

Первый язык, который мы с вами изучим, называется HTML (HyperText Markup Language, язык разметки гипертекста). Он позволяет указать, в какой части веб-страницы будет находиться тот или иной элемент: текст, таблица, изображение.

Раньше HTML применялся для форматирования текста и изменения цвета элементов страницы. Но сейчас для этих целей служат каскадные таблицы стилей (Cascading Style Sheets, CSS), о которых будет рассказано в главе 2.

После прочтения этой и следующей глав вы сможете называть себя верстальщиком сайтов, то есть человеком, выполняющим верстку HTML-документа.

Верстка — это создание структуры HTML-документа таким образом, чтобы во всех браузерах веб-страница выглядела точно так же, как и на представленном дизайнером макете.

Макет — разработанный в формате PSD вид будущей HTML-страницы, определяющий расположение элементов и изображения, которые должны на ней отображаться. Макеты формата PSD создаются в программе Adobe Photoshop.

Кроме того, в последнее время для создания макетов все чаще используется программа Adobe Fireworks. Такие макеты имеют формат PNG.

Реже макеты создают в векторных форматах AI и CDR. Еще реже в качестве макета выступает JPG-картинка или PDF-документ.

Подготавливаем шаблон

Самое сложное в любом деле — начать. При разработке сайта начало всегда одинаковое — нужно создать текстовый документ с расширением HTML, HTM или PHP, после чего внести в него общий каркас HTML-документа. Ведь HTML-документ — это не что иное, как обычный текстовый документ, но только со специальным расширением. Вы легко можете создать его в любом текстовом редакторе, даже в Блокноте.

Расширение HTML-документа

Между расширениями HTML и HTM нет разницы, поэтому вы можете использовать любое из них. А вот расширение PHP можно применять только в том случае, если вы создаете сайт на каком-либо хостинге (локальном или хостинге в Интернете) с установленным языком PHP.

Расширение PHP мы будем применять после главы 4, посвященной языку PHP. До этого все наши веб-документы будут иметь расширение HTML.

Имя HTML-документа

В принципе, HTML-документ может иметь любое имя, состоящее из латинских букв и цифр (но первой в имени должна идти буква). Но если вы хотите, чтобы HTML-документ открывался при вводе в адресной строке браузера адреса вашего сайта, нужно дать HTML-документу специальное имя.

Например, вы приобрели домен `example.com`, после чего разместили на нем набор HTML-документов. Какой HTML-документ будет открываться, если посетитель введет в адресной строке браузера URL-адрес `http://example.com` (то есть адрес сайта без указания документа, который хочет просмотреть)? В этом случае веб-сервер попытается открыть HTML-документ с именем `index.html` или `main.html`. Чаще всего главный HTML-документ называют `index.html`.

Вы, наверное, знакомы с таким понятием, как домашняя страница. На многих сайтах есть логотип со ссылкой на нее, или «хлебные крошки»¹, которые начинаются с нее. Как правило, домашняя страница — это и есть файл `index.html`, `index.htm` или `index.php`.

Итак, создайте текстовый документ с расширением HTML.

Версии HTML

Первая версия языка HTML — HTML 2.0 — была утверждена еще в 1995 году. Процесс создания сайтов в ней не подчинялся никаким правилам — разработчики могли использовать любой регистр, и сайты на HTML 2.0 грешили огромным количеством ошибок.

В 1997 году появились сразу две новых версии языка HTML — 3.2 и 4.

Еще через два года, в 1999 году, возникла версия HTML 4.01, содержащая большое количество нововведений.

А на следующий год была выпущена версия XHTML 1.0, которая долгое время использовалась для создания сайтов. И лишь недавно появилась и получила поддержку со стороны основных браузеров версия HTML5 — самая последняя версия языка разметки. Версия HTML5 является попыткой объединить XHTML 1.0 и HTML 4.01.

Чем же отличались старые версии HTML от более новых? Естественно, в новых версиях появлялись новые теги и атрибуты. Однако это не все изменения.

¹ Навигационная цепочка, где перечислены ссылки на все разделы сайта, начиная с главной страницы, которые нужно было последовательно посетить, чтобы попасть на открытую в данный момент страницу сайта.

Иногда использовавшиеся ранее теги или атрибуты признавались устаревшими и не рекомендуемыми для применения. От этого теги/атрибуты не теряли своей функциональности — иначе пострадала бы обратная совместимость. Но применять их не рекомендовалось.

Чаще всего теги и атрибуты признавались устаревшими по той причине, что для них появлялась более подходящая замена. Так стали устаревшими атрибуты изменения цвета элемента, а также тег `FONT` — на их место пришли каскадные таблицы стилей. Но изредка замены для устаревшего тега не находилось. Например, устаревший тег `CENTER` еще долгое время жил за гранью закона — разработчики вынуждены были его использовать, не находя адекватной замены.

