**Николай Прохоренок**
**Владимир Дронов**

# HTML JavaScript PHP и MySQL

## Джентльменский набор Web-мастера

## 5-е издание

Санкт-Петербург
«БХВ-Петербург»
2019

УДК 004.43+004.738.5
ББК 32.973.26-018.1
П84

**Прохоренок, Н. А.**

П84 HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера. — 5-е изд., перераб. и доп. / Н. А. Прохоренок, В. А. Дронов. — СПб.: БХВ-Петербург, 2019. — 912 с.: ил. — (Профессиональное программирование)

ISBN 978-5-9775-3986-9

Рассмотрены вопросы создания интерактивных Web-сайтов с помощью HTML, JavaScript, PHP и MySQL, форматирования Web-страниц при помощи CSS. Даны основы PHP и примеры написания типичных сценариев. Описаны приемы работы и администрирования баз данных MySQL при помощи PHP и программы phpMyAdmin. Особое внимание уделено созданию программной среды на компьютере разработчика и настройке Web-сервера Apache.

В 5-м издании содержится описание возможностей, предлагаемых PHP 7.2, новых инструментов JavaScript (включая рисование на холсте, средства геолокации и локальное хранилище данных) и всех нововведений, появившихся в актуальных на данный момент версиях HTML, CSS, Apache, MySQL и технологии AJAX.

Электронный архив содержит листинги примеров, а также руководство по созданию динамического сайта.

*Для Web-разработчиков*

УДК 004.43+004.738.5
ББК 32.973.26-018.1

**Группа подготовки издания:**

| | |
|---|---|
| Руководитель проекта | *Евгений Рыбаков* |
| Зав. редакцией | *Екатерина Сависте* |
| Компьютерная верстка | *Ольги Сергиенко* |
| Дизайн обложки | *Марины Дамбиевой* |

Подписано в печать 31.10.18.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 73,53.
Тираж 1500 экз. Заказ №
"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20.

Отпечатано с готового оригинал-макета
ООО "Принт-М", 142300, М.О., г. Чехов, ул. Полиграфистов, д. 1

ISBN 978-5-9775-3986-9

# Оглавление

# Введение

Если вы хотите научиться своими руками создавать сайты, свободно владеть HTML, CSS, JavaScript, PHP и MySQL, то эта книга для вас. Большинство подобных книг предлагают изучение или только клиентских технологий (HTML, CSS, JavaScript), или только серверных (PHP, MySQL). Но разделять эти технологии нельзя, т. к. они могут существовать только совместно, а значит, и изучать их нужно как единое целое.

Все главы книги расположены в порядке возрастания уровня сложности материала. Если вы начинающий Web-мастер, то книгу следует изучать последовательно, главу за главой. Если же материал какой-либо из глав был вами изучен ранее, то можно сразу переходить к следующей главе.

Что же можно создать с использованием описываемых технологий? Давайте кратко рассмотрим возможности этих технологий, а также содержание глав книги.

Язык разметки HTML, рассматриваемый в *главе 1*, позволяет задать местоположение элементов Web-страницы в окне Web-браузера. С помощью HTML можно отформатировать отдельные символы или целые фрагменты текста, вставить изображение, таблицу или форму, создать панель навигации с помощью карт-изображений, разделить окно Web-браузера на несколько областей, вставить гиперссылку и многое другое. А новая версия языка HTML — HTML 5 — даже позволяет поместить на страницу аудио- или видеоролик, который будет воспроизводиться самим Web-браузером, без необходимости устанавливать какие бы то ни было плагины.

При помощи каскадных таблиц стилей (CSS), о которых идет речь в *главе 2*, можно задавать точные характеристики практически всех элементов Web-страницы. Это позволяет контролировать внешний вид Web-страницы в окне Web-браузера и приближает возможности Web-дизайна к настольным издательским системам. Разработчик может указать параметры шрифта, цвет текста и фона, выравнивание, создать рамку и расположить элементы на странице произвольным образом. Новая версия CSS — CSS 3 — также предоставляет инструменты для задания градиентного фона, теней у текста и самого элемента страницы и даже для создания анимации.

У Web-страниц, созданных с использованием HTML и CSS, есть существенный недостаток — они являются статическими, т. е. не могут меняться, реагируя на действия пользователя. Внедрение в HTML-код программ, написанных на языке JavaScript, позволит «оживить» Web-страницу, сделать ее интерактивной или, дру-

гими словами, заставить взаимодействовать с пользователем. С помощью JavaScript можно обрабатывать данные формы до отправки на сервер, получать информацию о Web-браузере пользователя и его мониторе и соответствующим образом менять форматирование страницы, изменять любые элементы HTML-документа в ответ на какое-либо событие, создавать на Web-странице часы, показывающие текущее время с точностью до секунды, скрывать или отображать элементы Web-страницы и выполнять многие другие действия. Как все это сделать, рассказано в *главе 3*.

*Глава 4* повествует о том, как установить и настроить специальное программное обеспечение для тестирования сайтов: Web-сервер Apache, среду для выполнения серверных скриптов, написанных на языке PHP, и сервер баз данных MySQL. С его помощью можно проверить работоспособность создаваемого сайта непосредственно на своем компьютере, еще до его публикации в Интернете.

Огромные возможности открывают серверные технологии, среди которых для целей этой книги выбран язык программирования PHP. Используя его, можно изменять получаемый Web-браузером HTML-код в зависимости от вводимых пользователем данных, типа и версии установленного Web-браузера и других факторов. Большое количество расширений и готовых программных продуктов, а также легкость освоения, сделали PHP очень популярным языком программирования для Интернета. С помощью PHP можно работать с файлами и каталогами, обрабатывать данные формы на сервере, рассылать письма, загружать файлы на сервер, создавать для каждого пользователя личный кабинет, программировать гостевые книги, форумы, блоги, интернет-магазины и многое другое. Писать программы на PHP мы научимся в *главе 5*.

На сегодняшний день ни один крупный портал не обходится без использования баз данных. В Web-разработках чаще всего применяется быстрая и обладающая большими возможностями система управления базами данных (СУБД) MySQL. С помощью MySQL можно эффективно добавлять, изменять и удалять данные, получать нужную информацию по запросу. Работа с MySQL, в том числе с базами данных этого формата из программ, написанных на PHP, обсуждается в *главе 6*.

В *главе 7* описывается технология AJAX, позволяющая программно подгружать с сервера произвольные данные без перезагрузки самой страницы. Применение технологии AJAX позволяет значительно расширить функциональность создаваемых сайтов.

В сопровождающем книгу электронном архиве, размещенном на сайте издательства «БХВ-Петербург» (см. *приложение*), собраны следующие материалы:

- описание процесса разработки полнофункционального Web-сайта с использованием всех изученных технологий. Это каталог сайтов, включающий личный кабинет для пользователей с защитой средствами PHP, а также личный кабинет для администратора, защищенный средствами сервера Apache;
- все листинги, встречающиеся в тексте книги. Настоятельно рекомендуем обязательно рассматривать все примеры из книги и вначале самостоятельно набирать код. При наборе вы создадите множество ошибок. Но именно умение находить эти ошибки сделает из вас настоящего Web-мастера.

Авторы желают вам приятного чтения и надеются, что эта книга станет верным спутником в вашей программистской практике.

