

# Adobe Dreamweaver CS4

**+ ВИДЕОКУРС**



- Основные принципы Web-дизайна
- Фреймовый, табличный и контейнерный Web-дизайн
- Каскадные таблицы стилей
- Web-сценарии
- Использование эффектов и компонентов Spry
- Написание серверных Web-приложений
- Работа с наборами данных HTML и XML

**+ cd**

**Наиболее  
полное  
руководство**

**В ПОДЛИННИКЕ®**

**Владимир Дронов**

# **Adobe Dreamweaver CS4**

Санкт-Петербург

«БХВ-Петербург»

2009

УДК 681.3.06  
ББК 32.973.26-018.2  
Д75

**Дронов В. А.**

Д75 Adobe Dreamweaver CS4. — СПб.: БХВ-Петербург,  
2009. — 832 с.: ил. + Видеокурс (на CD-ROM) — (В подлиннике)

ISBN 978-5-9775-0412-6

Книга посвящена созданию Web-страниц и Web-сайтов в русской версии визуального Web-редактора Adobe Dreamweaver CS4. Подробно рассказывается об основных принципах Web-дизайна, о верстке Web-страниц с использованием фреймов, таблиц и контейнеров, использовании каскадных таблиц стилей CSS, Web-сценариев и эффектов и компонентов Spry. Особое внимание уделяется написанию в среде Dreamweaver серверных Web-приложений с использованием технологии ASP и работе с наборами данных HTML и XML. Подробно рассказывается о публикации готовых Web-сайтов в Интернете. Прилагаемый CD содержит видеокурс по основам работы в Adobe Dreamweaver CS4.

*Для Web-дизайнеров и Web-программистов*

УДК 681.3.06  
ББК 32.973.26-018.2

### **Группа подготовки издания:**

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Евгений Рыбаков</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Игорь Цырульников</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 22.02.09.

Формат 70×100<sup>1/16</sup>. Печать офсетная. Усл. печ. л. 67,08.

Тираж 2500 экз. Заказ №

"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию  
№ 77.99.60.953.Д.003650.04.08 от 14.04.2008 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 978-5-9775-0412-6

© Дронов В. А., 2009

© Оформление, издательство "БХВ-Петербург", 2009

# Оглавление

<b>Введение</b> .....	<b>1</b>
Dreamweaver CS4 — что нового? .....	2
Типографские соглашения .....	3
Как мы будем изучать Dreamweaver.....	3
Благодарности .....	4
<b>ЧАСТЬ I. СОЗДАЕМ ПРОСТЕЙШИЕ WEB-СТРАНИЦЫ</b> .....	<b>5</b>
<b>Глава 1. Введение в современные интернет-технологии</b> .....	<b>7</b>
Основные принципы работы Интернета .....	7
Что такое Интернет .....	7
Сервисы Интернета .....	9
Клиенты и серверы.....	9
Протоколы .....	13
Интернет-адреса .....	15
Основные понятия WWW .....	17
Web-страницы и Web-сайты.....	17
Web-обозреватели .....	21
Web-серверы.....	23
Публикация Web-сайта в Интернете. Хостинг-провайдеры.....	23
Как создаются Web-страницы? .....	25
Язык HTML и его теги.....	25
Вложенность тегов.....	27
Две секции Web-страницы .....	29
Гиперссылки .....	30
Интернет-адреса в WWW .....	32
Каскадные таблицы стилей CSS .....	33
Физическое и логическое форматирование .....	35
Будущее HTML.....	36
Что дальше?.....	37

<b>Глава 2. Основные принципы работы с Web-редактором Dreamweaver CS4.....</b>	<b>38</b>
Web-редакторы.....	38
Среда Adobe Dreamweaver CS4.....	40
Главное окно программы.....	40
Окна документов.....	42
Панели.....	46
Раскладки.....	52
Предварительная настройка Dreamweaver.....	56
Учим русский.....	56
Настраиваем параметры создаваемых Web-страниц.....	59
Задаем параметры создаваемого HTML-кода.....	60
Файловые операции.....	61
Инструменты для работы с Web-страницами.....	63
Инструменты, присутствующие в окнах документов.....	64
Просмотр Web-страницы.....	67
Поиск и замена текста.....	70
Вызов справки.....	80
Что дальше?.....	81
<b>Глава 3. Создание простейших Web-страниц.....</b>	<b>82</b>
Создание новой Web-страницы.....	82
Работа с текстом.....	84
Набор текста.....	84
Форматирование абзацев.....	87
Форматирование фрагментов текста.....	94
Вставка специальных символов.....	101
Работа с гиперссылками.....	108
Обычные гиперссылки.....	108
Почтовые гиперссылки.....	113
Якоря.....	114
Общие параметры Web-страницы.....	117
Дополнительные возможности Dreamweaver.....	123
Вставка и чтение комментариев.....	123
Вставка текущей даты.....	124
"Чистка" HTML-кода.....	126
"Чистка" HTML-кода, порожденного Microsoft Word.....	127
Что дальше?.....	131
<b>Глава 4. Работа с кодом HTML.....</b>	<b>132</b>
Основные средства Dreamweaver для работы с HTML-кодом.....	132
Три режима отображения Web-страницы.....	133
Вызов справки по HTML.....	135

Простейшие средства для работы с кодом HTML .....	136
Свертка кода HTML .....	140
Расширенные средства работы с HTML-кодом.....	142
Вставка тегов в HTML-код.....	142
Правка и удаление тегов.....	145
Настройки, влияющие на создаваемый HTML-код .....	147
Что дальше? .....	149

## **ЧАСТЬ II. ПРОДОЛЖАЕМ СОЗДАВАТЬ WEB-СТРАНИЦЫ ..... 151**

### **Глава 5. Графика и мультимедиа ..... 153**

Введение в интернет-графику .....	154
Внедренные элементы .....	154
Растровые и векторные изображения. Сжатие графики .....	155
Форматы интернет-графики .....	156
Простые графические изображения .....	157
Вставка графического изображения .....	157
Параметры графического изображения .....	162
Правка изображения в Dreamweaver .....	164
Специальные изображения.....	168
Изображения-гиперссылки.....	168
Активные изображения.....	170
Карты-изображения .....	172
Фоновые изображения .....	178
Мультимедиа .....	180
Поддержка мультимедийных данных.....	180
Графика Shockwave/Flash .....	183
Фильмы Flash Video .....	190
Прочее мультимедийное содержимое .....	195
Что дальше? .....	201

### **Глава 6. Таблицы ..... 202**

Текст фиксированного формата .....	202
Работа с таблицами .....	205
Создание таблицы .....	205
Как формируются таблицы.....	208
Задание размеров таблицы, ее строк и столбцов.....	211
Выделение элементов таблиц.....	214
Работа с таблицей и ее элементами .....	216
Форматирование таблиц .....	219
Объединение ячеек таблиц.....	223
Практикум по работе с таблицами.....	225
Специальные возможности по работе с таблицами .....	227
Сортировка таблиц.....	227
Вставка табличных данных .....	230

Составные изображения .....	232
Фиксация размеров ячеек таблицы .....	234
Недостатки таблиц и их преодоление .....	235
Что дальше? .....	237
<b>Глава 7. Работа с Web-сайтом .....</b>	<b>238</b>
Начала сайтостроения .....	238
Планирование сайта .....	239
Публикация Web-сайта .....	245
Регистрация сайта в Dreamweaver .....	247
Процесс регистрации сайта .....	248
Настройка прокси-сервера .....	255
Работа с локальной копией сайта .....	256
Управление файлами и папками локальной копии .....	256
Управление Web-сайтами, зарегистрированными в Dreamweaver .....	260
Взаимодействие панели <i>Файлы</i> и окна документа .....	263
Проверка гиперссылок и HTML-кода .....	264
Замена гиперссылок .....	268
Публикация сайта .....	269
Соединение с FTP-сервером .....	269
Простая публикация сайта и отдельных его файлов .....	270
Синхронизация копий сайта .....	271
Разъединение с FTP-сервером .....	275
Работа с удаленной копией сайта .....	275
Поиск обновленных файлов .....	278
Работа с Web-страницами на FTP-сервере напрямую .....	279
Дополнительные возможности работы с Web-страницами .....	280
Заметки .....	280
Активы .....	286
Избранные элементы .....	288
Библиотека .....	290
Что дальше? .....	293
<b>ЧАСТЬ III. ЗАНИМАЕМСЯ WEB-ДИЗАЙНОМ .....</b>	<b>295</b>
<b>Глава 8. Фреймовый Web-дизайн .....</b>	<b>297</b>
Введение во фреймы .....	297
Фреймы и наборы фреймов .....	298
Схемы наборов фреймов .....	299
Работа с фреймами в Dreamweaver .....	303
Создание фреймов .....	303
Как формируются фреймы .....	309
Параметры фреймов и наборов фреймов .....	311
Создание содержимого фреймов .....	315
Заполнение фреймов .....	316

Использование цели гиперссылки для открытия страниц в заданном фрейме .....	318
Создание полосы навигации.....	319
Оптимизация фреймов.....	324
Уменьшение объема и сложности HTML-кода фреймов .....	324
Ускорение обработки фреймов.....	326
Недостатки фреймов и их преодоление .....	327
Что дальше?.....	328
<b>Глава 9. Табличный Web-дизайн.....</b>	<b>329</b>
Схемы табличного дизайна .....	329
Построение таблиц разметки .....	333
Использование полосы навигации на страницах, построенных на основе таблиц разметки .....	337
Недостатки табличного Web-дизайна .....	338
Недостатки таблиц .....	339
Недостатки табличного Web-дизайна .....	342
Что дальше?.....	343
<b>Глава 10. Шаблоны.....</b>	<b>344</b>
Введение в шаблоны Dreamweaver.....	344
Работа с шаблонами.....	345
Создание шаблона.....	345
Правка шаблона.....	348
Создание изменяемых областей.....	350
Создание Web-страниц на основе шаблонов .....	352
Правка шаблонов, на основе которых уже созданы Web-страницы.....	355
Применение шаблонов к уже созданным Web-страницам .....	356
Использование полосы навигации в шаблонах и основанных на них страницах .....	359
Открепление Web-страниц от шаблонов.....	360
Управление шаблонами в списке панели <i>Активы</i> .....	361
Дополнительные возможности шаблонов.....	362
Изменяемые атрибуты .....	362
Необязательные области.....	365
Повторяющиеся области .....	368
Необязательные изменяемые области .....	371
Табличная повторяющаяся область.....	372
Вложенные шаблоны .....	373
Недостатки шаблонов и их преодоление .....	374
Что дальше?.....	375
<b>Глава 11. Работа со стилями CSS.....</b>	<b>376</b>
Введение в каскадные таблицы стилей .....	376
Стили.....	376
Таблицы стилей.....	379

Правила каскадности и приоритет стилей .....	381
Псевдостили .....	383
Работа с таблицами стилей в Dreamweaver .....	384
Создание стилей .....	385
Определение стиля .....	388
Привязка стилей .....	406
Панель <i>Стили CSS</i> .....	407
Управление стилями .....	410
Работа с CSS-кодом таблиц стилей .....	414
Управление таблицами стилей .....	417
Вызов справки по CSS .....	419
Контейнеры .....	420
Что дальше? .....	422
<b>ЧАСТЬ IV. ИСПОЛЬЗУЕМ НОВЕЙШИЕ ТЕХНОЛОГИИ.....</b>	<b>423</b>
<b>Глава 12. Плавающие контейнеры.....</b>	<b>425</b>
Простейший случай контейнерного Web-дизайна .....	426
Атрибуты стиля CSS, предназначенные для управления контейнерами.....	428
Пример контейнерного Web-дизайна.....	431
Пример реализации "резиновых" контейнеров .....	436
Стандартные разметки Dreamweaver.....	444
Недостатки контейнерного Web-дизайна .....	445
Что дальше? .....	446
<b>Глава 13. Свободно позиционируемые контейнеры .....</b>	<b>447</b>
Введение в свободно позиционируемые контейнеры.....	447
Что такое свободно позиционируемый контейнер.....	448
Как создаются свободно позиционируемые контейнеры .....	449
Работа со свободно позиционируемыми контейнерами .....	453
Создание свободно позиционируемых контейнеров.....	454
Параметры свободно позиционируемых контейнеров .....	457
Работа с группой свободно позиционируемых контейнеров .....	459
Использование панели <i>Элементы AP</i> .....	461
Пример использования свободно позиционируемых контейнеров .....	463
Прямое задание параметров свободно позиционируемых контейнеров.....	468
Инструменты позиционирования Dreamweaver .....	470
Преобразование свободно позиционируемых контейнеров в таблицы разметки и обратно.....	476
Недостатки свободно позиционируемых контейнеров и их преодоление .....	478
Что дальше? .....	479
<b>Глава 14. Использование Web-сценариев .....</b>	<b>480</b>
Начала Web-программирования .....	480
Web-сценарии .....	481

