СОДЕРЖАНИЕ

[Введение 4](#_Toc97972222)

[1 Основная часть 5](#_Toc97972223)

[1.1 Понятие и синтаксис языка программирования JavaScript 5](#_Toc97972224)

[1.2 Возможности и области применения JavaScript кода 7](#_Toc97972225)

[1.3 Библиотеки и фреймворки языка программирования JavaScript 9](#_Toc97972226)

[1.4 Межсайтовый скриптинг или XSS 14](#_Toc97972227)

[1.5 Определение редактора исходного кода для разработки 15](#_Toc97972228)

[1.6 Средства тестирования и отладки JavaScript кода 18](#_Toc97972229)

[2 Постановка задачи 19](#_Toc97972230)

[2.2 Входная информация 19](#_Toc97972231)

[2.3 Выходная информация 19](#_Toc97972232)

[3 Технические и инструментальные средства обеспечения задачи 20](#_Toc97972233)

[3.1 Обоснование выбора языка программирования 20](#_Toc97972234)

[3.2 Технические требования к компьютеру для решения задачи 21](#_Toc97972235)

[3.3 Технические требования к компьютеру пользователя для использования веб-сайта 21](#_Toc97972236)

[4 Программное обеспечения задачи 22](#_Toc97972237)

[4.1 Описание блок-схемы 22](#_Toc97972238)

[4.2 Описание интерфейса страницы 22](#_Toc97972239)

[Заключение 24](#_Toc97972240)

[Список использованных источников 26](#_Toc97972241)

[Приложение A 28](#_Toc97972242)

[Приложение Б 29](#_Toc97972243)

## 

Введение

Бесчисленное множество новых технологий, вызванных бурным ростом информатизации общества, делает нашу жизнь невозможной без быстрого доступа к информации. В наше время очень легко получить информацию, одним из способов быстрого доступа к ней является сайт.

Каждый сайт пишется c помощью языка гипертекстовой разметки HTML и каскадных таблиц стилей CSS. Так сложилось, что недостаточно знать только HTML и CSS, чтобы создать сайт — он получится статичным. Если добавить PHP, и страницы станут динамичными. Но если нужна интерактивность, то без JavaScript не обойтись.

Актуальность состоит в том, что JavaScript — постоянно развивающийся язык с продуманной инфраструктурой и большой экосистемой. Постоянно появляются новые технологии, которые могут оказать на него влияние и сдавать позиции в ближайшее время он не собирается.

Целью данного проекта является разработка JS кода и его оптимизации для улучшения интерактивности и функциональности веб-страниц.

Для достижения цели проекта необходимо решить ряд задач:

* Изучить понятие языка программирования JavaScript;
* Разобраться в синтаксисе языка;
* Изучить области применения и возможности JavaScrip;
* Изучить популярные библиотеки и фреймворки JavaScript;
* Определить, что такое XSS и как оно связано с JavaScript;
* Разработать оптимизировать и отладить итоговый JavaScript
* Рассмотреть средства тестирования и отладки;
* Определить редактор исходного кода для разработки;

Объектом курсового проекта является — чистый JavaScript код.

Предмет курсового проекта — процесс разработки и оптимизации JavaScript кода.

## 1 Основная часть

## 1.1 Понятие и синтаксис языка программирования JavaScript

JavaScript — это мультипарадигменный язык программирования, который обычно применяется в качестве встраиваемого инструмента для программного доступа к различным объектам приложений. Мультипарадигменныйязык программирования — язык, который поддерживает больше, чем одну парадигму программирования. Парадигма программирования — это совокупность идей и понятий, определяющих стиль написания компьютерных программ. [1]

Программы на JavaScript называются скриптами. Они могут встраиваться в HTML и выполняться автоматически при загрузке веб-страницы.

Скрипты распространяются и выполняются, как простой текст. Им не нужна специальная подготовка или компиляция для запуска.

Чтобы написать «Скрипт» нужно знать синтаксис языка.

Синтаксис JavaScript это набор правил, по которым строится программа. При написании программ используется набор символов Юникода — это надмножество кодировок и поддерживает практически все письменные языки, имеющиеся на планете.

JavaScript чувствителен к регистру символов. Это значит, что ключевые слова, имена переменных и функций и любые другие идентификаторы языка должны всегда содержать одинаковые наборы прописных и строчных букв.

JavaScript игнорирует пробелы, которые могут присутствовать в программе. Кроме того, язык по большей части игнорирует символы перевода строки. Поэтому пробелы и символы перевода строки могут без ограничений использоваться в исходных текстах программ для форматирования и придания им удобного и читаемого внешнего вида.

Как и в других языках программирования, для отделения инструкций друг от друга. Как и в других языках программирования, для отделения инструкций друг от друга в языке JavaScript используется точка с запятой.

Идентификатор — это просто имя. В JavaScript идентификаторы выступают в качестве имен переменных и функций, а также меток некоторых циклов. Идентификаторы в JavaScript должны начинаться с буквы, с символа подчеркивания или знака доллара. Далее могут следовать любые буквы, цифры, символы подчеркивания или знаки доллара. (Цифра не может быть первым символом, так как тогда интерпретатору трудно будет отличать идентификаторы от чисел.). [3]

JavaScript резервирует ряд идентификаторов, которые играют роль ключевых слов самого языка. Эти ключевые слова не могут служить идентификаторами в программах. JavaScript также резервирует некоторые ключевые слова, которые в настоящее время не являются частью языка, но могут войти в его состав в будущих версиях.

## 1.2 Возможности и области применения JavaScript кода

Для того чтобы понять, как и где можно применить JavaScript следует рассмотреть области, в которых применяется данный язык программирования.