# ГЛАВА 1

# Основы HTML 5. Создаем дизайн сайта

## 1.1. Первые шаги

*HTML* (*HyperText Markup Language*) — это язык разметки документа, описывающий форму отображения информации на экране компьютера. В настоящее время язык существует в нескольких версиях: HTML 4.01, XHTML и HTML 5.

Версию языка HTML 5 поддерживают сейчас следующие Web-браузеры:

- Microsoft Internet Explorer — начиная с версии 9;
- Mozilla Firefox — 4.0;
- Google Chrome — 6.0;
- Opera — 11.1;
- Apple Safari — 5.0.

Как можно видеть, HTML 5 поддерживается уже всеми современными Web-браузерами, поэтому мы будем изучать именно его, без оглядки на предыдущие версии.

Для новых тегов и параметров, появившихся в HTML 5, мы будем явно указывать версию языка. Поэтому, если у вас есть большое желание поддерживать старые версии Web-браузеров, то теги и параметры, обозначенные версией HTML 5, лучше не использовать или предусматривать стилизацию новых тегов — например, описывать их как блочные с помощью CSS-атрибута `display` (`display: block`) и/или создавать элементы в JavaScript.

> ***Примечание***
>
> Получить полную информацию о текущей поддержке тегов и параметров Web-браузерами можно на сайте **https://caniuse.com/**.

### 1.1.1. Основные понятия

При создании документа часто приходится выделять какую-либо часть текста полужирным шрифтом, изменять размер или цвет шрифта, выравнивать текст по центру страницы и т. д. В текстовом редакторе для этого достаточно выделить

нужный фрагмент и применить к нему форматирование. Например, чтобы пометить текст курсивом, нужно выделить его и нажать кнопку **Курсив**. На языке HTML тот же эффект достигается следующей строкой кода:

```
<i>Текст</i>
```

Комбинация символов `<i>` указывает, что текст надо выделить, начиная с этого места, а `</i>` отмечает конец выделенного фрагмента.

Комбинации символов `<i>` и `</i>` принято называть *тегами*. С помощью тегов описывается вся структура документа. Теги выделяются угловыми скобками `<` и `>`, между которыми указывается имя тега. Большинство тегов являются парными, т. е. состоят из открывающего тега (`<i>`) и соответствующего ему закрывающего (`</i>`). Закрывающий тег отличается наличием косой черты (`/`) перед его именем. Есть также теги, вообще не имеющие закрывающего тега, — например, тег переноса строки `<br>`.

Некоторые теги могут иметь *параметры* (иногда их называют *атрибутами*). Параметры указываются после имени тега через пробел в формате `параметр="значение"`. Если параметров несколько, то они приводятся через пробел:

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

В этом примере параметру `http-equiv` тега `<meta>` присвоено значение `Content-Type`, а параметру `content` — значение `text/html; charset=utf-8`.

Теги могут вкладываться друг в друга:

```
<p><i>Правильно</i></p>
```

При вложении тегов необходимо соблюдать последовательность их закрытия. Например, такой код использовать нельзя:

```
<p><i>Ошибка</p></i>
```

Теги бывают *блочными* и *строчными*. Блочный тег занимает всю ширину родительского элемента. Примером блочного тега является тег `<p>`, описывающий абзац. Строчный тег может быть расположен только внутри блочного тега. Примером строчного тега является тег `<i>`, указывающий, что фрагмент нужно выделить курсивом. Так как блочный тег `<p>` не может быть расположен внутри строчного тега `<i>`, следовательно, так вкладывать теги также нельзя:

```
<i><p>Ошибка</p></i>
```

Открывающие и закрывающие теги со всем их содержимым мы будем называть *элементами*. Например, в следующем примере описан элемент `p`:

```
<p>Текст абзаца</p>
```

> ***Примечание***
>
> В HTML названия тегов и параметров можно записывать в любом регистре, а в языке XHTML только в нижнем регистре. Рекомендуем всегда записывать теги в нижнем регистре. Именно так мы и будем записывать их в дальнейших примерах.

Просматривать HTML-документы можно с помощью специальных программ, которые называют *Web-браузерами*. Web-браузеры отображают документы с форматированием, выполненным на основе исходного кода, описывающего структуру документа.

Результат интерпретации HTML-документа, отображаемый в окне Web-браузера, называется *Web-страницей*. В отличие от HTML-документа Web-страница может содержать не только текст, но и графику, видео, звуковое сопровождение, может реагировать на действия пользователя и т. д. Кроме того, Web-страница может быть результатом интерпретации сразу нескольких HTML-документов.

Документы в формате HTML имеют расширение html или htm.

## 1.1.2. Редактор CKEditor

Прежде чем изучать язык HTML, советуем установить на компьютер редактор CKEditor. Этот редактор написан на языке программирования JavaScript и работает в Web-браузере.

Скачать CKEditor можно со страницы **https://ckeditor.com/ckeditor-4/download/**. Выберите полную версию (**Full Package**) и после закачки и распаковки архива запустите файл index.html (расположен в папке ckeditor\samples\) двойным щелчком левой кнопки мыши на значке файла. Можно также воспользоваться пунктом меню **Открыть** в Web-браузере.

На рис. 1.1 можно увидеть, как выглядит редактор CKEditor, запущенный в Web-браузере Firefox. Если вы раньше работали с текстовым редактором Microsoft Word, то большинство кнопок на панели инструментов будет вам знакомо. Принцип работы в CKEditor точно такой же, как и в Word. После ввода текста и его форматирования редактор автоматически генерирует HTML-код. Посмотреть исходный HTML-код можно, нажав кнопку **Источник** на панели инструментов (рис. 1.2).