События.....	483
Краткий курс языка JavaScript .....	484
Как пишутся Web-сценарии .....	496
Простейший Web-сценарий.....	497
Более сложный Web-сценарий.....	498
Web-сценарии — подход Dreamweaver. Поведения.....	500
Работа с поведением.....	502
Создание поведений.....	502
Поведения, поддерживаемые Dreamweaver .....	506
Написание своих Web-сценариев .....	530
Привязка Web-сценариев к фрагментам текста и произвольным тегам .....	532
Дополнительные инструменты Dreamweaver для работы со сценариями.....	533
Недостатки Web-сценариев и их преодоление .....	535
Альтернативные технологии. Апплеты Java.....	536
Что дальше?.....	539
<b>Глава 15. Использование эффектов и компонентов Spry .....</b>	<b>540</b>
Эффекты Spry.....	541
Эффект <i>Всплеск</i> .....	542
Эффект <i>Высвечивание</i> .....	542
Эффект <i>Жалюзи</i> .....	543
Эффект <i>Появление/Растворение</i> .....	545
Эффект <i>Расширить/Сжать</i> .....	546
Эффект <i>Скольжение</i> .....	548
Эффект <i>Тряска</i> .....	549
Компоненты Spry .....	550
Меню ( <i>Панель меню Spry</i> ).....	550
"Блокнот" ( <i>Панель со вкладками Spry</i> ).....	556
Связанные панели ( <i>Набор вкладок Spry</i> ) .....	560
Сворачиваемая панель ( <i>Сворачивающаяся панель Spry</i> ).....	563
Что дальше?.....	567
<b>Глава 16. Метатеги и серверные директивы.....</b>	<b>568</b>
Реклама в Интернете.....	568
Поисковые машины .....	568
Как работают поисковые агенты .....	572
Метатеги .....	573
Пассивная интернет-реклама .....	574
Работа с метатегами в среде Dreamweaver.....	576
Ключевые слова.....	577
Описание Web-страницы .....	578
Перезагрузка и перенаправление .....	579
Базовый интернет-адрес .....	580
Связи между Web-страницами.....	581
Специальные метатеги.....	584

Серверные директивы.....	585
Введение в серверные директивы.....	585
Вставка серверных директив в HTML-код .....	587
Стандартный набор серверных директив.....	587
Как использовать серверные директивы .....	589
Поддержка серверных включений в Dreamweaver .....	590
Что дальше? .....	591
<b>ЧАСТЬ V. ПИШЕМ СЕРВЕРНЫЕ WEB-СТРАНИЦЫ.....</b>	<b>593</b>
<b>Глава 17. Введение в серверное программирование.....</b>	<b>595</b>
Начала серверного программирования .....	595
Как работают серверные программы .....	595
Разновидности серверных программ.....	597
Как реализуется ввод данных. Web-формы .....	600
Как данные передаются по Сети.....	602
Серверное программирование — подход Dreamweaver. Серверные поведения .....	604
Какую технологию создания серверных страниц нам выбрать?.....	606
Введение в базы данных.....	607
Что дальше? .....	609
<b>Глава 18. Web-формы.....</b>	<b>610</b>
Работа с Web-формами в Dreamweaver .....	610
Создание Web-формы .....	611
Создание элементов управления.....	613
Основные принципы разработки Web-форм .....	631
Практикум по созданию Web-форм .....	633
Простейшая Web-форма .....	633
Использование таблиц и стилей для создания Web-форм .....	636
Список гиперссылок .....	638
Поведения, предназначенные для работы с Web-формами.....	640
Создание списка гиперссылок ( <i>Меню переходов</i> ) .....	641
Создание кнопки перехода для списка гиперссылок ( <i>Выполнение меню переходов</i> ).....	641
Задание нового значения поля ввода ( <i>Задать текст текстового поля</i> ).....	642
Проверка данных, введенных в форму ( <i>Проверить форму</i> ) .....	643
Компоненты Spry, предназначенные для работы с элементами управления.....	645
Проверка данных, введенных в поле ввода ( <i>Текстовое поле проверки Spry</i> ) .....	645
Проверка данных, введенных в область редактирования ( <i>Текстовая область проверки Spry</i> ) .....	651
Проверка данных, заданных с помощью флажков ( <i>Флажок проверки Spry</i> ) .....	654
Проверка данных, заданных в списке ( <i>Выбор проверки Spry</i> ) .....	658
Проверка данных, введенных в поле ввода пароля ( <i>Пароль проверки Spry</i> ) .....	661
Проверка подтверждения введенного пароля ( <i>Подтверждение проверки Spry</i> ).....	664
Что дальше? .....	667

<b>Глава 19. Простейшие серверные Web-страницы.....</b>	<b>668</b>
Подготовка к созданию серверных Web-страниц .....	669
Установка соединения с базой данных .....	673
Создание источника данных ODBC.....	674
Регистрация базы данных в Dreamweaver .....	677
Создание серверных страниц в Dreamweaver .....	681
Страница добавления записи.....	681
Простейшая страница для просмотра данных .....	686
Страница для просмотра нескольких записей .....	699
Создание динамических списков .....	702
Создание сложных наборов записей.....	705
Динамические атрибуты .....	708
Передача данных другой Web-странице .....	710
Создание фильтров.....	711
Необязательные области серверной страницы .....	714
Что дальше?.....	717
<b>Глава 20. Создание Web-сайтов, основанных на серверных Web-страницах.....</b>	<b>718</b>
Принципы создания динамических сайтов.....	718
База данных нашего сайта .....	720
Административные страницы сайта .....	721
Как администрируются Web-сайты .....	721
Инструменты для работы со списком категорий.....	722
Инструменты для работы со списком статей.....	730
Средства разграничения доступа .....	734
Страницы общего доступа.....	742
Страница списка категорий.....	742
Страница списка статей.....	745
Страница регистрации нового посетителя .....	747
Реализация поиска статей.....	749
Что дальше?.....	750
<b>ЧАСТЬ VI. ИСПОЛЬЗОВАНИЕ НАБОРОВ ДАННЫХ .....</b>	<b>751</b>
<b>Глава 21. Наборы данных HTML .....</b>	<b>753</b>
Понятие набора данных.....	754
Работа с наборами данных HTML в Dreamweaver .....	756
Подключение набора данных HTML.....	758
Простые случаи вывода данных из наборов HTML.....	763
Более сложные случаи вывода данных из наборов HTML.....	769
Достоинства и недостатки наборов данных.....	780
Что дальше?.....	781

---

<b>Глава 22. Наборы данных XML .....</b>	<b>782</b>
Язык XML.....	782
Краткий курс языка XML .....	782
Наборы данных XML.....	784
Два способа обработки данных XML. Таблицы стилей XSL и XSLT .....	785
Три способа обработки данных XML, поддерживаемые Dreamweaver.....	786
Вывод данных XML с помощью компонентов Spry .....	788
Вывод данных XML средствами Web-обозревателя.....	792
Вывод данных XML средствами Web-сервера.....	798
Вызов справки по XML и XSLT .....	801
<b>Заключение.....</b>	<b>803</b>
<b>Приложение. Описание компакт-диска.....</b>	<b>806</b>
<b>Предметный указатель .....</b>	<b>807</b>

# Введение

Программисты корпорации Adobe не зря едят свой хлеб. Совсем недавно из-под их клавиатур вышла очередная версия Dreamweaver — CS4. Значит, пора писать о ней книгу...

Что такое Dreamweaver? Это мощнейший Web-редактор — программа для создания Web-страниц и Web-сайтов, прекрасный помощник как начинающего, так и опытного Web-дизайнера. Сейчас на такие программы спрос велик, ведь Интернет растет и ширится, все больше и больше людей и организаций хотят в нем, что называется, "засветиться", и профессия Web-дизайнера популярна как никогда.

Ну а что нужно *Web-дизайнеру*, т. е. разработчику Web-страниц и сайтов? Разумеется, знание всех необходимых интернет-технологий, художественный вкус и мощное и удобное программное обеспечение. И таким программным обеспечением вполне может стать Adobe Dreamweaver CS4. Почему? Да потому что:

- ◆ для создания Web-страниц в его среде необязательно знать все необходимые интернет-технологии и интернет-стандарты;
- ◆ даже если Web-дизайнер, работающий в Dreamweaver, не знает ни технологий, ни стандартов, он, скорее всего, изучит их в процессе работы;
- ◆ а когда начинающий Web-дизайнер станет докой в своем деле, Dreamweaver все равно пригодится ему. Набор его возможностей велик, очень велик... Посмотрите хотя бы на толщину этой книги — а ведь в ней описано далеко не все.

Отличная программа этот Dreamweaver! Уж поверьте автору — он пользуется им, начиная с версии 2.0 (а может быть, и 1.0 — сейчас уже и не вспомнить), и весьма доволен. В частности, именно с помощью Dreamweaver он создал и поддерживает свои сайты.

Да что тут рассказывать!.. Лучше сами попробуйте Dreamweaver в действии! А в качестве учебника используйте эту книгу.

## Dreamweaver CS4 — что нового?

Dreamweaver — весьма "старый" пакет. Первая его версия была разработана еще в 1998 году и получила популярность в России благодаря своей "благо-склонности" к русскому языку и снисходительному отношению к множеству русскоязычных кодировок.

С тех пор утекло много воды. Dreamweaver рос и развивался. В версии 4.0 появилось увесистое электронное руководство по языкам HTML, CSS и JavaScript — истинное благодеяние для начинающих Web-дизайнеров, желающих повысить свою квалификацию. Dreamweaver MX получил средства для разработки серверных Web-страниц, ранее поставлявшиеся в виде отдельного продукта Macromedia UltraDev. Dreamweaver MX 2004 щеголял сильно улучшенным интерфейсом и кое-какими дополнительными инструментами, в частности, для мелкой правки изображений. Dreamweaver 8 мог похвастаться улучшением всего того, что было в нем раньше, и еще кое-какими мелкими нововведениями. Dreamweaver CS3 преподнес Web-дизайнерам набор весьма впечатляющих компонентов Spry — меню, панель с вкладками, сворачивающиеся панели, элементы управления с проверкой правильности введенных данных и пр., — а Web-программистов обрадовал средствами для работы с данными XML; плюс очередные улучшения всего уже имеющегося.

А что же Dreamweaver CS4? Что он может нам предложить?

- ◆ Режим интерактивного просмотра, позволяющий увидеть, как Web-страница будет выглядеть в Web-обозревателе "живьем".
- ◆ Удобный доступ к связанным файлам (внешним таблицам стилей, файлам сценариев, файлам с наборами данных и пр.).
- ◆ Поддержка наборов данных HTML.
- ◆ Расширенная поддержка графики Photoshop.
- ◆ Обновленный пользовательский интерфейс.

Что ж, новый Dreamweaver имеет много "плюсов". К сожалению, "минусов" у него тоже хватает...

