

Вадим Дунаев



HTML, СКРИПТЫ И СТИЛИ

3-е издание

- HTML 4, HTML 5, XHTML
- CSS 2, CSS 3
- JavaScript, PHP
- SVG



Наиболее
полное
руководство

В ПОДЛИННИКЕ®

Вадим Дунаев

**HTML,
СКРИПТЫ И СТИЛИ**
3-е издание

Санкт-Петербург

«БХВ-Петербург»

2011

УДК 681.3.06
ББК 32.973.26-018.2
Д83

Дунаев В. В.

Д83 HTML, скрипты и стили. — 3-е изд., переб. и доп. — СПб.: БХВ-Петербург, 2011. — 816 с.: ил. — (В подлиннике)

ISBN 978-5-9775-0502-4

Рассмотрены средства создания Web-сайтов — языки разметки гипертекста (XHTML, HTML 4 и HTML 5), каскадные таблицы стилей (CSS 2 и CSS 3), а также языки сценариев JavaScript и PHP. Изложены краткие теоретические сведения и приведены многочисленные примеры типичных задач разработки сайтов. Предлагаемые решения инвариантны относительно наиболее популярных браузеров (Microsoft Internet Explorer, Mozilla Firefox, Opera, Apple Safari и Google Chrome). Приложения книги содержат краткие сведения о тегах HTML и свойствах CSS. В третьем издании существенно расширены главы, посвященные языкам разметки гипертекста HTML 4, HTML 5 и XHTML и каскадным таблицам стилей CSS 2 и CSS 3. Добавлен материал по масштабируемой векторной графике (SVG) и динамической графике, управляемой скриптами.

Для Web-дизайнеров

УДК 681.3.06
ББК 32.973.26-018.2

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Игорь Шишигин</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Леонид Кочин</i>
Компьютерная верстка	<i>Натальи Караваевой</i>
Корректор	<i>Наталья Першакова</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 24.01.11.

Формат 70×100¹/₁₆. Печать офсетная. Усл. печ. л. 65,79.

Тираж 1600 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.

Оглавление

Предисловие к третьему изданию	1
Введение	1
Благодарности	2
ЧАСТЬ I. HTML и стили.....	5
Глава 1. Что такое HTML и таблицы стилей CSS	7
1.1. Языки разметки документа	7
1.2. Что такое таблицы стилей	18
Глава 2. Структура (X)HTML-документа.....	22
2.1. Определение типа документа: дескриптор <code><!DOCTYPE...></code>	22
2.2. Структура собственно (X)HTML-кода	26
2.3. Раздел заголовка документа <code><head></code>	29
2.3.1. Тег <code><meta></code>	29
Группа <i>HTTP-EQUIV</i> (HTTP-эквиваленты)	30
Группа <i>NAME</i> (имя)	31
2.3.2. Тег <code><base></code>	32
2.3.3. Другие теги внутри <code><head></code>	33
2.4. Раздел тела документа <code><body></code>	35
2.5. Основные атрибуты тегов	35
2.6. Отображение элементов в нормальном потоке	37
Глава 3. Основы CSS.....	41
3.1. Присоединение таблиц стилей к (X)HTML-документу	41
3.2. Правила форматирования	42
3.2.1. Селекторы	42
3.2.2. Контекстные селекторы	44
3.2.3. Псевдоселекторы и псевдоэлементы	45
3.3. Приоритеты определений параметров стилей	47
3.4. Размерность и цвета	50
3.5. Блоки: поля, отступы, границы и размеры	52
3.6. Наследование параметров	60

Глава 4. Позиционирование с помощью CSS	62
4.1. <i>position:static</i>	63
4.2. <i>position:relative</i>	63
4.3. <i>position:absolute</i>	65
4.4. <i>position:fixed</i>	68
4.5. Отсчет координат	69
4.6. Слои	70
4.7. Обтекание	72
4.8. Видимость	75
4.8.1. <i>overflow</i>	75
4.8.2. <i>clip</i>	75
4.8.3. <i>visibility</i>	76
4.8.4. <i>display</i>	77
4.9. Размеры.....	78
4.10. Практические примеры	80
4.10.1. Центрирование элемента.....	80
4.10.2. Управление положением элемента с помощью мыши	82
4.10.3. Раскрывающаяся панель.....	82
Глава 5. Фон элементов и границ	84
5.1. <i>background</i>	84
5.2. <i>opacity</i>	88
5.3. <i>border-image</i>	92
5.4. <i>border-radius</i>	94
5.5. <i>box-shadow</i>	95
Глава 6. Ссылки	97
6.1. Текстовые ссылки	98
6.1.1. Простое меню ссылок.....	98
6.1.2. Двухуровневое меню ссылок	102
6.2. Графические и комбинированные ссылки	105
6.3. Графические карты ссылок	106
6.3.1. Клиентский вариант графической карты ссылок	106
6.3.2. Серверный вариант графической карты ссылок	109
6.4. Внутренние ссылки.....	110
6.5. URL-адреса	112
6.5.1. Структура URL.....	112
6.5.2. Абсолютные и относительные пути	114
6.5.3. Кодирование URL	115
6.5.4. Псевдо-URL JavaScript	117
Глава 7. Тексты	118
7.1. Шрифты	118
7.2. Основные теги разметки текстов.....	121
7.3. Специальные символы.....	122
7.4. Форматирование текста.....	123
7.4.1. Красная строка	123

7.4.2. Выравнивание	124
7.4.3. Межстрочное расстояние	124
7.4.4. Межсловное расстояние	125
7.4.5. Межбуквенное расстояние	125
7.4.6. Декорация	126
7.4.7. Индексы	127
7.4.8. Выделение первой буквы строки и первой строки в блоке текста	128
7.4.9. Объемный текст	129
7.4.10. Преобразование регистра	130
7.4.11. Мультиколоночная верстка	131
7.5. Предварительно отформатированный текст	131
7.6. Генерируемое содержимое	132
Глава 8. Списки	137
8.1. Маркированный список	137
8.2. Нумерованный список	139
8.3. Автоматическая нумерация элементов списка	140
8.4. Иерархический раскрывающийся список	142
8.5. Меню на основе списка	148
8.6. Выравнивание элементов списка	152
8.7. Список определений	154
Глава 9. Таблицы	156
9.1. Табличные теги	156
9.2. Рамки таблицы	158
9.3. Размеры таблицы	162
9.4. Выравнивание содержимого ячеек таблицы	167
9.5. Задание параметров столбцов	169
9.6. Сложные таблицы	172
9.6.1. Расширение ячеек	172
9.6.2. Прокручиваемая таблица	176
9.7. Декорирование таблицы	178
Глава 10. Элементы пользовательского интерфейса и формы	180
10.1. Поля ввода, кнопки и переключатели: тег <code><input></code>	181
10.2. Кнопка: тег <code><button></code>	183
10.3. Раскрывающийся список: тег <code><select></code>	186
10.4. Текстовая область: тег <code><textarea></code>	189
10.5. Декорации	191
10.6. Форма: тег <code><form></code>	193
Глава 11. Вставка внешнего содержимого в (X)HTML-документ	196
11.1. Графические изображения	196
11.1.1. Основные форматы графики	196
Растровая графика	196
Векторная графика	199