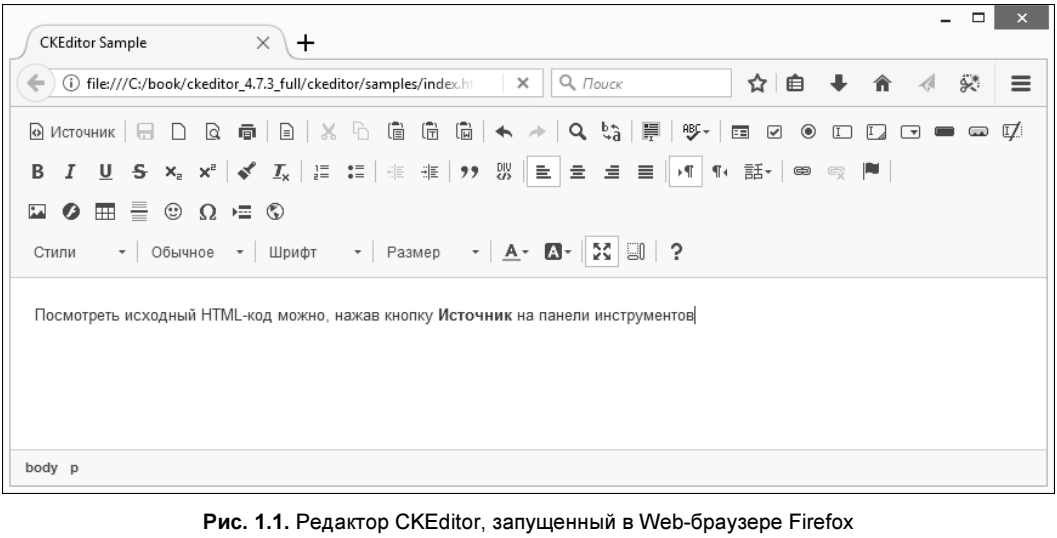


**Рис. 1.1.** Редактор CKEditor, запущенный в Web-браузере Firefox

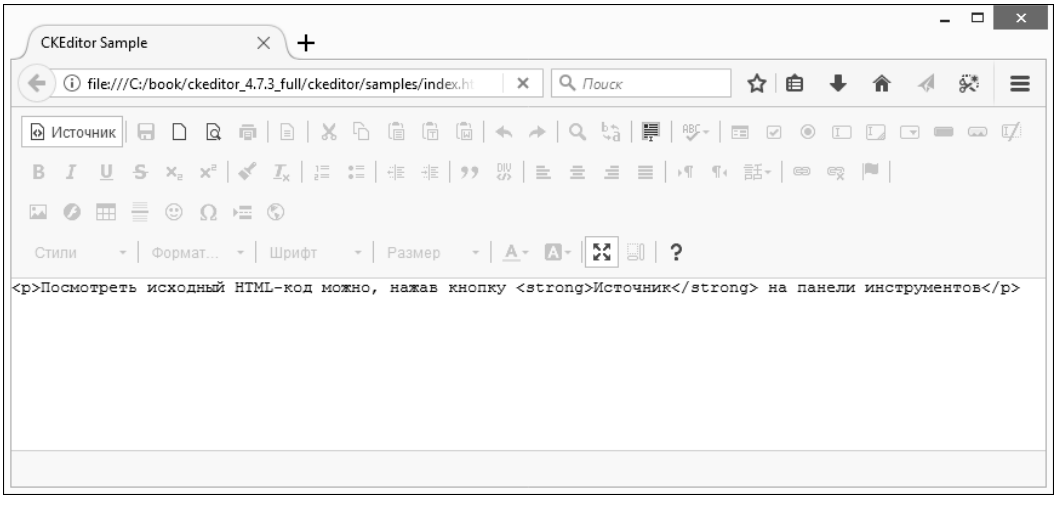


**Рис. 1.2.** Результат нажатия кнопки **Источник** в редакторе CKEditor

Следует заметить, что при изменении исходного HTML-кода автоматически изменяется и внешний вид документа.

## 1.1.3. Редактор Notepad++

Для создания HTML-документа можно воспользоваться любым текстовым редактором, например Блокнотом. Однако мы будем работать с кодировкой UTF-8, а Блокнот при сохранении в этой кодировке добавляет *метку порядка* байтов (от англ. Byte Order Mark, BOM). Эта метка может стать причиной ошибок при формировании заголовков ответа сервера из программы, написанной на языке PHP. Этот язык мы будем изучать далее, поэтому в качестве редактора кода на протяжении всего обучения воспользуемся программой Notepad++. Она позволяет корректно работать как с однобайтовой кодировкой windows-1251, так и с многобайтовой кодировкой UTF-8, а также имеет подсветку синтаксиса HTML, JavaScript, PHP и др.

Скачать программу Notepad++ можно абсолютно бесплатно со страницы **https://notepad-plus-plus.org/**. Из двух вариантов (архив и инсталлятор) советуем выбрать именно инсталлятор, т. к. при установке можно будет указать язык интерфейса программы. Установка Notepad++ предельно проста и в комментариях не нуждается.

Чтобы открыть какой-либо файл на редактирование, в меню **Файл** выбираем пункт **Открыть** или щелкаем правой кнопкой мыши на значке файла в Проводнике Windows и из контекстного меню выбираем пункт **Edit with Notepad++**.

> ***Примечание***
>
> Вместо Notepad++ можно воспользоваться редакторами PHP Expert Editor, Aptana Studio или NetBeans. Эти редакторы помимо подсветки синтаксиса предоставляют множество дополнительных функций. Тем не менее для быстрого редактирования файла удобнее пользоваться Notepad++.

## 1.1.4. Первый HTML-документ

Попробуем создать наш первый HTML-документ. Для его создания, как уже отмечалось, можно воспользоваться любым текстовым редактором. Мы же открываем редактор Notepad++ и создаем новый документ (меню **Файл | Новый**). В меню **Кодировки** устанавливаем флажок **Кодировать в UTF-8 (без BOM)**. Набираем код, представленный в листинге 1.1, а затем в меню **Файл** выбираем пункт **Сохранить как**. В открывшемся окне выбираем папку, например, C:\book, в строке **Имя файла** вводим `test.html`, а из списка **Тип файла** выбираем пункт **All types (*.*)**. Нажимаем кнопку **Сохранить**. После сохранения файла в заголовке окна редактора должен появиться путь: C:\book\test.html. Если окажется, что после фрагмента html стоит точка и какое-либо другое расширение, например txt, то при сохранении была допущена ошибка. Такая ошибка очень часто возникает при сохранении файла в редакторе Блокнот.

**Листинг 1.1. Первый HTML-документ**

```
<!DOCTYPE html>
<html lang="ru">
<head>
   <meta charset="utf-8">
   <title>Заголовок страницы</title>
</head>
<body>
<p>Привет, мир!</p>
</body>
</html>
```

Чтобы посмотреть результат, нужно открыть HTML-документ в Web-браузере. В этом издании книги мы будем пользоваться Web-браузером Mozilla Firefox (на момент подготовки издания текущей была версия 56). Можно пользоваться и другим Web-браузером, но процесс работы с другим Web-браузером будет отличаться, и вы станете путаться, не зная, что делать. Поэтому на время обучения установите Web-браузер Mozilla Firefox последней версии, а уже после обучения можете вернуться к своему любимому Web-браузеру.

Итак, запускаем Web-браузер Mozilla Firefox и переходим в главное меню. Выбираем пункт **Открыть файл**, в открывшемся окне находим наш файл test.html и нажимаем кнопку **Открыть**. Если все сделано правильно, то в окне Web-браузера вы увидите надпись «Привет, мир!», а в строке заголовка — надпись «Заголовок страницы». Теги в окне Web-браузера не отображаются!

> ***Примечание***
>
> Если в главном меню браузера вы не найдете пункт **Открыть файл**, то выберите пункт **Изменить**, найдите в левой части открывшегося окна пункт **Открыть файл** и перетащите его с помощью мыши в область меню. Точно таким же способом добавьте в меню пункт **Кодировка текста**. После чего нажмите кнопку **Выход из настройки** для применения изменений.

Если Web-браузер Firefox является Web-браузером по умолчанию, то открыть HTML-документ можно с помощью двойного щелчка левой кнопкой мыши на значке файла. Если это не так, то щелкаем правой кнопкой мыши на значке файла и из контекстного меню выбираем пункт **Открыть с помощью | Выбрать программу**. В открывшемся окне находим Web-браузер Mozilla Firefox и устанавливаем флажок **Использовать это приложение для всех файлов html**. Либо в настройках Web-браузера делаем его установленным по умолчанию.

Можно также в адресной строке Web-браузера набрать команду `file:///C:/book/test.html` и нажать клавишу <Enter>.