- ◆ Adobe продолжает работать по-крупному — размер дистрибутива Dreamweaver CS4 составляет 576 Мбайт, что вдвое больше, чем у Dreamweaver CS3.
- ◆ Исключены кое-какие возможности, присутствовавшие в предыдущих версиях Dreamweaver, как то: создание анимации, режим разметки таблиц, представление карты сайта, создание надписей и кнопок Flash и поддержка серверных технологий ASP.NET и JSP. Кому они помешали?..
- ◆ В окончательной версии Dreamweaver CS4 присутствуют досадные ошибки, мешающие работать. Подобными делами славилась фирма Macro-

media, первый владелец Dreamweaver, и сначала казалось, что после перехода под "крылышко" Adobe этот пакет избавился от ошибок. Но нет...

Так стоит ли переходить на Dreamweaver CS4. Автор перешел на него несколько месяцев назад и не жалеет. Интерактивный просмотр, доступ к связанным файлам, мелкие улучшения в интерфейсе — это, как и вообще все хорошее, быстро вызывает привыкание...

## Типографские соглашения

Прежде чем начать изучение Web-дизайна, давайте кое о чем условимся.

В этой книге будут приведены примеры на языках HTML, CSS, JavaScript (забегая вперед — эти языки используются при создании Web-страниц) и XML (а это универсальный язык описания данных). При написании примеров были использованы типографские соглашения, уже ставшие своего рода стандартами в компьютерном книгоиздании. Нам необходимо их знать.

- ◆ В угловые скобки (<>) заключаются названия параметров или фрагментов кода, которые, вместе со скобками, дополнительно выделяются курсивом. В код реального сценария, разумеется, должен быть подставлен реальный параметр или реальный код. Например:

```
<A HREF="<интернет-адрес>";
```

Здесь вместо строки "<интернет-адрес>" должен быть подставлен реальный интернет-адрес.

- ◆ В квадратные скобки ([ ]) заключаются необязательные фрагменты кода. Например:

```
htm[1];
```

Последняя буква 1 может присутствовать, а может и не присутствовать.

- ◆ Вертикальной чертой (|) разделяются несколько фрагментов кода, из которых в данном месте должен присутствовать только один. Например:

```
input|textarea
```

Здесь должно присутствовать либо слово `input`, либо слово `textarea`, но не оба сразу.

Весь остальной код HTML набирается "как есть".

## Как мы будем изучать Dreamweaver

Изучать Dreamweaver CS4 мы будем на конкретном примере. Мы создадим личный Web-сайт гипотетического Web-дизайнера Ивана Ивановича Иванова. Сначала этот сайт будет совсем простеньким; на его основе мы изучим

базовые инструменты программы и основные принципы Web-дизайна. Впоследствии сайт станет сложнее и красивее; мы используем фреймы, таблицы и контейнеры, чтобы придать ему профессиональный вид. Далее мы изучим таблицы стилей и Web-сценарии, позволяющие "оживить" страницы нашего сайта. А на самой последней ступени мы изучим серверное программирование и сделаем для нашего сайта гостевую книгу. Ну и, конечно же, мы узнаем, как собрать разрозненные страницы в сайт и опубликовать его в Сети.

Но прежде чем начать разговор об Adobe Dreamweaver CS4, дадим несколько основных понятий и ответим на несколько вопросов, которые могут возникнуть у неподготовленного читателя. Конечно, если вы на короткой ноге с Всемирной сетью, можете пропустить первую главу и начать читать сразу со второй. Но автор обязан позаботиться обо всех, кто будет читать эту книгу.

А теперь — еще немного "воды"...

## 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 — 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.
- ◆ Всем своим читателям и почитателям — за прекрасные отзывы о моих книгах.
- ◆ Всем, кого я забыл здесь перечислить, — за все хорошее.



# ЧАСТЬ I

## Создаем простейшие Web-страницы

- Глава 1.** Введение в современные интернет-технологии
- Глава 2.** Основные принципы работы с Web-редактором Dreamweaver CS4
- Глава 3.** Создание простейших Web-страниц
- Глава 4.** Работа с кодом HTML



# ГЛАВА 1



## Введение в современные интернет-технологии

У читателей наверняка скопилось много вопросов. Что представляют собой те красивые Web-странички, которые выводит нам Web-обозреватель? Как они создаются? Какие программы нужны для этого? Наверно, все это очень сложно...

Стоп-стоп-стоп! Разумеется, автор ответит на все эти вопросы. И начнет он с самой что ни на есть голой теории...

## Основные принципы работы Интернета

Говорят, в первой польской энциклопедии, изданной, кажется, в XVII столетии, термин "лошадь" описывался так: "что такое лошадь, знают все". То же самое можно сейчас сказать об Интернете. (Вот только можно ли сейчас сказать то же самое о лошади?..)

Но дать определение Интернету мы все-таки должны. Просто для того, чтобы понимать, о чем идет речь. А поняв это, можем приступить к изучению принципов, по которым этот самый Интернет работает.

## Что такое Интернет

В самом деле, что такое *Интернет*? Всемирная компьютерная сеть. Ее, кстати, так часто и называют: Всемирная сеть, или даже просто Сеть с большой буквы. Протянутая по всему земному шару паутина медных проводов, волоконно-оптических линий и радиоканалов, связывающих друг с другом многочисленные компьютеры, — вот что такое Интернет. Разумеется, все здесь подчиняется общим стандартам (о которых мы поговорим далее) — иначе эта суперсеть просто не будет работать.

Если же быть совсем точным, то Интернет — это не единая сеть, а совокупность более мелких сетей, связанных друг с другом общими каналами и стандартами. Таких сетей превеликое множество: огромные территориальные сети, раскинувшиеся на целые области, штаты и государства, и ведомственные сети, объединяющие родственные организации, и локальные компьютерные сети отдельных организаций, и так называемые кампусные сети — сети, объединяющие компьютеры одного или нескольких близлежащих районов города. Благодаря проложенным между ними каналам высокоскоростной связи они составляют единое целое, имя которому Интернет.

Даже частные пользователи, подключающиеся к Интернету по модему, выделенной линии, радиоканалу или поддерживающему такую возможность сотовому телефону, тоже по сути дела являются частью Сети. Так что когда мы включаем наш модем и дозваниваемся до нашего *интернет-провайдера* (организации, предоставляющей доступ в Интернет), то приобщаемся к единому целому. А что, разве это не повод для законной гордости?

Сеть Интернет имеет одну замечательную особенность — она очень устойчива к сбоям. Так, если где-то порвется провод, мы этого не заметим. А все потому, что данные, которые мы запрашиваем, пойдут в этом случае по другому проводу. Специалисты говорят, что Интернет децентрализован — он не имеет единого центра, из которого ведется управление пересылкой данных, поэтому в случае аварии автоматически переконфигурируется и продолжает нормально работать.

Еще одна замечательная особенность Интернета — его глобальность, всемирность. Не вставая из-за компьютера, мы можем совершить путешествие по всему миру, побывать в США, Австралии, Германии, Танзании, на Огненной Земле и даже в Антарктиде! Для этого нужно всего лишь набрать нужный нам интернет-адрес.

Интернет имеет достаточно долгую и бурную историю. Он появился еще в конце 60-х годов XX века, когда Министерство обороны США финансировало проект создания компьютерной сети, устойчивой к сбоям. Разумеется, создавалась эта сеть для нужд обороны, да и название имела другое — *ARPANET*. Позднее, в начале 80-х, эта сеть отошла к ученым, а военные приступили к созданию другой сети, которой пользуются до сих пор. И в то же самое время ARPANET был переименован в *Internet*, или, если по-русски, Интернет.

Первоначально, еще во времена ARPANET, эта сеть использовалась для пересылки электронной почты и обмена файлами. Web-странички, ради которых мы, в основном, и путешествуем по Сети, появились только в конце 80-х. Именно тогда Интернет и "пошел в народ", перестав быть сетью ученых и превратившись в сеть для всех.

В Россию, точнее, в СССР, Интернет официально пришел в 1991 году, но популярность среди широких масс компьютерщиков приобрел только в середине 90-х. В настоящее же время в России, наверно, и не найти человека, не слышавшего об Интернете. Вы такого встречали? Автор — еще нет.

## Сервисы Интернета

Раз уж мы заговорили об услугах, предоставляемых Интернетом, или, как говорят профессионалы, *сервисах* Интернета, то давайте узнаем о них побольше. В конце концов, нам ими пользоваться...

Самый старый и самый популярный до сих пор сервис Интернета — это электронная почта (e-mail). Ежедневно в мире отправляются и принимаются сотни миллионов электронных писем, и это количество в будущем будет только увеличиваться. В самом деле, электронная почта доступна, удобна, быстра и бесплатна, в отличие от почты "бумажной", которую пользователи Интернета уже успели презрительно прозвать "улиточной" (по-английски — snail mail). Конечно, эти доступность, удобство, быстрота и бесплатность имеют и некоторые недостатки, вроде спама — несанкционированных рекламных рассылок, но эти недостатки вполне можно стерпеть.

Еще один сервис Интернета, почти такой же старый, как почта, — это пересылка файлов. Пользователи Интернета называют его *FTP* (File Transfer Protocol, протокол передачи файлов; почему так — мы узнаем чуть позже). Сейчас FTP уже не имеет той популярности, как на заре существования Интернета, но все еще довольно часто используется. Так, все крупные корпорации — Microsoft, Adobe, Intel и др. — помимо Web-сайта, имеют и сервер FTP.

Третий сервис Интернета — это Всемирная паутина, или *WWW* (World Wide Web, повсеместно протянутая паутина), или просто *Web*, те самые Web-страницы и Web-сайты, которые мы просматриваем в Web-обозревателе. Появившийся значительно позже электронной почты и FTP, WWW стала самым популярным сервисом и, собственно, превратила Интернет из сети учебных в сеть для всех.

Об остальных сервисах Интернета (а их немало) мы только упомянем. Это потоковое вещание, интернет-пейджеры, чаты, нашумевшие в последние несколько лет файлообменные сети и некоторые другие, менее известные или устаревшие сервисы.

## Клиенты и серверы

Но каким образом мы пользуемся всем тем богатством, что дает нам Всемирная сеть? С помощью особых программ! Это Web-обозреватель, клиент элек-

тронной почты, программа просмотра интернет-телевидения и прослушивания интернет-радио, интернет-пейджер и "чатилка". Все они очень хорошо нам знакомы.

Но программ, используемых для предоставления нам сервисов Интернета, гораздо больше. И очень многие из них нам, если так можно сказать, "не видны", т. е. мы не общаемся с ними напрямую. Вообще, существуют два совершенно разных вида интернет-программ. И сейчас мы о них поговорим.

Программы, относящиеся к первому виду, — это Web-обозреватели, клиенты электронной почты, чатов, интернет-пейджеры, в общем, все те, с которыми мы имеем дело непосредственно. Мы получаем с их помощью различную информацию из Сети и работаем с ней. Такие программы называются программами-клиентами, а компьютеры, на которых они работают, — наши с вами компьютеры! — клиентскими.

Да, но как программы-клиенты получают из Сети нужную нам информацию (Web-страницы, файлы, письма и пр.)? Очень просто — для этого они обращаются к другим программам, относящимся ко второму виду. Это программы-серверы, работающие на серверных компьютерах, где также хранится и запрашиваемая клиентами информация. Существуют Web-, FTP-серверы, серверы электронной почты, чата, интернет-пейджеров, потокового вещания и пр.

### На заметку

Очень часто понятие "сервер" распространяется и на серверный компьютер, и на саму программу-сервер. Это, вообще-то, неправильно, т. к. на одном серверном компьютере может быть установлено несколько различных программ-серверов, но вошло в практику.

Процесс получения информации клиентами от сервера включает шесть шагов.