Верстка веб-страниц — Используя JavaScript, можно добавлять динамические функции, такие как изменение текста и содержимого, а также изображений, размер которых изменяется в соответствии с размером веб-сайта. Это изменения взаимодействия с пользователем, что означает, что они видны пользователю и тому, как он использует сайт в своём веб-браузере

Веб-приложения — JavaScript используется в клиентской части веб-приложений: клиент-серверных программ, в котором клиентом является браузер, а сервером — веб-сервер, имеющих распределённую между сервером и клиентом логику. Обмен информацией в веб-приложениях происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения являются кроссплатформенными сервисами.

AJAX— это Асинхронный JavaScript и XML. Это набор методов веб-разработки, которые позволяют веб-приложениям работать асинхронно — обрабатывать любые запросы к серверу в фоновом режиме. В результате любое веб-приложение, использующее AJAX, может отправлять и извлекать данные с сервера без необходимости перезагрузки всей страницы. [4]

XML (расширяемый язык разметки) — это язык программирования, который состоит из объявлений в виде информации и определяющих тегов. С его помощью удобно хранить и передавать любые данные.  
Язык не зависит от операционной системы и среды обработки. XML служит для представления неких данных в виде структуры, которую вы можете сами разработать или подстроить под программу или сервис. [2]

Букмарклет — это JavaScript-код, который сохраняется как закладка в браузере.

Браузерные операционные системы (WebOS) — веб-приложение, организующее платформу (операционную среду с набором готовых функций) для выполнения других веб-приложений. Главной составляющей WebOS является её графический интерфейс, его особенности — это внешний вид и возможности, присущие традиционным операционным системам, то есть наличие рабочего стола, ярлыков, окон и т. п. [6]

Мобильные приложения — Традиционно мобильные приложения разрабатывались с использованием языков, специфичных для операционной системы, однако в последние годы было разработано несколько технологий, позволяющих создавать мобильные приложения с использованием языка программирования JavaScript.

Виджет — вспомогательная мини-программа, графический модуль которой размещается в рабочем пространстве соответствующей родительской программы, служащая для украшения рабочего пространства, развлечения, решения отдельных рабочих задач или быстрого получения информации из интернета без помощи веб-браузера.

JavaScript используется для написания прикладного ПО. Например, 16,4 % исходного кода Mozilla Firefox написано на JavaScript. Google Chrome в качестве прикладного ПО использует веб-приложения.

JavaScript используется в офисных приложениях для автоматизации рутинных действий, написания макросов, организации доступа со стороны веб-служб.

Разработка игр — JavaScript используется в онлайн-играх. Многие из этих игр являются простыми платформенными играми, потому что JavaScript не так эффективен для создания игр, как другие языки. Существует множество ограничений на создание браузерной игры с использованием JavaScript. Но, тем не менее, этот язык часто используется разработчиками игр для создания простых игр. Сам по себе JavaScript не очень хорошо масштабируется, если вы хотите создавать большие игры.

## 1.3 Библиотеки и фреймворки языка программирования JavaScript

Библиотека JavaScript является многоразовой частью кода написанной на JavaScript, которая предлагает определенные функции. Это набор функций, объектов и классов, которые вы можете использовать в своем приложении. Библиотека абстрагирует различные слои, так что вам не нужно беспокоиться об их деталях реализации.

Вы можете вызвать библиотечную функцию и передать ей некоторые параметры, и библиотека выполнит ее и вернет вам элемент управления. Это не создает каких-либо структурных ограничений, которые ограничивают использование библиотеки.

Фреймворки JavaScript — это библиотеки программирования JavaScript, в которых есть предварительно написанный код для использования в стандартных функциях и задачах программирования. [8]

По сути, библиотека — это набор функций и способов решения функциональности проекта, для которого пишется код,

а фреймворк – это площадка для программы. Они облегчают разработку и объединение разных элементов кода одного большого проекта.

Главным отличием библиотеки от фреймворка является инверсия контроля. Если программист контролирует ситуацию при вызове функции или метода, то фреймворк самостоятельно вызывает код, написанный программистом.

React— это декларативная эффективная и гибкая JavaScript-библиотека для создания пользовательских интерфейсов. [12]

React может использоваться для разработки одностраничных и мобильных приложений. Его цель — предоставить высокую скорость, простоту и масштабируемость. В качестве библиотеки для разработки пользовательских интерфейсов React часто используется с другими библиотеками.

С помощью React можно создать интерактивный интерфейс с использованием декларативного подхода, в котором вы можете контролировать состояние приложения, говоря: «Представление должно выглядеть так». Он использует компонентную модель, в которой компоненты являются повторно используемыми элементами пользовательского интерфейса, и каждый компонент имеет свое собственное состояние.

React использует виртуальную модель DOM, так что вам не нужно беспокоиться о прямом манипулировании с DOM. Другие примечательные особенности React включают однонаправленный поток данных, дополнительный синтаксис JSX и инструмент командной строки для создания проекта React с нуля.

DOM (Document Object Model) — это фундаментальный прикладной программный интерфейс, обеспечивающий возможность работы с содержимым HTML и XML-документов. Прикладной программный интерфейс (API) модели DOM не особенно сложен, но в нем существует множество архитектурных особенностей, которые вы должны знать.

Прежде всего вложенные элементы HTML или XML-документов представлены в виде дерева объектов DOM. Древовидное представление HTML-документа содержит узлы, представляющие элементы или теги и узлы, представляющие строки текста. HTML-документ также может содержать узлы, представляющие HTML-комментарии.

JSX — это расширение синтаксиса JavaScript, которое позволяет использовать HTML-подобный синтаксис для описания структуры интерфейса. Как правило, компоненты написаны с использованием JSX, но также есть возможность использования обычного JavaScript.

jQuery — это библиотека, которая сделала JavaScript более доступным, а DOM-манипуляцией проще, чем раньше. Плавная кривая обучения jQuery и простой синтаксис породили на стороне клиента новое поколение новых разработчиков. Несколько лет назад jQuery считался прочным решением для создания надежных веб-сайтов с поддержкой кросс-браузерности (свойство веб-сайта отображаться и функционировать в используемых браузерах идентично. Под идентичностью функционирования подразумевается: отсутствие некорректной работы, ошибок в вёрстке и способность отображать материал с одинаковой степенью читабельности.). Основные функции jQuery такие, как манипулирование DOM на основе селекторов CSS, обработка событий и создание вызовов AJAX, подпитывали его популярность.