Если вместо русского текста в окне Web-браузера отобразились непонятные символы, то следует вначале проверить кодировку файла с HTML-документом. Для этого в редакторе Notepad++ отображаем содержимое файла и открываем меню **Кодировки**. Флажок должен быть установлен у пункта **Кодировать в UTF-8 (без BOM)**. Если это не так, то выбираем пункт **Преобразовать в UTF-8 (без BOM)**, сохраняем файл и пытаемся открыть его снова.

Если русские буквы все равно отображаются неправильно, то проверяем кодировку в следующей строке:

```
<meta charset="utf-8">
```

Для совместимости со старыми Web-браузерами эту строку можно записать так:

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

Если опять не помогло, то смотрим какую кодировку выбрал Web-браузер. Для этого открываем главное меню и выбираем пункт **Кодировка текста**. Флажок должен стоять у пункта **Юникод**.

Теперь попробуем изменить заголовок в окне Web-браузера. Открываем файл с HTML-документом в программе Notepad++ и изменяем строчку:

```
<title>Заголовок страницы</title>
```

на:

```
<title>Моя первая Web-страница</title>
```

Сохраняем файл (меню **Файл | Сохранить**), возвращаемся в Web-браузер и обновляем Web-страницу. Обновить страницу в Firefox можно следующими способами:

- нажатием кнопки **Обновить текущую страницу** в адресной строке;
- щелчком правой кнопкой мыши на заголовке текущей вкладки и выбором из контекстного меню пункта **Обновить вкладку**;
- нажатием на клавиатуре комбинации клавиш <Ctrl>+<R> или клавиши <F5>.

В результате надпись в строке заголовка заменится на «Моя первая Web-страница».

Таким образом, изменяя что-либо в исходном коде, можно визуально оценивать результаты произведенных действий. Алгоритм такой: открываем исходный код, вносим корректировку, сохраняем, а затем обновляем Web-страницу.

## 1.1.5. Просмотр исходного HTML-кода в Web-браузере

Чтобы посмотреть исходный HTML-код загруженной Web-страницы, следует щелкнуть правой кнопкой мыши в любом месте окна Web-браузера и выбрать из контекстного меню пункт **Исходный код страницы**. В некоторых случаях результат этого действия может быть разным. Так, если Web-страница состоит из нескольких HTML-документов, то от места щелчка зависит, код какого HTML-документа будет отображен.

Чтобы посмотреть код только какого-либо фрагмента, достаточно выделить его в окне, щелкнуть на выделении правой кнопкой мыши и из контекстного меню выбрать пункт **Исходный код выделенного фрагмента**.

Исходный HTML-код будет отображен на отдельной вкладке, причем его синтаксис подсвечивается. Если какой-либо фрагмент подсвечен красным цветом, то в нем содержится ошибка, которую нужно исправить. К ошибкам внутри HTML-кода Web-браузеры часто относятся снисходительно и исправляют их самостоятельно, но часто не так, как вам бы этого хотелось. Поэтому обращайте внимание на фрагменты, подсвеченные красным цветом — это подсказка для вас.

Web-браузеры запускают код в так называемой «песочнице». Это означает, что код внутри Web-страницы не может напрямую взаимодействовать с файловой системой компьютера. Сделано это из соображений безопасности. Поэтому, если HTML-документ опубликован в Интернете, то можно лишь созерцать его исходный код, а вот изменить его нельзя. Можно лишь скопировать его и вставить куда-либо или сохранить в файл, выбрав из контекстного меню пункт **Сохранить как**. Последнее действие выполняет лично пользователь, а не код внутри Web-страницы, поэтому ограничения «песочницы» здесь не действуют.

## 1.1.6. Инструменты разработчика

Web-браузер Mozilla Firefox содержит **Инструменты разработчика**, которые позволяют не только просматривать исходный код, но и изменять значения различных параметров тегов, атрибутов стиля и т. д. При этом все изменения сразу применяются в окне Web-браузера.

Чтобы открыть панель **Инструменты разработчика**, нужно в главном меню выбрать пункт **Разработка | Инструменты разработчика** или нажать комбинацию клавиш <Shift>+<Ctrl>+<I>. На этой панели сейчас нас интересует вкладка **Инспектор**. Поэтому, чтобы отобразить ее сразу, в главном меню выбираем пункт **Разработка | Инспектор** или нажимаем комбинацию клавиш <Shift>+<Ctrl>+<C>.

По умолчанию панель отобразится в нижней части окна. Это поведение можно изменить с помощью значков в правой части заголовка панели. Например, можно отобразить панель в отдельном окне или прикрепить ее к боковой части окна. В заголовке окна содержится также значок с изображением шестеренки (пункт **Настройки инструментов**). Щелчок на нем откроет панель с различными настройками.

На вкладке **Инспектор** отображается структура HTML-документа в виде дерева (рис. 1.3), а на вкладке **Разметка** можно увидеть блочную структуру каждого элемента (если вкладка не отображается, то нужно щелкнуть левой кнопкой мыши на значке **Развернуть панель**). При наведении указателя мыши на какой-либо элемент он подсвечивается в окне Web-браузера, что позволяет определить его местоположение визуально. Чтобы сразу перейти к элементу внутри HTML-кода, щелкаем на значке **Выбрать элемент из страницы** в заголовке панели, а затем наводим указатель мыши на элемент в окне Web-браузера и щелкаем на нем левой кнопкой мыши, — строка HTML-кода, соответствующая этому элементу, на панели **Инспектор** будет выделена.

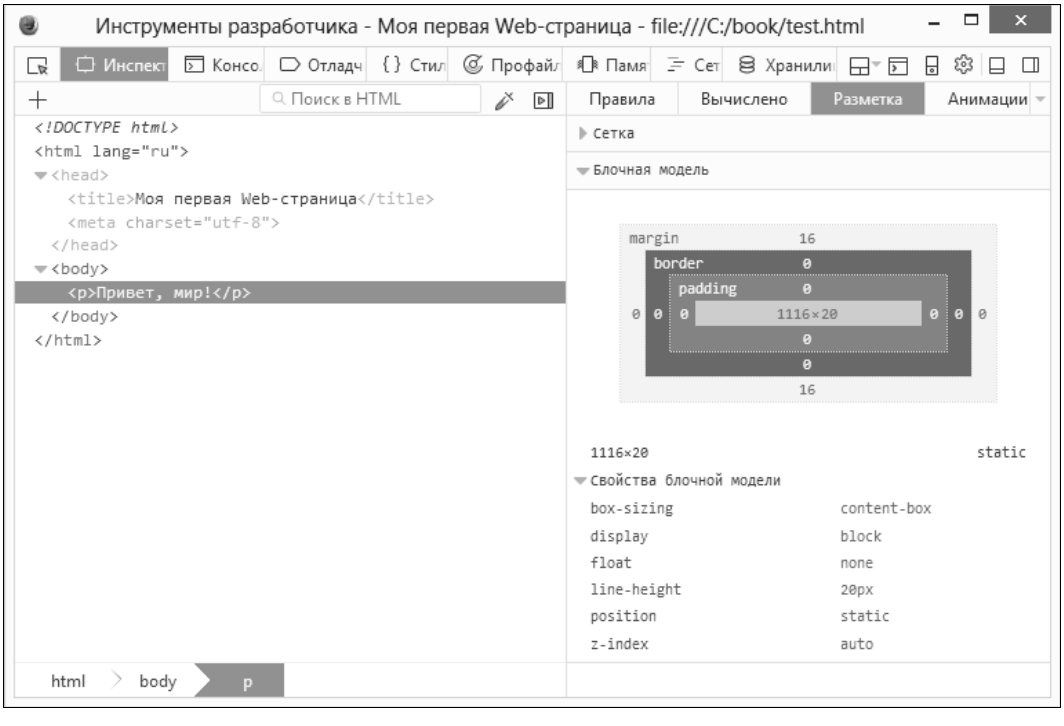


**Рис. 1.3.** Панель **Инструменты разработчика** в Web-браузере Firefox

Все значения на этой панели можно менять и сразу видеть результат в окне. Например, изменим текст абзаца. Для этого найдем элемент `p` и сделаем двойной щелчок левой кнопкой мыши на тексте — текст станет доступен для редактирования. Вводим любой другой текст, нажимаем клавишу <Enter> и любуемся проделанной работой в окне Web-браузера.