1. Пользователь запрашивает с помощью программы-клиента некую информацию, введя в нее интернет-адрес сервера. (Об интернет-адресах мы поговорим потом, а пока что будем знать, что это особый адрес, однозначно идентифицирующий нужную нам программу-сервер, работающую на определенном компьютере, который подключен к Интернету.)
2. Клиент устанавливает *соединение* (воображаемую линию связи) с сервером и посылает тому особый информационный блок, называемый *клиентским запросом*. Этот запрос содержит описание требуемых клиенту данных и должен быть определенным образом оформлен, чтобы сервер его понял.
3. Сервер принимает запрос и расшифровывает его.

4. Сервер извлекает запрошенный файл или фрагмент данных, записанных в файле, и посылает его клиенту в составе другого информационного блока — *серверного ответа*. Разумеется, этот ответ также должен быть составлен определенным образом. Если же запрашиваемые клиентом данные не были найдены, или сервер почему-то не смог понять клиентский запрос, он возвращает *сообщение об ошибке* — информационный блок, содержащий *код* (числовой номер) и, возможно, текстовое описание возникшей ошибки. Так, если Web-сервер не найдет запрошенную Web-обозревателем Web-страницу, он вернет сообщение об ошибке с кодом 404 — "запрошенная Web-страница не найдена".
5. Клиент получает ответ от сервера, расшифровывает его и выдает полученную информацию пользователю. Если получено сообщение об ошибке, клиент сообщает об этом пользователю либо предпринимает какие-то действия самостоятельно. Так, получив сообщение об ошибке с кодом 404, Web-обозреватель выведет соответствующее сообщение.
6. Клиент разрывает соединение с сервером.

Процесс отправки клиентом данных серверу также включает шесть шагов.

1. Пользователь вводит в программу-клиент информацию и интернет-адрес сервера, которому она должна быть отправлена.
2. Клиент устанавливает соединение с сервером и посылает тому отправляемую информацию в составе клиентского запроса. При этом отправляемая информация, как правило, особым образом кодируется.
3. Сервер принимает запрос, расшифровывает его и извлекает отправленную информацию.
4. Сервер записывает отправленную клиентом информацию в файл, помещает в базу данных или обрабатывает каким-то образом. В случае успешной записи или обработки он отправляет клиенту так называемое *подтверждение* — информационный блок, сообщающий о том, что все прошло нормально. Если у сервера возникли проблемы с приемом информации, он отправляет сообщение об ошибке.
5. Клиент получает ответ от сервера, расшифровывает его и уведомляет пользователя об успешной или неуспешной отправке данных либо предпринимает какие-то действия самостоятельно.
6. Клиент разрывает соединение с сервером.

Весь процесс "общения" клиента и сервера, начиная с отправки клиентом запроса и заканчивая принятием им ответа от сервера, называется *сеансом*. А соединение между клиентом и сервером, устанавливаемое на время этого сеанса и разрываемое после его окончания, называется *сеансовым*, или *временным*.

Любое соединение между клиентом и сервером устанавливается только клиентом. Сервер установить соединение с клиентом не может. Можно сказать, что серверу здесь отведена подчиненная роль.

Мы только что познакомились с особой *архитектурой* (принципом построения компьютерных систем), называемой *двухзвенной*, или архитектурой "*клиент-сервер*". Эта архитектура использует два вида программ — клиенты и серверы, — выполняющие разные роли. Она используется для реализации почти всех современных интернет-сервисов и пока что себя оправдывает.

### На заметку

Некоторые интернет-сервисы, в частности файлообменные сети (BitTorrent и др.), используют другую архитектуру — *однозвенную*. Здесь все компьютеры, подключенные к Интернету и реализующие этот сервис, фактически равны между собой; любой из них может выступать в роли как клиентского (запрашивать информацию у других компьютеров), так и серверного (предоставлять хранящуюся на нем информацию другим компьютерам). Само собой, здесь используется особое программное обеспечение, которое может работать и как клиент, и как сервер.

В отличие от клиента, "имеющего дело" с одним-единственным пользователем, сервер работает сразу с множеством пользователей, причем одновременно. Сведения о соединениях, данные, пересылаемые клиентам и принимаемые от клиентов, — все это активно отнимает системные ресурсы компьютера, и чем больше соединений и данных проходят через сервер, тем больше требуется ресурсов. Поэтому на серверных компьютерах, как правило, не экономят.

Серверные компьютеры — настоящие монстры, содержащие несколько процессоров, дисковые массивы впечатляющей емкости, быстрые каналы связи с Интернетом и специальное программное обеспечение, способное "ворочать" огромными массивами данных. Все в них нацелено на то, чтобы обслужить как можно больше клиентов за минимальное время. Но часто, если клиентов и запросов оказывается слишком много, ресурсов серверного компьютера не хватает, и начинаются проблемы. Они могут проявляться в том, что сервер просто отказывается обслужить "лишних" клиентов, предлагая им подождать немного, когда нагрузка немного снизится, а то и в том, что могучий серверный компьютер просто-напросто "зависает". Такое тоже случается, и не так уж редко...

Ну да не будем о грустном! Не стоит начинать знакомство с таким притягательным миром интернет-технологий со столь печальных вещей, как системные сбои. Чем их меньше, и чем реже они случаются, тем лучше для всех нас.

## Протоколы

Люди, чтобы понимать друг друга, должны разговаривать на одном языке. Точно так и с компьютерами, подключенными к сети, неважно какой — всемирной или локальной. Обмен данными по этим сетям должен проходить по единым стандартам, иначе начнется новое вавилонское столпотворение.

Стандарт, согласно которому организуются передаваемые по сети данные и команды, управляющие передачей этих данных, называется *протоколом*. В Интернете для обмена данными используются довольно много протоколов, и некоторые мы здесь вкратце рассмотрим.

Самый фундаментальный протокол Интернета — *IP* (Internet Protocol, межсетевой протокол). Он занимается тем, что разбивает подготовленные к передаче данные на порции (*пакеты*) определенной длины и определенного формата, помещает в каждую такую порцию интернет-адреса компьютера-отправителя и компьютера-получателя и предусматривает простейшие средства защиты от сбоев, которые могут возникнуть при пересылке данных.

Можно сказать, что протокол IP выполняет "грязную" работу по пересылке данных, работая на самом низком уровне. Поэтому его называют *протоколом низкого уровня*.

На IP базируется протокол *TCP* (Transfer Control Protocol, протокол управления передачей). Он обеспечивает гарантированную доставку данных, т. е. отвечает за то, чтобы все отправленные данные дошли до компьютера-получателя. Но это только первая из его обязанностей.

А вторая обязанность протокола TCP заключается в том, что он делит один реальный, физический, канал связи Интернета (кабель, волоконно-оптическую линию или радиоканал) на несколько воображаемых, виртуальных, "каналчиков", называемых *портами TCP*. Делается это для того, чтобы по одному физическому каналу можно было передавать сразу несколько потоков данных, принадлежащих разным программам, — для этого используются разные порты TCP. Всего таких портов предусмотрено 65 535, и все они пронумерованы.

TCP также относится к протоколам низкого уровня и так тесно связан с IP, часто эту парочку называют одним словом *TCP/IP*. А иногда даже считают за один протокол.

TCP/IP используется другими протоколами, уже *высокого уровня*. Эти протоколы описывают способы оформления клиентских запросов, серверных ответов, подтверждений и сообщений об ошибках, команды, пересылаемые клиентом серверу при запросе или передаче данных, и способ кодирования передаваемой информации.

### На заметку

Строго говоря, существуют еще *протоколы физического уровня*, располагающиеся "ниже" даже IP. Они определяют электрические параметры сигнала, кабелей, разъемов и пр.

Каждый сервис Интернета использует свой собственный протокол высокого уровня, а то и несколько, предназначенных для разных задач или разработанных конкурирующими организациями. Давайте рассмотрим протоколы, с которыми мы столкнемся в будущем.

Начнем мы, конечно, с WWW. Для передачи данных Всемирная паутина использует протокол *HTTP* (HyperText Transfer Protocol, протокол передачи гипертекста). Он задает набор команд, отправляемых клиентом (Web-обозревателем) Web-серверу, и способы представления пересылаемых данных. Пожалуй, это самый широкоизвестный протокол Интернета — всем более-менее грамотным интернетчикам знакомы эти четыре буквы.

Сервис пересылки файлов FTP использует протокол, который так и называется — FTP. Он также определяет набор команд для управления файлами на сервере (загрузка с сервера, отправка на сервер, копирование, перемещение, удаление, создание папки на сервере и т. д.) и способы кодирования файлов для пересылки по каналам связи. В этом смысле протоколы HTTP и FTP весьма похожи.

А вот электронная почта использует целых два протокола. Первый протокол — *SMTP* (Simple Mail Transfer Protocol, простой протокол пересылки почты) — используется для пересылки почты клиентом серверу. Для получения же почты от сервера клиент общается с ним по протоколу *POP3* (Post-Office Protocol ver. 3, протокол почты версии 3).

Существует еще один почтовый протокол — *IMAP* (Internet Message Access Protocol, протокол доступа к почте Интернета). "Коллега" и "наследник" более старого POP3, он предоставляет больше возможностей, но распространен не так широко.

Чуть раньше мы узнали о портах TCP. Так вот, каждый существующий протокол высокого уровня использует для передачи данных свой собственный порт (так называемый *порт по умолчанию*). В табл. 1.1 перечислены некоторые протоколы и используемые ими порты по умолчанию.

Порт по умолчанию может быть изменен — такую возможность предоставляют все более-менее серьезные серверы. Так, Web-сервер может быть настроен так, чтобы использовать для "общения" с клиентами не 80-й порт, а, скажем, 8000-й. Это применяется, например, если на одном серверном компьютере работают два Web-сервера; тогда один из них настраивают на порт по умолчанию — 80-й, — а другой — да хотя бы и на 8000-й.

**Таблица 1.1.** Порты TCP, используемые по умолчанию для передачи данных некоторых протоколов высокого уровня

Протокол	Используемый порт TCP
FTP	21
HTTP	80
POP3	110
SMTP	25

## Интернет-адреса

Теперь давайте поговорим о том, каким образом идентифицируются компьютеры, подключенные к Интернету. А именно — об интернет-адресах.

*Интернет-адрес* — это числовое или строковое значение, позволяющее точно идентифицировать компьютер в Сети. Именно такой интернет-адрес (точнее, два — отправителя и получателя) подставляется в каждый отправляемый по Сети пакет IP, чтобы он успешно дошел до места назначения.

### На заметку

Существует, правда, возможность дать одному компьютеру сразу несколько интернет-адресов. Но используется это нечасто и в особых случаях. И в дальнейшем для простоты мы будем считать, что один интернет-адрес — это один компьютер.

На заре эпохи Интернета в качестве интернет-адреса использовался *IP-адрес* — числовое значение, идентифицирующее компьютер для протокола IP. IP-адрес замечательно подходит для компьютеров, но очень плохо — для людей. Вот пример интернет-адреса:

### 192.168.1.10

Не очень-то наглядно, правда? Именно поэтому с расширением Интернета была введена в строй новая система интернет-адресов, которой мы пользуемся до сих пор. Это так называемые доменные имена, о которых стоит поговорить подробно.

Но прежде чем мы начнем разговор о доменных именах, давайте выясним, что такое домен. *Домен*, или *доменная зона*, — это участок Интернета, созданный для удобства управления им. Такой участок может быть крупным, мелким или вообще состоять из одного компьютера. Каждому домену присваивается имя, состоящее из латинских букв и цифр; также могут быть использованы символы дефиса, подчеркивания и некоторые другие.

Структура доменов похожа на матрешку: мелкие домены "вложены" внутрь крупных, а крупные, в свою очередь, — внутрь гигантских. Гигантские домены называются *доменами верхнего уровня*, а вложенные в них более мелкие — *доменами нижнего уровня*.

Домены верхнего уровня бывают интернациональными и национальными. *Интернациональные домены* объединяют компьютеры по какому-то признаку; к ним относятся домены с именами com и biz (коммерческие организации), edu (образовательные), mil (военные), org (организации, не занимающиеся компьютерами и Интернетом), net (организации, занимающиеся компьютерами и Интернетом), travel (туристические организации) и некоторые другие. *Национальные домены* объединяют компьютеры по территориальному признаку и выдаются отдельным странам; это домены с именами us (США), uk (Великобритания), fr (Франция), de (Германия), ru (Россия) и др.

