**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**Факультет: Информационные технологии и управление**

**Кафедра: Компьютерная инженерия**

**Предмет: «Веб-системы и технологии»**

Курсовая работа

**Тема** «Разработка онлайн-системы калькулятора обменных курсов»

**Группа:** 680.22

**Курc:** 3

**Специальность: 050616** Информационные технологии

**Студент**: Гадирова Эльвина

**Руководитель:** **препод.Халилов М. Э.**

**Зав.кафедрой: доц.Рагимова Н.А.**

**Баку – 2025**

**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**ЗАДАНИЕ ПО КУРСОВОЙ РАБОТЕ**

**ФАКУЛЬТЕТ**: Информационные технологии и управление

**КАФЕДРА**: Компьютерная инженерия

**Группа** 680.22 **курс** III

**Специальность** **050616** Информационные технологии

**Студент** Гадирова Эльвина

**Зачетная книжка**

**Руководитель курсовой работы**  **асс. Халилов М. Э.**

**Срок выдачи**

**Дата сдачи**

**Тема курсовой работы** «Разработка онлайн-системы калькулятора

обменных курсов »

**Отзыв руководителя курсовой**:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись студента**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись рук.курсовой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись зав.кафедрой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Дата защиты курсовой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Оценка**\_\_\_\_\_\_\_\_\_\_

**Пред.комиссии**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**Члены комиссии**: 1.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

Содержание

[ВВЕДЕНИЕ 4](#_Toc23160)

[1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 6](#_Toc2664)

[1.1.Понятие Web-сайта 6](#_Toc19836)

[1.1.1. Классификация Web-сайта 4](#_Toc2375)

[1.1.2. Web-программирование. 7](#_Toc3525)

[1.2. Основы языка HTML 8](#_Toc17367)

[1.3.Технология каскадных таблиц стилей CSS 13](#_Toc21868)

[1.4. Основы JavaScript 15](#_Toc18946)

[1.4.1.Базовые элементы JavaScript 17](#_Toc25139)

[2. ПРАКТИЧЕСКАЯ ЧАСТЬ 18](#_Toc8689)

[2.1 Описание предметной области 18](#_Toc19251)

[3. РЕЗУЛЬТАТ 19](#_Toc20907)

[4. ЛИТЕРАТУРА 20](#_Toc17528)

# **ВВЕДЕНИЕ**

В условиях глобализации и активного развития международных экономических отношений обменные курсы валют играют ключевую роль в финансовой деятельности компаний и частных лиц. Колебания валютных курсов оказывают значительное влияние на стоимость товаров, услуг, а также на инвестиционные решения. В связи с этим возникает необходимость в инструментах, позволяющих оперативно и точно рассчитывать обменные курсы.​

Современные технологии предоставляют возможность создания онлайн-систем, способных автоматически обновлять данные о курсах валют и производить необходимые расчеты. Разработка такого калькулятора обменных курсов требует интеграции с надежными источниками финансовой информации и обеспечения удобного пользовательского интерфейса.​

Цель данной курсовой работы заключается в разработке онлайн-системы калькулятора обменных курсов, которая обеспечит пользователям быстрый и точный доступ к актуальной информации о валютных курсах и позволит выполнять необходимые расчеты в реальном времени.

Разработка онлайн-системы калькулятора обменных курсов будет включать в себя следующие этапы:

1. **Анализ требований и постановка задачи:** На этом этапе определяется целевая аудитория системы, формулируются основные функциональные требования: поддержка актуальных обменных курсов, возможность конвертации различных валют, удобный и интуитивно понятный интерфейс.

2. **Проектирование архитектуры системы:** Создается архитектура приложения, включая определение клиентской и серверной частей, баз данных, методов интеграции с внешними API для получения курсов валют.

3. Разработка: Это предполагает создание веб-сайта с использованием HTML, CSS, JavaScript и других веб-технологий.

4. **Тестирование системы:** Проведение функционального, нагрузочного и пользовательского тестирования для выявления и устранения ошибок.