Иногда теги, которые ранее были признаны устаревшими, в новой версии HTML опять становились рекомендуемыми для применения. Как и любая развивающаяся система, язык HTML старался соответствовать текущему времени и текущим нуждам разработчиков.

Но основная тенденция развития языка HTML заключалась в стандартизации синтаксиса и верстки HTML-документа. Из-за многочисленных условностей при верстке допускалось большое количество ошибок. Было принято решение бороться с этим. В результате синтаксис языка HTML стал более строгим. Но об этом мы поговорим далее в книге.

Вернемся к нашим дням. На данный момент последняя версия HTML имеет номер 5. HTML5 отличается от XHTML 1.0 только новыми тегами. Причем эти теги не поддерживаются старыми версиями браузеров. Под старыми версиями имеются в виду версии, которые были выпущены более двух лет назад.

Несмотря на такой большой отрезок времени, в Интернете до сих пор существует огромное количество пользователей, которые работают с устаревшими версиями браузеров. Кроме того, большинство заказчиков требуют от разработчиков сайтов совместимости верстки с браузером Internet Explorer 7.0 или Internet Explorer 8.0 (то есть чтобы в устаревшем браузере Internet Explorer 7.0 верстка отображалась точно так же, как и в более новых версиях браузера).

По этим причинам однозначно сказать, что сейчас нужно применять только HTML5, никак нельзя. И каждый разработчик для себя сам решает, какую именно версию языка HTML применять — HTML5 или XHTML. А вот более старые версии языка HTML сейчас использовать действительно не рекомендуется.

В данной книге мы начнем изучение HTML с возможностей языка XHTML. Как было сказано выше — все, что можно в XHTML, можно и в HTML5. И только в последних главах начнем применять новые возможности HTML5.

Выбираем DOCTYPE

Каждый HTML-документ должен начинаться со строки DOCTYPE. Она говорит браузеру, какую версию HTML вы планируете использовать при создании HTML-страницы.

Это очень важный момент, поскольку строка DOCTYPE влияет на большое количество мелочей: от поведения различных тегов до размера границ, расстояния по умолчанию, высоты и ширины тегов. Поэтому указывать строку DOCTYPE следует осознанно. Но поскольку мы только начали изучение языка HTML, нам остается лишь выбрать DOCTYPE последней версии HTML и учиться верстать для нее.

Если вы не укажете строку DOCTYPE или сделаете это неверно, различные браузеры будут вести себя по-разному. Браузер Opera будет считать, что вы верстаете в самой новой версии HTML, а браузеры Internet Explorer и Mozilla, наоборот, — что для самой старой версии. Соответственно, ваша веб-страница будет по-разному выглядеть в разных браузерах, что категорически недопустимо.

Итак, откройте в Блокноте HTML-файл, который мы создали ранее, и в первой строке этого файла введите строку:

```
<!DOCTYPE HTML>
```

Это DOCTYPE для HTML5. Именно его мы будем использовать в дальнейшем.

Помимо данного DOCTYPE, можно встретить следующие:

- ❑ `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">` — для стандарта HTML 4.01 Strict (строгий);
- ❑ `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">` — для стандарта HTML 4.01 Transitional (переходный);
- ❑ `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">` — для стандарта HTML 4.01 Frameset (с фреймами);
- ❑ `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">` — для стандарта XHTML 1.0 Strict (строгий);
- ❑ `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">` — для стандарта XHTML 1.0 Transitional (переходный);
- ❑ `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">` — для стандарта XHTML 1.0 Frameset (с фреймами);

❑ `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">` — для стандарта XHTML 1.1.

Как видите, запись DOCTYPE для HTML5 самая простая: в ней нет никаких непонятных адресов. И для него, как и для XHTML 1.1, нет разновидностей Transitional, Frameset и Strict.

От Transitional, Frameset и Strict также зависит, как будут выводиться те или иные элементы языка HTML, а также какие из этих элементов будут считаться устаревшими.

- ❑ Разновидность Transitional, как следует из названия, является переходным вариантом от прошлой версии языка HTML к новой. Если в новой версии HTML какие-то теги были признаны устаревшими, то в стандарте Transitional они по-прежнему остаются рекомендуемыми для применения. Иначе говоря, при использовании варианта Transitional сайт, скорее всего, будет выводиться так же, как и при применении более старой версии HTML.
- ❑ Разновидность Frameset аналогична Transitional, кроме того, в Frameset также разрешено применять теги для создания фреймов.
- ❑ Разновидность Strict используется в том случае, если код документа полностью соответствует выбранной версии HTML. В Strict оформление и содержание полностью разделены между HTML и CSS.

Если по каким-то причинам вы не можете применить `<!DOCTYPE HTML>`, то лучше всего использовать DOCTYPE предыдущей версии HTML: `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">`.