Домены нижнего уровня выдаются, как правило, отдельным организациям или, опять же, по территориальному признаку. Их текстовое обозначение часто совпадает с названием этой организации или района.

Если теперь записать обозначения всех доменов, в которых находится нужный нам компьютер, в порядке от более мелких к более крупным, разделив их точками, мы получим *доменное имя* этого компьютера. Так, если у нас сам компьютер имеет имя comp45, отдел, в котором он стоит, — buh (бухгалтерия), организация, включающая этот отдел, — office, а страна — ru (Россия), то мы получим такое доменное имя:

**comp45.buh.office.ru**

Согласитесь — запомнить это гораздо проще, чем невразумительный IP-адрес.

Да, но проблема в том, что протокол IP не понимает доменные имена! Что делать? Как преобразовать доменное имя в понятный ему IP-адрес?

Для этого используется особый сервис Интернета, называемый *DNS* (Domain Name System, система доменных имен). Клиент отправляет *серверу DNS* запрос, содержащий доменное имя, и получает в виде ответа IP-адрес, соответствующий этому доменному имени. А уж с IP-адресом он знает, что делать.

Такие серверы DNS имеются в каждом домене; кроме того, несколько самых мощных в мире серверов DNS (*корневые серверы DNS*) находятся как бы "выше" всех доменов, даже доменов верхнего уровня.

Всем хороши доменные имена, кроме одного, — они не позволяют задать номер порта TCP или хотя бы протокол. Они только задают сам серверный компьютер, а ведь на одном серверном компьютере могут работать несколько программ-серверов. Что делать? Просто указать перед доменным именем

обозначение протокола, реализуемого нужным сервером, вот так (обозначение протокола подчеркнуто):

**http://comp45.buh.office.ru**

**ftp://comp45.buh.office.ru**

В первом случае мы обращаемся к Web-серверу, а во втором — к серверу FTP, находящемуся на одном и том же компьютере **comp45.buh.office.ru**.

Также имеется возможность указать номер порта TCP, через который производится обмен данными. Номер порта записывается после доменного имени серверного компьютера через двоеточие, вот так (подчеркнут):

**http://comp45.buh.office.ru:8000**

Многие серверы (почтовые, FTP и др.) требуют от пользователя ввода его имени и, возможно, пароля. Имя пользователя помещается между названием протокола и самим доменным именем и отделяется от последнего знаком амперсанда (@). Вот два примера задания имени пользователя в доменном имени сервера (подчеркнуто):

**ftp://user@comp45.buh.office.ru**

**account@server.ru**

Последний пример демонстрирует нам обычный адрес электронной почты. Заметим, что название протокола здесь не указывается — почтовый клиент и почтовый сервер сами знают, какой протокол использовать.

Ну а пароль пользователя помещается между именем и знаком @ и отделяется от имени двоеточием — вот так (подчеркнут):

**ftp://user:password@comp45.buh.office.ru**

Ну вот, с основными принципами работы Интернета мы ознакомились. Теперь давайте сосредоточимся на WWW — в основном, именно этим сервисом мы будем пользоваться на протяжении всей книги.

## Основные понятия WWW

Здесь мы узнаем все о Web-страницах и Web-сайтах, выясним, чем сайт отличается от страницы, поговорим о Web-обозревателях и Web-серверах и изучим множество новых терминов.

## Web-страницы и Web-сайты

Что такое Web-страница? Ответить на этот вопрос могут многие. Это интернет-документ, предназначенный для распространения через Интернет по-

средством сервиса WWW. А если уж говорить по-простонародному, это то, что показывает в своем окне программа-клиент для просмотра Web-страниц — Web-обозреватель.

С технической точки зрения Web-страница — это обычный текстовый файл, который можно создать в любом текстовом редакторе, например Блокноте, стандартно поставляемом в составе Windows. Этот файл содержит собственно текст Web-страницы и команды форматирования этого самого текста. Команды форматирования называются *тегами*, а описывает их особый язык *HTML* (HyperText Markup Language, язык гипертекстовой разметки). Файл Web-страницы обязательно должен иметь расширение `htm[l]`.

А что такое Web-сайт? Это набор Web-страниц, подчиненных общей тематике и объединенных в единое целое (как — будет рассказано далее в этой книге). Как видим, сугубо технических отличий у Web-страницы и Web-сайта не слишком много.

Web-сайт сохраняется на жестких дисках серверного компьютера, на котором работает Web-сервер (серверная программа, обеспечивающая работу сервиса WWW), в виде набора различных файлов. Прежде всего, это, разумеется, файлы Web-страниц, составляющих сайт. Многие сайты включают файлы графических изображений, помещенных на страницы (почему изображения хранятся отдельно от самих страниц, мы узнаем потом). Также сайт может содержать файлы архивов и дистрибутивов программ и некоторые другие файлы, о которых мы тоже поговорим потом.

Зачастую различные файлы, составляющие сайт, хранятся в папках. Конечно, папки использовать необязательно, но так удобнее, особенно если файлов много и все они разных типов.

Для хранения всех файлов, составляющих сайт, на диске серверного компьютера создается особая папка, называемая *корневой*. Все файлы и папки сайта должны находиться только в этой папке, без малейших исключений.

Корневую папку сайта на серверном компьютере создает человек, занимающийся настройкой и обслуживанием программы Web-сервера (или же всего серверного компьютера), — *администратор*. При этом он заносит полный путь этой папки в настройки Web-сервера, чтобы последний "знал", где ее найти.

### На заметку

Нужно отметить, что все серьезные программы Web-серверов позволяют создавать так называемые *виртуальные папки*. Виртуальная папка может находиться абсолютно в любом месте файловой системы компьютера, но Web-сервер считает ее частью сайта, словно она находится в его корневой папке. Виртуальные папки также создаются администратором Web-сервера.

Но как нам получить нужный файл (страницу, архив или дистрибутив) с Web-сайта? Правильно — для этого нужно указать Web-обозревателю интернет-адрес этого файла, введя его в специальное поле ввода. Web-обозреватель извлечет из введенного нами интернет-адреса путь к нужному файлу и отправит его Web-серверу, управляющему сайтом. Web-сервер получит этот путь, найдет корневую папку, отыщет в ней запрошенный файл и пришлет Web-обозревателю, т. е. нам.

Предположим, мы ввели в Web-обозреватель вот такой интернет-адрес:

**`http://www.somesite.ru/somepage.html`**

В этом случае Web-обозреватель сразу выделит из него интернет-адрес Web-сервера

**`http://www.somesite.ru`**

и путь к запрошенному нами файлу

**`/somepage.html`**

Как мы видим, в начале пути стоит символ слэша (/). Он обозначает корневую папку сайта. Именно в корневой папке Web-сервер будет искать файл `somepage.html`.

Далее Web-обозреватель отправит Web-серверу **`http://www.somesite.ru`** такой запрос:

**`/somepage.html`**

то есть путь к нужному нам файлу. Web-сервер, получив этот запрос, найдет в корневой папке файл `somepage.html`, загрузит его и отправит Web-обозревателю. Если же такого файла нет или Web-сервер почему-то не сможет его загрузить, он отправит Web-обозревателю сообщение об ошибке.

Мы уже знаем, что любой пакет IP содержит в себе, кроме всего прочего, интернет-адрес отправителя. Кроме того, интернет-адрес клиентского компьютера посылается в составе клиентского запроса HTTP. Так что Web-сервер всегда сможет узнать, куда ему отправить запрошенный файл.

Если же мы наберем в Web-обозревателе вот такой интернет-адрес:

**`http://www.somesite.ru/download/archive.zip`**

Web-обозреватель выделит из него такой путь:

**`/download/archive.zip`**

Сейчас мы запросили архивный файл `archive.zip`, находящийся в папке `download`, вложенной в корневую папку сайта. Web-обозреватель pošлет Web-серверу **`http://www.somesite.ru`** вот такой запрос:

**`/download/archive.zip`**

### На заметку

Для запроса файла, находящегося в виртуальной папке, используется аналогичный интернет-адрес:

**<http://www.somesite.ru/pictures/picture.jpg>**

Здесь pictures — виртуальная папка, в которой Web-сервер будет искать файл picture.jpg.

Так, все прекрасно, все замечательно и все исключительно ясно! Но мы ведь нечасто указываем Web-обозревателю интернет-адреса, содержащие пути конкретных файлов. Скажем, мы можем набрать вот такой интернет-адрес:

**<http://www.somesite.ru>**

Тогда Web-обозреватель отправит Web-серверу в клиентском запросе вот что:

/

то есть один символ слэша. Как поступит Web-сервер в таком случае?

Дело в том, что одна из страниц сайта задается в качестве так называемой *страницы по умолчанию*. Именно она отправляется Web-обозревателю, если он не указал в клиентском запросе путь конкретного файла. Имя файла страницы по умолчанию указывается администратором Web-сервера в его настройках — как правило, default.htm[1] или index.htm[1].

Так что в приведенном выше случае Web-сервер отправит нам страницу по умолчанию, хранящуюся в корневой папке сайта. Это может быть, например, страница default.html.

Мы можем набрать в строке ввода интернет-адреса Web-обозревателя и такой интернет-адрес — **<http://www.somesite.ru/articles>**. Тогда Web-обозреватель отправит в составе клиентского запроса следующее:

**[/articles/](#)**

И Web-сервер в ответ pošлет нам страницу по умолчанию, хранящуюся в папке articles, что вложена в корневую папку сайта.

Впоследствии мы еще вернемся к интернет-адресам, используемым в WWW. А пока что закончим на этом.

### На заметку

Отдельные Web-страницы и целые Web-сайты также могут быть сохранены на жестком диске клиентского компьютера и открываться в Web-обозревателе прямо с него. В этом случае роль своеобразного Web-сервера выполняет файловая система.

## Web-обозреватели

Мы уже знаем, что Web-обозреватели — это программы для просмотра Web-страниц и Web-сайтов. Основная их задача — отправить Web-серверу корректно, в соответствии со всеми стандартами сформированный клиентский запрос, принять серверный ответ и обработать его (вывести полученную страницу на экран, сохранить на диске полученный архивный файл и пр.). Для этого окно Web-обозревателя содержит поле ввода интернет-адреса и область, в которой отображается содержимое полученной Web-страницы. (Разумеется, оно также содержит заголовок, меню и панели инструментов, как и многие окна приложений Windows.)

Обычно после получения от Web-сервера файла Web-страницы (да и любого другого файла, составляющего содержимое сайта) Web-обозреватель сохраняет их на жестком диске клиентского компьютера в особой области, называемой *кэш*. Этот кэш может иметь вид как обычной папки (кэш Microsoft Internet Explorer или Opera), так и большого файла (кэш Mozilla или Firefox).

Зачем это нужно? Да хотя бы затем, чтобы мы смогли впоследствии просмотреть данную страницу, не подключаясь к Интернету. Все современные Web-обозреватели поддерживают так называемый *автономный режим* (offline mode), когда они отображают только те страницы, что находятся в кэше. (Кстати — исключительно удобная вещь!) Если же мы попытаемся просмотреть страницу, которой нет в кэше, Web-обозреватель предложит нам подключиться к Интернету и загрузить ее.

Но даже если мы и подключены в данный момент к Интернету, Web-обозреватель все равно активно использует кэш. Перед тем как загрузить какой-либо файл, он проверяет, не изменился ли данный файл по сравнению с тем, что находится сейчас в его кэше (если, конечно, он там уже есть). Если не изменился, он загружает нужный файл прямо из кэша, что намного быстрее.

Теперь познакомимся с программами Web-обозревателей, имеющими в настоящее время наибольшую популярность. Все они, в общем, следуют одним и тем же стандартам и отличаются друг от друга только деталями, не оговоренными в этих стандартах, и удобством для пользователей.