11.1.2. Вставка графики в (X)HTML-документ.....	201
Применение тега <code></code>	201
Применение тегов <code><iframe></code> и <code><object></code> для вставки изображений.....	203
Пример простой фотогалереи.....	205
11.2. Звук и видео.....	207
11.2.1. Основные форматы звуковых и видеофайлов.....	207
11.2.2. Вставка звука и видео в (X)HTML-документ.....	208
Теги <code><object></code> и <code><embed></code>	208
Теги <code><audio></code> и <code><video></code>	210
11.2.3. Вставка FLV-видео в (X)HTML-документ.....	212
11.2.4. Вставка Flash-фильмов.....	214
11.3. Вставка (X)HTML-документов.....	218
11.4. Вставка элементов управления ActiveX.....	220
11.4.1. Что такое ActiveX.....	220
11.4.2. Примеры элементов ActiveX.....	221
Flash-проигрыватель.....	221
Adobe SVGViewer.....	222
Календарь.....	223
Обслуживание табличных данных в текстовых файлах.....	223
11.4.3. О безопасности ActiveX.....	228
11.5. Вставка апплетов Java.....	230
11.5.1. Что такое апплет.....	230
11.5.2. Вставка апплета посредством тега <code><applet></code>	231
11.5.3. Вставка апплета посредством тега <code><object></code>	232
Глава 12. Фреймы.....	235
12.1. Установочный файл.....	236
12.2. Теги <code><frameset></code> , <code><frame></code> и <code><noframes></code>	237
12.3. Раскладки фреймов.....	239
12.4. Декорации фреймов.....	241
12.5. Применение CSS к фреймам.....	244
12.6. Пример макета фреймовой страницы.....	246
Глава 13. Применение SVG.....	249
13.1. Что такое SVG.....	249
13.2. Создание простых фигур.....	254
13.2.1. Прямоугольник.....	255
13.2.2. Круг.....	255
13.2.3. Эллипс.....	255
13.2.4. Многоугольник.....	256
13.2.5. Линии.....	257
13.3. Создание сложных фигур (тег <code><path></code>).....	259
13.4. Вставка растровых изображений.....	263
13.5. Применение CSS.....	264
13.6. Группировка элементов.....	266
13.7. Третье измерение, определения и клонирование элементов.....	267
13.8. Градиентная заливка.....	269
13.8.1. Линейный градиент.....	269
13.8.2. Радиальный градиент.....	271

13.9. Маски	275
13.9.1. Тег <code><mask></code>	275
13.9.2. Тег <code><clipPath></code>	278
13.10. Тексты	278
13.11. Трансформации	287
13.11.1. Перенос	289
13.11.2. Поворот	289
13.11.3. Наклон	291
13.11.4. Масштабирование	295
13.11.5. Отражение	295
13.11.6. Трансформация посредством матрицы	297
13.12. Анимация	298
13.12.1. Тег <code><animate></code>	300
13.12.2. Тег <code><animateTransform></code>	306
13.12.3. Тег <code><animateMotion></code>	308
13.12.4. Тег <code><animateColor></code>	312
13.12.5. Тег <code><set></code>	313
13.13. Интерактивность	313
13.13.1. Гиперссылки	314
13.13.2. Обработка событий	315
Применение тегов и атрибутов SVG	315
Применение JavaScript	319
13.14. Вставка в SVG-документ XHTML-кода	328
Глава 14. Графические фильтры CSS в Internet Explorer	332
14.1. Статические фильтры	333
14.1.1. Тень (<code>dropShadow</code> , <code>shadow</code>)	334
14.1.2. Волновое искажение (<code>wave</code>)	335
14.1.3. Размытие (<code>blur</code>)	335
14.1.4. Прозрачность (<code>alpha</code>)	335
14.1.5. Повороты	336
14.2. Динамические фильтры	337
14.2.1. Трансформация (<code>revealtrans</code>)	337
14.2.2. Освещение (<code>light</code>)	340
14.3. Применение нескольких фильтров одновременно	343
ЧАСТЬ II. СКРИПТЫ НА JAVASCRIPT	345
Глава 15. Что такое JavaScript	347
15.1. Немного истории	347
15.2. Общая характеристика языка	349
15.3. Вставка сценариев в (X)HTML-документ	350
Глава 16. Основы JavaScript	356
16.1. Ввод и вывод данных	356
16.1.1. Метод <code>alert()</code>	357
16.1.2. Метод <code>confirm()</code>	357

16.1.3. Метод <i>prompt()</i>	358
16.1.4. Метод <i>document.write()</i>	359
16.2. Типы данных	360
16.2.1. Примитивные типы данных	361
16.2.2. Составные типы данных	362
16.2.3. Автоматическое преобразование типов данных	365
Преобразование строк (<i>String</i>)	366
Преобразование чисел (<i>Number</i>)	366
Преобразование логических значений (<i>Boolean</i>)	366
Преобразование пустого значения (<i>null</i>)	367
Преобразование неопределенного значения (<i>undefined</i>)	367
16.2.4. Принудительное преобразование типов данных	367
16.3. Переменные и оператор присваивания	370
16.3.1. Имена переменных	370
16.3.2. Создание переменных	372
16.3.3. Операторы присваивания	378
16.3.4. Проверка типа переменной	379
16.4. Операторы	379
16.4.1. Комментарии	380
16.4.2. Арифметические операторы	381
16.4.3. Дополнительные операторы присваивания	383
16.4.4. Операторы сравнения	384
16.4.5. Логические операторы	386
16.4.6. Операторы условия	388
Оператор <i>if</i>	388
Оператор условия <i>?:</i>	390
Оператор <i>switch</i>	390
16.4.7. Операторы цикла	392
Оператор <i>for</i>	392
Оператор <i>while</i>	395
Оператор <i>do-while</i>	396
16.4.8. Об условиях в операторах условия и цикла	397
16.4.9. Побитовые операторы	397
16.4.10. Другие операторы	399
16.4.11. Приоритет операторов	399
16.5. Функции	401
16.5.1. Встроенные функции	401
16.5.2. Пользовательские функции	404
16.5.3. Объект <i>Function</i>	407
16.6. Строки	411
16.6.1. Кавычки и специальные символы	411
16.6.2. Объект <i>String</i>	413
16.6.3. Функции вставки и замены подстроки	419
16.6.4. Функции удаления ведущих и заключительных пробелов	421
16.7. Массивы	423
16.7.1. Создание массива	423
16.7.2. Многомерные массивы	425
16.7.3. Копирование массива	427

16.7.4. Объект <i>Array</i>	428
16.7.5. Функции обработки числовых массивов	433
16.8. Числа	434
16.8.1. Числа целые и с плавающей точкой.....	434
16.8.2. Объект <i>Number</i>	437
16.8.3. Объект <i>Math</i>	440
16.9. Дата и время	441
16.9.1. Создание объекта <i>Date</i>	442
16.9.2. Методы объекта <i>Date</i>	443
16.10. Объекты	449
16.10.1. Создание объекта	449
16.10.2. Свойства и методы объекта <i>Object</i>	453
16.10.3. Объектные операторы	454
16.10.4. JSON.....	457
16.11. Операторы обработки исключительных ситуаций.....	460
Глава 17. Объектная модель браузера и документа	463
17.1. Общие сведения	463
17.2. Доступ к объектам	467
17.3. Доступ к свойствам элементов документа.....	471
17.3.1. Доступ к атрибутам.....	471
17.3.2. Доступ к свойствам CSS.....	473
Параметры CSS, определенные в атрибуте <i>style</i>	473
Параметры CSS, определенные в теге <i><style></i> или во внешнем файле.....	474
17.3.3. Доступ к содержимому элемента.....	478
17.4. Обработка событий.....	480
17.4.1. Привязка обработчиков событий.....	481
Атрибуты-события.....	482
Регистрация обработчика события.....	482
Средства DOM	486
17.4.2. Область видимости обработчиков событий.....	487
17.4.3. Изменение поведения элементов по умолчанию	488
17.4.4. Программный вызов обработчика события.....	489
17.4.5. Прохождение событий.....	492
17.4.6. Информация о событии: объект <i>Event</i>	496
Доступ к <i>Event</i>	496
Кто является целевым объектом?	498
Какое событие произошло?.....	499
Основные свойства объекта <i>Event</i>	500
17.4.7. Основные события	502
17.5. Основные объекты браузера и документа	505
17.5.1. Объект <i>window</i>	506
Свойства объекта <i>window</i>	506
Методы объекта <i>window</i>	507
17.5.2. Объект <i>screen</i>	508
17.5.3. Объект <i>location</i>	508
Свойства объекта <i>location</i>	508
Методы объекта <i>location</i>	509