Если нужно изменить не только текст элемента, но и поменять HTML-код, то щелкаем правой кнопкой мыши на элементе и из контекстного меню выбираем пункт **Править как HTML**. После этого станет доступным поле, в котором можно произвести изменения. Например, изменим код абзаца следующим образом:

```
<p><i>Текст выделен курсивом</i></p>
```

Точно таким же образом можно отобразить в виде текста весь HTML-код. Для этого достаточно щелкнуть правой кнопкой мыши на открывающем теге `<html>` и из контекстного меню выбрать пункт **Править как HTML**. Чтобы скопировать весь код, вначале выделяем его, а затем нажимаем комбинацию клавиш <Ctrl>+<C>, — код будет скопирован в буфер обмена, и его можно будет вставить в любое другое место — например, сохранить в файл. Аналогичное действие позволяет выполнить пункт из контекстного меню **Копировать | Внешний HTML**.

Пользоваться панелью **Инструменты разработчика** мы будем очень часто, особенно при изучении языка программирования JavaScript, поэтому способы отображения этой панели нужно знать наизусть.

### 1.1.7. Комментарии в HTML-коде

Текст, заключенный между тегами `<!--` и `-->`, Web-браузером не отображается. Заметим, что это нестандартная пара тегов, т. к. открывающий тег не имеет закрывающей угловой скобки, а в закрывающем теге отсутствует открывающая угловая скобка:

```
<!-- Текст комментария -->
```

> ***Совет***
>
> Использование комментариев в исходном коде позволит быстро найти нужный фрагмент. Это особенно важно для начинающих Web-разработчиков.

## 1.2. Структура HTML-документа

Итак, мы изучили технологию создания HTML-документов, научились сохранять, отображать и изменять исходный код. Пришла пора вернуться к языку HTML. В листинге 1.2 представлена структура, характерная для любого HTML-документа.

**Листинг 1.2. Структура HTML-документа**

```
<!DOCTYPE> <!-- Объявление формата документа -->
<html>
   <head>
      <!-- Техническая информация о документе -->
   </head>
   <body>
      <!-- Основная часть документа -->
   </body>
</html>
```

## 1.2.1. Тег <!DOCTYPE>. Объявление формата документа

Тег <!DOCTYPE> позволяет Web-браузеру определить формат файла и правильно отобразить все его инструкции. Для HTML 5 инструкция выглядит очень просто:

```
<!DOCTYPE html>
```

Допустимые форматы для HTML 4.01:

- Strict — строгий формат. Не содержит тегов и параметров, помеченных как устаревшие или не одобряемые. Объявление формата:

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

- Transitional — переходный формат. Содержит устаревшие теги в целях совместимости и упрощения перехода со старых версий HTML. Объявление формата:

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

- Frameset — аналогичен переходному формату, но содержит также теги для создания фреймов. Объявление формата:

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

Если тег <!DOCTYPE> не указан, то некоторые Web-браузеры переходят в режим совместимости. В этом режиме может отличаться тип блочной модели. Поэтому при отсутствии тега <!DOCTYPE> разные Web-браузеры могут по-разному отображать Web-страницу.

## 1.2.2. Тег <html>

Весь текст HTML-документа расположен между тегами <html> и </html>. Тег <html> может иметь параметры. В большинстве случаев эти параметры универсальные, и мы их рассмотрим позже. Сейчас же будем указывать только параметр lang, задающий код языка Web-страницы:

```
<html lang="ru">
```

Значение ru параметра lang означает русский язык.

HTML-документ состоит из двух разделов: *заголовка* (между тегами <head> и </head>) и *содержательной части* (между тегами <body> и </body>).

## 1.2.3. Раздел HEAD. Техническая информация о документе

Раздел HEAD содержит техническую информацию о странице: заголовок, ее описание и ключевые слова для поисковых машин, данные об авторе и времени создания страницы, базовом адресе страницы, кодировке и т. д.

Единственным обязательным тегом в разделе HEAD является тег <title>. Текст, расположенный между тегами <title> и </title>, отображается в строке заголовка Web-браузера или в строке заголовка вкладки. Длина заголовка не должна превышать 60 символов, иначе он полностью не поместится:

```
<title>Заголовок страницы</title>
```

> ***Совет***
>
> Очень часто текст между тегами <title> и </title> используется в результатах, выдаваемых поисковыми порталами, в качестве текста ссылки на эту страницу. Соответственно, заголовок должен максимально полно описывать содержание страницы. Не следует писать что-то вроде «Главная страница», «Первая страница» и т. п.

С помощью одинарного тега <meta> можно задать описание содержимого страницы и ключевые слова для поисковых машин. Если текст между тегами <title> и </title> используется в качестве текста ссылки на эту страницу, то описание из тега <meta> может быть отображено под ссылкой:

```
<meta name="description" content="Описание содержимого страницы">
<meta name="keywords" content="Ключевые слова через запятую">
```

Также с помощью тега <meta> можно указать кодировку текста:

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

В HTML 5 указать кодировку можно гораздо проще:

```
<meta charset="utf-8">
```

Для автоматической перезагрузки страницы через заданный промежуток времени следует воспользоваться свойством refresh тега <meta>:

```
<meta http-equiv="refresh" content="30">
```

В этом примере страница будет перезагружена через 30 секунд. Если необходимо сразу перебросить посетителя на другую страницу, то можно указать ее интернет-адрес (URL) в параметре url:

```
<meta http-equiv="refresh" content="0; url=http://mail.ru/">
```

> ***Примечание***
>
> В разделе HEAD могут быть расположены также теги <base>, <link>, <script>, <style> и некоторые другие. Эти теги мы рассмотрим по мере изучения материала.

## 1.2.4. Файл *favicon.ico*

Когда посетители добавляют сайт в **Избранное**, Web-браузеры запрашивают с сервера файл favicon.ico. Этот файл должен содержать логотип сайта в виде значка 16×16 или 32×32 пиксела. В одном файле может находиться несколько форматов значка сразу. Если файл не найден, то в журнал регистрации ошибок сервера записывается сообщение об ошибке 404 (файл не найден), а в строке в **Избранном** отображается стандартный значок Web-браузера. Если значок найден, то он отобразится в **Избранном** (рис. 1.4) и в заголовке вкладки перед названием Web-страницы.