Однако все изменилось, и среда JavaScript постоянно развивается. Более того, новые библиотеки и фреймворки, используемые сегодня, имеют собственный способ связывания DOM, и поэтому простые методы манипуляции с DOM больше не требуются. Популярность jQuery находится на спаде, но я не вижу, чтобы эта библиотека исчезла в ближайшее время.

Особенность того, почему jQuery стала настолько успешной и популярной, это, вероятно, кроссплатформенные возможности. Она автоматически исправляет ошибки и работает таким же образом в наиболее часто используемых браузерах. Затем есть манипулирование DOM в котором есть несколько методов, как это сделать. Проще говоря, он позволяет вставлять и/или удалять элементы DOM на HTML-странице, а также упрощает перенос строк.

Создание анимации также упрощено с помощью jQuery. Как и в приведённом выше фрагменте кода об анимации, он покрыт несколькими строками кода, всё что вам нужно сделать, это вставить переменные.

Обход документов HTML, а также выполнение эффектов и обработка событий также улучшены с помощью jQuery.

Angular **—** это JavaScript-фреймворк с открытым исходным кодом. Предназначен для разработки одностраничных приложений. Его цель — расширение браузерных приложений на основе MVC-шаблона, а также упрощение тестирования и разработки. [10]

MVC (Model-View-Controller) — схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо:

* Модель (Model) предоставляет данные и реагирует на команды контроллера, изменяя своё состояние;
* Представление (View) отвечает за отображение данных модели пользователю, реагируя на изменения модели;
* Контроллер (Controller) интерпретирует действия пользователя, оповещая модель о необходимости изменений;

Angular — одна из самых мощных сред JavaScript. Google использует эту платформу для разработки одностраничного приложения (SPA). Эта среда разработки известна прежде всего потому, что она предоставляет разработчикам лучшие условия для объединения JavaScript с HTML и CSS.

Angular имеет компонентную структуру, как и React. Можно манипулировать, вкладывать и использовать их по мере необходимости. Нужно будет использовать TypeScript, чтобы написать приложение в Angular. Это расширенный набор JavaScript, который использует тот же синтаксис, но также поддерживает статическую типизацию и классы. Проще говоря, Angular — это фантастическая платформа, на которую можно взглянуть, если вы новый разработчик.

Vue.js — это среда JavaScript для создания пользовательских интерфейсов. Он построен на основе стандартных HTML, CSS и JavaScript и предоставляет декларативную и основанную на компонентах модель программирования, которая помогает эффективно разрабатывать пользовательские интерфейсы, будь то простые или сложные.

Vue — это фреймворк и экосистема, охватывающая большинство общих функций, необходимых для фронтенд-разработки. Но сеть чрезвычайно разнообразна — вещи, которые мы создаем в сети, могут сильно различаться по форме и масштабу. Имея это в виду, Vue спроектирован так, чтобы быть гибким и постепенно адаптироваться.

 В Vue разработчики просто привязывают представление к соответствующей модели, и Vue автоматически наблюдает за изменениями в модели и перерисовывает представление. Эта функция делает управление состоянием Vue довольно простым и интуитивно понятным.

В Vue.js можно использовать только знания JavaScript и HTML.У Vue.js есть собственная официальная достаточно богатая документация на многих языках, которая может послужить примером в объяснении проектирования и разработки в браузере.

Ember.js **—** это JavaScript-фреймворк для разработки клиентской части веб-приложений, амбициозный проект, который в последнее время привлекает к себе много внимания.

Основные внутренние концепты, используемые каркасом — маршруты, модели, контроллеры и шаблоны.

Маршруту объекта соответствует URL-адрес, который определяет текущее состояние приложения. Маршруты определены в единственном объекте маршрутизатора. Концепция маршрутов подчёркивают важность URL в управлении состоянием приложения.

Каждому маршруту соответствует модель, в которой содержатся данные, соответствующие текущему состоянию приложения. И несмотря на то, что есть возможность использовать jQuery, чтобы загружать с сервера JSON-объекты, большинство приложений все-таки использует для этих целей библиотеку с моделью данных.

## 1.4 Межсайтовый скриптинг или XSS

Общая проблема, касающаяся JavaScript — межсайтовый скриптинг или XSS — тип уязвимости веб-сайта, при которой вредоносный скрипт внедряется в сайт или приложение, затем устанавливает вредоносное ПО в браузер жертвы. Используя межсайтовый скриптинг, хакеры не нацеливаются на конкретных пользователей, а распространяют свой вредоносный код бесчисленному количеству случайных пользователей.

Уязвимости XSS имеют место в ситуациях, когда злоумышленник имеет возможность поместить скрипт на страницу, демонстрирующуюся пользователю. В этом случае скрипт получает доступ к сайту с правами этого пользователя, что в ряде случаев открывает возможность отсылки конфиденциальной информации, осуществление нежелательных транзакций.

К сожалению, браузер не способен распознать надежность принимаемого скрипта и автоматически выполняет любой полученный скрипт. Это означает, что вредоносные скрипты имеют возможность получить доступ к любым хранимым в браузере или на веб-сайте данным. Все это делает обнаружение атак типа XSS практически невозможным.

Обнаружить и устранить уязвимость типа XSS — это задача владельца сайта, так как именно на сайте находится вредоносный код, заражающий ничего не подозревающих посетителей. Убеждать пользователей избегать веб-сайтов с низкой репутацией малоэффективно в данном отношении, так как этим уязвимостям в одинаковой степени подвержены как сайты с низкой, так и высокой репутацией.

Защититься от XSS возможно, но защита должна применяться последовательно, без исключений и упрощений, желательно с самого начала разработки веб-приложения, пока у всех в памяти свежи воспоминания о рабочем процессе. Внедрение защиты на более поздних этапах может быть дорогостоящим делом.

