Колледж Автономной некоммерческой образовательной организации высшего образования

«Научно-технологический университет «Сириус»

***“Разработка веб-сайтов***

***с использованием HTML, CSS и JavaScript”***

Выполнил: студент 1 курса, группы к0709-24/2

направления подготовки «Информационные системы и программирование»

Симонова Виктория Константиновна

Проверил:

Яковлева Софья Вячеславовна

### **Содержание**

[Содержание](#_1e19b297l518) 1

[Введение](#_qvt2hw7dfgvl) 2

[1 ЦЕЛИ И ЗАДАЧИ РЕФЕРАТА](#_2qttlnvsw5mr) 3

[2 ОСНОВНЫЕ ТЕХНОЛОГИИ ВЕБ-РАЗРАБОТКИ](#_3a2qq4yys9ip) 4

[2.1 HTML (HyperText Markup Language)](#_vpcocoeev0d0) 4

[2.3 Анатомия HTML элемента](#_8w8i7pwryreq) 6

[2.4 Разметка текста](#_l30zwgyi3sug) 7

[2.4.1 Заголовки](#_p9av0j73uixx) 7

[2.4.2 Абзацы](#_338fho5pvtx) 8

[2.4.3 Списки](#_n64gaeyh48ul) 8

[2.4.4 Ссылки](#_7bns8jx024ua) 9

[3.1 CSS (Cascading Style Sheets)](#_psw1sxtd00el) 9

[3.2 CSS-вёрстка](#_wm1qqtvmdxtg) 10

[3.3 Подключение CSS к HTML](#_38fpp91d9y5x) 11

[3.3 Анатомия набора правил CSS](#_9t5188t8j0g) 12

[3.4 CSS-свойства](#_ojdtht7kkbwx) 13

[3.5 CSS-значения](#_m3m505kr6ek9) 13

[3.6 CSS-селекторы](#_oswb69vs665m) 14

[4 JavaScript](#_1t4194iro72r) 15

[4.1 Подключение JavaScript к HTML](#_8vbh41wrrupf) 16

[4.2 Переменные в JavaScript](#_74mep31jf54s) 16

[4.3 Работа с DOM](#_oxwrvji1m5c8) 17

[4.4.1 Получение элементов](#_8nq7wmuxzu57) 17

[4.4.2 Изменение элементов](#_vyq6j0gd4jgq) 18

[4.4.3 Обработка событий](#_2a61j76e6yws) 19

[ЗАКЛЮЧЕНИЕ](#_l207y7qlix8z) 19

#### **Введение**

Веб-разработка — это процесс создания веб-сайтов и веб-приложений для Интернета. Она включает в себя множество аспектов, таких как веб-дизайн, веб-программирование, управление базами данных и многое другое. Для новичков важно понять основные концепции и технологии, которые лежат в основе веб-разработки. Веб-дизайн фокусируется на визуальном аспекте сайта, включая макеты, цветовые схемы и типографику. Веб-программирование, с другой стороны, включает в себя написание кода, который позволяет веб-сайту функционировать. Управление базами данных необходимо для хранения и управления данными, которые используются на веб-сайте.

С развитием Интернета за последние десятилетия создание общественных сайтов стало одним из самых востребованных направлений в веб-разработке. HTML, CSS и JavaScript (JS) играют ключевую роль в этом процессе, обеспечивая возможность создания динамичных, интерактивных и эстетически привлекательных веб-страниц.

Применение этих технологий в разработке общественных сайтов способствует созданию удобного пользовательского интерфейса, а также добавлению интерактивных элементов и динамических эффектов. Такие сайты могут преследовать различные цели: социальное взаимодействие, распространение информации, предоставление общественно полезных ресурсов, участие граждан в общественной деятельности и многое другое.

#### 1 ЦЕЛИ И ЗАДАЧИ РЕФЕРАТА

Цель данного реферата — изучение значимости языков HTML, CSS и JavaScript в процессе создания веб-сайтов. Основные задачи исследования включают:

1. Анализ влияния HTML, CSS и JavaScript на разработку общественных сайтов. Будет рассмотрено, какие функции выполняет каждый из этих языков, какие аспекты веб-страниц они обеспечивают и каким образом взаимодействуют между собой.
2. Исследование возможностей HTML. Будут рассмотрены ключевые элементы и теги, их роль в структурировании информации на сайте и принципы организации веб-контента.
3. Определение значимости CSS в веб-разработке. CSS отвечает за оформление и внешний вид страниц. В рамках исследования будут изучены основные синтаксические конструкции, ключевые свойства и методы стилизации, а также примеры их использования для создания визуально привлекательного дизайна.
4. Анализ роли JavaScript в разработке общественных сайтов. JavaScript обеспечивает динамичность и интерактивность веб-страниц. Будут рассмотрены его ключевые концепции, включая переменные, условные операторы, работа с DOM.