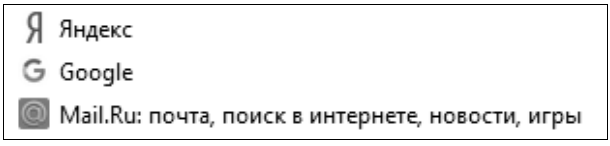


**Рис. 1.4.** Пользовательские значки в **Избранном**

Некоторые поисковые порталы (например, Яндекс) отображают значок в результатах поиска.

Файл favicon.ico по умолчанию должен находиться в корневой папке сайта и быть доступным по адресу **http://<Имя сайта>/favicon.ico**. В раздел `HEAD` HTML-документа можно добавить следующие строчки, задающие точное местоположение значка:

```
<link rel="icon" href="/favicon.ico" type="image/x-icon">
<link rel="shortcut icon" href="/favicon.ico" type="image/x-icon">
```

Файлы favicon.ico создаются в специализированных редакторах. Дополнительную информацию о файле favicon.ico и методах его создания можно получить на сайтах **http://favicon.ru/** и **http://favicon.com/**.

## 1.2.5. Файл *robots.txt*

В некоторых случаях следует запретить индексацию отдельных страниц сайта или всего сайта — например, если он предназначен только для членов семьи. Следует также учитывать, что поисковые роботы создают большой трафик, т. к. запрашивают все страницы сайта без передышки. Это обстоятельство может значительно нагрузить сервер.

Для управления индексацией конкретной страницы можно воспользоваться тегом `<meta>`:

```
<meta name="robots" content="<Индексация>, <Переход по ссылкам>">
```

В параметре `content` указывается комбинация следующих значений:

- `index` — индексация разрешена;
- `noindex` — индексация запрещена;
- `follow` — разрешено переходить по ссылкам, которые находятся на этой Web-странице;
- `nofollow` — запрещено переходить по ссылкам;
- `all` — комбинация `index` плюс `follow`;
- `none` — комбинация `noindex` плюс `nofollow`.

Приведем ряд примеров:

- индексация и переход по ссылкам разрешены:

```
<meta name="robots" content="index, follow">
```

- индексация разрешена, а переход по ссылкам запрещен:

```
<meta name="robots" content="index, nofollow">
```

- индексация и переход по ссылкам запрещены:

```
<meta name="robots" content="noindex, nofollow">
```

Но этот метод не снизит нагрузку на сервер. Чтобы этого добиться, необходимо в корневой папке сайта создать файл robots.txt (файл должен быть доступен по адресу **http://<Имя сайта>/robots.txt**). Содержимое файла выглядит следующим образом:

```
User-agent: *
Disallow: /file.html
Disallow: /user/
```

В директиве `User-agent` можно задать конкретное название робота или указать символ `*`, который означает, что приведенная информация относится ко всем роботам. Узнать название робота можно по полю `user-agent` в логах сервера.

Каждая строка `Disallow` определяет файл или каталог, который запрещен для индексации указанным роботом.

Перед индексацией сайта все роботы обязаны ознакомиться с содержимым этого файла. Даже если все страницы сайта разрешены для индексации, все равно следует создать пустой файл robots.txt и поместить его в корневой папке сайта, т. к. отсутствие этого файла приведет к ошибке 404 (файл не найден). Отсутствие файла означает, что индексация разрешена.

Чтобы запретить индексацию всего сайта, необходимо разместить файл со следующими директивами:

```
User-agent: *
Disallow: /
```

## 1.2.6. Раздел *BODY*. Основная часть документа

В этом разделе располагается все содержимое документа. Большинство тегов, рассмотренных в этой главе книги, должны находиться именно между тегами `<body>` и `</body>`.

Следует отметить, что в HTML 4.01 в формате `Strict` содержимое тега `<body>` должно быть расположено внутри блочных элементов, например: `<p>`, `<div>` или др.:

```
<p>Текст документа</p>
<div>Текст документа</div>
```

В HTML 5 этого не требуется, даже теги `<html>` и `<body>` являются необязательными. Например, следующий HTML-документ является валидным (однако так лучше не делать!):

```
<!DOCTYPE html>
<head>
   <title>Заголовок страницы</title>
```

```
   <meta charset="utf-8">
</head>
Привет, мир!
```

В HTML 4.01 в формате `Transitional` тег `<body>` имеет параметры `bgcolor` (задает цвет фона Web-страницы), `background` (позволяет задать фоновый рисунок для документа), `alink` (определяет цвет активной ссылки), `link` (устанавливает цвет еще не просмотренных ссылок), `vlink` (определяет цвет уже просмотренных ссылок), `text` (устанавливает цвет текста) и др. В HTML 5 все эти параметры отсутствуют. Вместо них следует пользоваться стилями CSS (листинг 1.3).

**Листинг 1.3. Тег <body>**

```
<!DOCTYPE html>
<html lang="ru">
<head>
   <meta charset="utf-8">
   <title>Заголовок страницы</title>
   <style>
      body {
         /* Цвет фона Web-страницы */
         background-color: green;
         /* Фоновый рисунок */
         background-image: url(photo.jpg);
         /* Цвет текста */
         color: red
      }
      /* Цвет еще не просмотренных ссылок */
      a:link { color: #0000EE }
      /* Цвет уже просмотренных ссылок */
      a:visited { color: #008000 }
      /* Цвет активной ссылки */
      a:active { color: #FF0000 }
   </style>
</head>
<body>
<p>Текст документа</p>
<p><a href="https://www.google.ru/">Текст ссылки</a></p>
</body>
</html>
```

В этом примере мы воспользовались парным тегом `<style>` и внутри него добавили различные стили как для тега `<body>`, так и для тега `<a>`. Более подробно CSS стили мы рассмотрим в следующей главе.

У тега `<body>` существуют и другие параметры, которые мы будем рассматривать по мере изучения языка.

# 1.3. Разделение Web-страницы на фрагменты

Элементы Web-страницы бывают *блочными* и *строчными*. Блочный элемент занимает всю ширину родительского элемента. Строчный элемент может быть расположен только внутри блочного элемента. С помощью блочных элементов мы можем разделить Web-страницу на логические фрагменты — например, выполнить форматирование текста с помощью заголовков и абзацев.

## 1.3.1. Заголовки

Заголовки могут иметь шесть различных размеров:

```
<hx>Заголовок</hx>
```

где `x` — число от 1 до 6.

Заголовок с номером 1 является самым крупным:

```
<h1>Самый крупный заголовок</h1>
```

Заголовок с номером 6 является самым мелким:

```
<h6>Самый мелкий заголовок</h6>
```

В HTML 4.01 в формате `Transitional` теги `<hx>` имеют параметр `align`, с помощью которого можно управлять горизонтальным выравниванием текста внутри заголовка. В HTML 5 этого параметра нет, вместо него нужно использовать атрибут стиля `text-align`:

```
<h1 style="text-align: center">Текст по центру</h1>
```

В этом примере мы воспользовались параметром `style`, который имеют почти все теги.