17.5.4. Объект <i>history</i>	509
Свойства объекта <i>history</i>	510
Методы объекта <i>history</i>	510
17.5.5. Объект <i>navigator</i>	510
Свойства объекта <i>navigator</i>	510
Коллекции объекта <i>navigator</i>	511
Методы объекта <i>navigator</i>	512
17.5.6. Объект <i>document</i>	512
Свойства объекта <i>document</i>	512
Коллекции объекта <i>document</i>	513
Методы объекта <i>document</i>	513
Глава 18. Работа с основными объектами	514
18.1. Управление окнами и фреймами	514
18.1.1. Создание окон	514
18.1.2. Взаимодействие окон.....	516
18.1.3. Работа с фреймами.....	518
Взаимодействие фреймов.....	518
Предотвращение загрузки в чужой фрейм.....	522
Проверка загрузки всех фреймов	522
Работа с плавающими фреймами	523
18.1.4. Окно PopUp в Internet Explorer	524
18.2. Работа с таблицами	525
18.3. Работа с табличными данными в текстовых файлах	529
18.3.1. Применение ActiveX Tabular Datae Control	529
Перемещение по строкам таблицы.....	530
Сортировка данных таблицы	533
Фильтрация данных таблицы.....	534
18.3.2. Применение объекта <i>XMLHttpRequest</i>	536
18.4. Работа с формами	537
18.4.1. Проверка данных перед отправкой	537
18.4.2. Баннер как форма.....	539
18.4.3. Переходы между полями по клавише <Enter>	541
18.5. Работа с cookie	542
18.6. Работа с графическими изображениями	548
18.6.1. Объект элемента 	548
18.6.2. Объект <i>Image</i>	549
18.6.3. Управление свойствами изображения.....	550
18.6.4. Предварительная загрузка изображений.....	552
18.6.5. Апокрифические применения объекта <i>Image</i>	555
Передача данных на сервер.....	555
Парольная защита страницы на стороне клиента.....	557
18.7. Взаимодействие с сервером: объект <i>XMLHttpRequest</i> и AJAX.....	560
18.7.1. Объект <i>XMLHttpRequest</i>	560
Свойства объекта <i>XMLHttpRequest</i>	561
Передача данных.....	562
18.7.2. AJAX	570
18.8. Управление во времени	571

Глава 19. Примеры клиентских сценариев	575
19.1. Подсветка кнопки	575
19.2. Меню	578
19.2.1. Моментально раскрывающееся вертикальное меню	578
19.2.2. Плавно раскрывающееся меню	579
19.3. Раскрывающийся комбинированный список	582
19.4. Иерархический раскрывающийся список	583
19.5. Эффект пишущей машинки	585
19.6. Отображение кода на странице	586
19.7. Перемещение элементов мышью	589
19.8. Движение по траектории	593
19.8.1. Движение по произвольной кривой	593
19.8.2. Движение по эллипсу	595
19.9. Рисование линий посредством <code><div></code>	597
19.9.1. Прямая линия	598
19.9.2. Произвольная линия	601
19.9.3. Графики зависимостей	605
19.9.4. Перерисовка линий	607
19.10. Рисование посредством <code><canvas></code>	608
19.10.1. Как вставить <code><canvas></code> в (X)HTML-документ	608
19.10.2. Фигуры и линии	610
Прямоугольник	611
Путь	612
Линии	613
Панель с закругленными углами	615
19.10.3. Градиенты	616
19.10.4. Трансформации	617
19.10.5. Импорт растровых графических изображений	619
19.10.6. Анимация	622
19.10.7. Композиция графики	624
19.10.8. Текст	627
19.11. Дата и время	628
19.11.1. Отображение даты и времени в виде текста	629
19.11.2. Часы	630

ЧАСТЬ III. СКРИПТЫ НА PHP

633

Глава 20. Что такое серверные сценарии и PHP	635
20.1. Общая характеристика языка PHP	636
20.2. Как установить модуль PHP	636
20.3. Настройка Web-сервера	639
20.4. Проверка работоспособности Web-сервера с PHP	639
20.5. Проба пера	640
20.6. Включаемые файлы	641
20.7. Сообщения об ошибках	642
20.8. Принудительный выход из сценария	642
20.9. Справочная информация по PHP	643

Глава 21. Основы PHP	644
21.1. Вывод данных	644
21.2. Типы данных	645
21.3. Переменные и оператор присваивания	648
21.3.1. Имена переменных	648
21.3.2. Создание переменных	648
21.3.3. Отображение значений переменных	650
21.3.4. Переменные переменные	652
21.3.5. Область действия переменных	653
21.3.6. Проверка существования переменных и их типов	654
21.4. Константы	655
21.5. Операторы	656
21.5.1. Комментарии	656
21.5.2. Арифметические операторы	657
21.5.3. Строковый оператор	658
21.5.4. Дополнительные операторы присваивания	658
21.5.5. Операторы сравнения	659
21.5.6. Логические операторы	660
21.5.7. Побитовые операторы	661
21.5.8. Операторы условного перехода	662
Оператор <i>if</i>	662
Оператор <i>switch</i>	663
Оператор условия <i>?:</i>	663
21.5.9. Операторы цикла	664
21.6. Строки	664
21.6.1. Двойные и одинарные кавычки	664
21.6.2. Склейка строк	666
21.6.3. Преобразование строк	667
21.6.4. Форматирование строк	671
21.7. Числа	674
21.7.1. Математические функции	675
21.7.2. Математические константы	676
21.7.3. Представление чисел в различных системах счисления	676
21.7.4. Форматирование чисел	678
21.8. Дата и время	679
21.9. Массивы	682
21.9.1. Создание массива	682
21.9.2. Многомерные массивы	685
21.9.3. Отображение массивов	686
21.9.4. Операции над массивами	687
Копирование массивов	687
Сортировка массивов	687
Перемещение по массиву	689
Запись значений элементов массива в переменные	691
Преобразование массива в текстовую строку	692
Преобразование текстовой строки в массив	692
Другие операции над массивами	693
21.10. Глобальные предопределенные переменные	695

21.11. Функции	697
21.11.1. Пользовательские функции	697
21.11.2. Переменные функции	701
21.11.3. Встроенные функции	702
21.11.4. Как узнать, есть ли такая функция?	702
21.12. Классы и объекты	702
21.12.1. Определение класса	703
Свойства и методы	704
Конструктор	705
21.12.2. Применение объектов	707
21.12.3. Ограничение доступа к свойствам и методам	708
21.12.4. Клонирование и удаление объектов	709
21.12.5. Использование методов несозданных объектов	710
21.12.6. Обработка исключений	710
21.12.7. Пример класса формы	712
21.13. Выполнение PHP-кода в текстовых строках	713
Глава 22. Примеры серверных сценариев	715
22.1. Получение данных из (X)HTML-форм клиента	715
22.1.1. Получение данных из HTML-форм	715
22.1.2. Передача файлов на сервер	723
22.2. Переходы и передача данных между Web-страницами	725
22.2.1. Вывод ссылок	726
22.2.2. Применение форм	726
22.2.3. Применение функции <i>header()</i> для переадресации	726
22.2.4. Добавление информации к URL-адресу	728
22.2.5. Применение cookie	729
22.2.6. Применение сеансов	730
Создание сеанса	730
Особенности сеансов	731
Пример организации сеанса	732
Защита страниц паролем	734
22.3. Работа с графикой	737
22.3.1. Создание и отправка изображения браузеру	737
22.3.2. Масштабирование изображения	738
22.3.3. Поворот изображения	739
22.3.4. Композиция нескольких изображений	739
22.3.5. Вставка текста в изображение	740
22.3.6. Рисование линий	742
22.4. Работа с файлами	743
22.4.1. Открытие файла	743
22.4.2. Закрытие и удаление файлов	744
22.4.3. Чтение файла	745
Чтение файла в переменную	745
Чтение файла в массив	746
Чтение файла с удалением тегов HTML	746
22.4.4. Запись в файл	747
22.4.5. Работа с папками	747

