МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский национальный исследовательский университет

информационных технологий, механики и оптики»

**Факультет программной инженерии и компьютерной техники**

Направление (специальность) ⎯ 09.04.02 Информационные системы и технологии

Специализация ⎯ Веб-технологии

Дисциплина — Проектирование и анализ языков веб-решений

Курсовой проект (работа)

ТЕМА: Исследование интерактивности на основе JavaScript

ВЫПОЛНИЛ

Студент группы   Р41622 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Елисеева Е.C.

№  группы подпись, дата ФИО

ПРОВЕРИЛ к.пед.н., доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Государев И.Б.

ученая степень, должность подпись, дата ФИО

САНКТ-ПЕТЕРБУРГ  
2019 г.

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**федеральное государственное автономное образовательное учреждение**

**высшего образования**

**«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,**

**МЕХАНИКИ И ОПТИКИ»**

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ (РАБОТУ)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Студент | | Елисеева Екатерина Сергеевна | | | | | | |
| (Фамилия, И., О.) | | | | | | | | |
| Факультет | ПИиКТ | | | | | | Группа | P41622 |
| Направление (специальность) | | | | | | 09.04.02 «Информационные системы и технологии» | | |
| Руководитель | | | | | Государев И.Б., Университет ИТМО, доцент, к.пед.н. | | | |
| (Фамилия, И., О., место работы, должность, ученое звание, степень) | | | | | | | | |
|  | | | | | | | | |
| Дисциплина | | | | Проектирование и анализ языков веб-решений | | | | |
|  | | | | | | | | |
| Наименование темы: | | | Исследование интерактивности на основе JavaScript | | | | | |

Задание

Исследовать технологии, применяемые при разработке интерактивных элементов веб-сайтов на JavaScript.

Краткие методические указания

Исследовать понятия интерактивности и интерактивных элементов веб-сайтов. Проанализировать и описать технологии и инструменты, применяемые при разработке интерактивных элементов на JavaScript. Представить результаты в виде отчета и статического сайта в составе веб-портфолио

Содержание пояснительной записки

Оглавление. Введение. Ход выполнения работы – иллюстрированное описание языка, грамматики, библиотек, фреймворков, действий и применяемых программ, команд, приёмов, параметров, а также обоснование используемых средств, моделей, параметров. Заключение. Список использованной литературы.

Рекомендуемая литература

1. Пьюривал С. Основы разработки веб-приложений.—СПб.: Питер, 2015.—272 с.

2. В.И. Бойков, С.В. Быстров, А.С. Кремлев, К.А. Сергеев. Правила оформления курсовых и квалификационных работ. — СПб: СППб ГУ ИТМО, 2007. – 39 с.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Государев И.Б.   
 (подпись) (Фамилия И.О.)

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Елисеева Е.С.  
 (подпись) (Фамилия И.О.)

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,   
МЕХАНИКИ И ОПТИКИ**

**ГРАФИК КУРСОВОГО ПРОЕКТА (РАБОТЫ)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Студент | | Елисеева Екатерина Сергеевна | | | | | |
| (Фамилия, И., О.) | | | | | | | |
| Факультет | ПИиКТ | | | | | Группа | P41622 |
| Направление (специальность) | | | | | 09.04.02 «Информационные системы и технологии» | | |
| Руководитель | | | | Государев И.Б., Университет ИТМО, доцент, к.пед.н. | | | |
| (Фамилия, И., О., место работы, должность, ученое звание, степень) | | | | | | | |
|  | | | | | | | |
| Дисциплина | | | Проектирование и анализ языков веб-решений | | | | |
|  | | | | | | | |
| Наименование темы: | | | | Исследование интерактивности на основе JavaScript | | | |
|  | | | | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Наименование этапа | Дата завершения | | Оценка и подпись руководителя |
| Планируемая | Фактическая |
| 1. | Исследование интерактивных элементов веб-сайтов, выявление технологий, применяемых при разработке интерактивных элементов. | Апрель 2019 |  |  |
| 2. | Исследование и описание технологий разработки интерактивных элементов на JavaScript. Написание отчета. Защита проекта. | Май 2019 |  |  |

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Государев И.Б.