Настоящий король виртуальных пространств — это, конечно, Microsoft Internet Explorer. Он имеется на любом компьютере, работающем под управлением Windows (что, как говорят злые языки, и обусловило его популярность). Однако это очень мощная, быстрая, весьма нетребовательная к ресурсам и исключительно удобная программа. Автор данной книги для просмотра Web-страниц пользуется именно Internet Explorer. В настоящее время доступна версия 7.0, которая входит в состав новой версии Windows — Windows Vista;

также существует отдельная версия для Windows XP и 2003. А скоро должна выйти версия 8.0.

### На заметку

Начиная с версии 7.0, Internet Explorer официально сменил имя. Теперь он называется Microsoft Windows Internet Explorer. Но автор на протяжении этой книги будет пользоваться старым наименованием этой программы как более привычным.

Второе место по популярности занимает весьма амбициозный Web-обозреватель Mozilla Firefox, или просто Firefox. Эта программа распространяется бесплатно, более того, ее исходные тексты открыты для изучения и модификации. Она весьма быстра и компактна, поддерживает все Web-стандарты, нетребовательна к системным ресурсам и имеет множество интересных и весьма полезных возможностей, которыми пока не может похвастаться ни один из его конкурентов. Самой последней версией Firefox на данный момент является 3.0.3.

Третье место оккупировано разработкой фирмы Apple, производящей широко известные в узких кругах компьютеры Macintosh, — Safari. В настоящее время имеет хождение версия Safari 3.1, которая доступна и для Macintosh, и для Windows. Утверждается, что это самый быстрый в мире Web-обозреватель. Пока трудно сказать, что в действительности этот Safari собой представляет — у автора нет под рукой данной программы, чтобы проверить это утверждение.

На четвертом месте отдыхает разработка норвежских программистов Opera. Эта достаточно мощная и очень быстрая программа (хотя и уступающая, по слухам, Safari), поддерживающая все официальные Web-стандарты, тем не менее, весьма требовательна к системным ресурсам и не всегда правильно отображает некоторые Web-страницы. Последняя вышедшая в свет версия носит номер 9.62 и, скорее всего, после выхода книги устареет, т. к. новые версии Opera появляются очень часто.

В настоящее время просторы WWW "бороздят" практически только четыре перечисленные выше программы. Существует, однако, еще несколько малоизвестных Web-обозревателей, а также довольно многочисленная когорта программ, построенных на основе программного ядра Internet Explorer и расширяющих его возможности. Мы не будем их рассматривать.

Осталось только сказать, что выбор Web-обозревателя — это личное дело каждого. Все они поддерживают одни и те же стандарты (правда, зачастую по-своему) и предоставляют пользователю примерно одинаковый набор возможностей. Так что, как в песне поется, "думайте сами, решайте сами..."

## Web-серверы

Поскольку мы не только пользователи, но и уже наполовину разработчики, нас будут интересовать также и Web-серверы. Давайте поговорим и о них.

"Зоопарк" Web-серверов ничуть не меньше "зоопарка" (или, если учесть, что Web-обозреватели жестоко конкурируют друг с другом, "серпентария") Web-обозревателей, так что мы можем подобрать себе программу по вкусу. И, в отличие от Web-обозревателей, среди Web-серверов нет безоговорочного лидера — даже самые распространенные из них не занимают больше половины рынка.

Начнем наш краткий обзор с двух программ фирмы Microsoft: Personal Web Server и Internet Information Services. Они поставляются в составе Microsoft Windows, первая — в составе Windows 98 и ME, а вторая — в составе Windows NT, 2000, XP, 2003 и Vista. Со своими обязанностями они справляются очень хорошо, не транжирят системные ресурсы, легко настраиваются, поддерживают множество передовых интернет-технологий и при надлежащей настройке легко затыкают за пояс конкурентов. Кроме собственно Web-сервера, они содержат также простейшие серверы FTP и почты, а также множество дополнительных программ.

Web-сервер Apache — пожалуй, самый распространенный. Среди его достоинств: полная бесплатность (более того — его исходные тексты открыты), легкость настройки, довольно высокая производительность, хорошая поддержка. По крайней мере, для Web-сайтов с небольшой загрузкой — это идеальный выбор.

Есть еще один весьма примечательный Web-сервер — Sambar. Он поддерживает такое количество интернет-технологий (многие из них — эксклюзивные, не доступные больше ни в одной программе), что просто оторопь берет — как же всем этим богатством воспользоваться и куда его применить? Недостатка у Sambar всего два: недостаточная известность и не очень удобная настройка. (Кстати, собственный сайт автора этой книги, доступный в сети института, где он работает, функционирует под управлением именно этого Web-сервера.)

Менее известные и специализированные Web-серверы мы рассматривать не будем, т. к. их довольно много. Поговорим лучше о том, как разместить созданный нами сайт в Интернете.

## Публикация Web-сайта в Интернете. Хостинг-провайдеры

Итак, предположим, что мы создали свой сайт (а мы его и создадим, пока будем изучать интернет-технологии по этой книге). Теперь нам нужно сделать

так, чтобы все желающие увидеть его собственными глазами, а именно — разместить, или, как говорят опытные интернетчики, *опубликовать* его в Интернете. А значит, нам нужно подключение к Интернету и Web-сервер.

Если наш компьютер подключен к Интернету по скоростному каналу или находится в локальной сети, работающей по тем же стандартам, что и Интернет (так называемый *интранет*), то мы можем просто установить на него Web-сервер и сюда же поместить наш Web-сайт. Это самый простой способ, хотя, конечно, нам придется попутно освоить профессию администратора.

Для тех "счастливчиков", что выходят в Интернет по телефонным каналам (как автор этой книги), существуют три способа донести свое Web-творение до страждущих масс. Давайте перечислим их в порядке от простых к более хлопотным.

Большинство солидных интернет-провайдеров, кроме собственно доступа в Интернет, предоставляют своим клиентам и другие услуги: электронную почту, собственный сайт с новостями, документацией и файловым архивом и пр. Так вот, среди этих "пр." есть и такая услуга, как выделение на жестких дисках серверного компьютера места для размещения Web-сайтов клиентов. В этом и заключается первый способ: выяснить условия публикации сайта на Web-сервере интернет-провайдера и, следуя этим условиям, опубликовать его.

Если же интернет-провайдер почему-то жадничает, можно прибегнуть ко второму способу. В Интернете существует довольно много Web-серверов, предоставляющих место для сайтов бесплатно. Процесс и в этом случае очень прост: заходим на сайт такого сервера, регистрируемся, выясняем условия публикации сайта и публикуем.

Всем хороши бесплатные Web-серверы: и денег не берут, и позволяют публиковать сайты. Но бесплатного сыра много не бывает... Как правило, объем предоставляемого под сайт дискового пространства сильно ограничен, значит, большой сайт таким образом не опубликуешь. Еще администратор может ограничить количество пользователей, которые могут одновременно зайти на наш сайт. Да и с поддержкой некоторых технологий, используемых для создания Web-сайтов, дело может обстоять не очень хорошо.

Если же нам нужно нечто большее, чем предложения бесплатных серверов, то придется достать кошелек и пойти третьим путем — опубликовать сайт на платном Web-сервере. Благо сейчас их достаточно много, и услуги их довольно дешевы.

Кстати, организации, предоставляющие место на своих серверах для публикации Web-сайтов, называются *хостинг-провайдерами*.

# Как создаются Web-страницы?

Разобравшись с Интернетом, интернет-технологиями, серверами и клиентами, перейдем к рассмотрению Web-страниц. Как они создаются, какие средства и программы для этого используются — на все эти вопросы автор сейчас даст ответ.

## Язык HTML и его теги

Мы уже знаем, что Web-страницы — суть обычные текстовые файлы, созданные в любом простейшем текстовом редакторе (том же Блокноте) и сохраненные с расширением `htm[1]`. Эти файлы содержат текст, который составляет содержимое страницы, и особые команды, называемые тегами и используемые для задания внешнего вида или назначения тех или иных *элементов* Web-страницы. С помощью этих тегов можно, например, оформить фрагмент текста как отдельный абзац, выделить его полужирным шрифтом или курсивом или превратить в заголовок.

Также мы знаем, что для создания Web-страниц используется язык HTML. Этот язык определяет набор тегов, их назначение, правила расстановки в тексте страницы и их дополнительные параметры.

Язык HTML лучше всего изучать на примерах. Так что давайте не будем болтать впустую, а сразу же создадим нашу первую Web-страничку.

Операционная система Windows уже содержит все инструменты для этого. И первый из этих инструментов — текстовый редактор Блокнот. Запустим его и наберем приведенный ниже текст (или, как говорят программисты, код) на языке HTML. Выглядит он устрашающе, но настоящего Web-дизайнера так просто не испугаешь.

```
<HTML>
  <HEAD>
    <TITLE>Web-страница</TITLE>
  </HEAD>
  <BODY>
    <H1>Пример Web-страницы</H1>
    <P>Это простейшая Web-страничка, созданная в стандартном
    <EM>Блокноте</EM> и отображенная в <EM>Microsoft Internet
    Explorer</EM>.</P>
  </BODY>
</HTML>
```

После этого проверим набранный код HTML на ошибки и сохраним в файле с именем `1.1.htm`. Только когда будем вводить имя файла в стандартном окне

сохранения, заключим его в кавычки, иначе Блокнот по доброту душевной добавит расширение txt, и наш файл получит имя 1.1.htm.txt.

Теперь откроем полученный файл в Web-обозревателе Internet Explorer, для чего достаточно дважды щелкнуть по нему мышью. Internet Explorer — второй инструмент Web-дизайна, который припасла нам Windows. То, что мы увидим в его окне, показано на рис. 1.1.

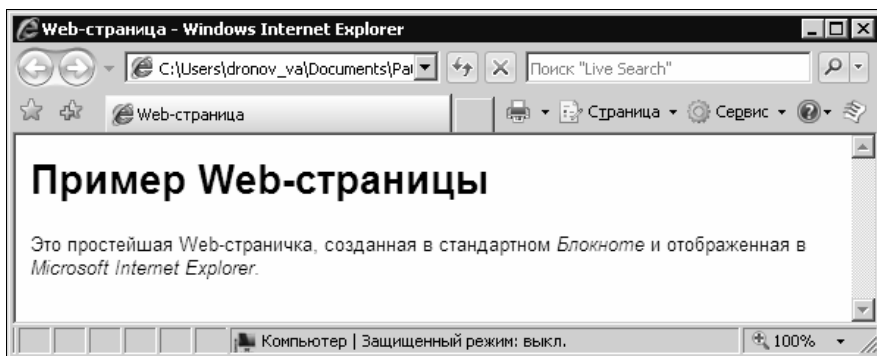


Рис. 1.1. Простейшая Web-страничка

Замечательно! Наша первая Web-страница, хоть и совсем простенькая, готова!

Настала пора разобраться с языком HTML. Снова откроем Блокнот, если уже закрыли его, и загрузим в него только что созданную Web-страницу. И найдем в ее HTML-коде вот этот фрагмент — как видно, он задает содержимое Web-страницы, видимое в окне Web-обозревателя:

```
<H1>Пример Web-страницы</H1>  
<P>Это простейшая Web-страничка, созданная в стандартном  
<EM>Блокноте</EM> и отображенная в <EM>Microsoft Internet  
Explorer</EM>.</P>
```

Помимо самого текста, здесь присутствуют какие-то слова, заключенные в символы < ("меньше") и > ("больше"). Это и есть теги HTML. Они задают форматирование текста. Скажем, строка "Блокноте" будет выведена курсивом, т. к. теги <EM> и </EM> задают курсивное начертание текста. Причем тег <EM> помечает начало курсивного фрагмента (*открывающий* тег), а тег </EM> — конец (*закрывающий*). А собственно фрагмент текста, заключенный между открывающим и закрывающим тегами, называется *содержимым* тега. Именно к содержимому применяется действие тега.

