**1 Введение в НTML**

**1.1 История развития языка HTML**

HTML (HyperText Markup Language) представляет язык разметки гипертекста, используемый преимущественно для создания документов в сети интернет. HTML начал свой путь в начале 90-х годов как примитивный язык для создания веб-страниц. В 2014 году официально была завершена работа над новым стандартом - HTML5, который произвел революцию, привнеся в HTML много нового:

1. HTML5 определяет новый алгоритм автоматизированного получения информации с веб-сайтов для создания структуры DOM;
2. добавление новых элементов и тегов, как например, элементы video, audio и ряд других;
3. переопределение правил и семантики уже существовавших элементов HTML.

Фактически с добавлением новых функций HTML5 стал не просто новой версией языка разметки для создания веб-страниц, но и фактически платформой для создания приложений, а область его использования вышла далеко за пределы веб-среды интернет: HTML5 применяется также для создания мобильных приложений под Android, iOS, Windows Mobile и даже для создания десктопных приложений для обычных компьютеров [2].

Развитием HTML5 занимается независимая международная организация под названием World Wide Web Consortium (сокращенно W3C, Консорциум Всемирной паутины), которая определяет стандарт HTML5 в виде спецификации. W3C продолжает работать над развитием HTML5 и публиковать обновления.

Большинство последних версий браузеров поддерживают большинство функциональности HTML5 (Yandex, Firefox, Opera, Internet Explorer 11, Microsoft Edge). В то же время многие старые браузеры, как например, Internet Explorer 8 и более младшие версии, не поддерживают стандарты, а IE 9, 10 поддерживает лишь частично. При этом даже те браузеры, которые в целом поддерживают стандарты, могут не поддерживать какие-то отдельные функции.

**1.2 Элементы и атрибуты HTML5**

Прежде чем переходить непосредственно к созданию своих веб-страниц на HTML5, рассматриваются основные строительные блоки, кирпичики, из которых состоит веб-страница.

Документ HTML5, как и любой документ HTML, состоит из элементов, а элементы состоят из тегов. Как правило, элементы имеют открывающий и закрывающий тег, которые заключаются в угловые скобки. Например:

**<div>Текст в элементе div</div>**

Здесь определен элемент div, который имеет открывающий тег <div> и закрывающий тег </div>. Между этими тегами находится содержимое элемента div. В данном случае в качестве содержимого выступает простой текст “Текст элемента div”.

Элементы также могут состоять из одного тега, например, элемент <br />, функция которого - перенос строки.

**<div>Текст <br/> элемента div</div>**

Такие элементы еще называют пустыми элементами (void elements) [1]. Помимо этого, в некоторых тегах наличие закрывающих слешей необязательно. К таким тегам относится тег перехода на новую строку: <br>.

Каждый элемент внутри открывающего тега может иметь атрибуты. Атрибут позволяют изменить свойства тега, заданные по умолчанию.

Например:

**<div style="color: blue;">Кнопка</div>**

**<input type="button" value="Нажать">**

В приведенном примере определено два элемента: div и input. Элемент div имеет атрибут style. После знака равно в кавычках пишется значение атрибута: style=“color: red;”, которое указывает, что цвет текста будет красным.

Второй элемент - элемент input, состоящий из одного тега, имеет два атрибута: type (указывает на тип элемента - кнопка) и value (определяет текст на кнопке).

Существуют глобальные или общие для всех элементов атрибуты, как например, style, а есть специфические, применяемые к определенным элементам, как например, type.

Кроме обычных атрибутов существуют еще булевы или логические атрибуты (boolean attributes). Подобные атрибуты могут не иметь значения. Например, у кнопки можно задать атрибут disabled:

**<input type="button" value="Нажать" disabled>**

Атрибут disabled указывает, что данный элемент отключен [3].

**1.2.1 Глобальные атрибуты**

В HTML5 есть набор глобальных атрибутов, которые применимы к любому элементу HTML5:

* accesskey: определяет клавишу для быстрого доступа к элементу;
* class: задает класс CSS, который будет применяться к элементу;
* contenteditable: определяет, можно ли редактировать содержимое элемента;
* contextmenu: определяет контекстное меню для элемента, которое отображается при нажатии на элемент правой кнопкой мыши;
* dir: устанавливает направление текста в элементе;
* draggable: определяет, можно ли перетаскивать элемент;
* dropzone: определяет, можно ли копировать переносимые данные при переносе на элемент;
* hidden: скрывает элемент;
* id: уникальный идентификатор элемента. На веб-странице элементы не должны иметь повторяющихся идентификаторов;
* lang: определяет язык элемента;
* spellcheck: указывает, будет ли для данного элемента использоваться проверка правописания;
* style: задает стиль элемента;
* title: устанавливает дополнительное описание для элемента;
* translate: определяет, должно ли переводиться содержимое элемента.

Но, как правило, из всего этого списка наиболее часто используются три: class, id и style [4].

**1.2.2 Пользовательские атрибуты**

В отличие от предыдущей версии языка разметки, в HTML5 были добавлены пользовательские атрибуты (custom attributes). Теперь разработчик веб-страницы сам может определить любой атрибут, предваряя его префиксом data*-*. Например:

**<input type="button" value="Нажать" data-color="blue" >**

Здесь определен атрибут data-color, который имеет значение “blue”. Хотя для этого элемента, в html не существует подобного атрибута. Он определяется разработчиком и устанавливается у него любое значение.

**1.2.3 Одинарные и двойные кавычки**

Нередко можно встретить случаи, когда в HTML при определении значений атрибутов применяются как одинарные, так и двойные кавычки. Например:

**<input type='button' value='Нажать'>**

И одинарные, и двойные кавычки в данном случае допустимы, хотя чаще применяются двойные кавычки. Однако иногда само значение атрибута может содержать двойные кавычки, и в этом случае все значение лучше поместить в одинарные:

**<input type="button" value='Кнопка "Привет мир"'>**

**1.3 Создание документа HTML5**

Элементы являются кирпичиками, из которых складывается документ html5. Для создания документа необходимо создать простой текстовый файл, назвать index,  а в качестве расширения файла указать \*.html (рисунок 1.1).

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.1 - Структура документа HTML5

Затем можно открыть этот файл в любом текстовом редакторе, например, в Notepad++. Добавить в файл следующий текст:

**<!DOCTYPE html>**

**<html>**

**...**

**</html>**

Для создания документа HTML5 необходимо в первую очередь два элемента: DOCTYPE и html. Элемент doctype или Document Type Declaration сообщает веб-браузеру тип документа. <!DOCTYPE html> указывает, что данный документ является документом html и что используется html5, а не html4 или какая-то другая версия языка разметки.

Элемент HTML между своим открывающим и закрывающим тегами содержит все содержимое документа. Внутри элемента HTML можно разместить два других элемента: HEAD и BODY. Элемент HEAD содержит метаданные веб-страницы - заголовок веб-страницы, тип кодировки и т.д., а также ссылки на внешние ресурсы - стили, скрипты, если они используются. Элемент BODY, собственно, определяет содержимое HTML-страницы.