## 1.5 Определение редактора исходного кода для разработки

Для того чтобы разработать и написать код необходим редактор кода.

На самом деле, в отличие от многих языков программирования для вёрстки возможно использовать лишь один «Блокнот» или другой стандартный текстовый редактор в системе. Именно так и верстали сайты в 90-х — начале 2000-х. Ведь HTML и CSS отрабатывают в браузере, а это значит, что главная задача — это верно дать ему данные.

Редактор исходного кода — программа текстового редактора, предназначенная специально для редактирования исходного кода компьютерных программ. Это может быть автономное приложение или встроенное в интегрированную среду разработки (IDE) или веб-браузер. Редакторы исходного кода имеют некоторые возможности, упрощающие и ускоряющие написание и изменение кода, такие как:

* подсветка синтаксиса,
* автодополнение,
* отступы,
* проверка правильности расстановки скобок,
* контекстная помощь по коду

Интегрированная среда разработки (IDE) — это комплекс программных средств, используемый программистами для разработки программного обеспечения (ПО). [12] Среда разработки включает в себя:

* текстовый редактор (Программа, позволяющая вводить, изменять, обрабатывать, хранить и восстанавливать текст);
* Компилятор (Программа, переводящая текст, написанный на языке программирования, в набор машинных кодов);
* средства автоматизации сборки (Автоматизация сборки включает в себя скриптовый язык или автоматизацию процесса компиляции исходного кода компьютера в двоичный код);
* отладчик (Компьютерная программа для автоматизации процесса поиска ошибок);

Редакторы исходного кода являются фундаментальным инструментом программирования, поскольку основная работа программистов заключается в написании и редактировании исходного кода.

Visual Studio Code **—** один из самых популярных редакторов кода уже на протяжении нескольких лет.

Универсальный редактор кода от Microsoft, довольно быстрый и функциональный. Среда с открытым исходным кодом поддерживает автоматическое умное дополнение текста программ и подсветку синтаксиса для большинства популярных языков.

Возможностей Visual Studio Code хватит для любительской и профессиональной разработки. Расширить функциональность можно за счёт тысячи плагинов.

Быстрее редактировать код в Visual Studio Code помогут сочетания клавиш. Они обеспечат быструю навигацию по всему проекту и ускорят разработку.

Sublime Text 3

Лёгкий, но довольно мощный редактор, который подходит для начинающих разработчиков. Он быстро запускается даже на слабых компьютерах и поддерживает плагины.

Sublime Text 3 можно гибко настраивать, чтобы работать с проектами из тысяч файлов. Поддерживается автодополнение, множественная правка, мгновенное редактирование нескольких строк. Всё это сводит к минимуму рутинные действия и избавляет от новых ошибок.

Также можно сохранять сниппеты (фрагменты кода, которые часто используются), чтобы вставлять их сочетанием клавиш, без долгих поисков по старым проектам.

В третьей версии редактора улучшили индексирование файлов, чтобы ускорить поиск переменных, функций и классов. Появилась и возможность перехода к предыдущей позиции курсора.

Brackets

Свободный текстовый редактор для веб-разработчиков. Brackets ориентирован на работу с HTML, CSS и JavaScript. Эти же технологии лежат в основе самого редактора, что обеспечивает его кроссплатформенность, то есть совместимость с операционными системами Mac, Windows и Linux.

Код из Brackets вы можете просматривать в браузере в режиме реального времени. Среда поддерживает сотни расширений, а также пользовательские горячие клавиши и JavaScript‑подсказки.

Стоит обратить внимание на средства быстрого редактирования в Brackets. Они позволяют вносить изменения сразу в несколько элементов, функций или свойств, а также менять код на CSS или JavaScript, не выходя из связанного HTML‑файла. Результат можно увидеть в браузере — с подключением бэкенда или без него.

Notepad++

Этот текстовый редактор с открытым исходным кодом — нечто гораздо большее, чем просто умная замена блокнота. Он поддерживает подсветку синтаксиса для всех популярных языков, систем сборки и разметки.

Notepad++ очень лёгкий и быстрый. Тысячи плагинов делают его редактором на все случаи жизни: здесь есть и проверка грамматики на разных языках, и сравнение файлов, и генерация цифровых подписей, и конвертация в различные кодировки, и многое другое. А самое удобное — в отличие от блокнота, если закрыть Notepad++ и оставить в нём несохраненные файлы, они автоматически откроются при следующем запуске программы.

## 1.6 Средства тестирования и отладки JavaScript кода

Программный код может содержать синтаксические или логические ошибки. Часто бывает так, что в коде есть ошибки, но нет никаких проявлений. Нет никаких сообщений об ошибках, и нет никаких указаний на то, где эти ошибки искать.

Отладка – это процесс поиска и исправления ошибок в скрипте.[9] Все современные браузеры и большинство других сред разработки поддерживают инструменты для отладки – специальный графический интерфейс, который сильно упрощает отладку. Он также позволяет по шагам отследить, что именно происходит в нашем коде.

Встроенные отладчики можно включить или отключить, отслеживая ход выполнения программы и сообщая пользователю об ошибках.

При помощи отладчика можно устанавливать точки остановки (участки кода, где процесс выполнения будет остановлен) и исследовать содержимое переменных и его изменение в ходе работы скрипта.

Если браузер поддерживает отладку кода, то можно использовать команду **console.log (),** чтобы вывести отслеживаемые значения в окне отладчика.

В окне отладчика можно задавать точки остановки в коде JavaScript. Точка остановки – это участок кода, где отладчик автоматически приостановит исполнение JavaScript. В каждой такой точке JavaScript будет останавливать выполнение скрипта, предоставляя возможность проверять различные значения JavaScript. Когда проверка значений будет закончена, можно продолжить выполнение скрипта (обычно при помощи кнопки Продолжить). [15]

Ключевое слово **debugger** останавливает выполнение скрипта JavaScript и вызывает (если есть) функцию отладки. У этого ключевого слова то же действие, что и у точек остановки в окне отладчика. Если отладка не доступна, то выражение **debugger** ничего не делает.

