

Лабораторная работа 1. Синтаксис HTML

Базовый синтаксис HTML

Обычно HTML-разметка страницы содержится в файле index.html.

HTML документ состоит из тегов. HTML нечувствителен к регистру в имени тегов!

HTML-документ состоит из дерева HTML-элементов и текста. Каждый элемент обозначается в исходном документе начальным (открывающим) и конечным (закрывающим) тегом (за редким исключением).

Начальный тег показывает, где начинается элемент, конечный — где заканчивается. Закрывающий тег образуется путем добавления слэша / перед именем элемента: <имя элемента>...</имя элемента>. Между начальным и закрывающим тегами находится содержимое элемента — контент.

Каждый HTML документ начинается с тега-декларации <!DOCTYPE html>. Таким образом браузер понимает, что он читает именно HTML. Далее идет корневой тег - <html></html>, в который вложены все остальные теги. В нем находятся два тега <head></head> и <body></body>.

Тег <head> используется для описания метаданных о веб-странице (информация об информации). Эта информация не видна пользователю, а используется браузерами и поисковыми движками.

Один из наиболее важных тегов, используемых в <head> это незакрывающийся <meta>. С его помощью можно задать используемую кодировку, добавить описание страницы, ключевые слова для поисковиков, имя автора.

Например:

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

```
<meta name="description" content="Free Web tutorials">
```

```
<meta name="keywords" content="HTML,CSS,XML,JavaScript">
```

```
<meta name="author" content="John Doe">
```

Внутри <head> можно задавать внутренние стили для документа в тег <style>, внешние стили с помощью тега <link>, указываются скрипты с помощью тега <script>, указать основной URL страницы, относительно которого разрешаются все остальные URL, например: <base href="https://www.w3schools.com/images/" target="_blank">.

Теги бывают с содержимым: <head>Содержимое</head> и пустыми:
. Теги можно вкладывать друг в друга.

Элементы могут вкладываться друг в друга, например, <p><i>Текст</i></p>. При вложении следует соблюдать порядок их закрытия (принцип «матрёшки»), например, следующая запись будет неверной: <p><i>Текст</p></i>.

Внутри тега можно задавать параметры или, что то же самое, атрибуты: <section class="main">. HTML-элементы могут иметь атрибуты (глобальные, применяемые для всех HTML-элементов, и собственные). Атрибуты прописываются в открывающем теге элемента и содержат имя и значение, указываемые в формате имя атрибута="значение". Атрибуты позволяют изменять свойства и поведение элемента, для которого они заданы.

Элементы, представленные одиночными тегами, не могут хранить в себе содержимого напрямую, оно прописывается как значение атрибута, например, элемент <input type="button" value="Кнопка"> создаст кнопку с текстом Кнопка внутри.

Каждому элементу можно присвоить несколько значений class и только одно значение id. Множественные значения class записываются через пробел, <div class="nav top">. Значения class и id должны состоять только из букв, цифр, дефисов и нижних подчеркиваний и должны начинаться только с букв или цифр.

Создание документа HTML5

Элементы являются кирпичиками, из которых складывается документ html5. Для создания документа нам надо создать простой текстовый файл, а в качестве расширения файла указать *.html.

Создадим текстовый файл, назовем его index и изменим его расширение на .html.

Структура документа HTML5

Затем откроем этот файл в любом текстовом редакторе, например, в Notepad++. Добавим в файл следующий текст:

1. `<!DOCTYPE html>`
2. `<html>`
- 3.
4. `</html>`

Для создания документа HTML5 нам нужны 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.

А элемент html между своим открывающим и закрывающим тегами содержит все содержимое документа.

Внутри элемента html мы можем 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.

Теперь изменим содержимое файла index.html следующим образом:

1. `<!DOCTYPE html>`
2. `<html>`
3. `<head>`
4. `<meta charset="utf-8">`
5. `<title>Документ HTML5</title>`
6. `</head>`
7. `<body>`
8. `<div>Содержание документа HTML5</div>`
9. `</body>`
10. `</html>`

В элементе head определено два элемента:

- элемент title представляет заголовок страницы
- элемент meta определяет метаинформацию страницы. Для корректного отображения символов предпочтительно указывать кодировку. В данном случае с помощью атрибута charset="utf-8" указываем кодировку utf-8.

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

Поскольку мы выбрали в качестве кодировки utf-8, то браузер будет отображать веб-страницу именно в этой кодировке. Однако необходимо чтобы сам текст документа также соответствовал выбранной кодировке utf-8. Как правило, в различных текстовых редакторах есть соответствующие настройки для установки кодировки. Например, в Notepad++ надо зайти в меню Кодировки и в открывшемся списке выбрать пункт Преобразовать в UTF-8 без BOM:

Установка кодировки html-файла

После этого в статусной строке будет можно будет увидеть UTF-8 w/o BOM, что будет указывать, что нужная кодировка установлена.

