#1 Введение в HTML и CSS

HTML

Как написать свою веб-страничку?

Сайт состоит из множества страниц и доступен по какому-то адресу. Например, [www.example.com](http://www.example.com/)

Каждая страница имеет свой какой-нибудь адрес и содержит какую-то информацию: приветствие, контактные данные, галерею картинок, ссылки на другие сайты и так далее.

Чтобы создать html-страницу, достаточно создать простой текстовый документ, расширение которого изменить с .txt на .html

Graphical user interface, application

Description automatically generated

Обратите внимание, что как только вы сохранили документ с расширением .html, его значок изменился на значок вашего браузера по умолчанию, в нашем случае это значок Chrome

Graphical user interface, text

Description automatically generated

Теперь нужно открыть этот документ в любом текстовом редакторе (мы рекомендуем вам sublime text, но вы можете воспользоваться даже самым простым блокнотом) и написать любой текст.

Graphical user interface, text

Description automatically generated

Всё! По сути, это веб-страница. Просто находится она не в интернете, а у нас на компьютере. Если открыть такой файл через веб-браузер, то можно увидеть текст “Hello world”.

Но это пока страница-калека. Чтобы стать настоящей взрослой веб-страницей, нужно сделать ее разметку  с помощью HTML-тегов.

**HTML-теги**

HTML-теги это метки, которые используются для указания браузеру, как он должен показывать web-сайт. С помощью тэгов верстальщик  пишет  разметку, “скелет” будущей странички.

Все тэги имеют одинаковый формат: они начинаются знаком "<" и заканчиваются знаком ">", например, если вы хотите создать заголовок для вашей статьи , используйте специальный тег заголовка h1.

В текстовом редакторе вы напишете  <h1>Hello, World!</h1>, и теперь браузер будет воспринимать ваш текст как заголовок, увеличив его размер

Graphical user interface, text, application

Description automatically generated

Обычно имеются два тэга - открывающий: <h1> и закрывающий: </h1>. Различие в том, что в закрывающем имеется слэш(наклонная черта) "/". Заметьте, что написанное в текстовом редакторе выводится в браузер не полностью - мы не видим самих тегов, и видим только текст, заключенный в тег.

Всё содержимое, находящееся между открывающим и закрывающим тегами, является содержимым тега. В нашем примере содержимым тега h1 является текст “Hello World!”, и только оно выводится пользователю, а сами теги являются служебной информацией для браузера.

В некоторых тегах содержимое не предусмотрено, и тогда им не нужен закрывающий тег. Примером может послужить тег перехода на новую строку <br>. Этот тег работает как клавиша Enter на вашей клавиатуре, т.е. переводит текст на новую строку. Давайте посмотрим, как этот тэг сработает в нашей страничке - попробуем вставить тег <br> в содержимое нашего заголовка:

Обратите внимание, что в браузере вместо текста <br> получился переход на новую строку - тег сработал. Также вы, наверное, заметили, что мы поставили тег <br> **внутрь** заголовка h1 и вы, конечно, догадались, что теги можно вкладывать друг в друга.

Graphical user interface, text, application

Description automatically generated

Давайте рассмотрим вложенность тегов подробнее на примере. Предположим, нам нужно, чтобы наш заголовок был не только заголовком, но и ссылкой, и при нажатии на него мы могли перейти куда-то (прочитать статью, посмотреть картинку, перейти на другую страницу)

Мы можем вложить тег ссылки <a></a> в тег заголовка  <h1></h1>:

Graphical user interface, application

Description automatically generated

Цвет нашего текста изменился, появилось подчеркивание, теперь мы можем на него нажать - все эти **свойства** добавил нашему тексту тег <a></a>. Это стандартные свойства для ссылок в браузере. Однако, если мы нажмем на наш текст, мы никуда не перейдем, не увидим новую страницу, а ведь мы так этого хотели.

Если вы достаточно внимательно посмотрите на разметку в текстовом редакторе, вы увидите, что в открывающем теге ссылки есть загадочный текст **href = “ ”**  - это **атрибут** тега. Атрибут - это специальный текст внутри тэга, который содержит служебную информацию для браузера, т.е. пользователь не видит этой информации.

В самом деле, на нашей страничке мы увидели только содержимое тэга, никаких лишних букв.  Атрибут href = “ ” в ссылке хранит адрес, по которому пользователь перейдет, если нажмет на ссылку. Добавим любой полный адрес между кавычками атрибута href (просто скопируйте из адресной строки вашего браузера адрес любого сайта)

Graphical user interface, text

Description automatically generated

Теперь по нажатию на ссылку ваш браузер переведет вас на указанную страницу - сработает атрибут.