WEB-сайт системы калькулятора обменных курсов– Это система, ориентированная в первую очередь на пользователя, которому требуется быстро и удобно произвести конвертацию валют. Она должна предоставлять сервисы актуального отображения курсов валют, легкость в выборе валютной пары и простоту выполнения расчета.  
 Таким образом, целью моей курсовой работы является создание веб-сайта калькулятора обменных валют, обеспечивающего пользователю удобство, надежность и интуитивно понятный интерфейс для выполнения валютных операций.

# **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

## **1.1.Понятие Web-сайта**

Информация, доступная пользователям Интернета, располагается на компьютерах (Web-серверах), на которых установлено специальное программное обеспечение. Значительная часть этой информации организованна в виде Web-сайтов. Каждый из них имеет свое имя (адрес) в Internet. Web-сайт – это информация, представленная в определенном виде, которая располагается на Web-сервере и имеет свое имя (адрес). Для просмотра Web-сайтов на компьютере пользователя применяются специальные программы, которые называются браузерами.

Наиболее распространенными браузерами в настоящее время являются Internet Explorer и Netscape Navigator. В зависимости от того, какое имя (адрес) сайта мы зададим в строке «Адрес», браузер будет загружать в свое окно соответствующую информацию.

Web-сайт состоит из связанных между собой Web-страниц. Webстраница представляет собой текстовый файл с расширением \*.htm, который содержит текстовую информацию и специальные команды – HTML-коды, определяющие, в каком виде эта информация будет отображаться в окне браузера. Вся графическая, аудио- и видеоинформация непосредственно в Web-страницу не входит и представляет собой отдельные файлы с расширениями \*.gif, \*.jpg (графика), \*.mid, \*.mp3 (звук), \*.avi (видео). В HTML-коде страницы содержатся только указания на такие файлы (рис. 1.1). Рис. 1. Структура Web-сайта.

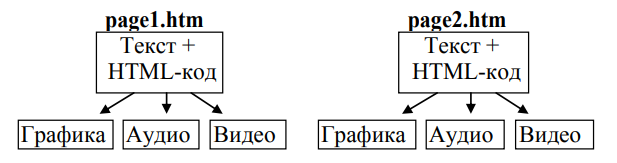


Рис.1.1. Структура Web-сайта

Каждая страница Web-сайта также имеет свой интернет адрес, который состоит из адреса сайта и имени файла, соответствующего данной странице. Таким образом, Web-сайт – это информационный ресурс, состоящий из связанных между собой гипертекстовых документов (Web-страниц), размещенный на Web-сервере и имеющий индивидуальный адрес.

### Классификация Web-сайта

В настоящее время во всемирной паутине размещено несколько миллионов Web-сайтов и их число постоянно растет.

1. **По доступности сервисов:**

* Открытые – все сервисы полностью доступны для любых посетителей. 
* Полуоткрытые – для доступа необходимо зарегистрироваться (обычно бесплатно). 
* Закрытые – полностью закрытые служебные сайты организаций, личные сайты частных лиц. Такие сайты доступны для узкого круга людей. Доступ новым людям даѐтся через инвайты (приглашения).

1. **По природе содержимого:**

* Статические – всѐ содержимое заранее подготавливается. Пользователю выдаются файлы в том виде, в котором они хранятся на сервере. 
* Динамические – содержимое генерируется специальными скриптами (программами) на основе других данных из любого источника.

1. **По физическому расположению:**

* Внешние сайты сети Интернет. 
* Локальные сайты – доступны только в пределах локальной сети. Это могут как корпоративные сайты организаций, так как и сайты частных лиц в локальной сети провайдера.

1. **По схеме представления информации, еѐ объѐму и категории решаемых задач:**

*  Сайт-визитка – содержит самые общие данные о владельце сайта (организация или индивидуальный предприниматель). Вид деятельности, история, прайс-лист, контактные данные, реквизиты, схема проезда. Специалисты размещают своѐ резюме. То есть подробная визитная карточка.
* Интернет-магазин – Web-сайт с каталогом продукции, с помощью которого клиент может заказать нужные ему товары.
* Промо-сайт – сайт о конкретной торговой марке или продукте, на таких сайтах размещается исчерпывающая информация о бренде, различных рекламных акциях (конкурсы, викторины, игры и т. п.).
* Веб-сервис – обычно решает конкретную пользовательскую задачу напрямую связанную с сетью Интернет: 

Поисковые сервисы (например, Яндекс, Google). 