Изучение данных аспектов позволит разработчикам и веб-дизайнерам глубже понять значение и роль языков программирования в веб-разработке общественных сайтов. Эти знания помогут им создавать более удобные, эстетически привлекательные и интерактивные пользовательские интерфейсы, улучшая качество и функциональность веб-ресурсов.

#### 2 ОСНОВНЫЕ ТЕХНОЛОГИИ ВЕБ-РАЗРАБОТКИ

#### 2.1 HTML (HyperText Markup Language)

HTML — это язык разметки, который используется для создания структуры веб-страниц. Он определяет элементы, такие как заголовки, абзацы, ссылки, изображения и другие компоненты веб-страницы. HTML является основой любого веб-сайта и позволяет разработчикам создавать структурированные документы.

Каждый раз заходя на любой сайт вы видите веб-страницу, которая состоит из блоков текста, картинок, кнопок и других элементов. Все эти элементы называются разметкой и описываются с помощью языка HTML (HyperText Markup Language) эта аббревиатура переводится как “язык гипертекстовой разметки”. Он представляет собой набор тегов, которые объявляют и описывают каждый элемент на странице. Браузеры интерпретируют HTML-код и отображают его в виде веб-страницы. Теги могут содержать в себе другие теги, формируя таким образом структуру страницы.

#### 2.2 Структура HTML-документа

1. Декларация типа документа (DOCTYPE)

Первая строка HTML-документа представляет собой декларацию типа документа (DOCTYPE), которая указывает браузеру, какую версию HTML использует данный документ (рис. 1).

1. Тег <html>

Тег <html> служит корневым элементом HTML-документа и включает в себя все остальные элементы. Он состоит из двух ключевых дочерних элементов: <head> и <body>.

1. Тег <head>

Тег <head> содержит метаинформацию о веб-странице, такую как заголовок, описание, ключевые слова и ссылки на внешние ресурсы (например, CSS-стили и JavaScript-файлы). Содержимое <head> не отображается в браузере.

1. Тег <body>

Внутри тега <body> располагается основное содержимое веб-страницы, включая текст, изображения, гиперссылки и формы.

1. Заголовки (<h1> – <h6>)

Заголовки применяются для структурирования контента и помогают пользователям и поисковым системам быстро понять содержание страницы. Существует шесть уровней заголовков: <h1> — наиболее важный, <h6> — наименее значимый.

1. Параграфы (<p>)

Тег <p> используется для форматирования текстовых блоков, обеспечивая удобное отображение информации.

1. Списки (<ul> и <ol>)

Теги <ul> и <ol> применяются для создания списков. <ul> формирует маркированный (неупорядоченный) список, а <ol> — нумерованный (упорядоченный).

1. Ссылки (<a>)

С помощью тега <a> создаются гиперссылки, позволяющие переходить на другие веб-страницы или ресурсы.

1. Изображения (<img>)

Тег <img> используется для вставки изображений на веб-страницу, обеспечивая визуальное оформление контента.

#### 3.1 CSS (Cascading Style Sheets)

CSS (Cascading Style Sheets) — это язык стилей, используемый для описания внешнего вида HTML-документов. С его помощью можно задавать цвета, шрифты, отступы, выравнивание и другие визуальные параметры элементов на веб-странице. CSS делает веб-страницы более привлекательными и удобными для пользователей.

CSS позволяет отделить структуру документа (HTML) от его представления (стилей). Это упрощает поддержку и обновление веб-страниц, а также улучшает их производительность. Например, изменив один файл CSS, можно изменить внешний вид всех страниц сайта. Это особенно полезно для крупных проектов, где множество страниц используют одни и те же стили. В результате, разработчики могут сосредоточиться на содержимом страниц, не беспокоясь о том, как они будут выглядеть.

Кроме того, CSS позволяет создавать адаптивные дизайны, которые автоматически подстраиваются под различные устройства и экраны. Это особенно важно в современном мире, где пользователи могут просматривать веб-сайты на компьютерах, планшетах и смартфонах. С помощью CSS можно настроить стили так, чтобы они выглядели хорошо на всех устройствах, обеспечивая лучший пользовательский опыт.

#### 3.2 CSS-вёрстка

До появления CSS оформление веб-страниц осуществлялось непосредственно внутри содержимого документа. Однако с появлением CSS стало возможным принципиальное разделение содержания и представления документа. За счёт этого нововведения стало возможным лёгкое применение единого стиля оформления для массы схожих документов, а также быстрое изменение этого оформления.