Для того, чтобы создавать полноценные html-страницы, нам нужно будет изучить около 50 тегов и около 20 атрибутов к ним.

Основные теги можно разбить на несколько групп:

* **блоки div**
* **текстовые теги:**
  + **заголовки h1, h2, h3, h4, h5, h6**
  + **списки ul-li, ol-li**
  + **текстовые span, a, b, strong, i, em**
* **табличные: table, tr, td, th**
* **картинки: img**
* **формы: form, input type=”text”, select, textarea…**

**Кроме того, для хорошей странички нам не хватает начальной разметки**

**Обычно верстальщик первым делом, еще до того, как вносить теги содержимого страницы, создает в начальную разметку. По сути это набор тегов, необходимых для корректной работы любой страницы, выглядит он примерно так:**

Text

Description automatically generated

**Рассмотрим каждый элемент по порядку:**

**<!DOCTYPE html> - описание типа документа, задает стандарт, по которому браузер отображает страницу. Просто запоминаем и всегда пишем этот текст в начале документа**

**<html></html> - тег, который содержит в себе всю страницу, обязательная обертка**

**<head></head> - “голова” страницы, содержит служебную информацию для браузера, например, кодировку (<meta**

**charset="UTF-8">) и название страницы (в теге <title></title>), которое отображается на вкладке**

**<body></body> - это тело страницы, в него мы будем вкладывать большинство тегов, атрибутов и содержимого. Так, мы можем вложить наш заголовок, и это уже будет почти настоящая сверстанная страничка:**

Text

Description automatically generated

Для настоящей странички нам не хватает только **стилей**

CSS

CSS (Cascading Style Sheet) - это язык разметки, ответственный за внешний вид нашей странички. Если до этого с помощью тегов мы создавали “скелет” странички, обозначали суть каждого элемента, то теперь мы будем описывать, как каждый элемент будет выглядеть, добавим скелету “мышц” и “одежды”. Сделаем страницу разноцветной, расположим блоки так, как нам удобно, “нарисуем” красивые кнопки и т.д.

Большая часть работы и искусства верстки заключается не столько в написании html,  сколько в умении правильно написать стили в css. Одна и та же разметка html может выглядеть совершенно по-разному, если написать для нее разные стили.

Diagram

Description automatically generated

Посмотрите, например на кнопки слева - их разметка идентична, это обычная ссылка <a href= “”>click me</a>, она выглядит, как первая синяя подчеркнутая ссылка.

Но они ведь совершенно не похожи ни друг на друга, ни на оригинальный вариант ссылки! Все дело в стилях css, для каждой кнопки используется разный набор css-правил  и в итоге, мы получаем из одной html-разметки три разных по внешнему виду элемента.

Как написать свой CSS для html-страницы?

Cоздаем в папке с html-файлом новый текстовый файл с расширением .css

Graphical user interface, application

Description automatically generated

До того, как мы напишем стили в этом файле, нам нужно соединить два файла в один проект - прописать путь к файлу css в файле html вот так:

Text

Description automatically generated

Мы написали в разделе head специальный тег link (он не требует закрывающего тэга)

<link rel="stylesheet" href="style.css">

В нем есть два атрибута:

rel= "", в котором мы указываем, что за документ мы “подключили” к файлу html, и

href= "", знакомый нам атрибут, в котором прописывается путь к файлу и название самого файла.

Теперь браузер сможет соединить информацию разметки и стилей вместе и выдать нам красивую страницу.

Что и как можно сделать в файле стилей?

Итак, давайте попробуем написать первые стили. Синтаксис у стилей очень простой, выглядит примерно так:

**селектор {свойство: значение;}**

Селектор (от англ select - выбирать) определяет, к какому элементу будут применены стили, написанные в фигурных скобках.

Свойство - основа языка css, определяет, какое именно свойство будет изменено. Разные свойства могут изменять цвет, размер, жирность текста, цвет заднего фона, цвет и размер границы блока, размер блока и еще много-много других вещей.

Значение обозначает, как именно будет изменено свойство - красный цвет текста, размер шрифта в пикселях, размер блока в половину ширины экрана и т.п.

**Селекторы:**

\* - универсальный селектор, применит свойства ко всем элементам на странице.