Почтовый сервис. 

Веб-форумы.

Блоговый сервис. 

Фото хостинг (например, Flickr, ImageShack, Panoramio, Photobucket).

Хранение видео (например, YouTube, RuTube). 

Доска объявлений. 

1. **По отношению к посетителю:**

* Вовлекающий сайт. 
* Безразличный к посетителю.

Компоненты веб-страницы:

− контент (текст, изображения, видео, аудио и т.п., в том числе из баз данных); − логическая структура (HTML);

− оформление (CSS);

− поведение (языки программирования).

Этапы разработки веб-сайтов представлены в таблице 1.

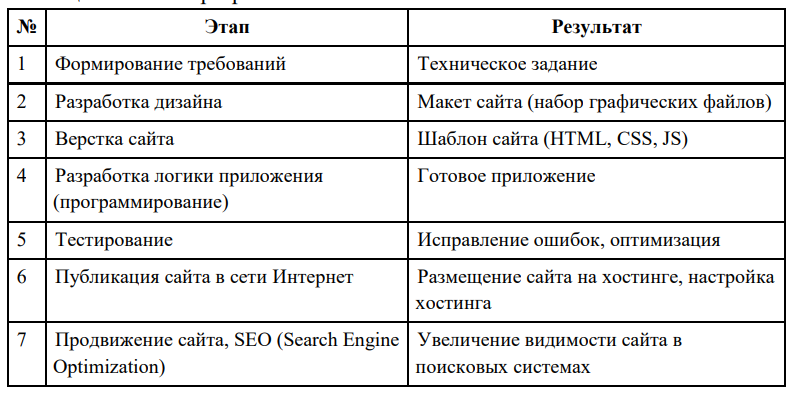


Таблица 1 - Этапы разработки веб-сайтов

Подходы к разработке сайтов:

1. Разработка «с нуля» – разработчик полностью пишет код верстки HTML, CSS, программы на языках JavaScript и PHP или других языков программирования, проектировать и создавать базу данных.
2. Использование конструкторов и систем управления контентом сайта (CMS – Content Management System), например, WordPress, Joomla, OpenCart, Bitrix, Netcat и т.д. 9

3. Использование фреймворков (это программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта), например, Symfony, Yii, Laravel, Aura и т.д.

### Web-программирование.

Web-программирование (Web-разработка) - это бурно развивающийся раздел программирования, ориентированный на разработку динамических Интернет-приложений.

Языки Web-программирования делятся на две группы: клиентские и серверные.

Клиентские языки обрабатываются на стороне пользователя (в основном в браузере). Соответственно, обработка скрипта зависит от браузера пользователя, и пользователь имеет полномочия настроить свой браузер так, чтобы тот вообще игнорировал скрипты. При этом если браузер старый, он может не поддерживать тот или иной язык или версию языка, на которую опирался разработчик. С современными браузерами таких проблем возникать не должно, к тому же языки программирования не так уж часто кардинально обновляются (раз в несколько лет) . Код клиентского скрипта может посмотреть каждый, выбрав в меню своего браузера «Исходный код страницы».

Самым распространенным из клиентских языков признан JavaScript, разработчиками которого является компания Netscape совместно с компанией Sun Microsystems. Еще один популярный язык - это VBScript.

Серверные языки программирования открывают перед программистом большие просторы для деятельности. Когда пользователь делает запрос на какую-либо страницу (переходит на нее по ссылке, или вводит адрес в адресной строке своего браузера), то вызванная страница сначала обрабатывается на сервере (то есть выполняются все программы, связанные со страницей) и только потом возвращается к посетителю по сети в виде файла. Этот файл может иметь расширения: HTML, PHP, ASP, Perl, SSI, XML, DHTML, XHTML.