## 1.3.2. Разделение на абзацы

Тег `<p>` позволяет разделить текст на отдельные абзацы. При этом перед абзацем и после него добавляется пустое пространство:

```
<h1>Заголовок</h1>
<p>Абзац с выравниванием по левому краю</p>
<p style="text-align: center">Абзац с выравниванием по центру</p>
<p style="text-align: justify">Абзац с выравниванием по ширине</p>
```

Закрывающий тег `</p>` не обязателен, поэтому и такой код является валидным:

```
<h1>Заголовок</h1>
<p>Абзац 1
<p>Абзац 2
<p>Абзац 3
```

В этом случае концом абзаца считается следующий открывающий тег `<p>` или любой другой блочный элемент.

Парный тег <blockquote> создает абзац с длинной цитатой. Его блок будет иметь меньшую ширину, чем обычный абзац, и выравнивание по центру.

```
<blockquote>Длинная цитата</blockquote>
```

Тег <blockquote> может содержать параметр cite, задающий ссылку на страницу с оригинальным текстом.

## 1.3.3. Тег <*div*>

Тег  является универсальным блочным элементом. С его помощью можно сгруппировать несколько элементов страницы в один и применить к группе различные стили. Кроме того, тег  используется для блочной верстки Web-страницы (листинг 1.4).

**Листинг 1.4. Тег **

```
<!DOCTYPE html>
<html lang="ru">
<head>
   <meta charset="utf-8">
   <title>Элемент DIV</title>
</head>
<body>
   <div id="header">
      <h1>&amp;quot;Шапка&amp;quot; страницы</h1>
   </div>
   <div id="nav">
      <p><a href="page2.html">Страница 2</a></p>
      <p><a href="page3.html">Страница 3</a></p>
   </div>
   <div id="main">
      <p>Основное содержимое страницы</p>
   </div>
   <div id="footer">
      <p>&amp;quot;Подвал&amp;quot; страницы</p>
   </div>
</body>
</html>
```

## 1.3.4. Семантическая разметка в HTML 5

Итак, в листинге 1.4 мы создали блочную верстку с помощью тегов . Каждый тег содержит параметр id, задающий уникальный идентификатор элемента. Этот параметр имеют практически все теги. Благодаря этому параметру мы можем обратиться к элементу из программы на JavaScript или применить к элементу стили.

Давайте посмотрим на первый элемент:

```
<div id="header">
  <h1>&quot;Шапка&quot; страницы</h1>
</div>
```

По значению параметра `id` человек догадается о предназначении элемента, а вот поисковый робот видит только безликий элемент `div` и не догадывается о его предназначении. Чтобы это исправить, в HTML 5 были добавлены средства *семантической разметки страниц*. Под семантической разметкой подразумевается разделение содержимого страницы на части, каждая из которых имеет особое значение: заголовок страницы, панель навигации, основное содержимое, часть основного содержимого, «подвал» и др.

Перепишем предыдущий пример:

```
<header>
   <h1>&quot;Шапка&quot; страницы</h1>
</header>
```

Мы заменили безликий элемент `div` блочным элементом `header`, имеющим определенный смысл. Теперь поисковый робот знает, с чем имеет дело, и сможет правильно обработать содержимое элемента.

Для семантической разметки в HTML 5 предусмотрены следующие блочные теги:

- `<header>` — «шапка» статьи или самой страницы;
- `<nav>` — набор гиперссылок для навигации по содержимому статьи или самому сайту (панель навигации);
- `<main>` — блок основного содержимого страницы;
- `<footer>` — «подвал» статьи или самой страницы;
- `<section>` — значимая часть материала (например, раздел документа). Раздел должен содержать заголовок;
- `<article>` — независимый и самодостаточный фрагмент основного содержимого страницы: отдельная статья, запись форума, блога или комментарий;
- `<aside>` — примечание к статье, обычно располагающееся сбоку от основного текста;
- `<figure>` — иллюстрация к статье: обычное графическое изображение, аудио-, видеоролик или даже фрагмент текста;
- `<figcaption>` — подпись к иллюстрации. Может присутствовать только в теге `<figure>`.

С помощью строчного тега `<mark>` можно пометить важный фрагмент текста. Web-браузер Firefox устанавливает для такого фрагмента желтый цвет фона (как бы выделяет фрагмент текста маркером).

Пример использования семантической разметки приведен в листинге 1.5.

**Листинг 1.5. Семантическая разметка**

```
<html lang="ru">
<head>
   <meta charset="utf-8">
   <title>Семантическая разметка</title>
</head>
<body>
   <header>
      <h1>&amp;quot;Шапка&amp;quot; страницы</h1>
   </header>
   <nav>
      <p><a href="page2.html">Страница 2</a></p>
      <p><a href="page3.html">Страница 3</a></p>
   </nav>
   <section>
      <h2>Заголовок раздела</h2>
      <p>Основное содержимое</p>
      <article>
         <h3>Заголовок статьи</h3>
         <p>Текст статьи.
            <mark>Важный фрагмент текста</mark>
         </p>
         <figure>
            <p><img src="photo.jpg" alt=""></p>
            <figcaption>Иллюстрация</figcaption>
            </figure>
         <aside>
            <p>Примечание к статье</p>
         </aside>
      </article>
   </section>
   <footer>
      <p>&amp;quot;Подвал&amp;quot; страницы</p>
   </footer>
</body>
</html>
```

Web-браузеры сейчас никак не выделяют визуально блочные теги семантической разметки. Однако мы всегда сможем привязать к ним стили, чтобы задать нужное нам оформление (о стилях будет рассказано в *главе 2*).

Чтобы старые версии Web-браузеров правильно отображали новые элементы, нужно объявить их блочными с помощью атрибута стиля `display`:

```
<style type="text/css">
main { display: block }
</style>
```

Для браузера Internet Explorer следует дополнительно создать элемент:

```
<script type="text/javascript">
document.createElement("main");
</script>
```

### 1.3.5. Тег <*details*>

Тег `<details>` в HTML 5 позволяет скрыть или отобразить какой-либо фрагмент страницы. По умолчанию содержимое элемента скрыто. Чтобы отобразить содержимое, нужно щелкнуть левой кнопкой мыши на заголовке, реализуемом с помощью тега `<summary>`, или добавить параметр `open`. Чтобы опять скрыть содержимое, нужно повторно щелкнуть мышью на заголовке:

```
<details open>
   <summary>Развернуть или свернуть</summary>
   <p>Скрытый текст</p>
</details>
```

Оба этих тега поддерживаются браузерами Chrome версии 12+, Opera версии 15+, Safari версии 6+ и Firefox версии 49+. Браузеры Internet Explorer и Microsoft Edge тег `<details>` не поддерживают.

### 1.3.6. Горизонтальная линия

Одинарный тег `<hr>` позволяет провести горизонтальную линию. По умолчанию линия занимает почти всю ширину родительского элемента и выравнивается по центру.

В HTML 4.01 в формате `Transitional` тег `<hr>` имеет параметры `size` (толщина линии), `width` (длина линии), `align` (выравнивание линии) и `noshade` (присутствие этого параметра отменяет рельефность линии). В HTML 5 этих параметров нет, вместо них нужно использовать стили:

```
<p>Со значениями по умолчанию</p>
<hr>
<p>Высота 5 px, ширина 90 процентов, с рамкой</p>
<hr style="height: 5px; width: 90%">
<p>Ширина 200 px, с выравниванием по правой стороне</p>
<hr style="width: 200px; margin: 0 0 0 auto">
<p>Высота 5 px, ширина 200 px, без рамки,
   с выравниванием по центру, красный цвет фона</p>
<hr style="height: 5px; width: 200px; border: 0; background: red">
```

## 1.4. Форматирование текста

Как уже говорилось, HTML — это *язык разметки*. Следовательно, важно уметь форматировать отдельные символы, а также целые фрагменты текста. Форматирование текста внутри блока выполняется строчными тегами.

