **user@mail.someserver.ru** мы видим интернет-адрес сервера электронной почты (**mail.someserver.ru**) и имя почтового ящика (**user**).

Разговор об интернет-адресах еще не закончен. Мы вернемся к нему в *главе 6*, когда будем рассматривать средства навигации по Web-сайту, в частности, гиперссылки. А пока что давайте подробнее поговорим о Web-серверах и их нелегкой "работе".

## Web-сайты и Web-серверы

Как мы только что выяснили, все интернет-программы делятся на клиенты и серверы. Клиенты работают на стороне пользователя, получают от них интернет-адреса и выводят им полученную с этих адресов информацию. Серверы принимают запросы от клиентов, находят запрашиваемую ими информацию на дисках серверных компьютеров и отправляют ее клиентам.

Во Всемирной паутине WWW в качестве клиентов используются Web-обозреватели, а в качестве серверов — Web-серверы. Это мы тоже знаем.

Любая информация на дисках компьютера хранится в файлах. Ну, это знает любой более-менее подкованный пользователь...

Web-страницы также хранятся в файлах с расширением htm или html (или, с учетом описанных во введении типографских соглашений, htm[1]). Одна Web-страница занимает один или более файлов.

Web-сайт — это совокупность множества Web-страниц, объединенных общей темой и связанных друг с другом посредством гиперссылок (о них мы поговорим в *главе 6*). Следовательно, Web-сайт — это также набор файлов, возможно, хранящихся в разных папках, — так ими удобнее управлять.

А теперь — внимание! Мы рассмотрим некоторые "интимные" подробности работы Web-серверов, которые знает не каждый интернетчик.

Прежде всего, для хранения всех файлов, составляющих Web-сайт, на диске серверного компьютера выделяется особая папка, называемая *корневой папкой Web-сайта*. Путь к этой папке указывается в настройках Web-сервера, чтобы он смог ее "найти".

Все, повторим — все файлы, составляющие Web-сайт, должны храниться в корневой папке или в папках, вложенных в нее. Файлы, расположенные вне корневой папки, с точки зрения Web-сервера не существуют. Так сделано для безопасности, чтобы злоумышленник не смог получить доступ к дискам серверного компьютера.

Когда в интернет-адресе указывается путь к запрашиваемому файлу, Web-сервер отсчитывает его относительно корневой папки. Это проще всего показать на примерах.

- **http://www.somesite.ru/page1.htm** — в ответ будет отправлен файл page1.htm, хранящийся в корневой папке Web-сайта.
- **http://www.somesite.ru/chapter2/page6.htm** — в ответ будет отправлен файл page6.htm, хранящийся в папке chapter2, которая вложена в корневую папку Web-сайта.
- **http://www.somesite.ru/downloads/others/archive.zip** — в ответ будет отправлен файл archive.zip, хранящийся в папке others, вложенной в папку downloads, которая, в свою очередь, вложена в корневую папку Web-сайта.

Но ведь мы нечасто набираем интернет-адрес, включающий путь к запрашиваемому файлу. Гораздо чаще интернет-адреса включают только адрес программы-сервера, например, **http://www.somesite.ru**. Что в таком случае делает Web-сервер? Какой файл он отправляет в ответ?

Специально для этого предусмотрены так называемые *Web-страницы по умолчанию*. Такая Web-страница выдается клиенту, если он указал в интернет-адресе только путь к файлу, но не его имя. Обычно файл Web-страницы по умолчанию имеет имя default.htm[1] или index.htm[1], хотя его можно изменить в настройках Web-сервера.

Так, если мы наберем интернет-адрес **http://www.somesite.ru**, Web-сервер вернет нам файл Web-страницы по умолчанию, хранящийся в корневой папке Web-сайта. Практически всегда это будет *главная Web-страница* — та, с которой начинается "путешествие" по Web-сайту.

Мы можем набрать и интернет-адрес вида **http://www.somesite.ru/chapter2/**. Тогда Web-сервер отправит нам файл Web-страницы по умолчанию, хранящийся в папке chapter2, вложенной в корневую папку Web-сайта.