Изменим содержимое файла index.html следующим образом:

**<!DOCTYPE html>**

**<html>**

**<head>**

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

**<title>Страница HTML5</title>**

**</head>**

**<body>**

**<div>Добро пожаловать! (Контент страницы HTML5)</div>**

**</body>**

**</html>**

В элементе head определено два элемента:

* элемент title, который представляет заголовок страницы;
* элемент meta, который определяет метаинформацию страницы.

Для корректного отображения символов предпочтительно указывать кодировку. В данном случае с помощью атрибута charset="utf-8" указана кодировка utf-8.

В пределах элемента body используется только один элемент - div, который оформляет блок. Содержимым этого блока является простая строка.

Поскольку выбрана в качестве кодировки utf-8, то браузер будет отображать веб-страницу именно в этой кодировке. Однако необходимо, чтобы сам текст документа соответствовал выбранной кодировке (рисунок 1.2).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.2 - Установка кодировки html-файла в VS Code

После этого в статусной строке будет можно будет увидеть UTF-8 w/o BOM, что будет указывать, что нужная кодировка установлена. Далее следует сохранить и открыть файл index.html в браузере (рисунок 1.3).

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.3 - Документ HTML5

Таким образом, создан первый документ HTML5. Так как указан в элементе title заголовок “Документ HTML5”, то именно такое название будет иметь вкладка браузера [1].

Так как указана кодировка utf-8, то веб-браузер будет корректно отображать кириллические символы. А весь текст, определенный внутри элемента body, можно увидеть в основном поле браузера.

Все, что видно в своем браузере при загрузке веб-страницы, основная часть документа html располагается между тегами <body> и </body>. Здесь размещаются большинство элементов html.

**1.4 Элемент head и метаданные веб-страницы**

Как правило, одним из первых элементов html-документа является элемент head, задача которого состоит в установке метаданных страницы и ряда сопроводительной информации. Метаданные содержат информацию о html-документе.

**1.4.1 Заголовок**

Для установки заголовка документа, который отображается на вкладке браузера, используется элемент title.

**<!DOCTYPE html>**

**<html>**

**<head>**

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

**<title>Заголовок - title</title>**

**</head>**

**<body> <p>Контент страницы HTML5</p></body>**

**</html>**

Реализация данного кода представлена на рисунке 1.4.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.4 – Использование заголовков в html

**1.4.2 Элемент base**

Элемент base позволяет указать базовый адрес, относительно которого устанавливаются другие адреса, используемые в документе:

**<!DOCTYPE html>**

**<html>  <head>**

**<base href="content/">**

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

**<title>Элемент base</title> </head>**

**<body> <a href="newpage.html">Перейти</a>  </body>**

**</html>**

Хотя для ссылки в качестве адреса указана страница newpage.html, но фактически ее адресом будет content/newpage.html. То есть в одной папке с текущей страницей должна быть подпапка content, в которой должен находится файл newpage.html

Можно также указывать полный адрес:  
**<base href="**[**http://www.microsoft.com/**](http://www.microsoft.com/)**">.** В этом случае ссылка будет вести на сайт <http://www.microsoft.com/newpage.html>.

**1.4.3 Элемент meta**

Элемент meta определяет метаданные документа. Чтобы документ корректно отображал текст, необходимо задать кодировку с помощью атрибута charset. Рекомендуемой кодировкой является utf-8:

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

При этом надо помнить, что указанная элементе meta кодировка должна совпадать с кодировкой самого документа. Как правило, текстовый редактор позволяет указать кодировку документа. Если ориентироваться на utf-8, то в настройках текстового редактора надо выбирать UTF-8.

Элемент meta также имеет два атрибута: name и content. Атрибут name содержит имя метаданных, а content - их значение.

По умолчанию в HTML определены пять типов метаданных:

* application name: название веб-приложения, частью которого является данный документ;
* author: автор документа;
* description: краткое описание документа;
* generator: название программы, которая сгенерировала данный документ;
* keywords: ключевые слова документа.

Надо отметить, что наиболее актуальным является тип description. Его значение поисковики часто используют в качестве аннотации к документу в поисковой выдаче. Далее представлен документ с использованием элементов meta.

**<!DOCTYPE html>**

**<html> <head>**

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

**<base href="content/">**

**<title>Заголовок - title</title>**

**<meta name="contents" content="Первая страница HTML5">**

**<meta name="author" content="Tagirova L.F. ">**

**</head>**

**<body> <a href="pagecontent.html">Контент страницы HTML5</a>**

**</body> </html>**

Далее в документ добавлен элемент meta descripton.

**<!DOCTYPE html>**

**<html> <head>**

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

**<base href="content/">**

**<title>Заголовок - title</title>**

**<meta name="contents" description="Контент страницы HTML">**

**</head> <body> <a href="pagecontent.html">Контент страницы HTML5</a>**

**</body> </html>**

**1.5 Элементы группировки в HTML5**

**1.5.1 Элемент div**

Элемент div служит для структуризации контента на веб-странице, для заключения содержимого в отдельные блоки [6]. Div создает блок, который по умолчанию растягивается по всей ширине браузера, а следующий после div элемент переносится на новую строку. Например:

**<body>**

**<div>Заголовок контента HTML5</div>**

**<div>Контент страницы HTML5</div>**

**</body>**

Реализация данного кода представлена на рисунке 1.5.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.5 - Блоки DIV в HTML5

**1.5.2 Параграфы**

Параграфы создаются с помощью тегов <p> и </p>, в которые заключают некоторое содержимое. Каждый новый параграф располагается на новой строке. Далее представлен пример применением параграфов [7]:

**<body>**

**<div>Заголовок контента HTML5</div>**

**<div> <p>Параграф 1</p>**

**<p>Параграф 2</p>  </div>**

**</body>**

Реализация данного кода представлена на рисунке 1.6.

Изображение выглядит как текст, снимок экрана, Шрифт, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.6 - Параграфы в html5

Если в рамках одного параграфа необходимо перенести текст на другую строку, то можно воспользоваться элементом <br>:

**<p>Первая строка. <br/> Вторая строка. </p>**

**1.5.3 Элемент pre**

Элемент pre выводит предварительно отформатированный текст так, как он определен.

**<body> <pre> Михаил Лермонтов**

**Один из самых известных русских поэтов**

**признание к нему пришло еще при жизни. </pre>**

**</body>**

Реализация данного кода представлена на рисунке 1.7.

Изображение выглядит как текст, снимок экрана, мультимедиа, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.7 - Элемент pre в HTML5

**1.5.4 Элемент span**

Элемент span обтекает некоторый текст по всей его длине и служит преимущественно для стилизации заключенного в него текстового содержимого. В отличие от блоков div или параграфов span не переносит содержимое на следующую строку.

**<body> <div>Заголовок контента HTML5</div>**

**<div> <p><span style="color: blue;">Параграф</span> 1</p>**

**<p><span>Параграф</span> 2</p>**

**</div>  </body>**

Реализация данного кода представлена на рисунке 1.8.