## 1.4.1. HTML-эквиваленты

Прежде чем изучать теги, рассмотрим возможность отображения специальных символов. Такими символами, например, являются знаки меньше (<) и больше (>), т. к. с помощью этих символов описываются HTML-теги. Для отображения специальных символов используются так называемые *HTML-эквиваленты*. Например, для вывода такого текста:

```
Текст между тегами <title> и </title> используется в результатах,
выдаваемых поисковым порталом.
```

необходимо написать так:

```
Текст между тегами &lt;title&gt; и &lt;/title&gt; используется
в результатах, выдаваемых поисковым порталом.
```

В этом примере мы заменили знак меньше (<) на `<`, а знак больше (>) — на `>`. Приведем наиболее часто используемые HTML-эквиваленты:

- `<` — знак меньше (<);
- `>` — знак больше (>);
- `&` — амперсанд (&);
- ` ` — неразрывный пробел;
- `&quot;` — кавычка (");
- `&copy;` — знак авторских прав (©);
- `&reg;` — знак зарегистрированной торговой марки (®);
- `&trade;` — торговая марка (™).

## 1.4.2. Перевод строки

Для разделения строк используется одинарный тег `<br>`.

Если в HTML-документе набрать текст:

```
<p>Строка1
Строка2
Строка3</p>
```

то Web-браузер отобразит его в одну строку: `Строка1 Строка2 Строка3`. Для того чтобы строки располагались друг под другом, необходимо добавить тег `<br>` в конец каждой строки:

```
<p>Строка1<br>
Строка2<br>
Строка3</p>
```

Заметьте, что символ переноса строки в предыдущем результате был заменен символом пробела. Аналогичная ситуация будет, если внутри HTML-кода идут несколько пробельных символов подряд, — все они преобразуются Web-браузером

в один пробел. Если нужно отобразить несколько пробелов подряд, то следует воспользоваться комбинацией  :

```
<p>Пункт1<br>
   Пункт1_1<br>
   Пункт1_2</p>
```

Результат в окне Web-браузера:

```
Пункт1
   Пункт1_1
   Пункт1_2
```

Для вывода текста в том же виде, что и в исходном коде, можно воспользоваться парным блочным тегом <pre>:

```
<pre>
Пункт1
   Пункт1_1
   Пункт1_2
</pre>
```

В этом примере строки также будут располагаться друг под другом, и пробелы будут сохранены, но текст выведется моноширинным шрифтом.

## 1.4.3. Выделение фрагментов текста

Тег <b> отображает текст полужирным шрифтом:

```
<b>Полужирный шрифт</b>
```

Вместо тега <b> можно использовать тег логического форматирования , который определяет важный фрагмент текста:

```
<strong>Важный фрагмент текста</strong>
```

Тег <i> отображает текст курсивом:

```
<i>Текст, выделенный курсивом</i>
```

Вместо тега <i> можно использовать тег логического форматирования , с помощью которого делается акцент на выделенном фрагменте:

```
<em>Текст, выделенный курсивом</em>
```

Тег <u> отображает текст подчеркнутым:

```
<u>Подчеркнутый текст</u>
```

Тег <s> отображает текст перечеркнутым:

```
<s>Перечеркнутый текст</s>
```

Отметить фрагмент текста как удаленный позволяет парный тег <del>. Текст отображается перечеркнутым. Тег <del> имеет следующие параметры:

- cite — задает ссылку на страницу с описанием причины удаления;
- datetime — дата и время удаления.

Текст, вставленный вместо удаленного, отмечается тегом <ins>. Текст отображается подчеркнутым:

```
<del>старый текст</del> <ins>новый текст</ins>
```

Для вывода текста шрифтом меньшего размера применяется парный тег <small>:

```
Текст <small>меньшего</small> размера
```

## 1.4.4. Теги логического форматирования

Теги, рассмотренные в предыдущем разделе, в основном являются тегами физического форматирования. Исключение составляют теги  и . Эти теги являются *тегами логического форматирования* текста и служат для выделения очень важных и просто важных фрагментов соответственно. Теги логического форматирования используются для структурной разметки документа и могут отображаться разными Web-браузерами по-разному. Приведем основные теги логического форматирования:

- <cite>...</cite> — применяется для отметки цитат, а также названий произведений;
- <code>...</code> — служит для отметки фрагментов программного кода;
- <abbr>...</abbr> — используется для отметки аббревиатур;
- <kbd>...</kbd> — отмечает фрагмент как вводимый пользователем с клавиатуры;
- <q>...</q> — используется для отметки коротких цитат. Фрагмент отображается в кавычках;
- <samp>...</samp> — применяется для отметки результата, выдаваемого программой;
- <var>...</var> — отмечает имена переменных;
- <dfn>...</dfn> — служит для выделения нового термина (текст отображается курсивом).

В HTML 5 доступны также следующие теги:

- <mark>...</mark> — помечает важный фрагмент текста. Браузер Firefox устанавливает для такого фрагмента желтый цвет фона (как бы выделяет фрагмент текста маркером);
- <time>...</time> — дата и время. Может включать параметр datetime, указывающий дату и время в формате:

  ```
  <год>-<месяц>-<число> <часы>:<минуты>
  ```

## 1.4.5. Создание нижних и верхних индексов

Тег <sub> сдвигает текст ниже уровня строки и уменьшает размер шрифта. Он используется для создания нижних индексов:

```
Формула воды H<sub>2</sub>O
```

Тег <sup> сдвигает текст выше уровня строки и уменьшает размер шрифта. Этот тег используется чаще всего для создания степеней:

```
Единица измерения площади — м<sup>2</sup>
```

### 1.4.6. Тег 

Тег  является универсальным строчным элементом, к которому можно прикрепить различные стили. Пример использования тега  приведен в листинге 1.6.

**Листинг 1.6. Тег **

```
<!DOCTYPE html>
<html lang="ru">
<head>
   <meta charset="utf-8">
   <title>Элемент SPAN</title>
   <style type="text/css">
   span { /* Стиль для всех элементов SPAN */
      font-size: 12pt;      /* Размер шрифта */
      color: blue;          /* Цвет шрифта */
      font-family: "Arial"; /* Название шрифта */
      font-weight: bold;    /* Жирность шрифта */
   }
   </style>
</head>
<body>
  <p>
    С помощью элемента <span>SPAN</span> можно отформатировать
    <span style="color: red">фрагмент</span> внутри абзаца.
  </p>
</body>
</html>
```

## 1.5. Списки

*Список* — это набор упорядоченных абзацев текста, помеченных специальными значками (*маркированные списки*) или цифрами (*нумерованные списки*). Рассмотрим каждый из вариантов в отдельности.

### 1.5.1. Маркированные списки

Маркированный список помещают внутри парного тега <ul>. Перед каждым пунктом списка необходимо поместить тег <li>. Закрывающий тег </li> не обязателен. В листинге 1.7 представлена структура маркированного списка.