Также в приведенном выше фрагменте HTML-кода имеются теги <P> и <H1> (и соответствующие им закрывающие теги </P> и </H1>). Они создают соот-

ветственно обычный текстовый абзац и заголовок; при этом Web-обозреватель отобразит их надлежащим образом — см. рис. 1.1.

Еще один полезный и часто употребляемый тег — `<STRONG>`. Он выделяет текст полужирным шрифтом. Так, если мы изменим HTML-код нашей Web-страницы таким образом:

```
<H1>Пример Web-страницы</H1>
```

```
<P>Это простейшая Web-страничка, созданная в стандартном  
<EM>Блокноте</EM> и отображенная в <STRONG>Microsoft Internet  
Explorer</STRONG>.</P>
```

слова "Microsoft Internet Explorer" отобразятся полужирным шрифтом.

Как видно, ничего особо сложного в языке HTML нет. Единственная сложность — это запомнить все его теги, но это вопрос времени, опыта и хорошей справочной литературы.

Для того чтобы различные Web-обозреватели отображали одну и ту же Web-страницу одинаково, язык HTML должен быть стандартизирован. Его стандартизацией (а также множеством других стандартов Интернета) занимается особая организация, называемая *World Wide Web Consortium* (сокращенно — *WWWС* или, что встречается чаще, *W3С*). Это название дословно можно перевести как "Комитет Всемирной паутины".

W3С издает весьма увесистые труды, описывающие различные версии стандарта HTML. Последняя версия этого языка — 4.01 — вышла в конце 90-х годов прошлого века; ее поддерживают все современные Web-обозреватели. В настоящее время идет разработка версии 5.0 языка HTML; она еще в полной мере никем не поддерживается.

## Вложенность тегов

Если внимательно посмотреть на HTML-код нашей страницы, можно заметить, что одни теги вложены в другие. В частности, тег `<EM>` вложен в тег `<P>`. Такая *вложенность* тегов в HTML — обычное дело.

Когда Web-обозреватель встречает тег, вложенный в другой тег, он как бы складывает эффект от применения этих тегов. Так, в нашем случае слово "Microsoft Internet Explorer" будет частью абзаца и отобразится курсивным шрифтом. То есть Web-обозреватель сложит эффект от применения тегов `<P>` и `<EM>`.

Если мы заключим в тег `<EM>`, скажем, тег `<STRONG>` (как мы уже знаем, он задает полужирное начертание текста), то содержимое последнего будет выведено полужирным курсивом. Давайте так и сделаем. Измененный фрагмент нашей Web-страницы будет выглядеть так:

```
<P>Это простейшая Web-страничка, созданная в стандартном
<EM>Блокноте</EM> и отображенная в <EM><STRONG>Microsoft</STRONG>
Internet Explorer</EM>.</P>
```

Сохраним полученный файл под именем 1.2.htm и откроем его в Web-обозревателе. То, что мы увидим, показано на рис. 1.2.

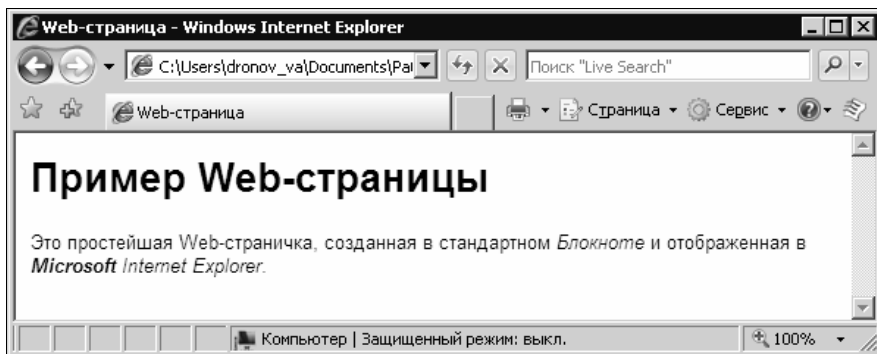


Рис. 1.2. Демонстрация эффекта вложенности тегов HTML

Обратим внимание на порядок, в котором следуют открывающие и закрывающие теги. 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>Это простейшая Web-страничка, созданная в стандартном
<EM>Блокноте</EM> и отображенная в <STRONG><EM>Microsoft</EM>
Internet Explorer</EM>.</P>
```

то Web-обозреватель может и не отобразить наше творение правильно (хотя Internet Explorer славится своим умением исправлять мелкие ошибки Web-дизайнера). Так что об этом желательно не забывать.

Осталось только выучить несколько новых терминов. Тег, в который непосредственно вложен данный тег, называется *родительским*. В свою очередь, тег, вложенный в данный тег, называется *дочерним*. Так, для тега `<EM>` в приведенном ниже 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).

Уровень вложенности того или иного тега показывает количество тегов, в которые он последовательно вложен. Так, если принять за точку отсчета тег `<P>`, то тег `<EM>` будет иметь первый уровень вложенности, т. к. он вложен непосредственно в тег `<P>`. Тег `<STRONG>` же будет иметь второй уровень вложенности, т. к. он вложен в тег `<EM>`, а тот, в свою очередь, — в тег `<P>`. В сложных же Web-страницах уровень вложенности иных тегов может составлять несколько десятков.

## Две секции Web-страницы

Теперь давайте рассмотрим еще несколько тегов, используемых для служебных целей и не отображаемых Web-обозревателем. Они так и называются — *невидимые* теги. (Все остальные уже рассмотренные нами теги были *видимыми*.)

Посмотрим снова на HTML-код нашей Web-страницы. Мысленно удалим из него фрагмент, "отвечающий" за видимое содержимое. Получится вот что:

```
<HTML>
  <HEAD>
    <TITLE>Web-страница</TITLE>
  </HEAD>
  <BODY>
    . . .
  </BODY>
</HTML>
```

Видно, что все теги, задающие содержимое Web-страницы, помещаются внутри парного тега `<BODY>`. Этот тег используется для выделения так называемой *секции тела* Web-страницы. Все видимое содержимое страницы должно находиться в секции тела, т. е. в теге `<BODY>`.

Другой парный тег — `<HEAD>` — задает *секцию заголовка* (не путайте с обычным текстовым заголовком, задаваемым тегом `<H1>`). В секции заголовка помещается служебная информация, используемая Web-обозревателем для внутренних нужд. Среди этой служебной информации может присутствовать *название* Web-страницы, показываемое в заголовке окна Web-обозревателя; оно задается парным тегом `<TITLE>`. Собственно, секция заголовка нашей Web-страницы только ее название — "Web-страница" — и содержит.

И секция заголовка, и секция тега страницы находятся внутри парного тега `<HTML>`. Этот тег находится на самом высшем (нулевом) уровне вложенности и не имеет родителя.

## Гиперссылки

Не только и не столько Web-страницы прославили Интернет. Свою роль внесло еще одно замечательное изобретение, буквально связавшее разрозненные документы в настоящую паутину. Благодаря этому изобретению мы можем с легкостью перемещаться по страницам, ведь любой уголок Всемирной сети находится от нас на расстоянии щелчка мыши.

Это *гиперссылки* — особые связи, ведущие от одной Web-страницы к другой. (Часто их называют просто *ссылками*.) Именно по ним мы щелкаем мышью, если хотим перейти на другую страницу.

Гиперссылки создаются с помощью особого парного тега `<A>` и имеют следующий вид:

```
<A HREF="http://www.somesite.ru/pages/page125.html">Страница №125</A>
```

Здесь мы столкнулись с так называемыми *атрибутами* — дополнительными параметрами тегов. В частности, тег `<A>` содержит атрибут `HREF`, который задает интернет-адрес страницы, на которую будет выполнен переход при щелчке на гиперссылке (*целевой* страницы). Сам интернет-адрес указывается в *значении атрибута* `HREF` — после знака равенства и в двойных кавычках.

Атрибут `HREF` обязательно должен присутствовать в теге `<A>`, задающем гиперссылку, — в данном случае это *обязательный* атрибут. (Как мы выясним в *главе 3*, тег `<A>` может задавать не только гиперссылки, и в этом случае атрибут `HREF` вообще не нужен.)

А теперь давайте создадим две простейшие Web-странички и свяжем их гиперссылкой. В качестве первой страницы мы возьмем уже написанную нами ранее и сохраненную в файле `1.2.htm` (рис. 1.2). Только добавим в этот файл следующий фрагмент кода (понятно, что добавить его нужно в секцию тела страницы):

```
<P><A HREF="1.4.htm">Сведения об авторе</A></P>
```

Сохраним исправленную страницу в файле `1.3.htm`.

HTML-код второй страницы приведен далее. Вряд ли его стоит комментировать.

```
<HTML>
  <HEAD>
    <TITLE>Сведения об авторе</TITLE>
  </HEAD>
  <BODY>
    <P>Эту страничку написал я. И очень горд этим!</P>
  </BODY>
</HTML>
```

Сохраним ее в файле `1.4.htm`.

Теперь откроем в Web-обозревателе файл 1.3.htm и щелкнем по гиперссылке **Сведения об авторе**. В окне Web-обозревателя появится страница со сведениями об авторе, сохраненная в файле 1.4.htm.

А теперь сделаем небольшой фокус. Откроем в Блокноте страницу 1.3.htm и добавим в HTML-код, задающий гиперссылку, еще один атрибут (он выделен полужирным шрифтом):