С Web-сайтами и Web-серверами пока все. Настала пора заглянуть внутрь Web-страниц и, чего уж тянуть резину, создать нашу первую, совсем простую Web-страничку. И по ходу дела начать изучение языка HTML 5, без которого в Web-дизайне не обойтись.

## Основные принципы создания Web-страниц. Язык HTML 5

Web-страницы выглядят зачастую очень пестро: разнокалиберные куски текста, таблицы, картинки, врезки, сноски и даже фильмы. Но описывается все это в виде обычного текста. Да-да, Web-страницы — суть текстовые файлы, которые можно создать с помощью хорошо знакомого нам редактора Блокнот, поставляемого в составе Windows! (Разумеется, подойдет любой аналогичный текстовый редактор.)

Для форматирования содержимого Web-страниц применяется особый язык — *HTML* (HyperText Markup Language, язык гипертекстовой разметки). С помощью

команд — *тегов* — этого языка создают и абзацы текста, и заголовки, и врезки, и даже таблицы.

Первая версия языка HTML появилась очень давно, еще в 1992 году. С тех пор по Сети утекло немало гигабайт... HTML также не стоял на месте. В данный момент готовится к выходу окончательная спецификация новой версии HTML под номером 5, и многие Web-обозреватели уже поддерживают некоторые ее возможности. Ее-то мы и будем изучать.

## Язык HTML и его теги

Изучать HTML лучше всего на примере. Так что давайте сразу же создадим нашу первую Web-страничку. Благо Windows уже содержит необходимый для этого инструмент — Блокнот.

### НА ЗАМЕТКУ

Вообще, для создания Web-страниц существует множество специальных программ — Web-редакторов. Они позволяют работать с Web-страницами, даже не зная HTML, — как с документами Microsoft Word, просто набирая текст и форматировав его. Также они следят за правильностью расстановки тегов, помогут быстро создать сложный элемент Web-страницы и даже опубликовать готовый Web-сайт в Сети. К таким программам принадлежит, в частности, известный Web-редактор Adobe Dreamweaver.

Однако мы пока что будем пользоваться простейшим текстовым редактором Блокнот. Это позволит нам лучше познакомиться с HTML.

Откроем Блокнот и наберем в нем текст (или, как говорят бывалые программисты, код), приведенный в листинге 1.1.

### Листинг 1.1

```
<!DOCTYPE html>
<HTML>
  <HEAD>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=utf-8">
    <TITLE>Пример Web-страницы</TITLE>
  </HEAD>
  <BODY>
    <H1>Справочник по HTML</H1>
    <P>Приветствуем на нашем Web-сайте всех, кто занимается Web-дизайном!
    Здесь вы сможете найти информацию обо всех интернет-технологиях,
    применяемых при создании Web-страниц. В частности, о языке
    <STRONG>HTML</STRONG>.</P>
  </BODY>
</HTML>
```

Проверим набранный код на ошибки и сохраним в файл с именем 1.1.htm. Только сделаем при этом две важные вещи.

1. Сохраним HTML-код в кодировке UTF-8. Для этого в диалоговом окне сохранения файла Блокнота найдем раскрывающийся список **Кодировка** и выберем в нем пункт **UTF-8**.

2. Заклучим имя файла в кавычки. Иначе Блокнот добавит к нему расширение txt, и наш файл получит имя 1.1.htm.txt.

Все, наша первая Web-страница готова! Теперь осталось открыть ее в Web-обозревателе и посмотреть на результат.

Мы можем использовать стандартно поставляемый в составе Windows Web-обозреватель Microsoft Internet Explorer. Но Internet Explorer на данный момент не поддерживает HTML 5; его поддержку обещают только в версии 9, которая пока находится в разработке. HTML 5 поддерживают последние версии Mozilla Firefox, Opera, Apple Safari и Google Chrome, поэтому предпочтительнее какая-либо из этих программ.

Откроем же Web-страницу в выбранном Web-обозревателе (автор выбрал Firefox) и посмотрим на нее (рис. 1.3).