22.4.6. Простой счетчик посещений страницы.....	749
22.4.7. Работа с CSV-файлами	750
Чтение CSV-файла	750
Функции работы с табличными данными.....	752
Сложный счетчик посещений страницы.....	756
Распространяемый счетчик посещений	758
Баннер	760
Гостевая книга.....	763
22.5. Работа с базами данных.....	769
22.5.1. Общие сведения о базах данных.....	769
22.5.2. Установка СУБД.....	771
22.5.3. Основные средства PHP для взаимодействия с базой данных.....	772
Подключение к базе данных	772
Передача запросов к базе данных.....	773
Обработка данных в сценарии	775
22.5.4. Создание гостевой книги.....	775
Создание базы данных.....	776
Сценарии для взаимодействия с посетителем	777
Сценарии для владельца гостевой книги	780
22.6. Другие возможности PHP	781

ПРИЛОЖЕНИЯ..... 783

Приложение 1. Перечень тегов HTML 5..... 785

Приложение 2. Перечень параметров CSS..... 791

Позиционирование.....	791
Размеры.....	791
Цвет и фон	792
Текст.....	792
Шрифты	792
Блоки (поля, отступы и границы)	792
Таблицы	793
Печать	793
Интерфейс.....	793
Звук	793
Прочее.....	794

Литература..... 795

Предметный указатель..... 796

Предисловие к третьему изданию

Третье издание настоящей книги кардинально отличается от предыдущих как по структуре, так и по содержанию. Существенно расширены главы, посвященные языкам разметки гипертекста HTML 4/5, XHTML и каскадным таблицам стилей CSS 2/3, которые теперь рассматриваются параллельно. Добавлен материал по масштабируемой векторной графике (SVG) и динамической графике, управляемой скриптами (элемент `<canvas>`), а главы по ASP и языку VBScript в данное издание не вошли.

Предыдущие издания книги в основном были ориентированы на господствовавший на рынке браузер Microsoft Internet Explorer версий от 5.0 до 6.0. Тогда коды Web-страниц создавали в основном под Internet Explorer, а их адаптацию к другим браузерам выполняли лишь для охвата весьма малочисленной (2—5%) аудитории, что было весьма хлопотно. Теперь программные коды пишут в соответствии со стандартами, а хлопоты пока доставляет Internet Explorer версий 6, 7 и 8, а не другие современные браузеры.

В данном издании, в отличие от предыдущих, большое внимание уделено соблюдению стандартов и межбраузерной совместимости кодов. По крайней мере, приведенные в книге примеры проверены на работоспособность во всех современных ведущих браузерах, таких как Microsoft Internet Explorer 8.0, Mozilla Firefox 3.6, Opera 11.0, Apple Safari 5.0 и Google Chrome 8.0.

Я старался раскрыть возможности (X)HTML, CSS, JavaScript и PHP для разработки Web-сайтов, не претендуя на полноту изложения. Это не справочник, а скорее расширенное практическое руководство по самостоятельному изучению разнообразных средств и приемов Web-программирования, доступное даже начинающим.

Введение

Как известно, Web-сайты создают с помощью языка разметки гипертекста (X)HTML, каскадных таблиц стилей (CSS) и скриптов (сценариев на языках JavaScript, PHP и др.). (X)HTML применяется для создания (разметки) структуры документа или Web-страницы, таблицы стилей используются для внешнего оформления (стилизации, декорирования) и позиционирования элементов документа, а скрипты — для обеспечения интерактивности и динамичности Web-страниц, а также обработки данных, которыми обмениваются браузер и Web-сервер. Всем этим средствам и посвящена данная книга. Для освоения изложенного материала никаких предварительных знаний Web-программирования не требуется. Наряду с краткими теоретическими сведениями читатель найдет здесь многочисленные примеры решения типичных задач разработки сайтов.

Данная книга не является справочником по (X)HTML, CSS и скриптовым языкам. Ее задача состоит в том, чтобы ввести начинающего разработчика в предметную область создания Web-сайтов, показать некоторые приемы обращения с теми или иными технологическими средствами. Каждая из тем ((X)HTML, CSS и языки сценариев) в действительности настолько обширна, что ее полному изложению можно посвятить отдельную книгу примерно такого же объема, что и данная. Поэтому мне потребовалось тщательно отбирать материал, необходимый на первых порах при решении задач разработки домашних страниц и сайтов. Разумеется, многие интересные вопросы пришлось оставить без ответа. Недостающие сведения можно получить из соответствующих справочников, которые нетрудно найти в Интернете. Вместе с тем хочу заметить, что я не навязываю читателю каких-либо технических приемов в качестве "правильных" и единственно возможных для достижения успеха. Практически любую цель можно достичь несколькими путями. И начинающие, и опытные разработчики должны самостоятельно выбирать средства, наиболее подходящие для решения стоящей перед ними задачи. В этом и состоит, на мой взгляд, главное в индивидуальном творчестве.

В первой части книги (*главы 1—14*) рассматриваются средства (X)HTML и CSS на примерах решения основных задач создания сайтов. При этом я старался не использовать тех тегов и атрибутов, главное назначение которых состоит в задании внешнего вида элементов, предпочитая им инструменты CSS. В *главе 11* рассматривается вставка в (X)HTML-документ содержимого из внешних файлов: графики, Flash-анимации, видео и аудио. *Глава 13* посвящена языку SVG для создания

масштабируемой векторной графики. Хотя SVG — особый язык, он устроен подобно языкам разметки гипертекста и описание на нем графики легко может быть встроено в (X)HTML-документ.

Вторая часть (*главы 15—19*) посвящена языку JavaScript программирования клиентских сценариев. Здесь, помимо прочего, рассматриваются AJAX и работа с элементом `<canvas>` для создания графики, управляемой скриптами.

В третьей части (*главы 20—22*) рассматриваются основы языка PHP программирования серверных сценариев.

В *приложениях 1 и 2* приведены перечни тегов HTML 5 и свойств CSS соответственно.

Все приведенные в книге примеры, за исключением некоторых особо оговоренных случаев, работают во всех современных браузерах.

Ваши отзывы и замечания я буду рад принять в гостевой книге на своем сайте по адресу <http://dunaevv1.narod.ru>. По этому же адресу можно найти множество примеров, в том числе и рассмотренных в данной книге.

Благодарности

Я благодарен многочисленным читателям, приславшим свои замечания и отзывы, которые я попытался учесть при подготовке данного издания.

Особую благодарность я выражаю своей жене Валентине за поддержку в нелегкий для меня период работы над этой книгой.



ЧАСТЬ I

HTML и стили



Глава 1

Что такое HTML и таблицы стилей CSS

В данной главе мы рассмотрим язык разметки и таблицы стилей. Здесь важно, не вникая в тонкости синтаксиса, понять их назначение, взаимосвязь и основные направления применения. Далее мы все проясним и поставим на свои места.