Сохраним и откроем файл index.html в браузере:

Документ HTML5

Таким образом, мы создали первый документ HTML5. Так как мы указали в элементе title заголовок "Документ HTML5", то именно такое название будет иметь вкладка браузера.

Так как указана кодировка utf-8, то веб-браузер будет корректно отображать кириллические символы.

А весь текст, определенный внутри элемента body мы увидим в основном поле браузера. 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.

Добавим в документ ряд элементов meta:

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="utf-8">
5. <base href="content/">
6. <title>Элемент title</title>
7. <meta name="description" content="Первый документ HTML5">
8. <meta name="author" content="Bill Gates">
9. </head>
10. <body>
11. Содержание документа HTML5
12. </body>
13. </html>

Параграфы

Параграфы создаются с помощью тегов <p> и </p>, которые заключают некоторое содержимое. Каждый новый параграф располагается на новой строке.

<p>Первый параграф</p>

<p>Второй параграф</p>

Если в рамках одного параграфа нам надо перенести текст на другую строку, то мы можем воспользоваться элементом
:

<p>Первая строка.
Вторая строка.</p>

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

<pre>

Первая строка

Вторая строка

Третья строка

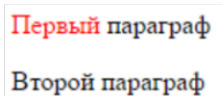
```
</pre>
```

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:

```
<p><span style="color:red;">Первый</span> параграф</p>
```

```
<p><span>Второй</span> параграф</p>
```

Итог:



Первый параграф

Второй параграф

Элементы `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>` и `<h6>` служат для создания заголовков различного уровня.

Ряд элементов `html` предназначены для форматирования текстового содержимого, например, для выделения жирным или курсивом и т.д. Рассмотрим эти элементы:

``: выделяет текст жирным

``: зачеркивает текст

`<i>`: выделяет текст курсивом

``: 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

``: выделяет текст жирным. В отличие от тега `` предназначен для логического выделения, чтобы показать важность текста. А `` не носит характера логического выделения, выполняет функции только форматирования

`<sub>`: помещает текст под строкой

`<sup>`: помещает текст над строкой

`<u>`: подчеркивает текст

`<ins>`: определяет вставленный (или добавленный) текст

`<mark>`: выделяет текст цветом, придавая ему оттенок важности

Для вывода изображений в `HTML` используется элемент `img`. Этот элемент представляет нам 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 по каким-то причинам не может отобразить изображение (например, если у атрибута `src` некорректно задан путь), то браузер показывает вместо самой картинки данное текстовое описание.

Атрибут `alt` еще важен тем, что поисковые системы по текстовому описанию могут индексировать изображение.

Например, положим в ту же папку, где у нас лежит файл `index.html`, какой-нибудь файл изображения. И затем отобразим его на веб-странице:

```

```

Для создания списков в `HTML5` применяются элементы `` (нумерованный список) и `` (нечисловой список):

```
<h2>Нумерованный список</h2>
```

```
<ol>
```

```
<li>iPhone 6S</li>
```

```

    <li>Galaxy S7</li>
    <li>Nexus 5X</li>
    <li>Lumia 950</li>
</ol>
<h2>Ненумерованный список</h2>
<ul>
    <li>iPhone 6S</li>
    <li>Galaxy S7</li>
    <li>Nexus 5X</li>
    <li>Lumia 950</li>
</ul>

```

Итог:



Элемент details позволяет создавать раскрываемый блок, который по умолчанию скрыт. Иногда такой элемент еще называют accordion.

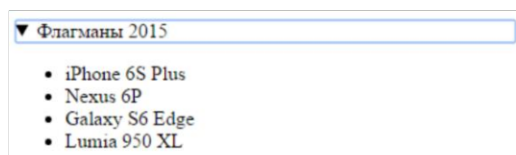
Данный элемент содержит элемент summary, который представляет заголовок для блока, и этот заголовок отображается в скрытом режиме. Например:

```

<details>
    <summary>Флагманы 2015</summary>
    <ul>
        <li>iPhone 6S Plus</li>
        <li>Nexus 6P</li>
        <li>Galaxy S6 Edge</li>
        <li>Lumia 950 XL</li>
    </ul>
</details>

```

Итог:



Элемент figure применяется для аннотации различных иллюстраций, диаграмм, фотографий и т.д. А элемент figcaption просто обортывает заголовок для содержимого внутри элемента figure.

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

```

<p>Lorem ipsum dolor ... </p>
<figure>
    <figcaption>Февраль 2013</figcaption>

```

```

</figure>
<p>Lorem ipsum dolor ... </p>
```

Итог:



Встроенный CSS

Встроенный CSS используется для применения уникального стиля к одному элементу HTML.

Встроенный CSS использует атрибут `style` элемента HTML.