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.8 - Элемент span в html5

При этом, сам по себе span ничего не делает. Так, во втором параграфе span не повлиял на внутренне текстовое содержимое. А в первом параграфе элемент span содержит атрибут стиля: style=“color: red: blue;”, который устанавливает для текста синий цвет. При этом элементы div и p являются блочными, элемент div может содержать любые другие элементы, а элемент p - только строчные элементы. В отличие от них элемент span является строчным, встраивает свое содержимое во внешний контейнер - тот же div или параграф. Не стоит помещать блочные элементы в строчный элемент span.

**1.6 Заголовки в HTML5**

Элементы <h1>, <h2>, <h3>, <h4>, <h5> и <h6> используются для создания заголовков различных уровней (рисунок 1.10):

**<body>**

**<h1>1-ый уровень</h1>**

**<h2>2-ой уровень </h2>**

**<h3>3-ий уровень </h3>**

**<h4>4-ый уровень </h4>**

**<h5>5-ый уровень </h5>**

**<h6>6-ой уровень </h6>**

**</body>**

Реализация данного кода представлена на рисунке 1.9.

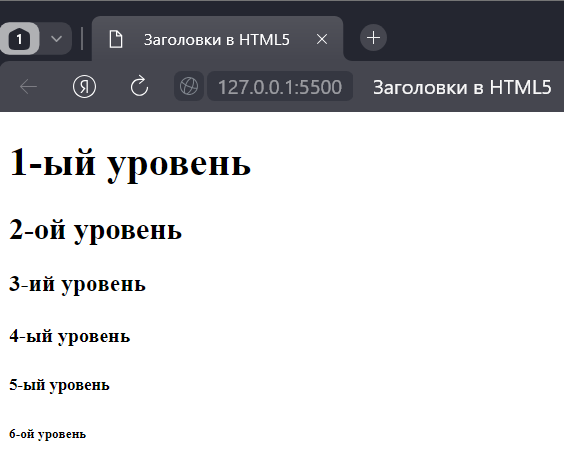


Рисунок 1.9 - Заголовки в HTML5

Заголовки выделяются жирным шрифтом и имеют определенный размер. При определении заголовков следует учитывать, что на странице должен быть только один заголовок верхнего уровня. Он функционирует как основной заголовок веб-страницы.

**1.7 Форматирование текста**

Ряд элементов HTML используется для форматирования текстового содержимого, например, для выделения жирным или курсивом и т.д. Далее описаны эти элементы:

* <b>: выделяет текст жирным;
* <del>: зачеркивает текст;
* <i>: выделение текста курсивом;
* <em>: выделяет текст курсивом, в отличие от тега <i> носит логическое значение, придает выделяемому тексту оттенок важности;
* <s>: зачеркивает текст;
* <small>: делает текст чуть меньше размером, чем окружающий;
* <strong>: выделяет текст жирным. В отличие от тега <b> предназначен для логического выделения, чтобы показать важность текста. А <b> не носит характера логического выделения, выполняет функции только форматирования;
* <sub>: помещает текст под строкой;
* <sup>: помещает текст над строкой;
* <u>: подчеркивает текст;
* <mark>: выделяет текст цветом, придавая ему оттенок важности.

Далее представлен пример применения всех этих элементов.

**<body>     <p>Форматирование в <mark>HTML5</mark></p>**

**<p> <b>Выделенный</b> контент</p>**

**<p><strong>Важный</strong> контент</p>**

**<p><del>Зачеркнутый</del> контент</p>**

**<p> <s>Недействительный</s> контент</p>**

**<p> <em>Важный</em> контент</p>**

**<p>Контент выделенный <i>курсивом</i> </p>**

**<p> <ins>Добавленный</ins> контент</p>**

**<p> <u>Подчеркнутый</u> контент</p>**

**<p>z<sub>i</sub> = y<sup><small>3</small></sup> + x<sup><small>3</small></sup> </p>**

**</body>**

Реализация данного кода представлена на рисунке 1.10.

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.10 - Использование тегов форматирования текста

**1.8 Escape-последовательности в HTML**

В HTML можно использовать экранирующий символ (escape-последовательность) для представления любого символа Unicode, используя только буквы ASCII [8].

Экранирующие символы очень полезны для представления символов, которые не являются очевидными или неоднозначны, в частности, это относится к символам «<», «>», «&», «\_», «©». В таблице 1.1 приведен список escape-последовательностей в HTML.

Таблица 1.1 - Некоторые популярные экранирующие символы

|  |  |  |  |
| --- | --- | --- | --- |
| Вид | Обозначение | Десятичный код | Название символа |
|  | &nbsp; | &#160; | неразрывный пробел |
| < | &lt; | &#60; | левая угловая скобка |
| > | &gt; | &#62; | правая угловая скобка |
| ¢ | &cent; | &#162; | цент |
| £ | &pound; | &#163; | фунт стерлингов |
| ¥ | &yen; | &#165; | йена |
| § | &sect; | &#167; | параграф |
| © | &copy; | &#169; | знак авторского права |
| ® | &reg; | &#174; | охраняемый знак |
| « | &laquo; | &#171; | открывающая двойная угловая кавычка |
| » | &raquo; | &#187; | закрывающая двойная угловая кавычка |
| ° | &deg; | &#176; | градус |
| ¿ | &iquest; | &#191; | перевернутый вопросительный знак |
| µ | &micro; | &#181; | знак микро |
| ¼ | &frac14; | &#188; | одна четвертая |
| ½ | &frac12; | &#189; | одна вторая |
| ¾ | &frac34; | &#190; | три четверти |
| & | &amp; | &#38; | амперсант |
| € | &euro; | &#8364; | евро |
| ↔ | &harr; | &#8596; | стрелка влево-вправо |
| ⇔ | &hArr; | &#8660; | двойная стрелка влево-вправо |
| ∞ | &infin; | &#8734; | бесконечность |
| ″ | &Prime; | &#8243; | знак двойной прим |
| ¶ | &para; | &#182; | конец абзаца |

В приведенном ниже примере представлена HTML-страница, на которой использованы различные экранирующие символы.

**<body> <h3> Использование&nbsp;Escape-последовательностей в HTML </h3>**

**<div> &copy; &laquo;Теги HTML - это команды, которые говорят браузеру, что и в каком порядке показывать на экране. </div>**

**<div> У каждого тега есть имя, которое расположено в угловых скобках. </div>**

**<div> Самая простая HTML-страница состоит из трех тегов: &lt;html&gt;, &lt;head&gt; и &lt;body&gt;&raquo; </div>**

**<div> </div> </body>**

Реализация данного кода представлена на рисунке 1.11.

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.11 – Использование Escape-последовательностей в HTML

**1.9 Работа с изображениями**

Для вывода изображений в HTML используется элемент img. Этот элемент представляет два важных атрибута:

* src: путь к изображению. Это может быть относительный или абсолютный путь в файловой системе или адрес в интернете;
* alt: текстовое описание изображения. Если браузер не может отобразить изображение, то он показывает вместо самой картинки данное текстовое описание.

Пример работы с рисунками представлен в нижеследующем примере.

**<body>**

**<img src="flowerss.jpg" alt="Цветы в вазе на подоконнике" />**

**</body>**

Реализация данного кода представлена на рисунке 1.12.