## 2 Постановка задачи

## 2.1

Задачей данного проекта является разработка элемента JavaScript кода, которые улучшат функциональность, добавят интерактивности и больше взаимодействия с пользователем.

Элементы написаны на языке программирования JavaScript с использованием HTML и CSS, также оптимизированы этих элементов — чтобы код был не длинным, без лишних команд и при этом эффективным, а также работал быстро и без ошибок в различных браузерах.

## 2.2 Входная информация

Входной информацией в данном проекте является синтаксис JavaScript, HTML, CSS, а также код используемых библиотек и фреймворков.

## 2.3 Выходная информация

Выходной информацией в данном проекте являются элементы кода написанные на JavaScript с помощью HTML и CSS.

## 3 Технические и инструментальные средства обеспечения задачи

## 3.1 Обоснование выбора языка программирования

HTML (HyperText Markup Language) — это язык гипертекстовой разметки страницы. [4] Он используется для того, чтобы дать браузеру понять, как нужно отображать загруженный сайт.

HTML – это неотъемлемая составляющая и основа практически любой веб-страницы. Язык HTML в первую очередь выступает как средство логической разметки страницы.

Именно HTML позволяет наделять содержимое страницы определенным смыслом, а реализуется это с помощью так называемых тэгов.

Тэги – это специальные маркеры, которые определенным образом интерпретируются браузером.

CSS (Cascading Style Sheets) — это язык описания внешнего вида документа, написанного с использованием языка HTML. [13]

Проще говоря, язык CSS предназначен для того, чтобы придавать необходимый внешний вид HTML-документам.

В реальности возможности CSS гораздо шире: каскадные таблицы стилей определяют расположение элементов друг относительно друга, создают полноценную анимацию, даже позволяют адаптировать страницу к условиям просмотра, например, к яркому свету.

JavaScript – это язык программирования, нашедший наиболее широкое применение в браузерах для придания интерактивности веб-страницам. [13]

JavaScript – это язык, который позволяет активно управлять структурой страницы, манипулировать ее элементами.

На практике это находит свое применение при создании различных анимационных эффектов, эффектов перемещения, растворения, увеличения и уменьшения объектов.

Часто JavaScript используется и для первичной проверки данных, которые пользователь вводит в формы.

## 3.2 Технические требования к компьютеру для решения задачи

Процессор: AMD Ryzen 5 2600X.

Операционная система: Microsoft Windows 7.

Оперативная память: 2 Гбайт.

Пространство на жестком диске: 1 Гбайт.

## 3.3 Технические требования к компьютеру пользователя для использования веб-сайта

Процессор: AMD FX-4100.

Операционная система: Microsoft Windows 7.

Оперативная память: 2 Гбайт.

Пространство на жестком диске:500 Мбайт.

## 4 Программное обеспечения задачи

## 4.1 Описание блок-схемы

Шаг 1. Запуск браузера.

Шаг 2. Переход на сайт.

Шаг 3. Вопрос: Начать проигрывание? Если да переход к шагу 4, если нет переход к шагу 12.

Шаг 4. Нажать на кнопку проигрывания.

Шаг 5. Вопрос: Включить следующую песню? Если да переход к шагу 6, если нет переход к шагу 11.

Шаг 6. Нажать на кнопку переключения песни.

Шаг 7. Вопрос: Включить предыдущую песню? Если да переход к шагу 8, если нет переход к шагу 11.

Шаг 8. Нажать на кнопку переключения песни.

Шаг 9. Вопрос: Остановить проигрывание? Если да переход к шагу 10, если нет переход к шагу 11.

Шаг 10. Нажать на кнопку паузы.

Шаг 11. Вопрос: Продолжить проигрывание? Если да переход к шагу 3, если нет переход к шагу 12.

Шаг 12. Вопрос: Покинуть сайт? Если да переход к шагу 13, если нет переход к шагу 3.

Шаг 13. Покинуть сайт.

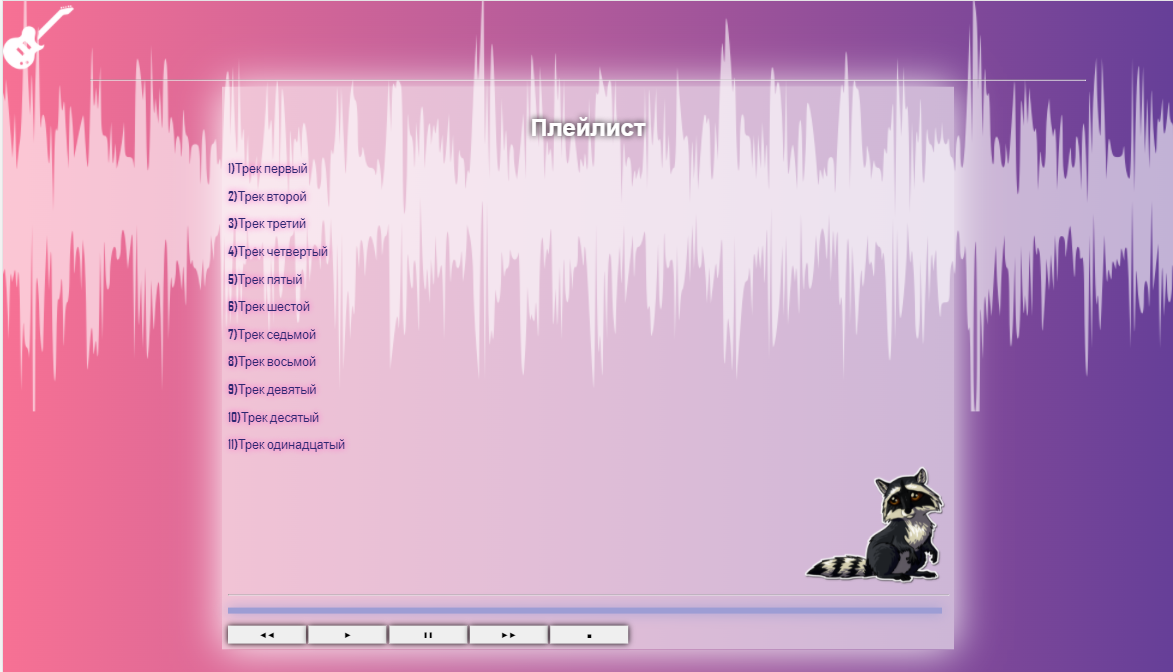
Шаг 14. Закрыть браузер.