(подпись) (Фамилия И.О.)

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Елисеева Е.С.

(подпись) (Фамилия И.О.)

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,   
МЕХАНИКИ И ОПТИКИ**

**АННОТАЦИЯ К КУРСОВОМУ ПРОЕКТУ (РАБОТЕ)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Студент | | Елисеева Екатерина Сергеевна | | | | | |
| (Фамилия, И., О.) | | | | | | | |
| Факультет | ПИиКТ | | | | | Группа | P41622 |
| Направление (специальность) | | | | | 09.04.02 «Информационные системы и технологии» | | |
| Руководитель | | | | Государев И.Б., Университет ИТМО, доцент, к.пед.н. | | | |
| (Фамилия, И., О., место работы, должность, ученое звание, степень) | | | | | | | |
|  | | | | | | | |
| Дисциплина | | | Проектирование и анализ языков веб-решений | | | | |
|  | | | | | | | |
| Наименование темы: | | | | Исследование интерактивности на основе JavaScript | | | |
|  | | | | | | | |

ХАРАКТЕРИСТИКА КУРСОВОГО ПРОЕКТА (РАБОТЫ)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **1. Цель и задачи работы** | | Предложены студентом | | Определены руководителем | |
| Цель работы — исследовать технологий разработки интерактивных элементов веб-сайтов на JavaScript.  Задачи работы: | | | | | |
| 1. ­­Определить понятия интерактивности и интерактивных элементов веб-сайтов | | | | | |
| 1. Проанализировать и описать технологии и инструменты, с помощью которых | | | | | |
| реализуется интерактивность на JavaScript | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| **2. Характер работы** | | | | | |
|  | | | | | |
| Расчет | Конструирование | | Моделирование | | Другое |
| **3. Содержание работы**  ­­Определены понятия интерактивности и интерактивных элементов веб-сайтов, | | | | | |
| исследованы и описаны технологии и инструменты, с помощью которых реализуется | | | | | |
| интерактивность на JavaScript | | | | | |
|  | | | | | |
| **4. Выводы** | |  | | | |
| Требования к проекту реализованы | | | | | |
|  | | | | | |

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Государев И.Б.   
 (подпись) (Фамилия И.О.)

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Елисеева Е.С.

(подпись) (Фамилия И.О.)

**ОГЛАВЛЕНИЕ**

[Введение 6](#_Toc10049175)

[1 Понятие интерактивности веб-сайтов 7](#_Toc10049176)

[2 Интерактивность на JavaScript 12](#_Toc10049177)

[2.1 DOM 12](#_Toc10049178)

[2.2 События 13](#_Toc10049179)

[2.3 AJAX 17](#_Toc10049180)

[2.4 Императивный и декларативный подходы 18](#_Toc10049181)

[2.5 Паттерны проектирования 18](#_Toc10049182)

[2.6 Фреймворки и библиотеки 21](#_Toc10049183)

[Заключение 23](#_Toc10049184)

[Список использованных источников 24](#_Toc10049185)

# Введение

**Актуальность темы.** Сегодня, когда пользователи интернета посещают веб-сайт, они ожидают определенного уровня интерактивности. Когда появились сайты второго поколения, известные как Web2.0, компании стали предлагать целый ряд интерактивных функций на своих сайтах, таких как видео, подкасты, формы, которые позволяют пользователям оставлять комментарии или отзывы. Сейчас интерактивные сайты могут предоставлять виртуальные туры по недвижимости или анимированные отображения погодных условий. Веб-сайт компании является ее лучшим маркетинговым инструментом [1]. Главная задача интерактивных сайтов в том, чтобы сделать взаимодействие с пользователем максимально привлекательным, что играет большую роль в показателях конверсии, лояльности и узнаваемости бренда.