1.1. Языки разметки документа

Допустим, вам требуется создать документ (письмо другу, резюме для работодателя, объявление о продаже какого-то имущества, прайс-лист, литературное произведение и т. п.), который вы намерены опубликовать — распечатать на принтере, а затем отправить в издательство или разместить на Web-сайте. Ваш документ будет содержать обычный текст и, возможно, картинки. Это — максимум из того, что вы можете позволить себе при публикации документа на бумажных носителях. Разумеется, все то же самое можно опубликовать и в Интернете, стремясь максимально расширить аудиторию читателей. Однако даже в этом простейшем случае нередко возникают порой нетривиальные задачи форматирования или, другими словами, оформления документа. Текст должен быть структурирован, т. е. разбит на части (разделы, подразделы, абзацы), иногда снабженные заголовками. Даже обычное письмо объемом всего в одну страницу, как правило, разбивают на несколько абзацев, не говоря уже о статьях и более крупных произведениях, таких как повести и романы. Заголовки и абзацы выделяются, по крайней мере, своим местоположением относительно друг друга. Кроме того, заголовки разделов и даже некоторые словосочетания обычно выделяют из общего текстового потока шрифтом — гарнитурой, размером, цветом и другими характеристиками. Если в тексте предусмотрены графические иллюстрации, то задача форматирования документа еще более усложняется: необходимо определить, каким образом текст будет "обтекать" вставленные картинки.

При публикации документа в Интернете вы можете сделать значительно больше, чем при печати: вставить в документ звуковые и видеоклипы, формы, заполняемые читателем и отправляемые на сервер, элементы интерфейса (ссылки на другие документы, кнопки, поля ввода, переключатели и т. п.), с помощью которых можно управлять отображением содержимого и решать другие задачи.

Даже в обычных бумажных документах нередко встречаются ссылки на разделы этого же или другого документа (например, "см. разд. 2.5", "в книге [7] можно найти...", "в приложении 1 приведен список ..." и т. п.). Встретив такую ссылку в книге,

читатель при желании должен либо перелистывать ее страницы, либо искать другую книгу, чтобы затем в поисках нужной информации "порыться" в ней. В Web-документах, просматриваемых в браузерах, щелчок левой кнопкой мыши на ссылке приводит к автоматическому поиску в Сети и отображению в окне браузера соответствующего (релевантного) документа. Такие ссылки в связи с их мощной и удобной функциональностью называют гиперссылками, а тексты, их содержащие, — гипертекстами (HyperText). Гиперссылки могут указывать на документы различного характера — обычные и гипертексты, графические изображения, звуковые и видеоклипы и др. Активизация гиперссылки означает запрос к Web-серверу указанного в ней документа и, если он найден в Сети, пересылку и отображение его в Web-браузере. Благодаря гиперссылкам множество различных документов можно связать в некую единую систему, образовав сайт или, другими словами, Web-узел. Так, вы можете создать, по крайней мере, один документ, содержащий гиперссылки на другие, чтобы не только представить самого себя в Интернете, но и послужить проводником к другим ресурсам Всемирной сети. Простая в своей основе идея и адекватный ей инструмент гиперссылки позволил создать и далее развивать технологии, лежащие в фундаменте Всемирной паутины (World Wide Web), преобразившей в последнее время все информационное пространство и коммуникации в нем.

Итак, при подготовке любого документа к публикации его требуется отформатировать или, иначе говоря, разметить — с помощью специальных символов явно указать, что и каким образом следует представить пользователю (читателю) этого документа. Набор специальных символов (меток, дескрипторов, команд или тегов), а также правила их употребления составляют то, что называют языком разметки (Markup Language). В настоящее время существуют несколько таких языков, среди которых наиболее популярны HTML (HyperText Markup Language — язык разметки гипертекста), XHTML (eXtensible HyperText Markup Language — расширяемый язык разметки гипертекста) и XML (eXtensible Markup Language — расширяемый язык разметки). HTML и XHTML применяются для разметки Web-документов, а XML — как средство структуризации информации для обмена между компьютерными программами. Между этими языками очень много общего, хотя имеются и различия, о которых будет рассказано позже.

Разметка документа, как уже упоминалось, осуществляется посредством специальных дескрипторов, называемых тегами (tag — метка, признак, ярлык). В языках разметки в большинстве случаев они имеют вид: `<имя_тега>` и `</имя_тега>`. Например, `<p>` и `</p>`, где `p` — имя тега. Эти две формы одного и того же тега играют роль подобно открывающей и закрывающей скобкам при написании математических формул. Между ними можно разместить некоторое содержимое — обычный текст и/или тег, содержащий, возможно, другие теги. Тег, имеющий открывающую и замыкающую части, называют еще контейнерным. Вот его синтаксис:

`< имя_тега > содержимое контейнерного тега </ имя_тега >`

Существуют и неконтейнерные теги (без замыкающей части), например `` — для вставки графического изображения и `
` — для указания перехода на новую строку.

Теги не отображаются в окне браузера, а лишь указывают ему, какие элементы входят в документ и как они между собой соотносятся. Так, текстовый документ состоит из абзацев, которые могут объединяться в разделы. Разделы, в свою очередь, могут входить в более крупные структурные единицы, например главы или части и т. д. Язык HTML обеспечивает поддержку иерархической декомпозиции документа довольно простыми средствами. Например, чтобы указать, что данный фрагмент текста относится к одному и тому же абзацу, достаточно просто перед этим фрагментом написать тег `<p>`, а после него — `</p>`. Другими словами, текст абзаца следует заключить в контейнер `<p>`, например,

```
<p>Здравствуйте, дорогой Иван Иванович.</p>
```

Чтобы несколько элементов документа (например, абзацев) объединить в один раздел, их следует заключить в какой-нибудь подходящий для этой цели контейнер, например `<div>` (листинг 1.1).

Листинг 1.1. Пример использования контейнера `<div>`

```
<div>
  <p>
    Здравствуйте, дорогой Иван Иванович.
  </p>
  <p>
    В первых строках своего письма передаю поклон Марье Ивановне.
  </p>
</div>
```

Здесь контейнер `<div>` содержит два контейнера `<p>`, каждый из которых заключает в себя некий текст.

Самый крупный контейнер для содержательной части документа — тег `<body>` (тело документа), он содержит другие контейнеры. Например, формируя письмо, мы могли бы написать фрагмент кода на языке разметки, приведенный в листинге 1.2.

Листинг 1.2. Фрагмент кода на языке разметки

```
<body>
  <div>
    <p>
      Здравствуйте, дорогой Иван Иванович.
    </p>
    <p>
      В первых строках своего письма передаю поклон Марье Ивановне.
    </p>
  </div>
</body>
```

Итак, мы уже упомянули несколько тегов: `<p>`, `
`, `<body>`. В HTML имеется много других тегов для группировки элементов документа в логические блоки, образующие в совокупности структуру документа.

На рис. 1.1 в окне обычного текстового редактора показан пример текста, размеченного тегами абзаца `<p>` и перевода строки `
`, а также его вид в окне браузера. Хотя данный документ в целом оформлен не по всем правилам и даже совсем не по правилам HTML, основные Web-браузеры все-таки отображают его, причем одинаково. Дело в том, что файл `text.htm`, содержащий теги HTML, был открыт в Web-браузере. Все современные браузеры при открытии в них файлов с расширениями `htm` или `html` пытаются интерпретировать последние как HTML-коды, т. е. как поток тегов языка разметки. Однако это еще не означает, что формировать Web-документ можно "как попало", вспоминая и используя те или иные теги. Правила создания Web-документов существуют, и соблюдать их настоятельно рекомендуется во избежание несовместимости с различными браузерами и других проблем.