Изображение выглядит как ваза, цветок, мультимедиа, текст

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.12 - Тег img в HTML5

В данном примере файл изображения flowerss.jpg находится в одной папке с веб-странице index.html. При этом рисунок задан в исходном размере, расстояние до окружающего текста и обтекание по умолчанию.

Для того, чтобы изменить значения рисунка по умолчанию у тега img можно применять дополнительные классические атрибуты:

- height – изменение высоты рисунка (задается по умолчанию в пикселях),

- width – изменение ширины рисунка (задается по умолчанию в пикселях),

- [usemap](https://hcdev.ru/html/img/#usemap) – ссылка на элемент [<map>](https://hcdev.ru/html/map/), содержащий координаты для клиентской карты-изображения,

- hspace – расстояние от рисунка до окружающего текста по горизонтали,

- vspace – расстояние от рисунка до окружающего текста по вертикали,

- border – рамка.

HTML5 предлагает современные настройки для работы с рисунками.

1. Атрибут srcset. Он представляет собой способ предложить браузеру версию картинки с повышенным разрешением:

**<img src="flowerss.jpg"**

**srcset=" flowerss.jpg2x.png 2x"**

**alt="Цветы в вазе на подоконнике>**

 Также атрибут srcset можно использовать в сочетании с атрибутом sizes, чтобы подсказать браузеру, какие варианты картинок есть, и помочь ему выбрать подходящие ситуации.

1. Атрибут sizes. HTML даёт возможность загружать разные изображения в зависимости от разных условий. Частая ситуация: разные картинки под разные ширины экранов. Можно делать это при помощи тега [<picture>](https://doka.guide/html/picture/), а можно задать атрибуты srcset и sizes прямо в теге <img>.

- srcset - атрибут, принимающий несколько строк, разделённых запятой. Каждая строка должна содержать ссылку на картинку и указание фактической ширины картинки, значения разделяются пробелом. При этом после ширины ставится единица измерения *w*, а не *px*.

- sizes - атрибут, в котором указывается одно или несколько условий через запятую. Каждая строка состоит из медиавыражения и ширины блока для картинки, разделённых пробелом. Ширину блока для картинки можно указывать в любых единицах измерения, кроме процентов.

**<img src="flowerss.jpg"**

**srcset=" flowers-small.jpg 320w, flowers-medium.jpg 480w,**

**flowers-large.jpg 800w" sizes=" (max-width: 320px) 280px,**

**(max-width: 480px) 440px, 800px" alt="Цветы в вазе на подоконнике">**

Далее приведен пример, в котором использовался тот же рисунок, однако были изменены его размер, добавлено обтекание и рамка.

**<body> <img src=flowerss.jpg alt="Цветы в вазе на подоконнике" style="float: left; margin-right:10px; width:250px; height:150px; border=2px;" />**

**<h2>Флористка</h2>**

**<p>Флористика - это искусство использовать растительные элементы для создания композиций или украшения интерьера/экстерьера. Профессию флориста нередко сравнивают с художником, только вместо красок мастер использует цветочные бутоны, зеленые растения, природные материалы.</p> <p>Создание букета требует определённых знаний. В букете нужно соединить цветы и зелень, которые будут дополнять друг друга, не конфликтуя между собой. Только так можно добиться единого шедевра. Специалисты флористики создали настоящую науку по комбинированию бутонов в букете. Важно учитывать не только цвет и другие параметры, но и природные данные цветов. </p>**

**</body>**

Реализация данного кода представлена на рисунке 1.13.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.13 – Использование дополнительных атрибутов тега img

**1.10 Мульмедиа**

**1.10.1 Видео**

Для воспроизведения видео в HTML5 используется элемент video. Чтобы настроить данный элемент, можно использовать следующие его атрибуты:

* src: источник видео, это может быть какой-либо видеофайл;
* width: ширина элемента;
* height: высота элемента;
* controls: добавляет элементы управления воспроизведением;
* autoplay: устанавливает автовоспроизведение;
* loop: задает повторение видео;
* muted: отключает звук по умолчанию.

Хотя можно установить ширину и высоту, но они не окажут никакого влияния на отношения ширины и высоты самого видео. Например, если видео имеет формат 375×240, то, к примеру, при настройках width=“375” height=“280” видео будет центрироваться на 280-пиксельном пространстве в HTML.

**<body>**

**<video src="flower.mp4" width="400" height="300" controls ></video></body>**

**</html>**

Реализация данного кода представлена на рисунке 1.14.

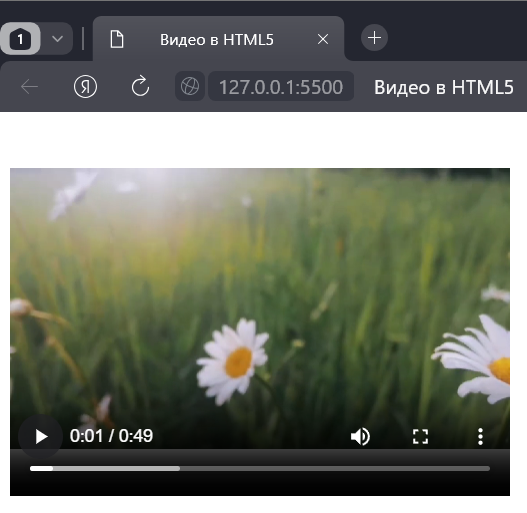
****

Рисунок 1.14 - Видео в HTML5

Далее были применены атрибуты autoplay и loop:

**<video src="flower.mp4" width="400" height="300" controls autoplay loop> </video>**

Теперь видео будет автоматически проигрываться бесконечное число раз.

Если при воспроизведении необходимо отключить звук, то можно воспользоваться атрибутом muted:

**<video src="flower.mp4" width="400" height="300" controls muted></video>**

**1.10.1.1** **Атрибут preload**

Еще один атрибут - preload призван управлять загрузкой видео. Он принимает следующие значения:

* auto: видео и связанные с ним метаданные будут загружаться до того, как видео начнет воспроизводиться;
* none: видео не будет загружаться в фоне, пока пользователь не нажмет на кнопку начала проигрывания;
* metadata: в фоне до воспроизведения будут загружаться только метаданные (данные о формате, длительности и т.д), само видео не загружается.

**<video src="flower.mp4" width="400" height="300" controls preload="auto"></video>**

**1.10.1.2 Атрибут poster**

Атрибут poster позволяет установить изображение, которое будет отображаться до запуска видео. Этому атрибуту в качестве значения передается путь к изображению:

**<video src="flower.mp4" width="400" height="300" controls poster="mycat.jpg"> </video>**

**1.10.1.3 Поддержка форматов видео**

Главной проблемой при использовании элемента video является поддержка различными веб-браузерами определенных форматов. С помощью вложенных элементов source можно задать несколько источников видео, один из которых будет использоваться:

**<video width="400" height="300" controls>**

**<source src="flower.mp4" type="video/mp4">**

**<source src="flower.webm" type="video/webm">**

**<source src="flower.ogv" type="video/ogg"> </video>**

Элемент source использует два атрибута для установки источника видео:

* src: путь к видеофайлу;
* type: тип видео (MIME-тип).