Например, мы хотим сделать весь текст на всей странице красным:

Text

Description automatically generated

В стилях мы записали

\* {color : red ;} - что в переводе с языка верстки на русский означает:  всем элементам страницы { цвет текста: красный; }

Но обращение ко всем тегам страницы не всегда удобно: что если мы хотим убрать подчеркивание у ссылки внутри заголовка и добавить параграфам желтый фон:

Text

Description automatically generatedText

Description automatically generated with medium confidence

Изменим наши стили:

p {background: yellow;}   -----------  (всем параграфам {фон: желтый} )

h1 a {text-decoration: none;} ------  (ссылке внутри заголовка {украшение текста: никакое} )

В этом случае селекторы - это **p** (все теги <p></p> на странице) и **h1 a** (все ссылки <a></a> внутри всех заголовков <h1></h1>). Также можно обращаться к любым тегам на странице по названию тэга.

Но это тоже не всегда удобно - представим, что первому параграфу нам нужно задать синий цвет текста. В этом случае мы не можем обратиться ко всем параграфам, и тут нам на помощь приходят классы (атрибут class="" ) - изменим немного html

A screenshot of a computer

Description automatically generated with medium confidence

Мы добавили в первый тэг параграфа атрибут class="myclass" (название класса myclass может быть любым), и обратились к нему в стилях .myclass. Мы могли также добавить атрибут id="my-id" (название класса my-id может быть любым), и обратиться к элементу по айди:  #my-id. Принципиальная разница между классом и айди в том, что один и тот же класс может быть у нескольких объектов, тогда как название айди - уникально и есть только у одного элемента на странице.  В верстке не стоит использовать айди, лучше пользоваться классами.

Несколько примеров обращения к элементам html-страницы в css:

|  |  |  |
| --- | --- | --- |
| HTML | Обращение в CSS | |
| Обращение по тегу: | | |
| <div></div> | div {color: red;} | найти на странице все дивы  и сделать текст в них красным |
| <b></b> | b{font-size: 16px;} | найти на странице все элементы <b>  и сделать размер текста в них 16px |
| <div class= “content”></div>    <section class=”content”></section> | .content {background: green;} | найти на странице все элементы с классом content и задать им зеленый фон (зеленый фон появится и у дивов с классом content и у <section> с классом content |
| <p class= “descr”></p>    <span class=”descr></span> | p.txt {color: yellow;} | найти все элементы <p> с классом descr и сделать текст в них желтым (желтым станет только текст размещенный внутри тегов <p> с классом descr, на <span> c таким же классом это css-правило не подействует) |
| <a id= “main-link”></a> | #my-id {font-weight: bold;} | найти элемент с id main-link и отобразить текст этого элемента жирным |
| <div class= “box”>      <span></span>  </div> | .box span {font-style: italic;} | Найти span, который размещен внутри элемента с классом box и изменить начертание текста внутри этого span на italic |

Синтаксис основных тегов

Основной блочный тег

|  |  |  |
| --- | --- | --- |
| Название тэга | HTML синтаксис | CSS синтаксис |
| блок | <div></div> | div |

Текстовые теги

|  |  |  |
| --- | --- | --- |
| Название тэга | HTML синтаксис | CSS синтаксис |
| абзац | <p></p> | p |
| универсальный текстовый элемент | <span></span> | span |
| жирный текст | <b></b>  <strong></strong> | b  strong |
| ссылка | <a href=""></a> | a |
| курсив | <i></i>  <em></em> | i |
| Заголовки 1,2,3,4,5,6 уровня | <h1></h1>  <h2></h2>  <h3></h3>  <h4></h4>  <h5></h5>  <h6></h6> | h1  h2  h3  h4  h5  h6 |
| Переход строки | <br> | - |

Маркированные и нумерованные списки:

|  |  |  |
| --- | --- | --- |
| Маркированный список | <ul></ul> | ul |
| Нумерованный список | <ol></ol> | ol |
| Один пункт нумерованного или ненумерованного списка | <li></li> | li |

Картинка:

|  |  |  |
| --- | --- | --- |
| Картинка | <img src="" alt=""> | img |

Пример:  <img src="images/logo.jpg" alt="Logo">

В примере src="images/logo.jpg" в кавычках написан путь к файлу и его название: файл с картинкой называется logo.jpg и находится в папке images. При этом папка images находится в той же директории(папке), что и файл html