Основополагающими технологиями, на которые опираются практически все веб-сайты являются HTML, CSS и JavaScript. HTML — это язык разметки, используемый для определения структуры веб-страниц, CSS используется для изменения стилей элементов, JavaScript добавляет поведение к веб-странице. JavaScript позволяет динамически изменять стили, анимировать меню, создавать функциональные кнопки, проверять данные, введенные в форму [2].

Многие из вещей, обычно выполняемых с помощью JavaScript, требуют большого количества кода. При усложнении интерактивных элементов и создании веб-приложений часто используются библиотеки и фреймворки JavaScript, позволяющие упорядочить и сократить код, необходимый для выполнения типичных задач [3].

**Цель работы** – исследование технологий разработки интерактивных элементов веб-сайтов на JavaScript.

Для достижения цели необходимо решить следующие **задачи**:

* определить понятия интерактивности и интерактивных элементов веб-сайтов;
* проанализировать и описать технологии и инструменты с помощью которых реализуется интерактивность на JavaScript.

1. Понятие интерактивности веб-сайтов

Применительно к веб-сайтам под понятием интерактивность подразумевается взаимодействие или коммуникация. В процессе такого взаимодействия сайт выступает посредником между своими создателями (компанией, организацией, сообществом, отдельным человеком) и посетителями, то есть, интерактивность сайта проявляется как реакция на действия пользователя в рамках предусмотренного сценария [4]. К таким сценариям можно отнести возможность показывать или скрывать информацию в ответ на запрос, передавать авторам сайта сообщение от посетителя, отображать комментарии или материал, добавленные пользователями.

Выделяют четыре основных формы интерактивности на веб-сайтах:

* встроенное интерактивное содержание – это видео и аудио элементы, флэш-игры, презентации, анимационные ролики, которые могут быть воспроизведены непосредственно на сайте;
* интерактивные элементы интерфейса – это элементы, так или иначе откликающиеся на действия пользователя и изменяющие в результате этих действий видимое содержание сайта. Примеры таких элементов – ссылки, вкладки, позволяющие переключаться между блоками содержания, слайдеры, модальные окна;
* получение информации от пользователей – это контактные формы, подписки, регистрации, опросы;
* прямая коммуникация и пользовательское содержание – это элементы, позволяющие пользователям вступить в прямой диалог с представителями организации или авторами сайта, изменить содержание сайта, разместив на нем свой контент. Примерами служат комментарии, системы вопрос-ответ, онлайн чаты [5].

Внимательный подход к разработке интерактивных элементов пользовательского интерфейса позволяет создать качественный цифровой продукт и обеспечить положительный опыт взаимодействия. Существует большое количество различных интерактивных элементов пользовательского интерфейса, далее будут рассмотрены наиболее распространенные из них.

Кнопка является одним из самых распространенных элементов любого интерфейса. Она представляет собой интерактивный элемент, который позволяет пользователю отправлять определенную команду системе. В большинстве случаев этот элемент взаимодействия напрямую связывается с веб-сайтом или приложением и передает необходимые команды для достижения определенной цели: отправить электронное письмо, купить продукт, загрузить данные, добавить предметы в корзину (рисунок 1).

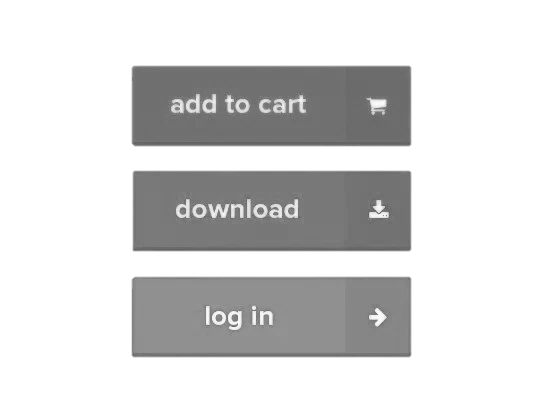


Рисунок 1 - Кнопки

Панель (bar) является основой, на которой расположены другие интерактивные элементы пользовательского интерфейса (рисунок 2). Панель позволяет пользователю быстро выполнить основные шаги взаимодействия с продуктом, а также может информировать о текущем этапе процесса. Среди основных типов баров можно выбелить следующие:

* панель вкладок — в мобильных приложениях отображается в нижней части экрана и обеспечивает возможность быстрого переключения между основными разделами содержимого;
* панель загрузки — это элемент управления, который информирует пользователя о текущем этапе действия или процесса. В большинстве случаев пользователи могут видеть поток с указанием времени или процента;
* индикатор выполнения обеспечивает обратную связь по результатам текущего процесса, например, показывая, сколько запланированной деятельности было выполнено. Например, часто применяется в музыкальных проигрывателях.