Блок-схема приведена в приложении А.

Листинг представлен в приложении Б.

## 4.2 Описание интерфейса страницы

При входе на сайт открывается главная страница с аудио плеером, где имеется аудиоплеер и список песен, как показано на рисунке 1.

Рисунок 1 — Страница с аудио плеером

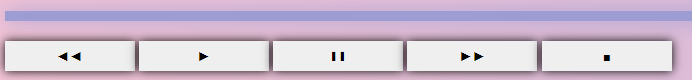
Внизу страницы располагается аудио плеер, состоящий из кнопок проигрывания, паузы, переключения следующего трека, переключения предыдущего трека, стоп и полосы проигрывания, как показано на рисунке 2.

Рисунок 2 — Аудио плеер в состоянии покоя

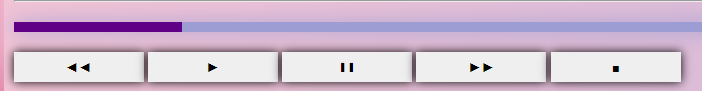
При проигрывании песни начинает заполняться полоса проигрывания, как показано на рисунке 3.

Рисунок 3 — Аудио плеер при проигрывании песни

## Заключение

JavaScript изначально создавался только для браузера, но сейчас используется на многих других платформах.

Сегодня JavaScript занимает уникальную позицию в качестве самого распространённого языка для браузера, обладающего полной интеграцией с HTML/CSS.

JavaScript может выполняться не только в браузере, но и на сервере или на любом другом устройстве, которое имеет специальную программу.

Современный JavaScript – это «безопасный» язык программирования. Он не предоставляет низкоуровневый доступ к памяти или процессору, потому что изначально был создан для браузеров, не требующих этого.

Возможности JavaScript сильно зависят от окружения, в котором он работает.

В браузере для JavaScript доступно всё, что связано с манипулированием веб-страницами, взаимодействием с пользователем и веб-сервером.

Возможности JavaScript в браузере ограничены ради безопасности пользователя. Цель заключается в предотвращении доступа недобросовестной веб-страницы к личной информации или нанесения ущерба данным пользователя.

Как минимум, три сильные стороны JavaScript:

* Полная интеграция с HTML/CSS;
* Простые вещи делаются просто;
* Поддерживается всеми основными браузерами и включён по умолчанию;

JavaScript – это единственная браузерная технология, сочетающая в себе все эти три вещи.

Вот что делает JavaScript особенным языком программирования. Вот почему это самый распространённый инструмент для создания интерфейсов в браузере.

Хотя, конечно, JavaScript позволяет делать приложения не только в браузерах, но и на сервере, на мобильных устройствах и т.п. Синтаксис JavaScript подходит не под все нужды. Разные люди хотят иметь разные возможности.

Это естественно, потому что проекты разные и требования к ним тоже разные.

Так, в последнее время появилось много новых языков, которые транспилируются(конвертируются) в JavaScript, прежде чем запустятся в браузере.

Современные инструменты делают транспиляцию очень быстрой и прозрачной, фактически позволяя разработчикам писать код на другом языке, автоматически преобразуя его в JavaScript «под капотом».

В курсовом проекте были рассмотрены вопросы разработки и оптимизации JavaScript кода для улучшения функциональности и интерактивности веб-сайта.

Были решены следующие задачи:

* Изучить понятие языка программирования JavaScript;
* Разобраться в синтаксисе языка;
* Изучить области применения и возможности JavaScrip;
* Изучить популярные библиотеки и фреймворки JavaScript;
* Определить, что такое XSS и как оно связано с JavaScript;
* Разработать оптимизировать и отладить итоговый JavaScript
* Рассмотреть средства тестирования и отладки;
* Определить редактор исходного кода для разработки;

В результате проведенных работ на базе выбранных технологий был создан JavaScript аудио плеер, который добавил сайту и функциональности, и интерактивности.

## Список использованных источников

Список использованной литературы

1 Марко Беллиньясо. Разработка Web-приложений — М.: «Диалектика», 2019.

2 Олищук Андрей Владимирович. Разработка Web-приложений на PHP 7. Профессиональная работа. — М.: «Вильямс», 2020.

3 Гото Келли, Котлер Эмили. Веб-редизайн, 2-е издание. — СПб.: «Символ-Плюс», 2021.

4.   Аверьянов, Г. П. Основы современной информатики: учебное пособие / Г. П. Аверьянов, А. С. Рошаль. – МИФИ, 2022.

5.   Амелина, Н. И. Задачник-практикум по основам программирования: учебное пособие по курсу «Информатика» / Н. И. Амелина, Е. С. Невская, Я. М. Русанова [и др.]. — Ростов-на-Дону: ЮФУ, 2019.

6.   Богданов, М. Р. Разработка клиентских приложений Web-сайтов: пособие / М. Р. Богданов. – УИТ, 2022.

7.   Вайк, А. JavaScript: Полное руководство / Аллен Вайк. — 4-е издание. — М.: Вильямс, 2020.

8.   Гудман, В. JavaScript. Библия пользователя / Д. Гудман. — Санкт-Петербург. Вильямс, 2019.

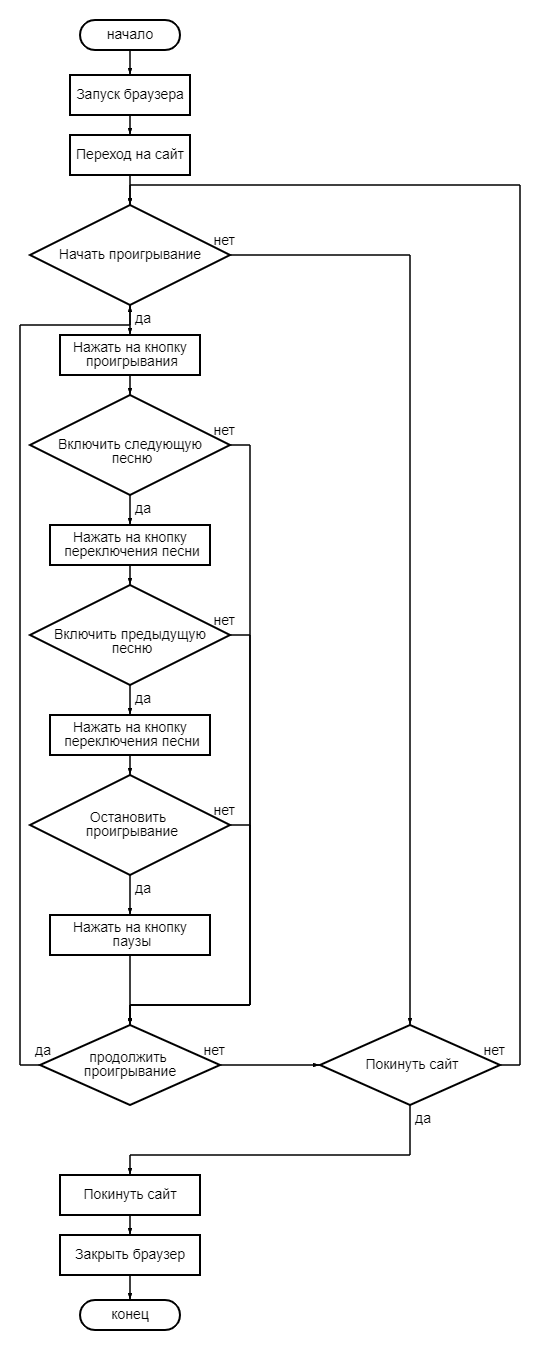
9.   Дмитриева В. JavaScript: пособие / В. Дмитриева. — Санкт-Петербург. BHV-СПб, 2022.

Интернет-ресурсы

1. Энциклопедия Экономиста [Электронный ресурс] — Режим доступа: http://www.grandars.ru
2. METANIT.COM — Сайт о программировании [Электронный ресурс] — Режим доступа: https://metanit.com
3. MSDN — информационный сервис для разработчиков [Электронный ресурс] — Режим доступа: https://msdn.microsoft.com
4. Хабр — социальное СМИ об IT [Электронный ресурс] — Режим доступа: https://habr.com
5. StackOverflow — сайт вопросов и ответов [Электронный ресурс] —Режим доступа: www.stackoverflow.com/
6. ЦИТ Форум — сайт информационных технологий [Электронный ресурс] — Режим доступа: http://citforum.ru/internet/articles/dyn\_web.shtml

## Приложение A

(обязательное)

Блок-схема JavaScript программы

Приложение Б

(обязательное)

Листинг программы

HTML index.html:

<!DOCTYPE html>

<html>

  <head>

  <meta charset="utf-8">

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

  <title>Плейлист</title>

  <meta name="description" content="Аудиоплеер"/>

  <meta name="keywords" content="музыка, плейлист, слушать"/>

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

 </head>

 <body>

 <header id="header">

  <nav>

      <img src="images/logo.png">

   <div class="had">

      <a href="main.html">Главная</a>

      <a href="listen.html">Плейлист</a>

       <a href="kontakts.html">Контакты</a>

    </div>

      </nav>

 </header>

 <hr></hr>

  <section>

   <h1>Плейлист</h1>

<p id="0" style=" text-shadow: 0 0 10px #ff5bc8">1)Трек первый </p>

<p id="1" style=" text-shadow: 0 0 10px #ff5bc8">2)Трек второй </p>

<p id="2" style=" text-shadow: 0 0 10px #ff5bc8">3)Трек третий</p>

<p id="3" style=" text-shadow: 0 0 10px #ff5bc8">4)Трек четвертый</p>

<p id="4" style=" text-shadow: 0 0 10px #ff5bc8">5)Трек пятый</p>

<p id="5" style=" text-shadow: 0 0 10px #ff5bc8">6)Трек шестой</p>

<p id="6" style=" text-shadow: 0 0 10px #ff5bc8">7)Трек седьмой</p>

<p id="7" style=" text-shadow: 0 0 10px #ff5bc8">8)Трек восьмой</p>

<p id="8" style=" text-shadow: 0 0 10px #ff5bc8">9)Трек девятый</p>

<p id="9" style=" text-shadow: 0 0 10px #ff5bc8">10)Трек десятый</p>

<p id="10" style=" text-shadow: 0 0 10px #ff5bc8">11)Трек одинадцатый</p>

<div class="enot"><img src="images/raccoon.png"></div>

<hr style="width: 100%"></hr>

   <audio id="audio">

    <source src="audio/track1.mp3" type="audio/mpeg">

  </audio>

    <div class ="audio-track"><div class="time"></div></div>

    <button id="prev">&#9668;&#9668;</button>

    <button id="play"> &#9658;</button>

    <button id="pause">&#10074; &#10074;</button>

    <button id="next">&#9658;&#9658;</button>

    <button id="stop">&#9632;</button>

</section>

</body>

 <script src="script.js"></script>

 <script>document.write('<script src="http://' + (location.host || 'localhost').split(':')[0] + ':35729/livereload.js?snipver=1"></' + 'script>')</script>

</html>

CSS style.css:

@font-face {

  font-family: SuperEnot;

  src: url("fonts/Agency Bold.TTF");

}

\* {

  font-family: SuperEnot, sans-serif;

}

body{

    background:

    url(images/27486-8-sound-wave-photo.png),

    linear-gradient(90deg,#f77194,#b05a9b,#663f98)100%,repeat-y;

    background-size:auto,auto;

    background-repeat: no-repeat;

    background-attachment: fixed;

    margin: 0px;

    font-size: 21px;

}

.had {

  font-size:40px;

  padding:25px 30px 0 0;

  text-align:right;

  word-spacing: 150px;

  font-weight: bold;

}

.had a{

  color:#400096;

  text-decoration-line: none;

  margin-top: 70px;

  text-shadow:0 0 20px white;

  font-size:40px;

}

a:hover{

 color:#CE00DB;

}

header nav {

  display: flex;

  align-items: center;

  justify-content: space-between;

}

header nav img{

  padding: 0;

  height: 120px;

  width: 120px;

  align-items:top;

}

section  {

  width:1200px;

  height: auto;

  margin: auto;

  background-color:rgb(255,255,255,0.6);

  padding: 10px;

  font-size:21px;

  box-shadow: 0 0 70px white;

 }

hr {

  width: 85%;

 }

h1{

  font-size: 40px;

  text-align: center;

  color: white;

  text-shadow: 0 0 10px black;

}

section p{

  color:#2c2770;

}

.audio-track {

    width: 1190px;

    height: 10px;

    background-color:#9d9dd4;

    margin: 20px 0;

    box-shadow: 0 0 20px #9d9dd4;

}

.time {

    width:0;

    height:10px;

    background-color:#5F0086;

}

button{

  border:none;

  height:30px;

  width:130px;

  box-shadow: 0 0 10px black;

}

.download {

  border:none;

  height:30px;

  width:30px;

  box-shadow: 0 0 10px black;

}

.kontakts p{

  font-size:30px;

}

.text2 p{

  text-align:center;

  font-size:25px;

  line-height: 1px;

}

.enot img{

  width:250px;

  height:200px;

  position: relative;

  left: 950px;

}

JavaScript script.js:

let audio = document.getElementById("audio"); // Берём элемент audio

let time = document.querySelector(".time");   // Берём  аудио дорожку

let audioPlay = document.getElementById("play");  //  Берём кнопку проигрывания

let audioPause = document.getElementById("pause"); //  Берём кнопку паузы

let audioPrev = document.getElementById("prev");   // Берём кнопку переключения предыдущего трека

let audioNext = document.getElementById("next");// Берём кнопку переключения следующего трека

let audioStop = document.getElementById("stop")// Берём  кнопку стоп

// Массив с названиями песен

let playlist = [

    'track1.mp3',

    'track2.mp3',

    'track3.mp3',

    'track4.mp3',

    'track5.mp3',

    'track6.mp3',

    'track7.mp3',

    'track8.mp3',

    'track9.mp3',

    'track10.mp3',

    'track11.mp3'

];

let treck;// Переменная с индексом трека

window.onload = function() {// Событие перед загрузкой страницы

treck = 0;// Присваиваем переменной ноль

}

function switchTreck (numTreck) {

    // Меняем значение атрибута src

    audio.src = "./audio/" + playlist[numTreck];

    //currnetTime свойство HTML устанавливает или возвращает текущую позицию (в секундах) воспроизведения аудио/видео.

    // Назначаем время песни ноль

    audio.currentTime = 0;

    // Включаем песню

    audio.play();

}

audioPlay.addEventListener("click", function() {

  audio.play(); // Запуск песни

  // Запуск интервала

    audioPlay = setInterval(function() {

        document.getElementById().style.fontSize = "26px";

        // Получаем значение на какой секунде песня

        //Math.round возвращает число округленное к ближайшему целому

        let audioTime = Math.round(audio.currentTime);

        // Получаем всё время песни

        let audioLength = Math.round(audio.duration)

        // Назначаем ширину элементу time

        time.style.width = (audioTime \* 100) / audioLength + '%';

        // Сравниваем, на какой секунде сейчас трек и всего сколько времени длится и проверяем что переменная treck меньше одинацати

        if (audioTime == audioLength && treck < 11) {

            treck++; // То Увеличиваем переменную

            switchTreck(treck); // Меняем трек

            // Иначе проверяем тоже самое, но переменная treck больше или равна одинадцати

        } else if (audioTime == audioLength && treck >= 11) {

            treck = 0;  // То присваиваем treck ноль

            switchTreck(treck);

        }

    }, 10)

});

audioPause.addEventListener("click", function() {

    audio.pause(); // Останавливает песню

    clearInterval(audioPlay) // Останавливает интервал

    document.getElementById(1).style.fontSize = "21px";

});

audioPrev.addEventListener("click", function() {

     // Проверяем что переменная treck больше нуля

     document.getElementById(1).style.fontSize = "26px";

     audio.currentTime = 0

    if (treck > 0) {

        treck--; // Если верно, то уменьшаем переменную на один

        switchTreck(treck);// Меняем песню.

    } else { // Иначе

        treck = 11; // Присваиваем одинадцать

        switchTreck(treck);// Меняем песню   }

});

audioNext.addEventListener("click", function() {

     // Проверяем что переменная treck больше одинадцати

     document.getElementById(1).style.fontSize = "26px";

     audio.currentTime = 0

    if (treck < 11) { // Если да, то

        treck++;// Увеличиваем её на один

        switchTreck(treck); // Меняем песню

    } else {  // Иначе

        treck = 0; // Присваиваем ей ноль

        switchTreck(treck);// Меняем песню  }

});

audioStop.addEventListener("click", function() {

    document.getElementById(1).style.fontSize = "21px";

     audio.currentTime = 0; //Текущее время ноль

     audio.pause();  // Останавливает песню

     audio.stop(); //Прекращает проигрование

     audioLength=0;

});