Geekbrains

Специальность: Frontend-программист Цифровые профессии

Тема дипломного проекта: «Профессиональная верстка: HTML и СSS»

ФИО студента: Дьячков Антон Геннадьевич

г. Северодвинск

2023

Содержание дипломного проекта

Введение 4

1. HTML 5
   1. Основы HTML 5
   2. Работа с атрибутами 8
   3. Работа с текстом 9
   4. Работа со ссылками 11
   5. Работа с изображениями 14
   6. Работа с таблицами 15
   7. Работа со списками 18
   8. Полезные ссылки 19
2. CSS 20
3. Javascript
4. Заключение
5. Список использованной литературы и ресурсов
6. Приложения
7. Презентацию к дипломному проекту

Введение

Данный дипломный проект - небольшая практическая работа. В ней предполагается использовать все полученные знания и закрепить теорию практикой, а результат положить в портфолио. Защита дипломного проекта является обязательной итоговой аттестацией, после прохождения технологических специализаций (Frontend-программист. Цифровые профессии) для получения диплома после обучения.

Тема проекта была выбрана автором самостоятельно с целью объединить знания о самых основных и минимально необходимых инструментах профессиональной верстки сайтов, полученных на курсе, интересы и некоторый опыт работы по верстке сайтов с соблюдением следующих условий: тема актуальна, реализуема и практична.

Используемые инструменты - HTML, CSS, Git, Visual Studio Code.

Проект представляет из себя сжатую инструкцию по использованию основных инструментов, необходимых для верстки сайтов - HTML, CSS. Выбранная тема призвана дать самое общее представление о вышеуказанных инструментах и возможность использования в практической работе имеющихся сведений для верстки сайтов. Если же рассматривать каждый инструмент обстоятельно и подробно, то общий объем данного проекта выйдет далеко за свои рамки. Для более глубокого и подробного понимания предлагается использовать список литературы и источников, приведенных в Разделе 5.

1. HTML

1.1. Основы HTML

HTML (от англ. HyperText Markup Language - «язык гипертекстовой разметки») это как известно основной язык Всемирной сети INTERNET. В данном разделе содержатся основные правила языка HTML, описание структуры HTML-страницы, отношения в структуре документа между его элементами.

Гипертекст – система текстовых страниц, соединенных между собой ссылками.

HTML-документ - это текстовый документ, который можно создать в обычном текстовом редакторе (Блокнот) или в специализированном, с подсветкой кода (Notepad++, Visual Studio Code и т.п.). HTML-документ имеет расширение .html. Создав файл с таким расширением можно открыть его через любой веб браузер.

HTML-документ состоит из дерева HTML-элементов и текста. Каждый элемент обозначается в исходном документе начальным (открывающим) и конечным (закрывающим) тегом (с редкими исключениями).

Начальный тег показывает, где начинается элемент, конечный - где заканчивается. Например, как представлено на Рисунке 1.1.

Рисунок 1.1

<p> Какой то текст</p>

Где <p> является открывающим тегом, а </p> - закрывающим. Браузер при этом выведет только все содержимое или контент (текст, название, картинка, видео и т.д.).

Элементы могут вкладываться друг в друга по принципу «матрёшки».

HTML-элементы могут иметь атрибуты (применяемые для всех HTML-элементов, и собственные). Атрибуты позволяют изменять свойства и поведение элемента, для которого они заданы.

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.

Браузер просматривает (интерпретирует) HTML-документ, выстраивая его структуру и отображая ее в соответствии с инструкциями, включенными в этот файл (CSS). Если разметка правильная, то в окне браузера будет отображена HTML-страница, содержащая HTML-элементы - заголовки, таблицы, изображения и т.д.

Процесс интерпретации начинается прежде, чем веб-страница полностью загружена в браузер. Браузеры обрабатывают HTML-документы последовательно, с самого начала, при этом обрабатывая CSS и соотнося таблицы стилей с элементами страницы.

HTML-документ состоит из двух разделов - заголовка - содержимое элемента <head> и содержательной части - содержимое <body>.

Создадим файл index.html в редакторе кода Visual Studio Code и сформируем самую базовую структуру html-документа как представлено на Рисунке 1.2

Рисунок 1.2

<!DOCTYPE html><!-- Объявление формата документа -->

<html lang="en">

<head><!-- Техническая информация о документе -->

    <meta charset="UTF-8"><!-- Определяем кодировку символов документа -->

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title><!-- Задаем заголовок документа -->

    <link rel="stylesheet" type="text/css" href="style.css"> <!-- Подключаем внешнюю таблицу стилей (CSS) -->

    <script src="script.js"></script> <!-- Подключаем сценарии JavaScript -->

</head>

<body><!-- Основная часть документа -->

</body>

</html>

Запись «<!-- Объявление формата документа -->» является комментарием и не отображается непосредственно на странице веб сайта и служит пояснением для самого программиста или других разработчиков кода.