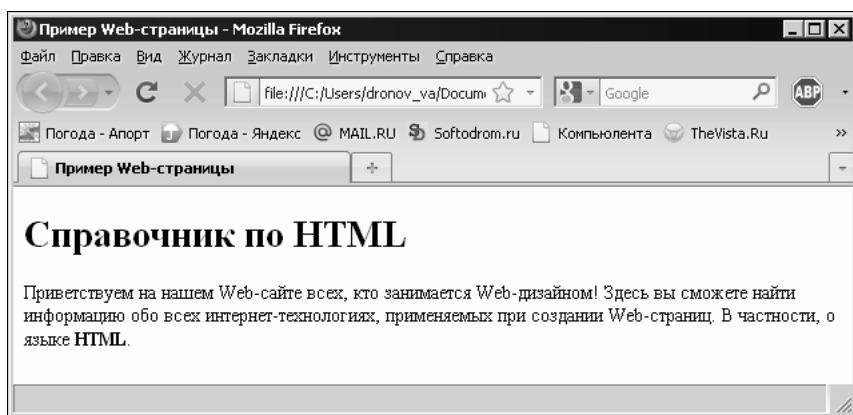


Рис. 1.3. Наша первая Web-страница

Видите? Мы создали Web-страницу, содержащую большой "кричащий" заголовок, абзац текста, который автоматически разбивается на строки и содержит фрагмент текста, выделенный полужирным шрифтом (аббревиатура "HTML"). И все это — в "голом" тексте, набранном в Блокноте!

Теперь посмотрим, что же мы такое написали в файле 1.1.htm. Пока что ограничимся небольшим фрагментом HTML-кода (листинг 1.2).

#### Листинг 1.2

```
<h1>Справочник по HTML</h1>
<p>Приветствуем на нашем Web-сайте всех, кто занимается Web-дизайном!
Здесь вы сможете найти информацию обо всех интернет-технологиях,
применяемых при создании Web-страниц. В частности, о языке
<strong>HTML</strong>.</p>
```

Здесь мы видим текст заголовка и абзаца. И еще странные слова, взятые в угловые скобки — символы < и >. Что это такое?

Это и есть теги HTML, о которых упоминалось ранее. Они превращают тот или иной фрагмент HTML-кода в определенный элемент Web-страницы: абзац, заголовок или текст, выделенный полужирным шрифтом.

Начнем с тегов `<h1>` и `</h1>`, поскольку они идут первыми. Эти теги превращают фрагмент текста, находящийся между ними, в заголовок. Тег `<h1>` помечает начало фрагмента, на который распространяется действие тега, и называется *открывающим*. А тег `</h1>` устанавливает конец "охватываемого" фрагмента и называется *закрывающим*. Что касается самого фрагмента, заключенного между открывающим и закрывающим тегами, то он называется *содержимым тега*. Именно к содержанию применяется действие тега.

Все теги HTML представляют собой символы `<` и `>`, внутри которых находится *имя тега*, определяющее назначение тега. Закрывающий тег должен иметь то же имя, что и открывающий; единственное отличие закрывающего тега — символ `/`, который ставится между символом `<` и именем тега.

Рассмотренные нами теги `<h1>` и `</h1>` в HTML фактически считаются одним тегом `<h1>`. Такой тег называется *парным*.

Поехали дальше. Парный тег `<p>` создает на Web-странице абзац; содержимое тега станет текстом этого абзаца. Такой абзац будет отображаться с отступами сверху и снизу. Если он полностью помещается по ширине в окне Web-обозревателя, то отобразится в одну строку; в противном случае сам Web-обозреватель разобьет его на несколько более коротких строк. (То же справедливо и для заголовка.)

Парный тег `<strong>` выводит свое содержимое полужирным шрифтом. Как мы видим, тег `<strong>` вложен внутрь содержимого тега `<p>`. Это значит, что содержимое тега `<strong>` будет отображаться как часть абзаца (тега `<p>`).