## **1.2. Основы языка HTML**

Web-страницы – это текстовые файлы, содержащие собственно текст содержимого страницы и команды форматирования, называемые тегами (tag) или дескрипторами разметки языка HTML. Язык разметки гипертекста HTML (HyperText Markup Language) является базовой технологией разработки web-страниц, которые также называют HTML-документами.

Код HTML интерпретируется браузером (средством просмотра webстраниц), который выводит отформатированное содержимое webстраницу на экран. Просмотр кода загруженной страницы доступен во всех браузерах, например, в MS Internet Explorer это можно сделать с помощью команды Вид/Просмотр HTML-кода.

**Основные элементы HTML**

Основной элемент языка HTML – тег. Тег – указатель разметки, который задается следующим образом:

*[Анатомия HTML элемента](https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Creating_the_content" \l "%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)*

Главными частями нашего элемента являются:

1. Открывающий тег (Opening tag): Состоит из имени элемента (в данном случае, "p"), заключённого в открывающие и закрывающие угловые скобки. Открывающий тег указывает, где элемент начинается или начинает действовать, в данном случае — где начинается абзац.
2. Закрывающий тег (Closing tag): Это то же самое, что и открывающий тег, за исключением того, что он включает в себя косую черту перед именем элемента. Закрывающий элемент указывает, где элемент заканчивается, в данном случае — где заканчивается абзац. Отсутствие закрывающего тега является одной из наиболее распространённых ошибок начинающих и может приводить к странным результатам.
3. Контент (Content): Это контент элемента, который в данном случае является просто текстом.
4. Элемент(Element): Открывающий тег, закрывающий тег и контент вместе составляют элемент.

Атрибуты содержат дополнительную информацию об элементе, которую вы не хотите показывать в фактическом контенте.

Атрибут всегда должен иметь:

1. Пробел между ним и именем элемента (или предыдущим атрибутом, если элемент уже имеет один или несколько атрибутов).
2. Имя атрибута, за которым следует знак равенства.
3. Значение атрибута, заключённое с двух сторон в кавычки.

*Пустые элементы*

Некоторые элементы не имеют контента, и называются пустыми элементами. Возьмём элемент [<img>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/img)

<img src="images/firefox-icon.png" alt="Моё тестовое изображение" />

Он содержит два атрибута, но не имеет закрывающего тега </img>, и никакого внутреннего контента. Это потому, что элемент изображения не оборачивает контент для влияния на него.

Его целью является вставка изображения в HTML страницу в нужном месте.

*[Анатомия HTML документа](https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Creating_the_content" \l "%D0%B0%D0%BD%D0%B0%D1%82%D0%BE%D0%BC%D0%B8%D1%8F_html_%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0)*

Теперь мы посмотрим, как отдельные элементы объединяются в целую HTML страницу. Пример простого документа представлен на рис.1.2

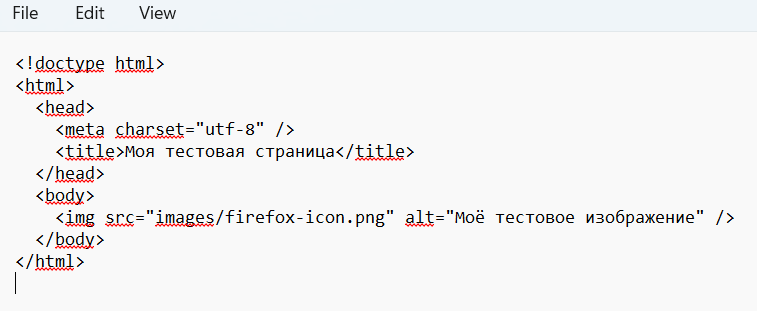


Рис. 1.2. Простой документ, показывающий анатомию HTML документа

Здесь мы имеем:

<!DOCTYPE html> — доктайп.

<html></html> — элемент [<html>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/html). Этот элемент оборачивает весь контент на всей странице, и иногда известен как корневой элемент.

<head></head> — элемент [<head>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/head). Этот элемент выступает в качестве контейнера для всего, что вы пожелаете включить на HTML страницу, но не являющегося контентом, который вы показываете пользователям вашей страницы.