Таблицы:

|  |  |  |
| --- | --- | --- |
| Тег таблицы | <table></table> | table |
| Одна строка в таблице | <tr></tr> | tr |
| Один столбец в строке таблицы | <td></td> | td |

Пример таблицы в 4 строки и 2 столбца:

<table>

<tr>

<td>Name</td>

<td>John</td>

</tr>

<tr>

<td>Lastname</td>

<td>Doe</td>

</tr>

<tr>

<td>Profession</td>

<td>Ruby programmer</td>

</tr>

<tr>

<td>Phone</td>

<td>+007007007</td>

</tr>

</table>

Теги форм:

|  |  |  |
| --- | --- | --- |
| Название тэга | HTML синтаксис | CSS синтаксис |
| форма | <form></form> | form |
| поле ввода текста | <input type="text"> | input[type="text"] |
| поле ввода электронной почты | <input type="email"> | input[type="email"] |
| поле ввода пароля | <input type="password"> | input[type="password"] |
| выпадающие списки | <select name="">    </select> | select |
| пункт выпадающего списка | <option value="">Text</option>  <option value="">Text</option> | option |
| пункт списка с множественным выбором | <input type="checkbox" name="" value=""> | input[type="checkbox"] |
| пункт списка с единственным выбором | <input type="radio" name="" value=""> | input[type="radio"] |
| область комментария | <textarea name="" id="" cols="30" rows="10"></textarea> | textarea |
| кнопка отправки формы | <input type="submit"> | input[type="submit"] |

Синтаксис основных CSS свойств

Свойства текста:

|  |  |  |
| --- | --- | --- |
| **font-family** | font-family: Arial, Helvetica, sans-serif  font-family: Times New Roman, serif | семейство шрифта |
| **font-size** | font-size: 30px | размер шрифта |
| **font-weight** | font-weight: bold  font-weight: normal | вес шрифта, его жирность |
| **font-style** | font-style: normal  font-style: italic | начертание |
| **font** | font: Arial, sans-serif italic bold 12px | все свойства в одном |
| **color** | color: blue  color: rgb(0,0,255)  color: #666 | цвет текста |
| **line-height** | line-height: 20px  line-height: 2 | междустрочный интервал |
| **text-decoration** | text-decoration: none  text-decoration: underline | подчеркивание текста |
| **text-transform** | text-transform: uppercase  text-transform: lowercase  text-transform: capitalize | Трансформирует текст - все строчные, все заглавные.. |
| **text-align** | text-align: left  text-align: right  text-align: center | Выравнивание текста |

Свойства фона и границ:

|  |  |  |
| --- | --- | --- |
| **background** | background: red  background: #fff  background: rgb(150, 250, 250)  background: rgba (250, 250, 250, 0.5)  background: url(../images/bg.png) center center no-repeat  background: url(bg.png) 30% 50% repeat-x  background url(bg.png) 0px 0px repeat | Задний фон |
| **border** | border: 1px solid #ccc  border: 1px dotted #ccc  border: 1px dashed #ccc | граница |
| **border-radius** | border-radius: 5px  border-radius: 50% | скругление углов |
| **border-bottom**  **border-top**  **border-right**  **border-left** | border-bottom: none  border-top: 5px solid red  border-right: 2px dashed #ccc  border-left: 1px dotted yellow | каждая отдельная граница |

Свойства блоков:

|  |  |  |
| --- | --- | --- |
| **padding** | padding: 10px  padding: 10px 20px  padding: 10px 20px 15px  padding: 10px 20px 15px 25px | Отступы внутри блока |
| **margin** | margin: 10px  margin: 10px 20px  margin: 10px 20px 15px  margin: 10px 20px 15px 25px | Отступы снаружи блока |
| **width** | width: 100%  width: 300px | Ширина |
| **height** | height: 100%  height: 300px | Высота |
| **display** | display: block  display: inline  display: table  display: inline-block  display: none | Отображение элемента:  блочным, строчным, таблицей, строчно-блочным |

Свойства списков ul (ol):

|  |  |  |
| --- | --- | --- |
| **list-style** | list-style: none  list-style: square  list-style: url(image.jpg) | Задает свойства маркера спискам |
| **list-style-type** | ul.circle {list-style-type: circle;}  ul.square {list-style-type: square;}  ol.upper-roman {list-style-type: upper-roman;}  ol.lower-alpha {list-style-type: lower-alpha;} | Задает свойства маркера спискам |
| **list-style-position** | list-style-position: inside  list-style-position: outside | Положение маркеров списка относительно самого списка |