Элементы, находящиеся внутри элемента <html>, образуют дерево (**модель) документа**. При этом сам элемент <html> является корневым элементом.

Взаимодействие элементов веб-страницы, происходит в отношениях между элементами, подразделяющиеся на родительские, дочерние и сестринские:

- предок - элемент, который заключает в себе другие элементы (на Рисунке 1.2 предком для всех элементов является <html> и в то же время элемент <body> является предком для всех содержащихся в нем самом элементов);

- потомок - элемент, расположенный внутри одного или более типов элементов (<body> является потомком <html>, а все элементы в нем являются потомками одновременно для <body> и <html>);

- родительский элемент - элемент, связанный с другими элементами более низкого уровня, и находящийся на дереве выше их (на рисунке 1.1  <html> является родительским только для <head> и <body>);

- дочерний элемент - элемент, непосредственно подчиненный другому элементу более высокого уровня (на Рисунке 1.2 все элементы внутри <body> являются дочерними по отношению к нему);

- сестринский элемент - элемент, имеющий общий родительский элемент с рассматриваемым, так называемые элементы одного уровня (на рисунке 1.1 <head> и <body> являются элементами одного уровня).

Кроме элемента html в любом HTML-документе также обязательно присутствуют:

- head (<head>…</head>) – в нем содержится техническая информация о странице: заголовок, описание, ключевые слова для поисковых машин, кодировку и т.д., а введенная в нем информация не отображается в окне браузера, но указывает браузеру, как следует обрабатывать страницу;

- title (<title>…</title>) – отображает текст в строке заголовка веб-браузера и содержит максимально полное описание содержимого веб-страницы;

- meta (<meta>) – здесь задается описание содержимого страницы и ключевые слова для поисковых машин, автора HTML-документа и прочие свойства метаданных и при этом элемент <head> может содержать несколько элементов <meta>, поскольку в зависимости от используемых атрибутов они несут различную информацию т.е. может иметь множество различных атрибутов;

- style (<style>…</style>) - здесь задаются стили, которые используются на странице, но в HTML-документе лучше всего использовать все же CSS;

- link (<link>) - подключение к html-документу других исполняемых файлов – например файл со стилями style.css, где href является основным атрибутом элемента, где в качестве значения выступает путь к файлу со стилями;

- script (<script>) - присоединяет к документу различные сценарии, написанные на языке программирования. Текст сценария может располагаться либо внутри этого элемента, либо во внешнем файле;

- body (<body>…</body>) - здесь располагается все содержимое документа или контент.

Это лишь часть основных элементов стандартной html-страницы.

1.2. Работа с атрибутами

В HTML существует значительное количество различных атрибутов. Многие из них дублируются в различных тегах (например, title - добавляет всплывающую подсказку и используется в большинстве HTML тегов) и запоминаются по мере изучения HTML.

HTML-атрибуты это специальные слова, которые управляют поведением HTML-элемента. Они добавляют дополнительную функциональность, либо меняют поведение элемента по умолчанию. Атрибуты элемента выражаются внутри начального тега элемента.

Некоторые из самых необходимых атрибутов:

- class (class="…") - список классов элемента с учетом регистра (позволяют CSS и Javascript выбирать и получать доступ к элементам с помощью селекторов классов или функций);

- id (id="…") - уникальный идентификатор элемента для идентификации элемента;

- lang (lang="en-GB") - указывает основной язык для содержимого элемента и для любого из атрибутов элемента, содержащих текст;

- style (style="color: blue; background: transparent") - содержит объявления стилей CSS, которые следует применить к конкретному элементу (имеется также тег style, который задает стиль для всей html страницы);

- title (title="Hypertext Transport Protocol") - содержит дополнительную информацию об элементе, задавая всплывающую подсказку для страницы.

1.3. Работа с текстом

Текст является одним из основных типов контента (содержимого) сайта и сделать его ярче и лучше для восприятия можно за счет добавления к нему специальных HTML тегов.

В HTML можно создавать заголовки за счет тегов h1-h6. Они позволяют создать заголовки разных размеров (от большего к меньшему) как показано на Рисунке 1.3.