Задний фон

Задний фон элемента задается с помощью CSS свойства `background-color`, которое в качестве значения может принимать любое доступное значение цвета (про различные значения цветов смотрите ниже), после свойства обязательно должно идти двоеточие и значение, после значения обязательно ставится точка с запятой, эти правила распространяются на все свойства атрибута `style`. Если нужно задать несколько свойств стиля для одного элемента, то каждое последующее свойство записывается после `;` предыдущего свойства.

Цвет текста

Цвет текста элемента задается с помощью CSS свойства `color`. Благодаря этому свойству можно задать любой цвет для текстового содержимого. В качестве значения свойство `color` может принимать имена цветов, RGB значения или шестнадцатеричные коды.

Установка цвета по имени

Наиболее простым способом задать цвет в CSS является указание его имени.

Предположим, вы хотите задать серебристый цвет для текста в элементе:

```
color: silver;
```

Итак, чтобы задать цвет таким образом, нужно просто указать его название в качестве значения свойства. При этом не имеет значения, пишете вы названия строчными или прописными буквами, поэтому можно написать `silver`, `Silver` или `SILVER`, и все это будет работать.

Установка цвета с помощью RGB

Система RGB использует три числа, которые описывают относительное количество красного, зеленого и синего цветов, которые смешаны вместе для получения любого оттенка. Числа могут варьироваться от 0 до 255 для числовых значений или от 0% до 100%

Можно установить цвет, указав сочетание красного, зеленого и синего в определенной пропорции. Допустим, вам нужно задать оранжевый цвет, который состоит из 80% красного, 40% зеленого и 0% синего. Вот как это можно сделать:

```
color: rgb(80%, 40%, 0%);
```

Можно также задавать значение красного, зеленого и синего числами от 0 до 255. Например, вместо 80% красного, 40% зеленого и 0% синего можно написать 204 красного, 102 зеленого и 0 синего:

```
color: rgb(204, 102, 0);
```

Шестнадцатеричные коды

Теперь перейдем к шестнадцатеричным кодам. Откроем вам небольшой секрет: каждый набор двух цифр такого кода представляет красную, зеленую и синюю составляющую цвета. Так, первые две цифры представляют красный цвет, следующие две - зеленый и последние две - синий:

```
color: #cc6600;
```

Шрифт

Шрифты могут очень сильно влиять на дизайн страниц. В CSS они разделены на семейства, в которых вы можете выбрать, какой шрифт лучше подходит для определенного элемента на странице.

```
font-family: Verdana, Geneva, Arial, sans-serif;
```

Свойство font-family дает возможность задать список предпочтительных шрифтов, которые указываются через запятую. Если имя шрифта состоит из нескольких слов, то такое название следует заключить в двойные кавычки, например: "Courier New".

Размер

Размер шрифта очень сильно влияет на дизайн веб-страницы и читабельность ее текста. В CSS есть несколько единиц измерения, с помощью которых можно задавать размер для текста, все они подробно описаны в разделе Единицы измерения CSS. Также имеется возможность задавать размер с помощью ключевых слов:

```
font-size: medium;
```

Все доступные ключевые слова, задающие размер, вы можете посмотреть в нашем справочнике по CSS в описании свойства font-size.

Выравнивание текста

Выравнивание текста в HTML документе задаётся с помощью свойства text-align, которое позволяет выравнивать текст по правой или левой стороне, а так же задать выравнивание текста по ширине. Свойство text-align работает только с блочными элементами, выравнивая все строчные элементы внутри блочного:

```
<html>
```

```
<body style="background-color: DarkGray; color: white;">
```

```
<h1 style="font-family: verdana;">Заголовок</h1>
```

```
<p style="font-size: 10px;">Очень маленький размер текста.</p>
<p style="text-align: right;">Этот текст будет выровнен по правому краю.</p>

</body>
</html>
```

В этом примере устанавливается цвет текста элемента <h1> синим цветом:

```
<h1 style="color:blue;">This is a Blue Heading</h1>
```

Другие примеры:

```
<h1 style="font-size: 36px; font-family: Times, serif; color: red">Заголовок 1</h1>
```

```
<p style="font-size: 120%; font-family: monospace; color: #cd66cc">Пример текста</p>
```

Критерии оценки:

«5» - создан файл HTML, использованы пять типов метаданных, на странице присутствует минимум: 3 разных заголовка, 2 параграфа, 2 изображения, нумерованный список, ненумерованный список, 2 figure, 2 details.

Все элементы имеют минимум по 3 встроенных CSS.

«4» - создан файл HTML, использованы три типа метаданных, на странице присутствует минимум: 2 разных заголовка, 2 параграфа, 2 изображения, нумерованный список, 2 figure, 2 details.

Все элементы имеют минимум по 2-3 встроенных CSS.

«3» - создан файл HTML, использованы три типа метаданных, на странице присутствует минимум: 2 разных заголовка, 2 параграфа, 2 изображения, нумерованный список, 2 figure, 1 details.

Все элементы имеют минимум по 1-2 встроенных CSS.