Преимущества:

· Несколько дизайнов страницы для разных устройств просмотра. Например, на экране дизайн будет рассчитан на большую ширину, во время печати меню не будет выводиться, а на КПК и сотовом телефоне меню будет следовать за содержимым.

· Уменьшение времени загрузки страниц сайта за счет переноса правил представления данных в отдельный CSS-файл. В этом случае браузер загружает только структуру документа и данные, хранимые на странице, а представление этих данных загружается браузером только один раз и могут быть закешированы.

· Простота последующего изменения дизайна. Не нужно править каждую страницу, а лишь изменить CSS-файл.

· Дополнительные возможности оформления. Например, с помощью CSS-вёрстки можно сделать блок текста, который остальной текст будет обтекать (например для меню) или сделать так, чтобы меню было всегда видно при прокрутке страницы.

Недостатки:

· Различное отображение вёрстки в различных браузерах (особенно устаревших), которые по разному интерпретируют одни и те же данные CSS.

· Часто встречающаяся необходимость на практике исправлять не только один CSS-файл, но и теги HTML, которые сложным и ненаглядным способом связаны с селекторами CSS, что иногда сводит на нет простоту применения единых файлов стилей и значительно удлиняет время редактирования и тестирования.

#### 3.3 Подключение CSS к HTML

Самый простой способ передать информацию о стилях браузеру, чтобы он отобразил страницу так, как мы хотим, — это прописать стили прямо в HTML-документе. Но в реальном мире веб-разработки необходимо хранить код HTML, CSS и JavaScript в отдельных файлах — так проще и удобнее поддерживать проекты.

Сохраним CSS-код в файл styles.css в ту же папку, в которой лежит HTML-документ, и подключим стили в него:

*<link rel="stylesheet" href="styles.css">*

#### 3.4 CSS-свойства

Свойства CSS определяют атрибуты стиля, которые нужно применить к элементам HTML. Они управляют различными аспектами внешнего вида:

* *font-family, font-size, font-weight, font-style* — свойства шрифта;
* *color, background-color, border-color* — свойства цвета;
* *text-align, text-decoration, text-transform* — свойства текста;
* *display, float, position, margin, padding* — свойства макета;
* *background-image, background-repeat, background-position* — свойства фона.

#### 3.5 CSS-значения

Значения CSS определяют желаемые настройки стиля для свойств. Это могут быть отдельные значения, числа, цветовые коды, ключевые слова или более сложные выражения. Вот несколько примеров значений:

* *font-family: Arial, sans-serif,*
* *color: #ff0000,*
* *background-image: url(image.png),*
* *margin: 10px 20px,*
* *text-align: center.*

На практике это работает так. Допустим, мы делаем свой список задач в браузере и для этого создали страницу для дальнейшей работы. Если не использовать стили, то браузер возьмёт все значения шрифтов, отступов и прочего оформления по умолчанию.

#### 3.6 CSS-селекторы

CSS-селектор — это часть CSS-правила, которая указывает, к каким элементам HTML будут применяться заданные стили. Если проще, они идентифицируют элементы: «такой элемент должен отображаться вот так». Простые селекторы указывают на элементы на основе их имён:

* *h1, p, ul, a, img* — селекторы тегов;
* *.highlight, .button, .image-container* — селекторы классов;
* *#main-content, #header, #footer* — селекторы идентификаторов;
* *\** — универсальный селектор, который выбирает все элементы;
* *type="text"* — селектор атрибута (в данном случае текста).

Сложные селекторы объединяют простые и позволяют более точно выбирать элементы на основе их отношений, состояния или атрибутов.

#### 4 JavaScript

JavaScript — это язык управления элементами на сайте. Он умеет красить кнопки, запускать анимации, загружать эффекты, и всё это — без перезагрузки страницы. Например, вы нажали кнопку «Нравится» — вспыхнуло красное сердечко. Или навели курсор на три точки, и выпало меню. В обоих случаях на странице сработал скрипт. А JavaScript как раз главный по ним в интернете — отсюда и его название.

#### 4.1 Подключение JavaScript к HTML

Подключение JavaScript к HTML может быть выполнено несколькими способами.