Рисунок 1.3

<h1>Заголовок 1 уровня</h1>

<h2>Заголовок 2 уровня</h2>

<h3>Заголовок 3 уровня</h3>

<h4>Заголовок 4 уровня</h4>

<h5>Заголовок 5 уровня</h5>

<h6>Заголовок 6 уровня</h6>

Напоминают разделы и подразделы в данном дипломном проекте. Они упорядочивают текст, формируя его визуальную структуру. Элементы должны использоваться только для выделения заголовков нового раздела или подраздела. От самого высокого до самого меньшего.

Основным тегом для разделения текста является тег абзаца - <p>. В него можно поместить любое количество текста и все будет выделено в формате абзаца с отступами как на Рисунке 1.4

Рисунок 1.4

<p> Элемент р - Разбивает текст на отдельные абзацы, отделяя друг от друга пустой строкой.</p>

Дополнительно в HTML можно делать текст более разнообразным при помощи специальных тегов. Можно выделить текст жирным при помощи тега <b> или же сделать его курсивом при помощи тега <em>

HTML-текст представлен в спецификации элементами для форматирования и группировки текста. Данные элементы являются контейнерами для текста и не имеют визуального отображения.

Элементы для форматирования текста несут смысловую нагрузку и обычно задают для текста, заключенного внутрь, стилевое оформление, например, выделяют текст жирным начертанием или отображают его шрифтом другого семейства (свойство font-family).

Грамотно отформатированный текст дает понять поисковым системам, какие слова несут важную смысловую нагрузку, по каким из них предпочтительно ранжировать веб-страницу в поисковой выдаче. Вся текстовая информация, отображаемая на сайте, размещается внутри элемента <body>.

Все HTML-элементы для работы с текстом представлены в Таблице 1.1

Таблица 1.1

|  |  |
| --- | --- |
| Наименование элемента | Описание элемента |
| Элементы для форматирования текста | |
| <b> | Задаёт полужирное начертания шрифта текста. Выделяет текст без акцента на его важность |
| <em> | Отображает шрифт текста курсивом, придавая тексту значимость |
| <i> | Отображает шрифт курсивом |
| <small> | Уменьшает размер шрифта на единицу по отношению к обычному тексту |
| <strong> | Задаёт полужирное начертание шрифта, относится к элементам логической разметки, указывая браузеру на важность текста |
| <sub> | Используется для создания нижних индексов. Сдвигает текст ниже уровня строки, уменьшая его размер |
| <sup> | Используется для создания степеней. Сдвигает текст выше уровня строки, уменьшая его размер |
| <ins> | Выделяет текст в новой версии документа, подчёркивая его |
| <del> | Перечёркивает текст. Используется для выделения текста, удаленного из документа |
| <mark> | Применяется для выделения фрагментов текста в справочных целях, окрашивая блок символов желтым цветом |
| Абзацы, средства переноса текста | |
| <p> | Разбивает текст на отдельные абзацы, отделяя друг от друга пустой строкой |
| <br> | Переносит текст на следующую строку, создавая разрыв строки |
| <hr> | Используется для разделения контента на веб-странице. Отображается в виде горизонтальной линии |

1.4. Работа со ссылками

Ссылка - перенаправление пользователя не только на новые страницы, но также на определенное место на странице. Ссылка будет являться активной и отображаться в формате ссылки только при условии прописанного атрибута href. Если не добавить значение в атрибут «href», то при клике на ссылку страница будет перезагружаться. Ссылки создаются с помощью элементов <a>, <area> и <link>.

Ссылки разделяются на ссылки на внешние ресурсы - создаются с помощью элемента <link> и используются для расширения возможностей текущего документа при обработке браузером и гиперссылки - ссылки на другие ресурсы, которые пользователь может посетить или загрузить.

Гиперссылки создаются с помощью элемента <a></a>. Внутрь помещается текст, который будет отображаться на веб-странице. Текст ссылки отображается в браузере с подчёркиванием, цвет шрифта - синий, при наведении на ссылку курсор мыши меняет вид.

Обязательным параметром элемента <a> является атрибут href, который задает URl-адрес веб-страницы.

Код, представленный на Рисунке 1.5 даст вид при загрузке браузером - Это сайт Geekbrains и при нажатии на него переадресует пользователя на сайт gb.ru.

Рисунок 1.5

<a href="https://gb.ru ">Это сайт Geekbrains</a>

Ссылка состоит из двух частей -  указателя («Это сайт Geekbrains») и адресной части (https://gb.ru).

Когда же в ссылке указывается только имя файла, браузер предполагает, что файл находится в той же папке, что и документ, содержащий гиперссылку. На практике веб-сайты содержат множество документов, которые размещают в отдельные папки, чтобы ими было легче управлять. Чтобы создать ссылку на файл, находящийся вне папки, содержащей текущий документ, необходимо указать расположение файла или путь: абсолютный или относительный.