Если браузер не поддерживает первый тип видео, то он пытается загрузить второй видеофайл. Если же и тип второго видеофайла не поддерживается, то браузер обращается к третьему видеофайлу.

**1.10.2 Аудио**

Для воспроизведения звука без видео в HTML5 применяется элемент audio. Он во многом похож на элемент video. Для настройки элемента audio можно использовать следующие его атрибуты:

* src: путь к аудиофайлу;
* controls: добавляет элементы управления воспроизведением;
* autoplay: устанавливает авто воспроизведение;
* loop: задает повторение аудиофайла;
* muted: отключает звук по умолчанию;
* preload: устанавливает режим загрузки файла.

Действие всех этих атрибутов будет аналогично их действию в элементе video.

**<body>**

**<audio src="sound.mp3" controls></audio>**

**</body>**

Реализация данного кода представлена на рисунке 1.15.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.15 - Аудио в HTML5

Ключевым моментом для работы с аудио является поддержка браузером тех или иных форматов. На данный момент подавляющее большинство браузеров поддерживают mp3. Однако если есть неуверенность, что аудио в определенном формате будет поддерживаться браузером пользователя, то можно использовать вложенный элемент source и указать аудио в иных форматах:

**<audio width="400" height="300" controls>**

**<source src="sound.mp3" type="audio/mpeg">**

**<source src="sound.m4a" type="audio/aac">**

**<source src="sound.ogg" type="audio/ogg">**

**</audio>**

Как и в случае с элементом video, здесь у элемента source устанавливается атрибут src с ссылкой на файл и атрибут type - тип файла.

**1.11 Списки в HTML**

Для создания списков в HTML5 применяются элементы <ol> (нумерованный список) и <ul> (ненумерованный список). В нумерованном списке для нумерации элементов по умолчанию используется стандартные цифры от 1. В ненумерованном списке каждый элемент обозначается закрашенным диском.

Далее представлен пример использования этих тегов.

**<body> <h2> Нумерованный список</h2>**

**<ol> <li>Розы</li> <li>Хризантемы</li> <li>Тюльпаны</li> <li>Гладиолусы</li> </ol> <h2> Ненумерованный список</h2>**

**<ul><li>Пионы</li> <li>Лилии</li> <li>Герберы</li> <li>Ромашки</li></ul>**

**</body>**

Реализация данного кода представлена на рисунке 1.16.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.16 – Использование списков в HTML5

При необходимости можно настроить нумерацию или отражаемый рядом с элементом символ с помощью стиля list-style-type. Данный стиль может принимать множество различных значений. Для нумерованных списков стиль list-style-type может принимать следующие значения:

* decimal: десятичные числа, отсчет идет от 1;
* decimal-leading-zero: десятичные числа, которые предваряются нулем, например, 01, 02, 03, … 98, 99;
* lower-roman: строчные римские цифры, например, i, ii, iii, iv, v,…;
* upper-roman: заглавные римские цифры, например, I, II, III, IV, V,…;
* lower-alpha: строчные римские буквы, например, a, b, c,… z;
* upper-alpha: заглавные римские буквы, например, A, B, C, … Z.

Для нумерованных список с помощью атрибута start можно дополнительно задать символ, с которого будет начинаться нумерация.

Далее представлен пример работы с атрибутами нумерованных списков для вывода наименований пород кошек.

**<h2>list-style-type = decimal</h2><ol style="list-style-type: decimal;" start="3">**

**<li>Сиамская</li> <li>Британская </li> <li>Мейн-кун</li></ol>**

**<h2>list-style-type = upper-roman</h2><ol style="list-style-type: upper-roman;">**

**<li>Персидская </li><li>Русская голубая </li> <li> Сфинкс </li></ol>**

**<h2>list-style-type = lower-alpha</h2><ol style="list-style-type: lower-alpha;">**

**<li> Тонкинская</li><li>Шиншилла</li> <li>Сибирская </li></ol>**

Реализация данного кода представлена на рисунке 1.17.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.17 - Нумерованный список в HTML5

Для ненумерованного списка атрибут list-style-type может принимать следующие значения:

* disc: черный диск;
* circle: пустой кружочек;
* square: черный квадратик.

Далее представлен пример работы с нумерованными списками для вывода наименований пород собак.

**<h2>list-style-type = disc</h2><ul style="list-style-type: disc;">**

**<li>Австралийская овчарка</li><li>Алабай</li><li>Спаниель</li></ul>**

**<h2>list-style-type = circle</h2><ul style="list-style-type: circle;">**

**<li>Бигль</li> <li>Боксер</li> <li>Болонка </li></ul>**

**<h2>list-style-type = square</h2><ul style="list-style-type: square;">**

**<li> Корги</li><li>Чихуахуа</li> <li>Доберман </li></ul>**

Реализация данного кода представлена на рисунке 1.18.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.18 - Неупорядоченный список в HTML5

Еще одну интересную возможность по настройке списков предоставляет стиль list-style-image. Он задает изображение, которое будет отображаться рядом с элементом списка.

**<ul style="list-style-image: url(./birds.png)">**

**<li>Лебедь</li> <li>Орел</li>**

**<li>Соловей</li> <li>Павлин</li></ul>**

Реализация данного кода представлена на рисунке 1.19.

Изображение выглядит как снимок экрана, текст, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.19 - Картинки в списке в HTML5

Стиль list-style-image в качестве значения принимает url(./birds.png), где “birds” - это название файла изображения, который находится в одной папке с веб-страницей index.html.

Одним из распространенных способов стилизации списков представляет создание горизонтального списка. Для этого для всех элементов списка надо установить стиль display:inline.

**<head> <style>   ul#menu li {display: inline;}  </style> </head>**

**<body> <ul id="menu">**

**<li>Пушкин</li> <li>Лермонтов</li><li>Пришвин</li><li>Маяковский</li> </ul>**

**</body> </html>**

Реализация данного кода представлена на рисунке 1.20.

**Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 1.20 - Горизонтальный список

**1.12 Элемент details**

Элемент details позволяет создавать раскрываемый блок, который по умолчанию скрыт. Иногда такой элемент еще называют accordion. Данный элемент содержит элемент summary, который представляет заголовок для блока, и этот заголовок отображается в скрытом режиме. Далее представлен пример использования аккордиона.

**<body> <details> <summary>Поэты</summary>**

**<ul> <li>Пушкин</li>**

**<li>Лермонтов</li>**

**<li>Тютчев</li>**

**<li>Блок</li> </ul>**

**</details> </body>**

По умолчанию видно только заголовок summary (рисунок 1.21).

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.21 - Элемент summary в HTML5

Нажав на стрелку или заголовок, можно раскрыть блок (рисунок 1.22).

Изображение выглядит как текст, снимок экрана, электроника, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.22 - Элемент details в HTML5

**1.13 Обозначение цветов в HTML**

В HTML используется цветовая модель RGB (Red, Green, Blue) цвета представляют тремя значениями, описывающими цветовые каналы - соответственно - красный, зелёный и синий канал.