Встроенный JavaScript

* Встроенный JavaScript означает, что код написан прямо внутри HTML-документа. Для этого используется тег <script>

Внешний JavaScript

* Внешний JavaScript означает, что код хранится в отдельном файле и подключается к HTML-документу с помощью атрибута src тега <script>

#### 4.2 Переменные в JavaScript

Переменные в JavaScript используются для хранения данных. Они могут быть объявлены с помощью ключевых слов *var, let* или *const.*

Переменные позволяют хранить и манипулировать данными в вашем коде. Ключевое слово *let* используется для объявления переменных, которые могут изменяться, в то время как *const* используется для объявления констант, значения которых не могут быть изменены после инициализации. Ключевое слово *var* также может использоваться для объявления переменных, но его использование не рекомендуется из-за особенностей области видимости.

#### 4.3 Работа с DOM

DOM (Document Object Model) — это интерфейс, который позволяет JavaScript взаимодействовать с HTML-документом. С помощью DOM можно изменять структуру, стиль и содержание веб-страницы.

DOM представляет собой дерево объектов, где каждый элемент HTML-документа является узлом дерева. Это позволяет JavaScript получать доступ к элементам страницы и изменять их. Например, можно добавлять новые элементы, изменять существующие или удалять ненужные.

#### 4.4.1 Получение элементов

Для работы с элементами страницы их сначала нужно получить. Это можно сделать с помощью методов *getElementById, getElementsByClassName, getElementsByTagName и querySelector*.

Пример кода:

*let element = document.getElementById("myElement");*

*let elements = document.getElementsByClassName("myClass");*

*let tags = document.getElementsByTagName("div");*

*let query = document.querySelector(".myClass");*

Методы получения элементов позволяют находить элементы на странице по их идентификаторам, классам, тегам или селекторам. Это первый шаг для манипуляции элементами страницы. Например, метод *getElementById* возвращает элемент с указанным идентификатором, а метод *querySelector* позволяет использовать CSS-селекторы для поиска элементов.

#### 4.4.2 Изменение элементов

После получения элемента можно изменять его свойства и содержимое. Например, изменить текст внутри элемента:

*let element = document.getElementById("myElement");*

*element.textContent = "New text!";*

Или изменить стиль элемента:

*element.style.color = "red";*

*element.style.fontSize = "20px";*

Изменение элементов позволяет динамически обновлять содержимое и стиль страницы. Это делает веб-сайты более интерактивными и отзывчивыми. Например, можно изменять текст в зависимости от действий пользователя или изменять стиль элементов для улучшения визуального восприятия.

#### 

#### 4.4.3 Обработка событий

JavaScript позволяет обрабатывать различные события, такие как клики, ввод текста и т.д. Для этого используются обработчики событий:

*let button = document.getElementById("myButton");*

*button.addEventListener("click", function() {*

*alert("Button clicked!");*

*});*

Обработка событий позволяет реагировать на действия пользователя. Это делает веб-сайты более интерактивными и удобными в использовании. Например, можно обрабатывать клики на кнопки, ввод текста в поля формы или наведение курсора на элементы.

#### ЗАКЛЮЧЕНИЕ

В ходе исследования была изучена роль языков HTML, CSS и JavaScript в веб-разработке. HTML формирует основу веб-страницы, обеспечивая её структуру и содержание. CSS отвечает за визуальное оформление, позволяя сделать сайт привлекательным и удобным для восприятия. JavaScript добавляет интерактивность и динамичность, что делает пользовательский опыт более комфортным и функциональным.

Анализ основных возможностей этих технологий позволил понять, как они взаимодействуют между собой, создавая современные веб-ресурсы. HTML задаёт каркас страницы, CSS определяет её внешний вид, а JavaScript добавляет динамическое поведение. Благодаря их совместному использованию разработчики могут создавать удобные, эстетически привлекательные и технологически продвинутые сайты.

Таким образом, понимание основ HTML, CSS и JavaScript является ключевым аспектом для успешной работы в сфере веб-разработки. Их грамотное применение позволяет улучшить функциональность, производительность и доступность веб-ресурсов, что особенно важно в условиях постоянно развивающегося цифрового пространства.

#### Список использованных источников

1. Дженнифер Роббинс HTML5, CSS3 и JavaScript ИСЧЕРПЫВАЮЩЕЕ РУКОВОДСТВО. 4-е издание.
2. <https://learn.javascript.ru/>
3. <https://thecode.media/baza-po-vyorstke-osnovy-html/?ysclid=m7dma6b57z154220354>
4. [https://developer.mozilla.org/ru/docs/Learn\_web\_development/Getting\_started/Your\_first\_website/Creating\_the\_content#](https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Creating_the_content#%D0%B0%D0%BD%D0%B0%D1%82%D0%BE%D0%BC%D0%B8%D1%8F_html_%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D0%B0)
5. <https://thecode.media/baza-po-vyorstke-samye-osnovy-css/?ysclid=m7eeq250q2997680267>
6. <https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Styling_the_content>