<body></body> — элемент [<body>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/body). В нем содержится весь контент, который вы хотите показывать пользователям, когда они посещают вашу страницу, будь то текст, изображения, видео, игры, проигрываемые аудиодорожки или что-то ещё.

<meta charset="utf-8"> — этот элемент устанавливает UTF-8 кодировку вашего документа, которая включает в себя большинство символов из всех известных человечеству языков. По сути, теперь документ может обрабатывать любой текстовый контент, который вы в него вложите. Нет причин не устанавливать её, так как это может помочь избежать некоторых проблем в дальнейшем.

<title></title> — элемент [<title>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/title). Этот элемент устанавливает заголовок для вашей страницы, который является названием, появляющимся на вкладке браузера загружаемой страницы, и используется для описания страницы, когда вы добавляете её в закладки/избранное.

*[Изображения](https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Creating_the_content" \l "%D0%B8%D0%B7%D0%BE%D0%B1%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F)*

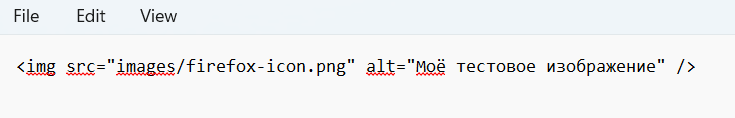


Рис. 1.3. Тег<img> и его атрибуы

Код встраивает изображение на нашу страницу в нужном месте.

Это делается с помощью атрибута

src (source, источник) - который содержит путь к нашему файлу изображения.

И атрибут

alt(alternative, альтернатива) - В атрибуте, указывается поясняющий текст для пользователей, которые не могут увидеть изображение, возможно, по следующим причинам:

* У них присутствуют нарушения зрения.
* Что-то пошло не так, в результате чего изображение не отобразилось. Например, попробуйте намеренно изменить путь в вашем атрибуте src, сделав его неверным.

**[Разметка текста](https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Creating_the_content" \l "%D1%80%D0%B0%D0%B7%D0%BC%D0%B5%D1%82%D0%BA%D0%B0_%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%B0)**

*[Заголовки](https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Creating_the_content" \l "%D0%B7%D0%B0%D0%B3%D0%BE%D0%BB%D0%BE%D0%B2%D0%BA%D0%B8)*

Элементы заголовка позволяют вам указывать определённые части контента в качестве заголовков или подзаголовков. HTML включает шесть уровней заголовков <h1>–<h6>, хотя обычно вы будете использовать не более 3-4 :

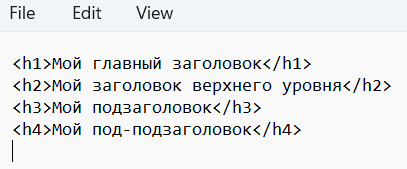


Рис.1.4. Заголовки и подзаголовки

*Абзацы*

Элемент [<p>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/p) предназначен для абзацев текста; вы будете использовать их регулярно при разметке текстового контента

*[Списки](https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Creating_the_content" \l "%D1%81%D0%BF%D0%B8%D1%81%D0%BA%D0%B8)*

Большая часть веб-контента является списками и HTML имеет специальные элементы для них. Разметка списка всегда состоит по меньшей мере из двух элементов. Наиболее распространёнными типами списков являются нумерованные и ненумерованные списки:

* Ненумерованные списки - это списки, где порядок пунктов не имеет значения, как в списке покупок. Они оборачиваются в элемент [<ul>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/ul).
* Нумерованные списки - это списки, где порядок пунктов имеет значение, как в рецепте. Они оборачиваются в элемент [<ol>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/ol).

Каждый пункт внутри списков располагается внутри элемента [<li>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/li) (list item, элемент списка).

*[Ссылки](https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Creating_the_content" \l "%D1%81%D1%81%D1%8B%D0%BB%D0%BA%D0%B8)*

 Чтобы добавить ссылку, нам нужно использовать простой элемент

[<a>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/a) — a это сокращение от "anchor" ("якорь").