Абсолютный путь указывает точное местоположение файла в пределах всей структуры папок на компьютере (сервере). Он даёт доступ к файлу со сторонних ресурсов и содержит различные компоненты (протокол, домен, папка, файл).

Когда необходимо сослаться на страницу на другом сайте, то можно использовать только полный абсолютный путь.

Если файл находится в корневой папке, то путь к нему - http://mysite.ru/index.html.

Обычно в качестве индексного файла выступает документ с именем index.html.

Относительный путь описывает путь к указанному документу относительно текущего. Он определяется с учётом местоположения веб-страницы, на которой находится ссылка. Относительные ссылки используются при создании ссылок на другие документы на одном и том же сайте.

Главное отличие относительного пути от абсолютного в том, что относительный путь не содержит имени корневой папки и родительских папок, что делает адрес короче, и в случае переезда с одного домена на другой не нужно прописывать новый абсолютный адрес. Но если сторонний ресурс будет ссылаться на файлы (изображения, видео, другой контент) с относительными адресами, то они не будут отображаться на другом сайте.

Якорь - это внутренняя ссыла, создающая переход на различные разделы текущей веб-страницы, позволяя быстро перемещаться между разделами. Это очень удобно в случае, когда на странице слишком много текста. Внутренние ссылки также создаются при помощи элемента <a> с разницей в том, что атрибут href содержит имя указателя, а не URl-адрес и перед именем указателя всегда ставится знак # как показано на Рисунке 1.6.

Рисунок 1.6

<a href="#p1">HTML</a> <!--создаём якорь, указав #id элемента-->

<a href="#p2">CSS</a>

<p id="p1">...</p> <!--добавляем соответствующий id элементу-->

<p id="p2">...</p>

При нажатии на созданную ссылку можно сразу перейти к параграфу текста или разделу, к которому он был привязан.

Если же нужно сделать ссылку с одной страницы сайта на определенный раздел другой страницы, то необходимо задать id для этого раздела страницы, а затем добавить его к абсолютному адресу ссылки как представлено на Рисунке 1.7.

Рисунок 1.7

<p id="about me">About me</з>

<a href="https://mysite.ru/css/#about me" class="site" target="\_blank">color</a>

Аналогичным образом можно создать ссылку из изображения поместив элемент <img> внутрь элемента <a>.

По клику можно не только переходить на другие страницы и скачивать файлы, но и совершать звонки на телефоны, отправлять сообщения или звонить по скайпу (Рисунок 1.8).

Рисунок 1.8

ссылка на телефонный номер

<a href="tel:+74951234567">+7 (495) 123-45-67</a>

ссылка на адрес электронной почты

<a href="mailto:example@mail.ru">example@mail.ru</a>

ссылка на скайп (позвонить)

<a href="skype:имя-пользователя?call">Skype</a>

ссылка на скайп (открыть чат)

<a href="skype:имя-пользователя?chat">Skype</a>

ссылка на скайп (добавить в список контактов)

<a href="skype:имя-пользователя?add">Skype</a>

ссылка на скайп (отправить файл)

<a href="skype:имя-пользователя?sendfile">Skype</a>

1.5. Работа с изображениями

Изображения добавляются на веб-страницы с помощью одинарного тега <img>. Вообще использование какой-либо графики делает веб-страницы значительно привлекательнее и позволяет лучше передать суть и содержание веб-документа (Рисунок 1.9).

Рисунок 1.9

<img src="image.png" alt="Какая то картинка">

Как видно для тега <img> доступны основные атрибуты src (задает путь к изображению либо полный к картинке, либо относительный, что находится на сервере или же в одной папке с файлом index.html.) и alt (добавляется альтернативный текст для изображения и выводится на месте появления изображения до его загрузки или при отключенной графике).

Кроме них используется и другие атрибуты для задания высоты изображения, его размера и ширины.

Возможные форматы изображений для их загрузки на сайт – JPEG, GIF, PNG, APNG, **SVG, BMP, ICO.**

# 1.6. Работа с таблицами

Таблицы упорядочивают и выводят на экран данные с помощью строк или столбцов. Таблицы состоят из ячеек, образующихся при пересечении строк и столбцов.

Ячейки таблиц могут содержать любые HTML-элементы, такие как заголовки, списки, текст, изображения, элементы форм, а также другие таблицы. К каждой таблице можно добавить связанный с ней заголовок, расположив его перед таблицей или после неё.