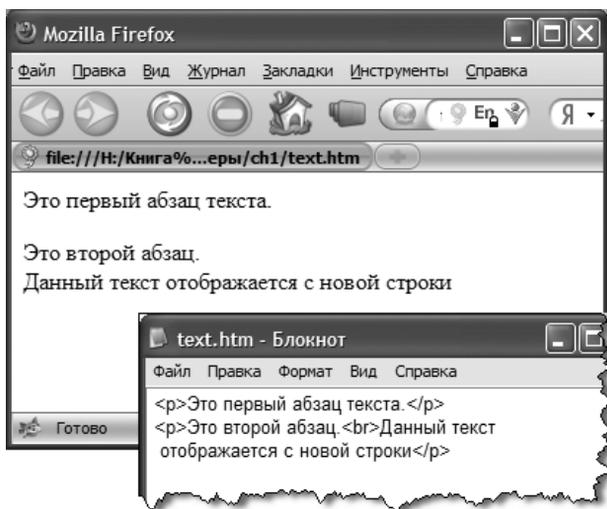


Рис. 1.1. Текст, размеченный тегами абзаца `<p>` и перевода строки `
`, в текстовом редакторе и в браузере

Обратите внимание, что в рассматриваемом примере расположение текста в окнах редактора и браузера различается. В текстовом редакторе символы могут располагаться в одной или нескольких строках, что еще не определяет однозначно их расстановку в окне браузера. При наборе символов в текстовом редакторе нажатие на клавиши `<Enter>`, `<Tab>` и т. п. приводит к форматированию текста, визуально контролируемому с помощью дисплея. Текстовый редактор отвечает на нажатие указанных клавиш расстановкой в вводимом тексте специальных управляющих символов, которые не видны в общем потоке печатаемого текста. Для управления расположением текста в окне браузера требуется явное задание управляющих символов в виде специальных тегов языка разметки. В рассматриваемом примере тег абзаца `<p>` указывает браузеру отобразить заключенный в него текст с новой строки

и с отступом в одну строку. В обычном текстовом редакторе этого же эффекта можно добиться, дважды нажав на клавишу <Enter> в конце введенного текста. Далее, переход на новую строку внутри второго абзаца задан с помощью тега
. Без него этот переход либо не произошел бы, либо мог произойти лишь при достаточно малой ширине окна браузера. Результат зависит от используемого браузера. Дело в том, что браузеры пытаются, так или иначе, перенести поток символов на новую строку, если ширина окна не позволяет расположить текст вдоль одной горизонтальной линии.

Чтобы отображение текста и других элементов документа было инвариантно относительно различных браузеров, необходимо принудительно указывать переносы, отступы и другие управляющие символы с помощью тегов языка разметки и специальных средств (атрибутов тегов и параметров CSS, о чем будет рассказано далее).

О правилах оформления HTML-документов мы поговорим позже, а сейчас остановимся на одной важной особенности HTML. Не все, но многие теги определяют не только структуру документа, но и внешний вид его элементов в окне браузера, т. е. задают еще и способ отображения соответствующей информации в браузере. Например, тег абзаца <p> определяет, что все следующее за ним будет отображено с новой строки и с пропуском одной строки; текст, заключенный между тегами <h1> и </h1>, будет выведен с новой строки максимальным по размеру шрифтом, а последующая информация будет размещена с новой строки (иначе говоря, контейнер <h1> задает заголовок первого уровня); текст в контейнере <i> выводится курсивом.

Таким образом, разметка тегами HTML в большинстве случаев предопределяет не только структурную декомпозицию документа, но и его внешний вид в окне браузера. Разумеется, структурная декомпозиция, в конечном счете, должна быть как-то выражена внешним образом. Например, листовую книгу, мы видим ее структуру в виде иерархии глав, разделов, подразделов и абзацев. Вместе с тем нетрудно понять различие между собственно структурной декомпозицией и внешним представлением ее элементов. В самом деле, главы, разделы и подразделы можно отобразить как линейно следующие друг за другом, так и расположенные на страницах в двух и более колонках; заголовкам можно придать тот или иной вид, манипулируя такими параметрами шрифта, как цвет, размер и начертание. При этом логическая структура документа остается постоянной, а изменяется только ее представление на бумаге или в окне браузера. Однако идея четко разделить структурный и представительский аспекты языка разметки не реализована в полной мере в HTML. В HTML средства структурной декомпозиции документа и его внешнего представления оказались изначально сильно связанными между собой. Именно это обстоятельство и сыграло, на мой взгляд, решающую роль в чрезвычайно широкой популярности данного языка разметки: разработчики Web-страниц хотели сразу видеть результат разметки документа в браузере и получили это (см. рис. 1.1).

Для определения элементов, вставляемых в документ, одних только тегов бывает недостаточно. Так, для вставки графического изображения служит тег . Однако требуется еще указать источник изображения, т. е. адрес или, точнее, URL (Uniform Resource Locator — унифицированный указатель ресурса) файла. В подобных случаях используются параметры тегов, называемые еще атрибутами. Один или несколько атрибутов записывают в открывающей части тега в произвольном

порядке в следующем виде: *имя_атрибута="значение"*. Атрибуты отделяют друг от друга пробелами. Вот пример тега для вставки графического изображения:

```

```

Тег `` не имеет заключительной части вида ``.

Чтобы указать размеры графического изображения на странице (в пикселах), можно записать следующее выражение:

```

```

Здесь `src`, `width` и `height` — имена атрибутов тега ``, за которыми следует знак равенства и, далее, значения, указанные в кавычках, двойных или одинарных.

Атрибуты позволяют указать, например, цвет фона, размеры, начертание и цвет шрифта, характеристики выравнивания элементов документа и т. п. Так, в контейнерном теге `<body>`, задающем основную часть документа, с помощью атрибутов `text` и `bgcolor` можно определить цвет текста и фона:

```
<body text="#ff0000" bgcolor="#00ffee">
```

Таким образом, развитие HTML пошло в сторону добавления средств, определяющих внешнее представление документов. Стали появляться как новые теги, так и новые их атрибуты. На первых порах производители браузеров (Microsoft, Netscape, Sun Microsystems и др.) создавали свои варианты HTML, отличающиеся не только наборами тегов, но и способами их интерпретации. В сложившихся условиях один и тот же документ мог выглядеть по-разному в зависимости от браузера, что явно противоречило цели всемирного распространения информации. Очевидно, стандарты стали насущной необходимостью.

Разработкой стандарта для HTML занялась международная организация World Wide Web Consortium (W3C, Консорциум Всемирной паутины), в состав которой вошли крупнейшие производители программного обеспечения для работы в Интернете (в том числе Microsoft, Sun Microsystems, Netscape и др.). Сведения о стандартах (спецификациях) можно найти на официальном сайте W3C по адресу www.w3.org. Схема подготовки спецификаций консорциумом примерно такая: сначала выпускается проект или черновик (draft) спецификации, в результате обсуждения которого появляется ее рабочий вариант. Последний предлагается к обсуждению в течение некоторого времени, по окончании которого рабочий вариант может стать рекомендацией — официальным вариантом спецификации. Браузеры должны интерпретировать язык разметки соответственно стандарту, хотя на практике это происходит не в полной мере. Одно дело идея, другое — ее реализация.

В июле 1997 г. консорциум W3C выпустил проект спецификации HTML 4.0, который уже в декабре стал официальной рекомендацией для производителей Web-браузеров. С декабря 1999 г. по настоящее время существует версия 4.01. Тем не менее, различия в интерпретации основными браузерами некоторых тегов и атрибутов официальной спецификации HTML все же остались. Для обеспечения дополнительных возможностей визуального представления, программирования поведения, динамичности и интерактивности документов появились каскадные таблицы стилей (CSS — Cascading Style Sheets) и скрипты (сценарии), написанные на специальных языках, наиболее популярный из которых — JavaScript.