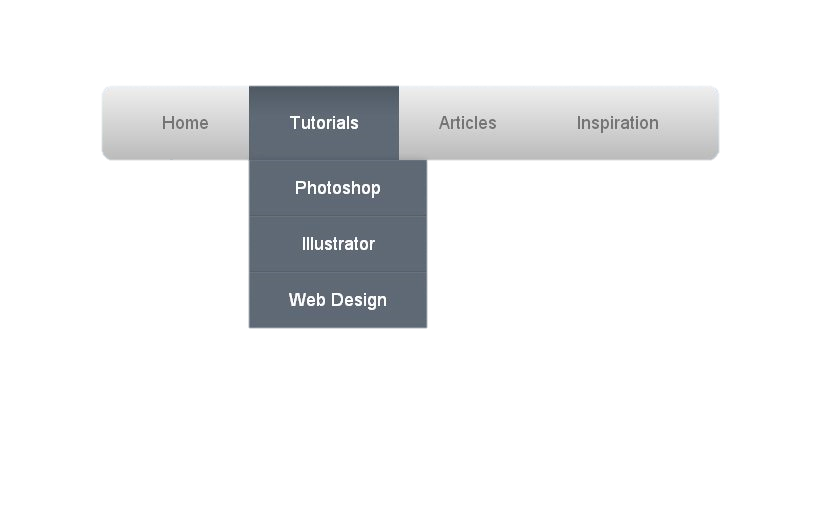


Рисунок 2 – Панель вкладок

Переключатели — это элемент управления, позволяющий пользователю включить или выключить отдельные опции (рисунок 3). Как и кнопки, переключатели эффективно применяются в современных интерфейсах, поскольку представляют собой имитацию реальных переключателей. Важным моментом является то, что состояния переключателя должны быть явными и легко различимыми для пользователя, чтобы он мог легко понять, активен параметр или нет.

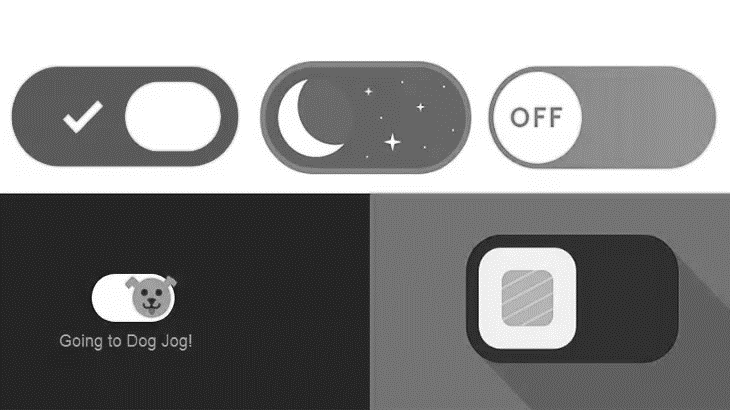


Рисунок 3 - Переключатели

Пикер (Picker) позволяет выбрать одну точку из ряда опций. Обычно он может включать один или несколько прокручиваемых списков различных значений, таких как часы, минуты, даты, измерения, валюты и т. д. Пользователи, прокручивая список, могут выбрать и установить необходимое значение. Данный тип элементов широко применяется в интерфейсах, имеющих функциональность установки времени и дат (рисунок 4).