Таблицы больше не используются для вёрстки веб-страниц и компоновки отдельных элементов, потому что такой приём не обеспечивает гибкость структуры и адаптивность сайта, существенно увеличивая HTML-разметку.

Таблица создаётся при помощи элемента <table></table>, который является контейнером для элементов таблицы и все элементы должны находиться внутри него (Рисунок 1.10).

Рисунок 1.10

<table>

    <tr><th>HTML</th><th>CSS</th></tr>

    <tr><td>Теги</td><td>Таблицы стилей</td></tr>

</table

По умолчанию таблица и ячейки не имеют видимых границ, и они задаются с помощью свойств CSS в файле style.css (Рисунок 1.12).

Рисунок 1.12

/\* внешние границы таблицы серого цвета толщиной 1px \*/

table {

    border: 1px solid grey;

 }

/\* границы ячеек первого ряда таблицы \*/

th {

    border: 1px solid grey;

 }

 /\* границы ячеек тела таблицы \*/

 td {

    border: 1px solid grey;

 }

Строки или ряды таблицы создаются с помощью элемента <tr>. Количество горизонтальных строк таблицы определяется количеством элементов <tr></tr>.

Элемент <th> создаёт заголовок столбца - специальную ячейку, текст в которой выделяется полужирным. Количество ячеек заголовка определяется количеством элементов <th></th>. Для элемента доступны атрибуты colspan, rowspan, headers.

Элемент <td> создаёт ячейки таблицы, внутрь которых помещаются данные таблицы. Элементы <td></td>, расположенные в одном ряду, определяют количество ячеек в строке таблицы. Количество пар ячеек <td> должно быть равно количеству пар ячеек <th>. Для элемента также доступны атрибуты colspan (задает количество ячеек, объединенных по горизонтали), rowspan (задает количество ячеек, объединенных по вертикали), headers.

Элемент <caption> создает подпись таблицы. Добавляется непосредственно после тега <table>, вне строки или ячейки.

Код и отображение кода браузером представлены на Рисунке 1.13 и Рисунке 1.14 соответственно.

Рисунок 1.13

<table>

    <caption>Перечень изделий</caption>

      <tr>

        <th>№ п/п</th>

        <th>Наименование изделия</th>

        <th>Ед. изм.</th>

        <th>Количество</th>

        <th>Цена за ед. изм., руб.</th>

        <th>Стоимость, руб.</th>

      </tr>

      <tr>

        <td>1.</td>

        <td>Изделие № 1</td><td>кг</td><td>15</td><td>10</td><td>150</td>

      </tr>

      <tr>

        <td>2.</td>

        <td>Изделие № 2</td><td>кг</td><td>2</td><td>50</td><td>100</td>

      </tr>

      <tr>

        <td colspan="5" style="text-align:right">ИТОГО:</td><td>250</td>

      </tr>

    </table>

Рисунок 1.14

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Перечень изделий | | | | | |
| **№ п/п** | **Наименование изделия** | **Ед. изм.** | **Количество** | **Цена за ед. изм., руб.** | **Стоимость, руб.** |
| 1. | Изделие № 1 | кг | 15 | 1 | 150 |
| 2. | Изделие № 2 | кг | 2 | 50 | 100 |
| ИТОГО: | | | | | 250 |

# 1.7. Работа со списками

Для создания списков используются теги ol и ul, которые создают пронумерованный (каждый элемент отмечается цифрой) и маркированный (каждый элемент отмечается маркером) список соответственно. Для создания конкретного элемента в списке потребуется использовать тег li. Если мы хотим вписать один список внутрь другого, то необходимо использовать следующую конструкцию:

<ol>

    <li>HTML</li>

    <li>CSS

        <ul>

            <li>Стили таблиц</li>

            <li>Стили текста</li>

        </ul>

    </li>

    <li>Javascript</li>

</ol>

Получим следующее:

1. HTML
2. CSS
   * Стили таблиц
   * Стили текста
3. Javascript

Каждый список представляет собой контейнер, внутри которого располагаются элементы списка или пары термин-определение.

С помощью различных атрибутов можно задать различные виды отображения списков: reversed (задает отображение списка в обратном порядке), start (задает начальное значение, от которого пойдет отсчет нумерации), type (задает вид маркера для использования в списке (в виде букв или цифр)/

1.8 Полезные ссылки:

1. <https://stepik.org/course/100971/syllabus>;
2. <https://html5book.ru/html-html5/>;
3. [http://htmlbook.ru](https://vk.com/away.php?to=http%3A%2F%2Fhtmlbook.ru&post=11193554_493&cc_key=);
4. <https://htmlacademy.ru>;
5. https://code-basics.com/ru/languages/html/

2. CSS