```
<P><A HREF="1.4.htm" TARGET="_blank">Сведения об авторе</A></P>
```

Это атрибут `TARGET`, задающий *цель гиперссылки*. Цель гиперссылки задает, где будет открыта целевая Web-страница. Так, если атрибуту `TARGET` присвоить значение `"_blank"`, целевая страница будет открыта в новом окне Web-обозревателя. Чтобы задать обычное поведение гиперссылки (целевая страница открывается в том же окне), нужно присвоить атрибуту `TARGET` значение `"_self"` или вообще убрать данный атрибут из тега `<A>`.

В отличие от атрибута `HREF`, атрибут `TARGET` не является обязательным. Он так и называется — *необязательный* атрибут.

Сохраним измененную Web-страницу и повторно откроем ее в Web-обозревателе, после чего щелкнем по гиперссылке. На экране появится новое окно Web-обозревателя, в котором мы увидим страницу со сведениями об авторе. Оба этих окна показаны на рис. 1.3.

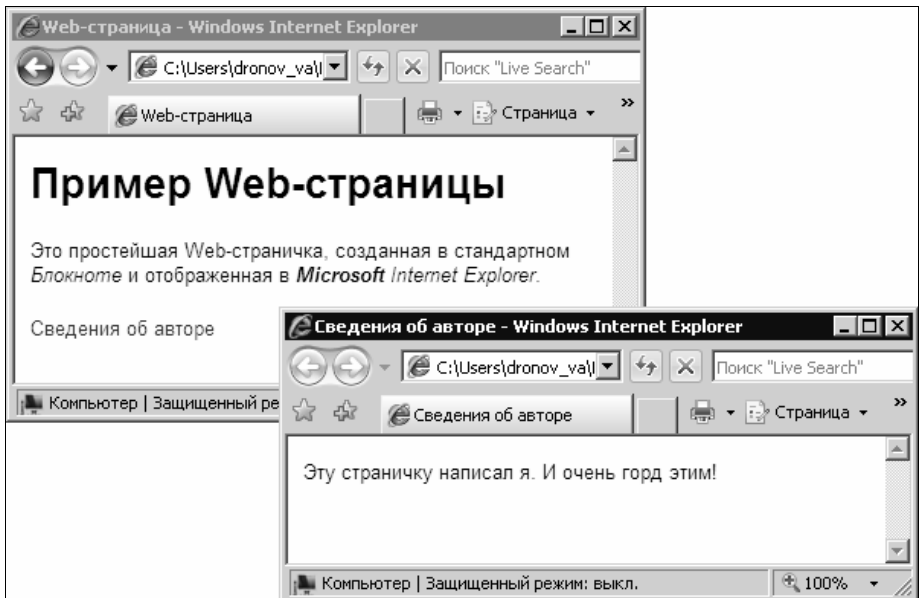


Рис. 1.3. Два окна Web-обозревателя, в которые загружены разные Web-страницы

## Интернет-адреса в WWW

Ранее мы рассмотрели два примера гиперссылок. Интернет-адреса целевых страниц (и любых других файлов) в них задаются двумя разными способами. Давайте их рассмотрим.

Вот первая гиперссылка. В ней присутствует такой интернет-адрес:

**<http://www.somesite.ru/pages/page125.html>**

Он содержит и интернет-адрес Web-сервера, и путь файла, который нам требуется (*целевого* файла). Поэтому он называется *полным*. Полные интернет-адреса используются, если нужно создать гиперссылку, указывающую на файл, что входит в состав другого сайта.

Однако если гиперссылка указывает на страницу, входящую в состав того же сайта, что и открытая в данный момент в Web-обозревателе (*текущая*), будет сподручнее использовать *сокращенный* интернет-адрес. Такие интернет-адреса не содержат интернет-адреса Web-сервера (он и так известен Web-обозревателю), а только имя нужного файла. Они бывают двух типов, которые мы сейчас рассмотрим.

Сокращенные интернет-адреса первого типа задают путь к целевому файлу относительно корневой папки сайта. Они содержат в своем начале символ слэша (/), который и говорит Web-серверу, что путь нужно отсчитывать относительно корневой папки.

Например, интернет-адрес

**[/page3.html](#)**

указывает на файл `page3.html`, хранящийся в корневой папке сайта. А интернет-адрес

**[/articles/article1.html](#)**

указывает на файл `article1.html`, хранящийся в папке `articles`, вложенной в корневую папку сайта.

Такие интернет-адреса называются *абсолютными* и используются, если нужно создать гиперссылку на файл, хранящийся в "глубине" сайта (скажем, в другой папке, нежели файл текущей страницы).

Сокращенные интернет-адреса второго типа задают путь к целевому файлу относительно файла текущей Web-страницы. Они не содержат в начале символа слэша — и в этом их важное отличие от абсолютных интернет-адресов.

Рассмотрим несколько примеров подобных интернет-адресов.

**[1.4.html](#)**

Этот интернет-адрес указывает на Web-страницу, хранящуюся в файле `1.4.html` в той же папке, что и файл текущей страницы. Собственно, этот интернет-адрес взят из второй гиперссылки, которую мы рассмотрели ранее.

### chapter3/page1.html

Этот интернет-адрес указывает на страницу page1.html, хранящуюся в папке chapter3, вложенной в папку, в которой хранится текущая Web-страница.

### ../article3.html

А этот интернет-адрес указывает на страницу article3.html, хранящуюся в папке, в которую вложена папка, где хранится текущая Web-страница. (Обратим внимание на две точки в начале пути — так обозначается папка предыдущего уровня вложенности.)

Такие интернет-адреса называются *относительными*. Они используются, если нужно создать гиперссылку на файл, хранящийся в той же папке, что и текущая страница, одной из вложенных в нее папок или папке предыдущего уровня вложенности.

#### На заметку

Во многих случаях лучше поэкспериментировать с разными интернет-адресами, чтобы выяснить, какой именно подойдет — абсолютный или относительный.

#### Внимание!

В Web-страницах, которые не должны быть опубликованы на Web-серверах, а будут открываться с дисков клиентских компьютера, следует использовать только относительные интернет-адреса. Дело в том, что файловая система компьютера не знает, какую папку считать корневой, поэтому не сможет правильно интерпретировать абсолютные интернет-адреса.

## Каскадные таблицы стилей CSS

Возможности HTML по оформлению текста, скажем прямо, не впечатляют. Всего-то и можем мы, что выделить текст полужирным шрифтом (тег `<STRONG>`) или курсивом (тег `<EM>`). Плюс еще несколько тегов, нам пока не знакомых, позволяют подчеркнуть текст, сделать отступ слева и пр. Все равно негусто...

Что же делать? Отказаться от всех красотостей? Как говорится в популярной рекламе, есть способ лучше. Мы можем использовать *каскадные таблицы стилей* (или просто *таблицы стилей*). По-английски они называются *CSS* (Cascading Style Sheets).

Каскадные таблицы стилей не относятся к языку HTML. Для их написания используется другой язык, который так и называется — *CSS*. Сейчас мы с ним познакомимся.

Пусть, например, нам нужно выделить какой-то фрагмент текста красным шрифтом, чтобы привлечь к нему внимание. Используя CSS, мы напишем такой HTML-код:

```
<P STYLE="color: red;">Внимание!</P>
```

Здесь мы использовали необязательный атрибут `STYLE`, поддерживаемый всеми видимыми тегами HTML. В качестве значения этого атрибута мы задали *стиль* абзаца, содержащий *атрибут стиля* `color`, задающий цвет текста. А *значение* этого атрибута стиля равно "red" — красный. В результате абзац "Внимание!" будет выделен красным цветом.

Отметим, что после значения атрибута стиля `color` стоит знак точки с запятой. Его следует указать, иначе некоторые Web-обозреватели могут не отобразить страницу правильно.

Стиль, который помещается прямо в тег с помощью атрибута `STYLE`, называется *встроенным*. Пожалуй, это самая простая разновидность стилей (а всего их несколько, в чем мы убедимся в *главе 11*, когда будем говорить о стилях CSS подробно).

Можно поступить по-другому. А именно, создать полновесную таблицу стилей. Она формируется с помощью парного тега `<STYLE>` и помещается в секцию заголовка Web-страницы.

```
<HEAD>
  <STYLE>
    .redtext { color: red; }
  </STYLE>
</HEAD>
```

Так будет выглядеть наша таблица стилей (выделена полужирным шрифтом). Мы определили в ней всего один стиль — `redtext`. Этот стиль называется *стилевым классом* и может быть применен (*привязан*) к любому тегу. Обратим внимание, что он обязательно должен иметь уникальное имя, по которому его будет искать Web-обозреватель, и имя это обязательно должно предвостанавливаться точкой.

Теперь, чтобы привязать созданный нами стиль к нужному тегу, применим необязательный атрибут `CLASS`, который также поддерживается всеми видимыми тегами HTML:

```
<P CLASS="redtext">Внимание!</P>
```

Видно, что имя стилевого класса указывается в значении атрибута `CLASS`. Причем указывается уже без точки.

Если нам нужно выделить красным цветом не весь абзац, а только его фрагмент, воспользуемся парным тегом `<SPAN>`. Этот тег служит только для того, чтобы выделить фрагмент содержимого тега, над которым мы собираемся выполнить некие действия (например, привязать стиль):

```
<P><SPAN CLASS="redtext">Красный</SPAN> текст.</P>
```

Здесь красным цветом будет выделено только слово "Красный".

А вот более сложный пример задания стиля:

```
<P CLASS="redtext">Красный и <SPAN STYLE="font-weight:
☛bold">полужирный</SPAN> текст.</P>
```

Здесь все содержимое тега `<P>` будет выделено красным цветом, а слово "полужирный" — еще и полужирным шрифтом. Атрибут стиля `font-weight` задает степень "жирности" шрифта, а его значение "bold" — полужирное начертание.

Более подробно таблицы стилей CSS будут описаны в *главе 11*. Здесь мы ограничимся только кратким введением.

## Физическое и логическое форматирование

Рассматривая приведенные выше фрагменты HTML-кода, вы, наверно, уже удивились. В самом деле, зачем нужен атрибут стиля `font-weight`, если существует прекрасный тег HTML `<STRONG>`, который делает то же самое?

Чтобы ответить на этот вполне резонный вопрос, нужно ввести еще несколько терминов.

Что делает уже знакомый нам тег `<P>`? Он превращает свое содержимое в текстовый абзац. При этом он выводит его отдельно от других абзацев, выделяя его визуально. Если мы загрузим созданную нами ранее Web-страницу в специальный Web-обозреватель для незрячих, который читает текст Web-страниц вслух, последний перед чтением абзаца сделает паузу. Иначе говоря, с помощью тега `<P>` мы логически оформили фрагмент текста в виде отдельного абзаца.

Точно так же действуют и теги `<STRONG>` и `<EM>`. Они не просто помечают фрагмент текста полужирным шрифтом и курсивом — они логически выделяют его, придавая тексту важность. (При этом тег `<STRONG>` придает тексту бóльшую важность, чем тег `<EM>`.) Тот же "говорящий" Web-обозреватель для незрячих может выделить этот текст, скажем, интонацией.

Говорят, что форматирование, применяемое тегами `<P>`, `<STRONG>` и `<EM>` к своему содержимому, называется *логическим*. А сами эти теги носят название *тегов логического форматирования*.

Что касается стилей, то они просто применяют к тексту заданное нами форматирование, не давая ему никакого особого значения. То есть, если мы определим для абзаца какой-либо стиль, например:

```
<P STYLE="font-weight: bold">Полужирный текст.</P>
```

то фактически скажем Web-обозревателю: "Просто выведи этот абзац полужирным шрифтом, но не давай ему никакого особого значения". Мы таким образом применим к абзацу *физическое форматирование*. Именно физическим форматированием занимаются таблицы стилей CSS.

Логическое форматирование задает *структуру* Web-страницы: расположение и порядок следования абзацев и заголовков и особое значение отдельных фрагментов текста. Физическое же форматирование задает *представление* Web-страницы: каким шрифтом будет набран обычный текст абзацев, каким цветом выделить заголовки и пр.

Правила хорошего тона Web-дизайна требуют, чтобы представление Web-страницы было отделено от ее структуры. Поэтому профессиональные Web-дизайнеры стараются выносить определения стилей CSS в отдельные таблицы стилей. К тому же, HTML-код, не загроможденный определениями стилей, становится более читабельным.

Вы спросите: если есть теги HTML логического форматирования, то, может быть, есть теги, выполняющие и форматирование физическое? Да, есть. Их довольно много. Так, существует тег *физического форматирования* <B>, аналогичный тегу <STRONG>, и тег <I>, аналогичный тегу <EM>. Однако все эти теги объявлены комитетом W3C устаревшими и не рекомендованными к использованию (хотя они все еще входят в стандарт языка HTML и поддерживаются всеми Web-обозревателями). Вместо них рекомендуется использовать стили CSS.

## Будущее HTML

Язык HTML существует уже два десятка лет. Срок немалый, особенно если вспомнить, сколько лет в среднем "живут" компьютерные технологии и стандарты. Что же ожидает его в будущем?

Как мы уже знаем, последней на данный момент версией языка HTML является 4.01. Она существует уже лет десять и заметно устарела. Что идет ей на смену?

Изначально комитет W3C вообще хотел "похоронить" HTML, проча ему в преемники новый язык под названием *XHTML* (eXtensible Hypertext Markup Language, расширяемый язык гипертекстовой разметки). Этот язык основан на популярном языке описания данных *XML* (eXtensible Markup Language,

расширяемый язык разметки), позволяет расширять доступный набор тегов и считается более строгим и более подходящим для машинной обработки. К тому же, из XHTML выброшено множество устаревших тегов (а именно все теги физического форматирования), все еще поддерживаемых HTML в целях совместимости.

Однако, судя по всему, операция "Преемник" в этом случае провалилась — XHTML не снижал популярности среди Web-сообщества. Поэтому комитет W3C срочно "реанимировал" HTML, начав разработку его новой версии, которая получила номер 5.0. В настоящее время доступна только черновая спецификация, которую можно найти на соответствующей странице сайта комитета (<http://www.w3.org/html/wg/html5/>). Когда выйдет окончательная спецификация, пока неизвестно.

## Что дальше?

Вот и закончился наш краткий курс интернет-технологий. Конечно, многое здесь не описано — все, что нам будет необходимо для создания и публикации Web-страниц, будет представлено в дальнейшем. А сейчас давайте перейдем к нашей главной цели — к работе с замечательным пакетом Web-редактора Adobe Dreamweaver CS4.