Параллельно с HTML развивался другой язык разметки — XML (eXtensible Markup Language — расширяемый язык разметки). Он был создан как средство структуризации информации для обмена между программами. Web-браузер — одна из них, но далеко не единственная. При этом на способ внешнего представления структурированных данных никаких ограничений не накладывалось. Главная задача, которая ставилась при разработке этого языка, состояла в обеспечении совместимости между различными системами обработки структурированных данных. Передающей стороне не важно, как будет отображен XML-документ получателем, требуется лишь, чтобы он однозначно разобрался в структуре принятого документа.

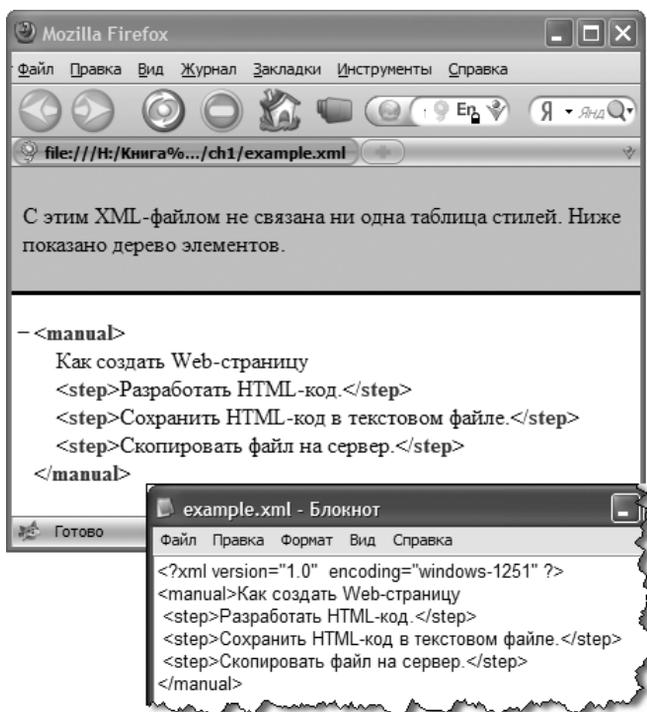


Рис. 1.2. Внешний вид XML-кода, загруженного в браузер

В XML, так же как и в HTML, есть теги и атрибуты. Однако, в отличие от HTML, в языке XML они не предопределены изначально, а могут создаваться автором документа по своему усмотрению. Теги XML задают лишь структуру документа, но не его представление. На рис. 1.2 показан фрагмент XML-кода в текстовом редакторе и результат его обработки Web-браузером. Теги `<manual>` и `<step>` были придуманы автором данной книги для представления структуры инструкции по созданию сайта. Web-браузер не "знает" таких тегов, а потому просто показывает XML-код без какой-либо его интерпретации. Внешний вид элементов XML-документа определяется отнюдь не тегам, а специальными программами-анализаторами, часто называемыми парсерами (parser). Основные Web-браузеры имеют такие встроенные парсеры, но чтобы подключить их к работе и заставить

отобразить собственно информацию (без тегов), необходимо указать, что содержимое документа является XML-кодом и, кроме того, достаточно сослаться на каскадную таблицу стилей (CSS).

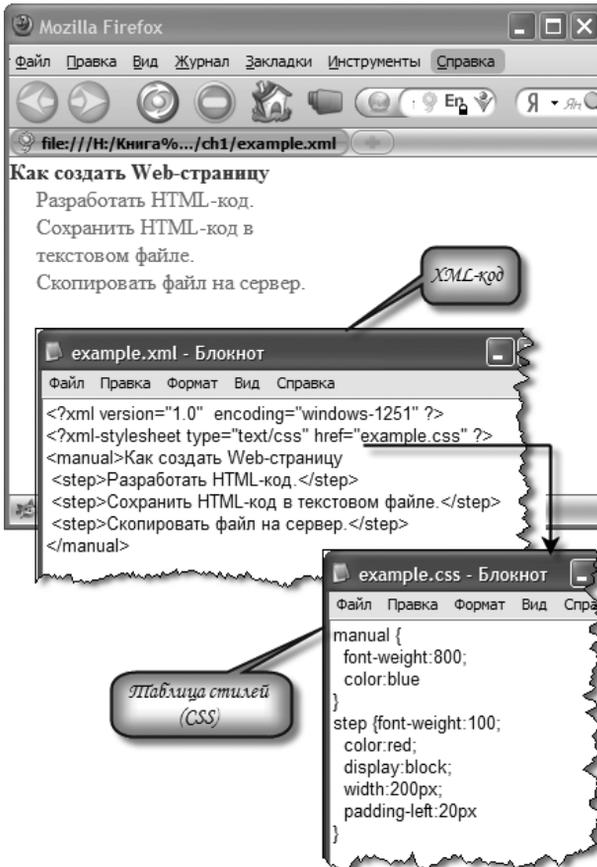


Рис. 1.3. Пример использования таблицы стилей для отображения XML-документа в окне браузера

Таблицы стилей сохраняются в отдельных файлах и содержат параметры визуализации элементов (тегов) XML-документа. На рис. 1.3 показан пример, в котором XML-документ, представляющий инструкцию, в первой строке содержит дескриптор, указывающий тип документа, а во второй строке — ссылку на файл `example.css` с правилами внешнего представления тегов `<manual>` и `<step>`. В данном случае браузер не отображает теги, а выводит результат их интерпретации. Так, в таблице стилей указаны цвет и "жирность" шрифта, а также требование, согласно которому каждый элемент `<step>` (шаг инструкции) занимает по горизонтали 200 пикселей (`width:200px`) с отступом 20 пикселей слева (`padding-left:20px`) и начинается с новой строки (`display:block`).

При желании можно модифицировать параметры отображения элементов документа, оставляя неизменной его логическую структуру.

Итак, в XML воплотилась хорошая идея разделения между структурным и внешним представлением содержания документа. В HTML 4 это не так, но его можно немного модернизировать, чтобы существенно приблизить к XML. На пути к достижению этой цели появился XHTML (Extensible Hypertext Markup Language — расширяемый язык разметки гипертекста).

ПРИМЕЧАНИЕ. XML — подмножество более сложного языка SGML (Standard Generalized Markup Language — стандартный обобщенный язык разметки). На основе XML созданы более специализированные языки разметки, например XHTML, SVG, RSS, MathML и др. Такие языки еще называют словарями данных языка XML. С другой точки зрения XML — текстовый формат описания иерархических баз данных.

Еще до появления XHTML многие разработчики Web-сайтов практиковали следующий технологический прием. С помощью HTML-тегов создавалась лишь структура документа, при этом теги и атрибуты, предназначенные только для задания внешнего вида элементов, практически не использовались (насколько это было возможно), а отображение элементов документа в окне браузера или при печати определялось только в каскадных таблицах стилей. Данный прием можно назвать "применением HTML в стиле XML".

Разметку Web-страницы можно реализовать на основе языка как HTML, так и XHTML. Однако XHTML более строг, ближе к универсальному языку XML и должен поддерживаться одинаковым образом всеми современными браузерами. Поэтому создавать Web-страницы сейчас рекомендуется с помощью языка XHTML. В данной книге все примеры написаны на языке XHTML. Однако во многих случаях я применял следующий шаблон: в приводимых примерах задан тип документа, указывающий на стандарт HTML 4.01, а собственно код написан в соответствии со стандартом XHTML. Хотя дескриптор `<!DOCTYPE ...>` типа документа указывает на его соответствие стандарту HTML 4.01, а не XHTML, ничего крамольного в этом нет. Объявив в дескрипторе `<!DOCTYPE ...>` свою приверженность HTML 4.01, вы можете писать свои коды разметки в соответствии с официальной спецификацией как HTML 4.01, так и XHTML.