Давайте ради интереса выделим слова "Web-дизайном" курсивом. Для этого поместим соответствующий фрагмент текста абзаца в парный тег `<em>`:

```
<p>Приветствуем на нашем Web-сайте всех, кто занимается  
<em>Web-дизайном</em>! Здесь вы сможете найти информацию обо всех  
. . .
```

Сохраним исправленную Web-страницу и обновим содержимое окна Web-обозревателя, нажав клавишу `<F5>`. Получилось! Да мы уже стали Web-дизайнерами!

Осталось рассмотреть важнейшие правила, согласно которым пишется HTML-код.

- ❑ Имена тегов можно писать как прописными (большими), так и строчными (малыми) буквами. Традиционно в языке HTML имена тегов пишут прописными буквами.
- ❑ Между символами `<`, `>`, `/` и именами тегов, а также внутри имен тегов не допускаются пробелы и переносы строк.
- ❑ В обычном тексте, не являющемся тегом, не должны присутствовать символы `<` и `>`. (Эти символы называют *недопустимыми*.) В противном случае Web-обозреватель сочтет фрагмент текста, где встречается один из этих символов, тегом и отобразит Web-страницу некорректно.

На этом пока закончим. Впоследствии, изучив другие языковые элементы HTML, мы пополним список этих правил.

## Вложенность тегов

Если мы снова посмотрим на приведенный в листинге 1.2 фрагмент HTML-кода, то заметим, что одни теги вложены в другие. Так, тег `<STRONG>` вложен в тег `<P>`, являясь частью его содержимого. Тег `<P>`, в свою очередь, вложен в тег `<BODY>`, а тот — в "глобальный" тег `<HTML>`. (Теги `<BODY>` и `<HTML>` мы рассмотрим чуть позже.) Такая *вложенность тегов* в HTML — обычное явление.

Когда Web-обозреватель встречает тег, вложенный в другой тег, он как бы накладывает действие "внутреннего" тега на эффект "внешнего". Так, действие тега `<STRONG>` будет наложено на действие тега `<P>`, и фрагмент абзаца окажется выделенным полужирным шрифтом, при этом оставаясь частью этого абзаца.

Давайте для примера текст "Web-дизайн", который мы недавно поместили в тег `<EM>`, заключим еще и в тег `<STRONG>`. Вот так:

```
<P>Приветствуем на нашем Web-сайте всех, кто занимается  
<EM><STRONG>Web-дизайном</STRONG></EM>! Здесь вы сможете найти  
. . .
```

В этом случае данный текст будет выделен полужирным курсивом. Иными словами, действие тега `<STRONG>` будет наложено на действие тега `<EM>`.

Теперь — внимание! Порядок следования закрывающих тегов должен быть обратным тому, в котором следуют теги открывающие. Говоря иначе, теги со всем их содержимым должны полностью вкладываться в другие теги, не оставляя "хвостов" снаружи.

Если же мы нарушим это правило и напишем такой HTML-код (обратите внимание на специально перепутанный порядок следования открывающих тегов):

```
<P>Приветствуем на нашем Web-сайте всех, кто занимается  
<EM><STRONG>Web-дизайном</EM></STRONG>! Здесь вы сможете найти  
. . .
```

Web-обозреватель может отобразить нашу Web-страницу неправильно.

### НА ЗАМЕТКУ

Нужно сказать, что современные Web-обозреватели "умеют" исправлять мелкие ошибки Web-дизайнера. Но именно мелкие!

Осталось выучить несколько новых терминов. Тег, в который непосредственно вложен данный тег, называется *родительским*, или *родителем*. В свою очередь, тег, вложенный в данный тег, называется *дочерним*, или *потомком*. Так, для тега `<EM>` в приведенном далее примере тег `<P>` — родительский, а тег `<STRONG>` — дочерний. Любой тег может иметь сколько угодно дочерних тегов, но только один родительский (что, впрочем, понятно — не может же он быть непосредственно вложен одновременно в два тега).

Элемент Web-страницы, в который вложен элемент, создаваемый данным тегом, называется *родительским*, или *родителем*. А элемент Web-страницы, который