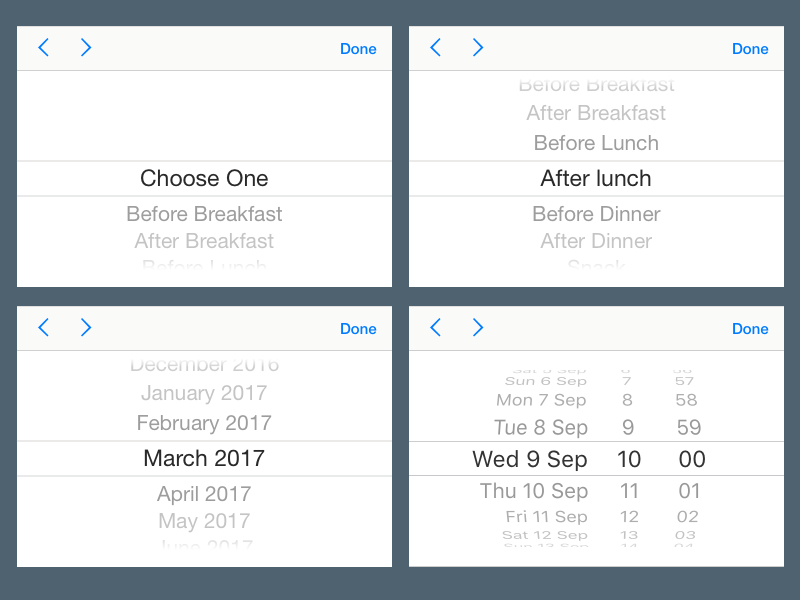


Рисунок 4 – Date picker

Чекбокс — это графический элемент пользовательского интерфейса, который позволяет отмечать определенный фрагмент контента, обычно устанавливающий выбор для двоичных опций. Чекбоксы и переключатели применяются в пользовательских интерфейсах любого типа, особенно часто в настройках. Кроме того, чекбоксы могут представлять собой общую часть навигации в приложениях и веб-сайтах с функциональностью менеджеров задач, списков дел, таймеров.

Поле или панель поиска представляет собой элемент интерфейса, позволяющий пользователям вводить ключевые слова и находить необходимые фрагменты контента. Поле поиска является одним из основных элементов навигации для веб-сайтов или приложений с большим количеством контента: в блогах, в электронной коммерции и новостных веб-сайтах. Поле поиска дает пользователю возможность перейти к нужной точке, не просматривая многочисленные страницы и меню. Такой подход востребован при разработке удобных интерфейсов, поскольку учитывает время и усилия пользователя.

Тег — это интерактивный элемент, состоящий из ключевого слова или фразы, позволяющий пользователям быстро перемещаться по элементам, отмеченным текстом. Теги обеспечивают быстрый доступ к определенным категориям контента и позволяют осуществить навигацию с дополнительным способом классификации контента. Взаимодействие с тегом перемещает пользователя на веб-страницу, которая содержит все материалы, отмеченные этим тегом.

Фильтры — это графические элементы управления, которые позволяют выставить необходимые настройки. Они являются эффективными в аспекте персонализации для лучшего UX: пользователи могут выбрать и настроить необходимые им параметры (рисунок 5).