<a>Интересный журнал</a>

Задавая элементу [<a>](https://developer.mozilla.org/ru/docs/Web/HTML/Reference/Elements/a) атрибут href, нужно заполнить значение этого атрибута веб-адрессом, на который нужно указать ссылку:

<a href="https://vartumashvili.livejournal.com"

>Интересный журнал</a>

## **1.3.Технология каскадных таблиц стилей CSS**

Таблицы каскадных стилей CSS (Cascade Style Sheets) – это простая технология определения и присоединения стилей оформления к документам HTML.

Стиль оформления – это все то, что определяет внешний вид документа: размер, цвет и вид шрифта текста, цвет фона текста, наличие границ, подчеркивания, выравнивание текста и т. д. Стиль определяется набором правил отображения тегов, задаваемых таблицей стилей.

Таблица стилей – это шаблон, который управляет форматированием тегов HTML в web-документе. Таблица стилей состоит из набора правил описания стиля.

Любое правило каскадных таблиц стилей состоит из двух частей: селектора и определения. Селектор – это любой элемент или группа элементов web-страницы, для которых определяется форматирование. Определение описывает конкретный вид форматирования и состоит из двух частей: свойства и значения, разделенных знаком двоеточия (рис.3.1 ).

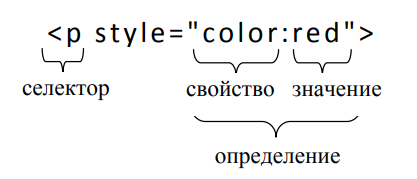
Например, цвет текста абзаца может быть задан с помощью стиля как:

Рис.3.1. Правило таблицы стилей

В ряде случаев использование CSS позволяет создать более компактный и эффективный код HTML-документа.

В одном правиле можно задать несколько определений, разделенных знаком точки с запятой (;).

Например, задание красного цвета, жирного написания и выравнивания «по ширине» текста абзаца с помощью стиля выглядит как:

<p style="color:red; font-weight:bold; text-align:justify"> Текст </p>

Некоторые свойства стиля приведены в табл. 2

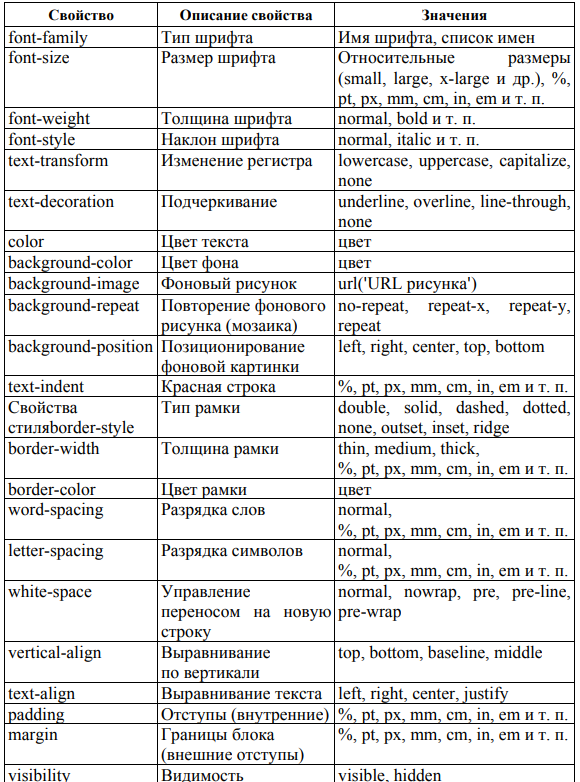
.

Таблица 2 - Свойства стилей

Указание правил CSS в HTML-документе может производиться четырьмя разными способами (табл. 3)

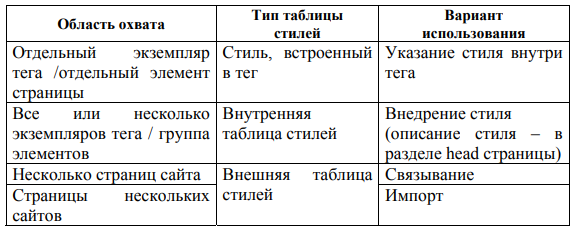


Таблица 3 - Варианты использования стилей CSS

Встраивание стиля в теги документа производится с помощью параметра style и позволяет изменить форматирование конкретных экземпляров тегов страницы.