Запоминать, как пишется DOCTYPE, не стоит. Лучше создать шаблон HTML-документа с нужным DOCTYPE, после чего использовать этот шаблон в своих проектах.

Подготавливаем HTML-шаблон

После того как вы определились с нужным DOCTYPE, следует составить базовую структуру HTML-документа. В HTML все теги имеют иерархическую систему. Они могут быть вложены только в определенные теги.

Так, самым первым тегом в HTML-документе должен быть HTML, объявление которого идет сразу после DOCTYPE. В этом теге могут находиться только теги HEAD или BODY. Сам документ должен располагаться внутри тега BODY, а информация, описывающая документ, — в теге HEAD.

Если вы не создаете страницу с фреймами, то базовая структура всегда будет одинакова. Для HTML5 базовая структура HTML-документа представлена в листинге 1.1.

Листинг 1.1. Базовая структура HTML-документа версии HTML5

```
<!DOCTYPE HTML>
<html>
<head>
  <title>Заголовок</title>
  <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
</head>
<body>
</body>
</html>
```

Для XHTML базовая структура отличается наличием других атрибутов (листинг 1.2).

Листинг 1.2. Базовая структура HTML-документа версии XHTML 1.0 Strict

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://
www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
  <title>Заголовок</title>
  <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
</head>
<body>
</body>
</html>
```

Сохраняем HTML-документ

Есть одна особенность, о которой не следует забывать, если вы создаете HTML-документ в кодировке UTF-8, то есть если вы указали в коде следующий тег:

```
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
```

Если ваша HTML-страница должна работать в кодировке UTF-8, то и HTML-документ нужно сохранить в этой кодировке. Для этого в Блокноте выберите пункт меню **Файл** ▶ **Сохранить как** и в раскрывающемся списке **Кодировка** диалога **Сохранить** как укажите пункт **UTF-8** (рис. 1.1).

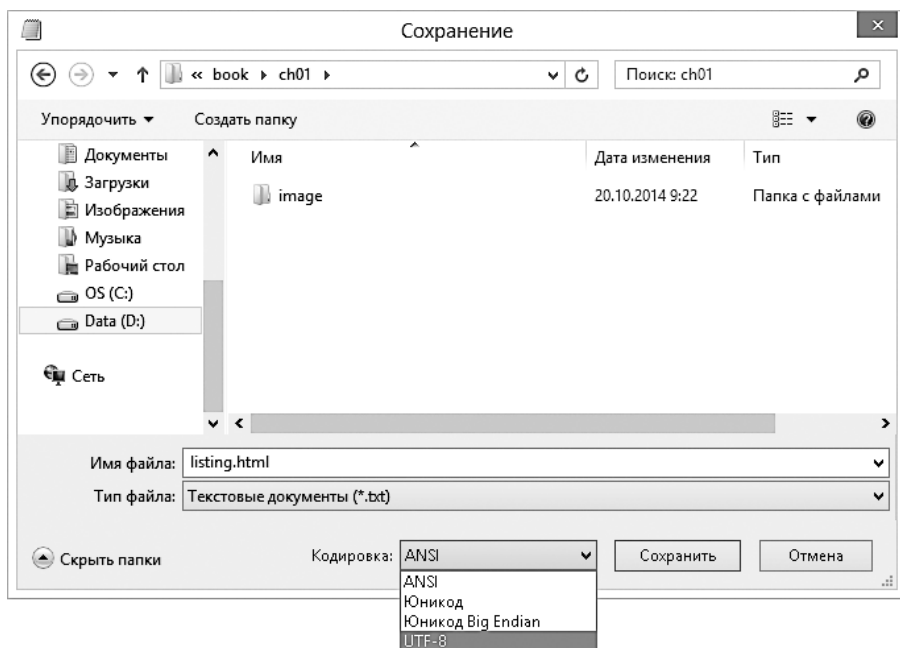


Рис. 1.1. Сохраняем HTML-документ в кодировке UTF-8

Если вы этого не сделаете, то русские символы в вашем документе при просмотре через браузер будут отображаться в виде «кракозябров».

Просматриваем наш документ

После того как HTML-документ сохранен, его можно просмотреть в любом доступном браузере (рис. 1.2). Если вы верстаете на заказ, проверять правильное отображение веб-страницы следует во всех популярных браузерах: Opera, Mozilla Firefox, Safari, Google Chrome, «ЯндексБраузер» и Internet Explorer версий, начиная с 7 или 8, поскольку одна и та же страница может выглядеть по-разному, в зависимости от браузера, в котором вы ее просматриваете.



ПРИМЕЧАНИЕ

В приложении 2 дополнительных материалов к книге приведена информация о том, какие теги работают в популярных версиях браузеров.