Читатели, знакомые только с HTML 4, увидят не слишком много отличий этого языка от XHTML. Перейти от HTML-кодов к XHTML-кодам довольно легко. Языки HTML и XHTML очень схожи, но различия все же есть. Вот основные (но далеко не все) из них:

- язык HTML регистронезависимый: не имеет значения, строчными или прописными буквами записаны имена тегов и атрибутов. В XHTML для записи имен тегов и атрибутов рекомендуется указывать только строчные буквы. Например, вместо

```
<IMG SRC="mypicture.jpg">
```

следует писать

```
;
```

- в HTML большинство тегов контейнерные (например, `<h1>...</h1>`), т. е. имеют открывающий и закрывающий дескрипторы. Вместе с тем есть и неконтейнерные теги: для дескриптора `<тег>` нет закрывающего дескриптора `</тег>`. Например, для вставки графического изображения в HTML применяется тег ``, а в XHTML — ``. Неконтейнерные теги еще называют

пустыми. В XHTML для таких тегов необходимо указывать слэш (косую черту) перед закрывающей уголкового скобкой: `<ter ... />`, например

`` или `
`;

- в некоторых тегах HTML имеются атрибуты без значений (так называемые булевы или логические атрибуты), например `<option selected>`. В XHTML для подобных атрибутов следует явно указывать строковое значение, совпадающее с именем соответствующего атрибута, например `<option selected="selected" />`;
- в XHTML значения атрибутов тегов нужно заключать в кавычки. В HTML это делать не обязательно. Например, в XHTML следует писать ``, а не ``, как допустимо (хотя тоже не рекомендуется) в HTML. Некоторые атрибуты принимают числовые значения, которые в HTML допустимо (хотя и не рекомендуется) указывать без кавычек, но в XHTML их следует заключать в кавычки, двойные или одинарные, например ``;
- в содержательной текстовой части XHTML-документа нельзя непосредственно использовать символы, которые применяются в коде, такие как `<`, `>` и `&`, причем их нельзя записывать даже в значении URL-адреса. Эти специальные символы следует заменять соответствующими буквенными или числовыми эквивалентами `<`, `>` и `&` (или `<`, `>` и `&`;) соответственно. Об эквивалентах специальных символов рассказано в *разд. 7.3*;
- кодировка файла по умолчанию для HTML-документа — ISO 8859-1, а для XHTML-документа — UTF-8.

Ошибки в HTML-коде браузеры пытаются преодолеть тем или иным образом, так что даже весьма небрежно написанный код все же как-то отображается (во многих случаях вполне удовлетворительно). Более других терпим к ошибкам в HTML-коде и лоялен к разработчику браузер Microsoft Internet Explorer. По рекомендации W3C в случае ошибки в XHTML-коде браузеры должны сообщать об этом и прекращать дальнейшую обработку. Таким образом, XHTML-код подвергается более тщательному анализу прежде, чем начнется его интерпретация — отображение в окне браузера. HTML-код, напротив, не анализируется предварительно на правильность, а отображается в порядке следования тегов. При этом сообщения об ошибках не выводятся, а отображение продолжается, даже если ошибки все же возникли. Очевидно, HTML более дружелюбен разработчику, чем XHTML. Но более других эту "дружелюбность" ценят малоопытные и небрежные разработчики своих домашних страниц, а не авторы серьезных Web-проектов. Для последних однозначность интерпретации и дисциплина важнее "угодливости" браузеров, вредность которой проявляется при поиске трудноуловимых смысловых ошибок и при попытках обеспечить межбраузерную инвариантность представления документов.

В XHTML сохранилось большинство тегов HTML 4. Однако некоторые старые HTML-теги, введенные в свое время для обеспечения дополнительных возможностей отображения, теперь применять не рекомендуется.

В настоящее время существуют официальные версии XHTML 1.0 и 1.1. и черновая спецификация XHTML 2.0. Однако работа над последней остановлена в пользу создания нового стандарта — HTML 5.

Начиная с 2004 г. совместными усилиями рабочих групп W3C HTML WG и WHATWG с привлечением таких компаний, как Apple, Mozilla, Opera, Google, Microsoft и др. ведется работа по созданию спецификации HTML 5. К середине 2010 г. существовал лишь черновой ее вариант (www.w3.org/TR/html5, www.whatwg.org/specs/web-apps). От W3C было заявление, что официальная рекомендация должна появиться к концу 2010 г. по другим источникам — 2012 г. Полная поддержка HTML 5 браузерами осуществится, видимо, еще позднее, но некоторые элементы Firefox, Opera, Safari и Chrome воспринимают уже сейчас. Браузер Internet Explorer приступит к поддержке HTML 5, начиная с версии 9.

HTML 5 призван стать преемником как HTML 4, так и XHTML 1.0, обеспечивая совместимость с ними, по крайней мере, в самом важном. В HTML 5 можно использовать синтаксис как обычного HTML, так и XHTML. Вместе с тем ряд тегов и атрибутов, не рекомендованных к использованию в прежних версиях, в новом стандарте отсутствуют. Например, теги `<applet>` для вставки Java-апплетов, `<center>` для центрирования содержимого по горизонтали и `<frameset>`, `<frame>` для разбиения окна браузера на фреймы в HTML 5 не поддерживаются. С целью обеспечения обратной совместимости в спецификации HTML 5 описывается, как должен поступить браузер в случае использования непредусмотренных тегов. Уже существующие сайты не должны пострадать из-за появления нового языка разметки.

В HTML 5 вводится в оборот ряд новых тегов с отчетливо выраженной семантикой, предназначенных для повышения эффективности разработки Web-приложений, а также для помощи поисковым системам. Так, для создания логической структуры и упрощения типовой верстки документа предлагаются теги `<header>` (заголовок), `<nav>` (навигационная панель), `<article>` (статья), `<section>` (раздел, секция статьи), `<aside>` (боковая колонка), `<footer>` ("подвал"), `<menu>` (меню) и др. В прежних версиях данные элементы обычно создаются тегами `<div>` и сразу понять смысл их применения довольно трудно. Иначе говоря, семантика для `<div>` в спецификации неоднозначна.

Элементы `<input>` полей ввода данных в HTML 5 приобрели новые атрибуты, заметно облегчающие создание пользовательского интерфейса: `time`, `email`, `url`, `required` и др. Эти атрибуты во многих случаях позволяют обойтись без скриптов, проверяющих правильность введенных данных.

Особого внимания заслуживает элемент `<canvas>`, предоставляющий область на странице, в которой с помощью скриптов на языке JavaScript можно рисовать изображения, подобные тем, которые создаются посредством SVG или Flash. Для вставки мультимедиа в HTML 5 предусмотрены теги `<audio>` и `<video>`, обеспечивающие внедрение на страницу соответствующего ресурса и возможность управления его воспроизведением.

HTML 5 предусматривает и другие полезные и удобные возможности. В данной книге будут описаны лишь некоторые из них, которые уже поддерживаются ведущими браузерами.

ВНИМАНИЕ! В данной книге описываются возможности HTML 4 и XHTML, а также применение некоторых элементов HTML 5. Коды примеров написаны в соответствии с синтаксисом XHTML, хотя дескриптор типа документа `<!DOCTYPE...>` в некоторых из них указывает на стандарт HTML 4.01. Такая эклектика вполне допустима. Разумеется,