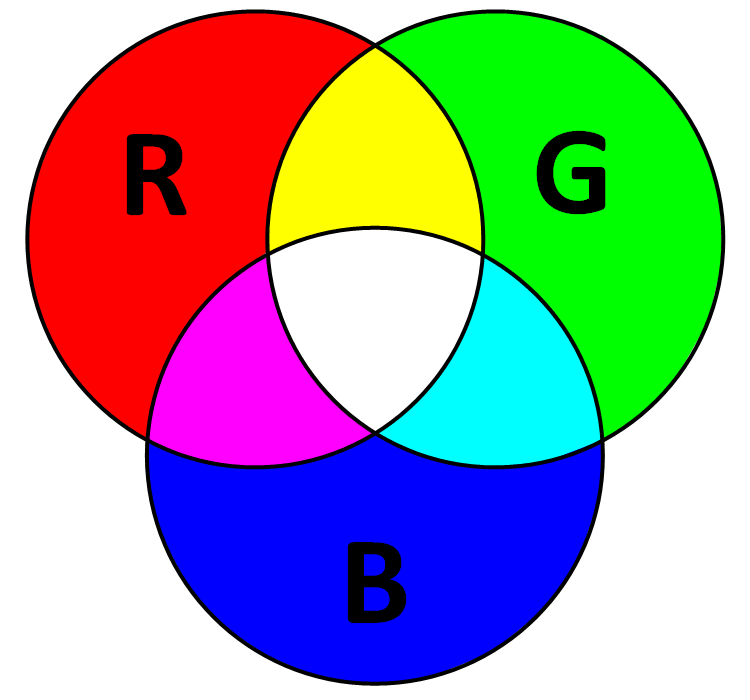


Рисунок 1.23 - Цветовая модель RGB

Каждое из значений можно представить числом, находящимся в диапазоне от 0 до 255, или процентным значением - от 0 до 100%. Если три значения равны 0, то получится чёрный цвет. А если все три цветовых компонента установлены в значение 255, то получится белый цвет. То же самое справедливо и при использовании процентных значений.

Для спецификации цвета в HTML существуют два варианта:

* символьная нотация, например, maroon;
* цифровое обозначение, которое контролирует, каким образом цвет формируется из основных цветов - красного, зеленого и голубого - в так называемое [RGB цветовое пространство](http://www.w3.org/pub/WWW/Printing/motta/W3Color.html), при этом обозначение должно быть взято в [кавычки](C:Лилькины%20ДокументыРАБОТААППАРАТНЫЙСетиСетевые%20информационные%20технологииИзучение_HTML_3-2_на_примерахgl_9.htm#quotes) (комбинация RGB: “#RRGGBB”). Для задания цифрового обозначения цвета в HTML используются шестнадцатеричная и десятичные формы записи. Например, “#800000” и rgb (128,0,0).

Символьные нотации намного легче и более понятны. С другой стороны, численные обозначения дают больше возможностей. К ним, в частности относится возможность подбора наиболее близких по цветовой гамме цветов при создании единого стиля страницы.

Далее представлены примеры использования этих вариантов.

**div {color: purple;}**

**H1 {color: rgb(255, 0, 255);}**

**P {color: #800080;}**

При описании цветов с использованием цветовой модели RGB можно, помимо цветовых компонентов, указывать и значение, соответствующее альфа-каналу, которое позволяет управлять прозрачностью цвета. Это помогает в работе с насыщенностью цветов, что весьма полезно. Ниже рассматриваются некоторые варианты использования цветов, при описании которых используется альфа-канал. Для работы с такими цветами применяется функция rgba():

**div { /\* Чёрный фон с 50% прозрачностью \*/ background-color: rgba(0, 0, 0, 0.5);}**

Если значение, соответствующее альфа-каналу, равно нулю, тогда основной или фоновый цвет элемента окажется полностью прозрачным, то есть - невидимым.

Если пара значений в описании в шестнадцатеричном формате одного компонента цвета идентична, одно из таких значений можно убрать. Например:

**body {color: #222222; /\* Это #222 \*/**

**body {color: #000000; /\* Это #000 \*/**

Значение #222222 эквивалентно значению #222. Если представить исходный шестнадцатеричный код в виде 22, 22, 22, то, чтобы получить из него код 2, 2, 2, достаточно взять из каждой пары значений по одному.

Наиболее популярные цвета, используемые в HTML, их символьные и числовые эквиваленты представлены в таблице 1.2.

Таблица 1.2 - Таблица HTML цветов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя цвета | HEX | RGB | Имя цвета | HEX | RGB |
| Black | #000000 | 0, 0, 0 | Coral | #FF7F50 | 255, 127, 80 |
| Gray | #808080 | 128, 128, 128 | Tomato | #FF6347 | 255, 99, 71 |
| Silver | #C0C0C0 | 192, 192, 192 | OrangeRed | #FF4500 | 255, 69, 0 |
| White | #FFFFFF | 255, 255, 255 | DarkOrange | #FF8C00 | 255, 140, 0 |
| Fuchsia | #FF00FF | 255, 0, 255 | Gold | #FFD700 | 255, 215, 0 |
| Purple | #800080 | 128, 0, 128 | LightYellow | #FFFFE0 | 255, 255, 224 |
| Red | #FF0000 | 255, 0, 0 | LemonChiffon | #FFFACD | 255, 250, 205 |
| Maroon | #800000 | 128, 0, 0 | Lavender | #E6E6FA | 230, 230, 250 |
| Yellow | #FFFF00 | 255, 255, 0 | Thistle | #D8BFD8 | 216, 191, 216 |
| Olive | #808000 | 128, 128, 0 | Plum | #DDA0DD | 221, 160, 221 |
| Lime | #00FF00 | 0, 255, 0 | Indigo | #4B0082 | 75, 0, 130 |
| Green | #008000 | 0, 128, 0 | SlateBlue | #6A5ACD | 106, 90, 205 |
| Aqua | #00FFFF | 0, 255, 255 | Goldenrod | #DAA520 | 218, 165, 32 |
| Teal | #008080 | 0, 128, 128 | DarkGoldenRod | #B8860B | 184, 134, 11 |
| Blue | #0000FF | 0, 0, 255 | Peru | #CD853F | 205, 133, 63 |
| Navy | #000080 | 0, 0, 128 | Chocolate | #D2691E | 210, 105, 30 |
| FireBrick | #B22222 | 178, 34, 34 | SpringGreen | #00FF7F | 0, 255, 127 |
| DarkRed | #8B0000 | 139, 0, 0 | MediumSeaGreen | #3CB371 | 60, 179, 113 |
| Salmon | #FA8072 | 250, 128, 114 | SeaGreen | #2E8B57 | 46, 139, 87 |
| DeepPink | #FF1493 | 255, 20, 147 | ForestGreen | #228B22 | 34, 139, 34 |
| Aquamarine | #7FFFD4 | 127, 255, 212 | MediumAquamarine | #66CDAA | 102, 205, 170 |

Далее приведен пример использования цветов при создании HTML страниц.

**<body style="background-color: gainsboro">**

**<h2>Использование цвета в HTML</h2>**

**<h1 style="background-color: cyan">Имя Цвета</h1>**

**<h3 style="background-color: #F067FF;">HEX</h3>**

**<h3 style="background-color: #0F0;">Сокращение HEX</h3>**

**<h3 style="background-color: rgb(255, 124, 30)">RGB</h3>**

**<h3 style="background-color: rgba(255, 0, 0, 0.5);">RGBA</h3> </body>**

Реализация данного кода представлена на рисунке 1.24.