Внедрение стиля позволяет определять форматирование для всех экземпляров тега или для группы элементов страницы. Таблица стилей внедряется в текст HTML-документа в раздел заголовка head с помощью тега style.

Связывание с таблицей стилей – таблица стилей сохраняется в виде внешнего файла, а в HTML-документ помещается ссылка на нее. Ссылка на файл с таблицей осуществляется тегом link, который указывается в разделе заголовка head. Связывание позволяет использовать одну таблицу CSS для форматирования элементов сразу нескольких web-страниц

## **1.4. Основы JavaScript**

JavaScript – объектно-ориентированный сценарный язык программирования.

Существует два способа добавления кода JavaScript на

веб-страницы: добавление прямо в код используемой страницы

или использование отдельного файла.

В первом варианте код пишется непосредственно на вебстранице, внутри пары тегов <script></script>. Тег <script> может быть помещен либо в секцию <head>, либо в секцию <body> страницы. Обычно скрипт, который выводит данные

прямо на страницу, помещен в секцию <body> веб-страницы.

Скрипты, помещенные в секцию <head>, обычно используются

другими скриптами, расположенными в секции <body>. В примере ниже демонстрируется использование обоих способов.

<html>

<head>

<meta http-equiv="Content-Type" content="text/html;

charset=windows-1251">

<script language="javascript" type="text/javascript">

document.title = "Пример кода JavaScript"

</script>

</head>

<body>

<script language="javascript" type="text/javascript">

document.write ("Заголовок данной веб-страницы - "+ document.title)

</script>

</body>

</html>

**Общие правила написания кода**

Наиболее важными правилами являются следующие:

– некоторые элементы JavaScript чувствительны к регистру, в том числе команды и имена переменных JavaScript;

– каждая инструкция (команда) в JavaScript обычно заканчивается точкой с запятой (инструкция – это строчка кода,

законченная и синтаксически правильная для выполнения

определенной задачи);

– текст, то есть любое сочетание букв и цифр, которое

также называют строкой, должен быть заключен в одинарные

(") или двойные ("") кавычки – разница в их использовании

7

разъяснена в нижеследующем примечании. Кавычки обязательно должны быть одинаковыми и парными;

– большинство функций требуют, чтобы их параметры

были заключены в круглые скобки;

– несколько параметров должны разделяться запятыми (,).

### 1.4.1.Базовые элементы JavaScript

**2. ПРАКТИЧЕСКАЯ ЧАСТЬ**

**2.1 Описание предметной области**

**3. РЕЗУЛЬТАТ**

За время выполнения курсовой работы были изучены принципы разработки web-сайта:

1. Мы создали веб-страницу используя HTML.
2. Стили выполнены с использованием CSS
3. Создана адаптивная веб-страница с использованием медиа-запросов.

В результате была создана современная, функциональная и эстетичная веб-страница, которая удовлетворяет требованиям заказчиков, удобна для пользователей и соответствует современным трендам в веб-дизайне.

Ссылка на веб-страницу: <https://medina003.github.io/Coursework/>

**4. ЛИТЕРАТУРА**

1. <https://studfile.net/preview/21325586/>
2. <https://books.ifmo.ru/file/pdf/878.pdf>
3. ЯКУБОВИЧ Д.А. ЕРОПОВА Е.С. ЕРОПОВ И.А.Основы WEB-разработки
4. Магомедова Динара Сахратулаевна Бекбулатова Зайнаб Абдулмуслимовна.Учебное пособие по дисциплине «Web-программирование»
5. <https://skillbox.ru/media/code/chto_takoe_html/>
6. Е.В. Киргизова, А.В. Рубцов - WEB-ТЕХНОЛОГИИ: от теории к практике
7. <https://sky.pro/wiki/html/osnovy-html-struktura-html-dokumenta/>
8. <https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Creating_the_content>

**5. ПРИЛОЖЕНИЯ**

Репозиторий:

Страница сайта на хостинге GitHub Pages: