**2 Использование JavaScript и сторонних библиотек для выполнения индивидуального задания**

**2.1 Работа с элементами DOM средствами JavaScript**

Цель**:** Ознакомление с принципами работы с DOM-деревом средствами JavaScript. Научиться создавать и управлять элементами DOM-дерева. Изучить организацию работы с событиями в JavaScript. Научиться обрабатывать пользовательские сценарии с помощью обработчиков событий в JavaScript[3].

Для реализации открытия или закрытия модальных окон в HTML документ подключаются файлы скриптов, представленных на рисунке 2.1:



Рисунок 2.1 – подключаемые файлы скриптов

Источник: собственная разработка

С помощью JavaScript был реализован скрипт для реализации открытия, а также закрытия модального окна:



Рисунок 2.2 – Реализация открытия или закрытия модального окна

Источник: собственная разработка

Также для индивидуального задания был разработан скрипт, реализовывающий перенаправление пользователей при нажатии на кнопку в разделе популярных товаров. Скрипт отображен на рисунке 2.3:

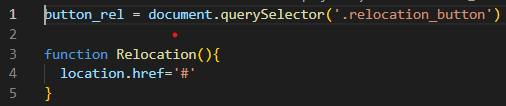


Рисунок 2.3 – Реализация функции перенаправления пользователя

Источник: собственная разработка

Контрольные вопросы:

что такое DOM;

что такое цель события или целевой элемент (event.target);

что такое текущая цель события (event.currentTarget);

в чем разница между ключевыми словами «var», «let» и «const»;

что такое «стрелочные функции» (Arrow Functions);

для чего необходимо удалять неиспользуемые обработчики событий;

что такое область видимости;

что такое анонимная функция;

что такое callback-функция? Приведите пример использования callback-функции;

зачем нужен метод «preventDefault»;

как выполняется поиск элементов по DOM-дереву.

Ответы на контрольные вопросы:

DOM – независящий от платформы и языка программный интерфейс, позволяющий программам и скриптам получить доступ к содержимому HTML-, XHTML- и XML-документов, а также изменять содержимое, структуру и оформление таких документов;

самый глубокий элемент, который вызывает событие, называется целевым элементом;

цель события определяет элемент, в котором в данный момент обрабатывается событие, при движении события внутри DOM. Всегда совпадает элементом, на котором обработчик события был назначен;

оператор «var» объявляет переменную, инициализируя её, при необходимости. Директива «let» объявляет переменную с блочной областью видимости с возможностью инициализировать её значением, а «const» создаёт константу (новую именованную ссылку на область памяти), доступную только для чтения;

стрелочные функции – анонимные функции, привязанные к значению «this» (но не привязаны к собственному «this», «arguments», «super», или «new.target»);

очищать обработчики событий важно, потому что каждый обработчик занимает место в памяти и выполняется всякий раз, когда срабатывает событие. Если не убирать неиспользуемые обработчики событий, то можно столкнуться с неожиданным поведением проекта;

область видимости – важная концепция, определяющая доступность переменных. Данная концепция лежит в основе замыканий, разделяя переменные на глобальные и локальные;

анонимная функция – функция без имени, которая вызывается по ссылке на функцию.

callback-функция – функция, которая должна быть выполнена после того, как другая функция завершила выполнение.

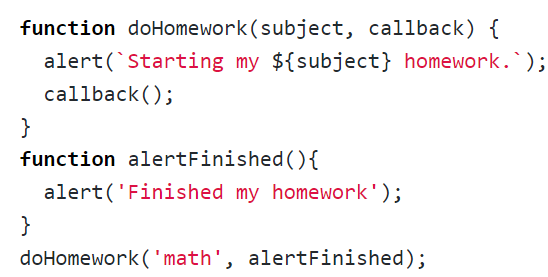


Рисунок 2.4 - callback-функция

Источник: собственная разработка

На рисунке 2.4 отображен пример реализации callback-функции;

метод «preventDefault()» интерфейса «Event» сообщает «User agent», что если событие не обрабатывается явно, его действие по умолчанию не должно выполняться так, как обычно;

наиболее универсальным способом поиска является поиск по селектору.

**2.2 Назначение и особенности использования библиотеки jQuery и jQuery UI. Современные библиотеки и их использование**

Цель: Ознакомление с назначением и особенностями использования библиотеки jQuery[4] и jQuery UI. Изучить официальную документацию по данным библиотекам. Изучить организацию работы с библиотеками и сферу их применения. Ознакомиться с особенностями работы библиотеки SwiperJS[5]. Реализовать функционал слайдера с использованием сторонней библиотеки SwiperJS.

Для индивидуального задания реализован функционал слайдера с использованием библиотеки SwiperJS, представленный на рисунке 2.5:

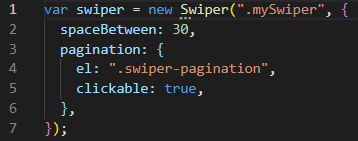


Рисунок 2.5 – реализация функционала слайдера

Источник: собственная разработка

Контрольные вопросы:

что такое jQuery и для чего она была создана;

какие преимущества и недостатки у jQuery;

что такое jQuery UI;

для чего используются сторонние библиотеки;

укажите назначение библиотеки SwiperJS.

Ответы на контрольные вопросы:

jQuery – набор функций JavaScript, фокусирующийся на взаимодействии JavaScript и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими. Также библиотека jQuery предоставляет удобный API для работы с AJAX;

преимущества jQuery: оптимизация кода, ускорение работы с JavaScript. Недостатки: для простых пользовательских интерфейсов, большой объем файла для импорта, устаревший API;

jQuery UI – библиотека JavaScript с открытым исходным кодом для создания насыщенного пользовательского интерфейса в веб-приложениях, часть проекта jQuery. Построена поверх главной библиотеки jQuery и предоставляет разработчику упрощенный доступ к её функциям взаимодействия, анимации и эффектов, а также набор виджетов;

для ускорения разработки;

создание на сайте слайдеров с информацией.

**2.3** **Подключение библиотеки jQuery к приложению и получение доступа к элементам документа. Работа с элементами DOM при помощи библиотеки jQuery. Создание анимации при помощи библиотеки**

Цель: Ознакомление с подключеним библиотеки jQuery к приложению и полученим доступа к элементам документа. Изучить официальную документацию по библиотеке. Изучить организацию работы с элементами DOM при помощи библиотеки jQuery. Ознакомиться с особенностями работы библиотеки AOS.js. Реализовать анимации на странице сайта с использованием библиотеки AOS.js.

Преподавателем был предоставлен проект «win-calc», использующей библиотек jQuery.

При ознакомлении с скриптом «calc.js», были выявлены мметоды, использующие jQuery, пример метода приведен на рисунке 2.6:

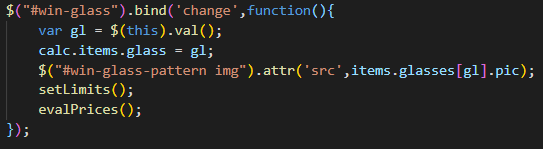


Рисунок 2.6 – реализация функции

Источник: проект «win-calc»

Для индивидуального задания была реализована анимация элементов, для инициализации использовался скрипт, отображенный на рисунке 2.7:



Рисунок 2.7 – инициализация библиотеки AOS

Источник: собственная разработка

Для работы анимации элемента необходимо прописать определенные атрибуты, пример элемента с анимацией приведен на рисунке 2.8:



Рисунок 2.8 – элемент с заданной анимацией

Источник: собственная разработка

Контрольные вопросы:

для чего используются сторонние библиотеки;

укажите назначение следующих методов (команд) библиотеки jQuery: html(), val(), find(), append(), attr(). Приведите аналог этих команд на синтаксисе современного JavaScript;

перечислите и опишите работу методов jQuery, предназначенных для работы с css классами;

укажите назначение библиотеки AOS.js;

назовите способы подключения библиотеки AOS.js к странице;

опишите процесс использования библиотеки AOS.js;

назовите основные настройки, применяемые для управления видом анимации элементов с помощью AOS.js.

Ответы на контрольные вопросы:

для ускорения процесса разработки;

html() – возвращает строку. Получает HTML-содержимое первого элемента в наборе. Метод val() – возвращает или устанавливает значение атрибута value. Метод find() – находит всех потомков выбранного элемента. Метод append() – позволяет вставить содержимое, указанное в параметре метода в конец каждого выбранного элемента в наборе совпавших элементов. Метод attr() - получает или устанавливает значение атрибутов выбранных элементов;

метод добавления класса к элементу – «.addClass», метод получения класса элемента – «.hasClass», метод удаления класса элемента – «.removeClass»;

добавление анимации элементам;

с использованием локального файла, через CDN в файле c расширением «.html»;

для работы с анимацией у элемента указываются определенные атрибуты;

«data-aos», «data-aos-easing», «data-aos-delay», «data-aos-offset».

В индивидуальном задании реализована работа с библиотекой «Cleave.js».

**2.4 Использование виджетов jQuery UI. Проверка данных форм при помощи библиотеки jQuery**

Цель**:** Ознакомиться с использованием UI виджетов и библиотек. Изучить официальную документацию по библиотекам Cleave.js и Parsley.js, ознакомится с примерами их использования. Реализовать форматирование вводимых данных с использованием Cleave.js и проверку данных форм при помощи библиотеки Parsley.js.

Скрипт, реализовывающий инициализацию и форматирование определенных элементов формы представлен на рисунке 2.9:

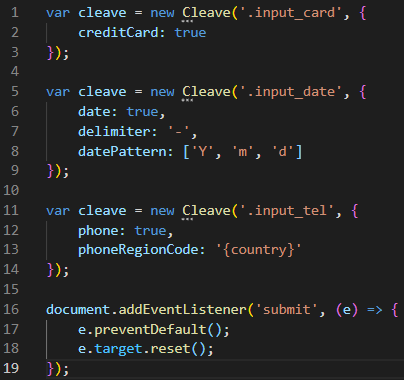


Рисунок 2.9 – реализация библиотеки «Cleave.js»

Источник: собственная разработка

Для проверки корректности введенных данных в форму реализовано использование библиотеки «Parsley.js». Подключение библиотеки осуществляется через CDN, скрипт, реализовывающий инициализацию библиотеки для определенной формы представлен на рисунке 2.10:



Рисунок 2.10 – реализация скрипта для инициализации

Источник: собственная разработка

Для указания обязательных полей для заполнения следует использовать атрибут «required». Пример реализации обязательных полей представлен на рисунке 2.11:



Рисунок 2.11 – реализация обязательных полей формы

Источник: собственная разработка

Контрольные вопросы:

для чего используются сторонние библиотеки;

укажите назначение библиотеки Cleave.js;

поясните процесс форматирования телефонного номера, даты, номера кредитной карты. Приведите примеры настроек формата;

для чего телефонные коды стран разделены на отдельные файлы;

укажите назначение библиотеки Parsley.js;

опишите процесс использования библиотеки Parsley.js;

какой компонент является обязательным для работы Parsley.js.

Ответы на контрольные вопросы:

для ускорения процесса разработки;

библиотека «Cleave.js» предназначена для форматирования полей формы;

после инициализации и настройки определенных полей, библиотека форматирует введенный текст (ставит «-», пробел), а также может ограничивать количество символов в полях;

ускорение работы, уменьшение размера файла;

валидация полей форм;

после подключения библиотеки для выбора обязательных полей нужно элементам задать атрибут «required», затем библиотека автоматически будет проводить валидацию полей с атрибутом «required»;

jQuery.