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

AI-generated content may be incorrect.

Рисунок 1.24 - Работа с цветами в HTML

**1.14 Таблицы в HTML5**

Для создания таблиц в html используется элемент table. Каждая таблица между тегами <table> и </table> содержит строки, который представлены элементом tr. А каждая строка между тегами <tr> и </tr> содержит ячейки в виде элементов td. Далее представлен пример простой таблицы.

**<body><table border=2px><tr>**

**<td>Жанр</td> <td>Наименование</td> <td>Автор</td></tr>**

**<tr> <td>Мистика</td> <td>Дракула</td> <td>Брэм Стокер</td> </tr>**

**<tr> <td>Приключения</td> <td>Вокруг света в восемьдесят дней</td> <td>Жюль Верн</td> </tr> <tr> <td>Детская литература</td> <td>Алиса в Стране Чудес</td> <td>Льюис Кэрролл</td> </tr>**

**<tr> <td>Русская классика</td> <td>Мастер и Маргарита</td> <td>Михаил Булгаков</td> </tr></table></body>**

Реализация данного кода представлена на рисунке 1.25.

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

AI-generated content may be incorrect.

Рисунок 1.25 - Таблицы в HTML5

Здесь представлено в таблице пять строк, каждая строка имеет по три столбца. При этом первая строка выполняет роль заголовка, а остальные три строки являются содержимым таблицы. Разделения заголовков, футера и тела таблицы в html предусмотрены соответственно элементы thead, tfoot и tbody. Для их применения таблица была изменена следующим образом.

**<body><table border=2px> <caption><b>Популярные книги</b></caption>**

**<thead><tr><th>Жанр</th><th>Наименование</th><th>Автор</th></tr></thead>**

**<tbody><tr> <td>Мистика</td><td>Дракула</td><td>Брэм Стокер</td></tr> <tr>**

**<td>Приключения</td><td>Вокруг света в восемьдесят дней</td><td>Жюль Верн</td></tr> <tr><td>Детская литература</td><td>Алиса в Стране Чудес</td><td>Льюис Кэрролл</td> </tr> <tr> <td>Русская классика</td><td>Мастер и Маргарита</td><td>Михаил Булгаков</td></tr> </tbody><tfoot>**

**<tr> <th colspan="3">Информация 2024 год</th></tr> </tfoot></table></table>**

В элемент thead заключается строка заголовков. Для ячеек заголовок используется не элемент td, а th. Элемент th выделяет заголовок жирным. А все остальные строки заключаются в tbody. Элемент tfoot определяет подвал таблицы или футер. Здесь обычно выводится некоторая вспомогательная информация по отношению к таблице. Реализация данного кода представлена на рисунке 1.26.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

AI-generated content may be incorrect.

Рисунок 1.26 - Заголовки таблиц в HTML5

Кроме заголовков столбцов с помощью элемента caption можно задать общий заголовок для таблицы. Для объединения столбцов таблицы используется атрибута colspan, который указывает на какое количество столбцов раздвигается данная ячейка. В данном примере футер таблицы содержит только один столбец, который раздвигается по ширине трех столбцов с помощью атрибута colspan="3".

Также с помощью атрибута rowspan можно раздвигать ячейку на определенное количество строк. Далее представлен пример объединения ячеек по столбцам и строкам.

**<style> td { width: 80px; height:80px;**

**border: solid 2px silver;**

**text-align: center; } </style> </head>**

**<body> <h2>Мультфильмы для детей</h2> <table>**

**<tr> <td colspan="2" style="background-color: #bb84e8;"> Лунтик. Возвращение домой</td> <td> Гадкий Я 4</td> </tr>**

**<tr> <td> Головоломка 2</td> <td rowspan="2" style="background-color: #00ffff;"> Большое путешествие. Вокруг света</td> <td> 10 жизней </td> </tr>**

**<tr> <td> Пушистый вояж </td> <td> Кунг-фу Панда 4</td> </tr> </table>**

**</body>**

Реализация данного кода представлена на рисунке 1.27.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.27 - Растяжение ячеек таблицы в html

**1.15 Гипертекстовые ссылки в HTML**

Ссылки, которые представлены элементом <a></a>, играют важную роль - они обеспечивают навигацию между отдельными документами. Основным атрибутом тега является атрибут href, в котором задается определяет адрес ссылки.

Тег а(anchor) делится на видимую часть, то есть то что отображается на экране и скрытую часть.

**<a href=index.html> Ссылка<a>**

Тег <a> помимо атрибута href может иметь следующие:

* hreflang: указывает на язык документа, на который ведет данная ссылка;
* media: определяет устройство, для которого предназначена ссылка;
* rel: определяет отношение между данным документом и ресурсом, на который ведет ссылка;
* target: определяет, как документ по ссылке должен открываться;
* type: указывает на mime-тип ресурса по ссылке.

Наиболее важным из перечисленных является атрибут href. В зависимости от того, какой указан адрес ресурса в атрибуте href ссылки разделяют на абсолютные, относительные и внутренние. Далее представлен пример абсолютной ссылки, в которой в качестве адреса ресурса указан [www.osu.ru](http://www.osu.ru) (рисунок 1.28).

**<body> <a href="http://www.osu.ru/">**

**Сайт Оренбургского государственного университета</a> </body>**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.28 - Ссылки в HTML5

В следующем примере представлен пример использования относительных ссылок, где в качестве адреса ресурса используется относительный путь до файлов glava1.html, glava2.html, glava3.html, которые находятся в той же папке.

**<body> <h2> Содержание курса по Веб-программированию</h2>**

**<a href="glava1.html">Глава 1. Основы HTML.</a>**

**<a href="glava2.html">Глава 2. Введение в JavaScript.</a>**

**<a href="glava3.html">Глава 3. Каскадные таблицы стилей CSS. </a> </body>**

Реализация данного кода представлена на рисунке 1.29.

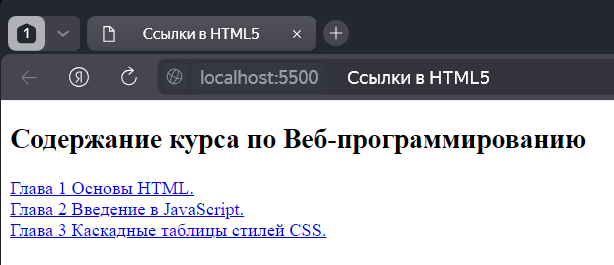


Рисунок 1.29 - Ссылки в HTML5

**1.15.1 Навигация внутри документа**

С помощью тега <а> также можно задать внутренние ссылки, которые будут переходить к определенным блокам внутри элементов.

**<body> <a href="#paragraph1">Глава 1</a> | <a href="#paragraph2">Глава 2</a> | <a href="#paragraph3">Глава 3</a>**

**<h2 id="paragraph1"> Глава 1</h2><p>Содержание главы 1</p>**

**<h2 id="paragraph2"> Глава 2</h2> <p>Содержание главы 2</p>**

**<h2 id="paragraph3"> Глава 3</h2><p>Содержание главы 3</p> </body>**

Реализация данного кода представлена на рисунке 1.30.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.30 - Навигация внутри документа

Чтобы определить внутреннюю ссылку, указывается знак решетки (#), после которого идет id того элемента, к которому надо осуществить переход. В данном случае переход будет идти к заголовкам h2.

**1.15.2 Атрибут target**

По умолчанию ресурсы, на которые ведут ссылки, открываются в том же окне. С помощью атрибута target можно переопределить это действие. Атрибут target может принимать следующие значения:

* \_blank: открытие html-документа в новом окне или вкладке браузера;
* \_self: открытие html-документа в том же фрейме (или окне);
* \_parent: открытие документа в родительском фрейме, если ссылка расположена во внутреннем фрейме;
* \_top: открытие html-документа на все окно браузера;
* framename: открытие html-документа во фрейме, который называется framename.

Например, открытие документа по ссылке в новом окне:

**<a href="http://osu.com/web/html5/" target="\_blank">Учебник по HTML5</a>**

**1.15.3 Стилизация ссылок**

По умолчанию ссылка уже имеет некоторый цвет (один из оттенков синего), кроме того, она имеет подчеркивание. При нажатии на ссылку она становится активной и приобретает красный цвет, а после перехода по ссылке эта ссылка может окраситься в другой цвет (как правило, в фиолетовый). Подобная стилизация задается многими браузерами по умолчанию, но ее можно переопределить. Определяются стили для ссылок в различных состояниях:

a: link применяется для ссылок в обычном состоянии, когда они не нажаты и на них не наведен указатель мыши,

a: visited - указывает на состояние ссылки, по которой уже был осуществлен переход,

a: hover - указывает на состояние ссылки, на которую навели указатель мыши,

a: active - указывает на ссылку в нажатом состоянии,

Далее представлен пример использования всех видов ссылок.

**<head> <style>**

**a:link {color:#5fffaa; text-decoration:none};**

**a:visited {color:#33c1ff; text-decoration:none};**

**a:hover {color:#020096; text-decoration:underline};**

**a:active {color:#aqua; text-decoration:underline}; </style>**

**</head>**

**<body> <h1>Фильмы для всей семьи</h1>**

**<a href="index.html"> Сто лет тому вперед </a>**

**<a href="index.html">Чебурашка</a>**

**<a href="index.html">По щучьему веленью</a>**

**<a href="index.html">Последний богатырь</a> </body>**

Реализация данного кода представлена на рисунке 1.31.

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.31 - Стилизация ссылок

Например, для стиль color: aqua устанавливает цвет для активной ссылки. А стиль text-decoration устанавливает подчеркивание: если значение underline, то ссылка подчеркнута, если none, то подчеркивание отсутствует.

**1.15.4 Ссылка-картинка**

Поместив внутрь элемента <a> элемент <img>, можно сделать ссылку-изображение.

**<a href="aboutcat.html">**

**<img src="cat.jpg" alt="HTML tutorial" style="height: 100px;**

**border: 3px solid black">**

**</a>**

Реализация данного кода представлена на рисунке 1.32.

Изображение выглядит как снимок экрана, мультимедиа, программное обеспечение, Мультимедийное программное обеспечение

AI-generated content may be incorrect.

Рисунок 1.32 – Рисунок-ссылка

В данном примере при нажатии на рисунок-ссылку будет реализован переход на страницу aboutcat.html, в которой описывается информация о кошках.

**1.16 Элементы figure и figcaption**

Элемент figure применяется для аннотирования различных иллюстраций, диаграмм, фотографий и т.д. А элемент figcaption заключает подпись к содержимому внутри элемента figure.

Для использования элемента figure нам надо поместить в него некоторое содержимое, например, изображение.

**<body> <div>**

**<p>Летний пейзаж … </p>**

**<figure>**

**<figcaption>Лето 2024</figcaption>**

**<img src="picture.png" alt="Летний пейзаж">**

**</figure>**

**<p>Летний пейзаж … </p>**

**</div>**

**</body>**

Реализация данного кода представлена на рисунке 1.33.

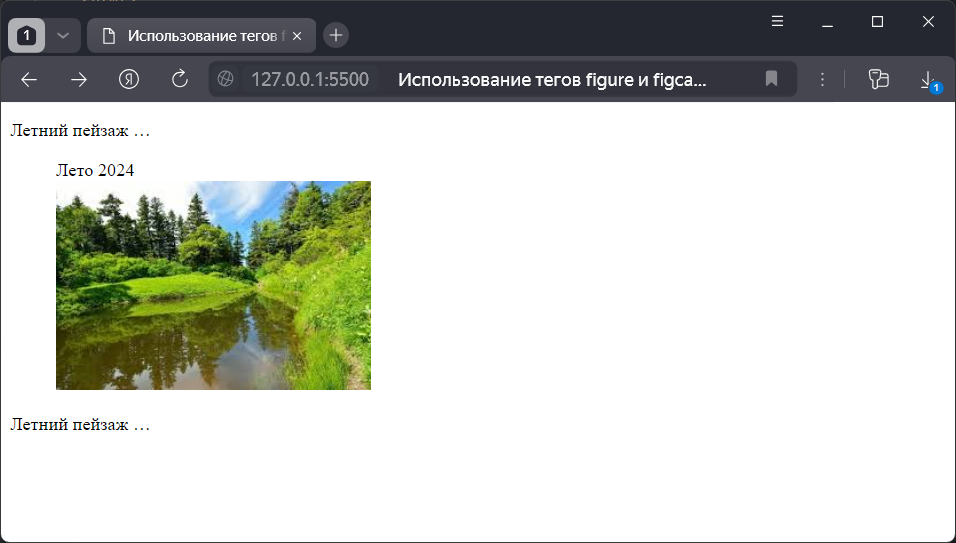


Рисунок 1.33 - Использование тегов figure и figcaption в HTML5

**1.17 Использование плавающих фреймов в HTML**

Плавающие фреймы позволяют встраивать на веб-страницу еще какую-нибудь другую веб-страницу. Фреймы представлены элементом iframe. Элемент iframe не содержит в себе никакого содержимого. Вся его настройка производится с помощью атрибутов:

* src: устанавливает полный путь к загружаемому ресурсу;
* width: ширина фрейма;
* height: высота фрейма.

Например, с помощью использования плавающего фрейма можно встроить на страницу карту размером 560 на 400 пикселей.

**<body> <iframe id="inlineFrameExample"**

**title="Inline Frame Map"**

**width="560"**

**height="400"**

**frameborder="1"**

**allowfullscreen="true"**

**src="https://yandex.ru/map-widget/v1/-/CBFkaYSE0A"> </iframe> </body>**

Реализация данного кода представлена на рисунке 1.34.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.34 – Использование плавающих фреймов

До недавнего времени помимо плавающих фреймов (iframe) использовались фреймы (frames), которые разбивали окно браузера на несколько областей, в каждую из которых загружался свой документ. Однако, на сегодняшний день использование фреймов не рекомендовано из-за сильной перегрузки трафика, а также затруднения работы поисковых машин.