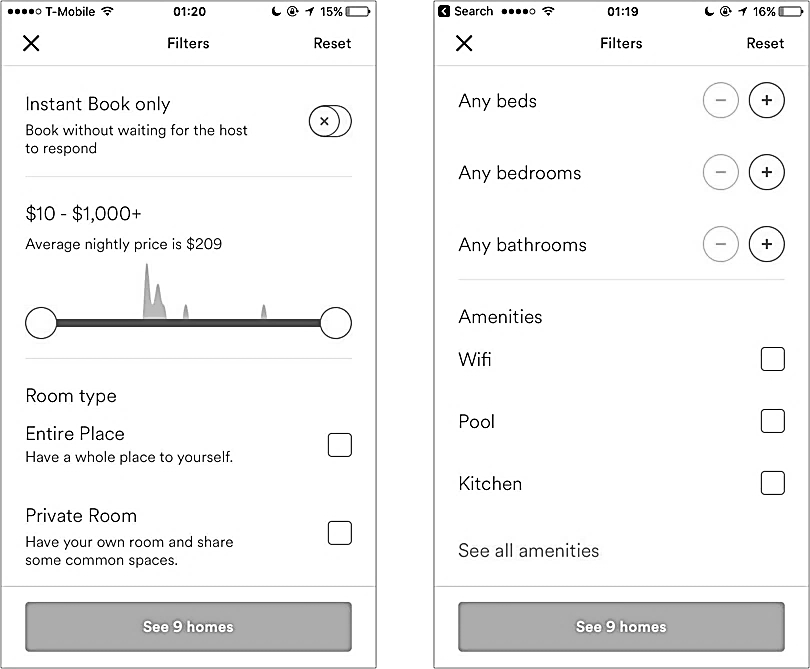


Рисунок 5 – Фильтр

Планирование интерфейсов требует хороших знаний психологии и моделей взаимодействия, тестирования пользователей и анализа с начальных этапов разработки приложения или веб-сайта. Интерактивные элементы пользовательского интерфейса позволяют реализовать взаимодействие, необходимое для решения проблем пользователей и побудить их возвращаться к продукту.

Наиболее распространенный вариант добавления интерактивности на сайт — использование JavaScript. JavaScript — это полноценный динамический язык программирования, который применяется к HTML документу, и может обеспечить динамическую интерактивность на веб-сайтах. Во втором разделе работы представлено исследование разработки интерактивных элементов на JavaScript.

1. Интерактивность на JavaScript

JavaScript — это кроссплатформенный объектно-ориентированный скриптовый язык программирования, который используется для реализации интерактивных элементов на странице [6]. Поддержка JavaScript есть во всех современных браузерах, но иногда она может быть по умолчанию неактивна. В таких случаях браузер уведомляет, что для полноценного взаимодействия со страницей сайта необходимо включить поддержку скриптов.

В современной веб-разработке язык JavaScript используется для следующих целей:

* организация взаимодействия с пользователями на веб-странице;
* реализация логики поведения элементов веб-страницы;
* добавление различных эффектов, в особенности анимации и графических, а также тех, что требуют математических вычислений;
* обработка данных в HTML-формах;
* взаимодействие с различными HTML-элементами и CSS-стилями [7].

1. DOM

Важной частью создания интерактивных веб-сайтов является объектная модель документа, обычно называемая DOM. Объектная Модель Документа (DOM) – это программный интерфейс (API) для HTML и XML документов. DOM предоставляет структурированное представление документа и определяет то, как эта структура может быть доступна из программ, которые могут изменять содержимое, стиль и структуру документа. Представление DOM состоит из структурированной группы узлов и объектов, которые имеют свойства и методы. По существу, DOM соединяет веб-страницу с языками описания сценариев либо языками программирования [8].

Это интерфейс, который позволяет языку программирования управлять содержимым, структурой и стилями веб-сайта. Когда веб-сайт выполняет какое-либо действие, например, переключение слайд-шоу изображений, отображение ошибки, когда пользователь пытается отправить неполную форму, или переключение навигационного меню, это является результатом доступа JavaScript к DOM и манипулирования им.

1. События

Реакция веб-сайтов на действия пользователей реализуется с помощью событий. События — это действия, которые происходят в браузере и могут быть инициированы пользователем или самим браузером, например: произошла загрузка страницы, пользователь нажимает кнопку, наводит курсор на выпадающий список, отправляет форму, нажимает клавишу на клавиатуре.

События не являются частью основного JavaScript — они определены в веб-интерфейсах браузера (Web API). Написав код JavaScript, который выполняется в ответ на событие, разработчики могут реализовать отображение сообщения для пользователей, проверку данных, реакцию на нажатие кнопки и выполнение других действий [9].

Когда пользователь нажимает кнопку на сайте или нажимает клавишу на клавиатуре происходит событие. Это называется событие нажатия (click event) или событие нажатия клавиши (keypress event), соответственно.

Обработчик события (event handler) является функцией JavaScript, которая запускается при возникновении события.

Прослушиватель событий (event listener) навешивает интерфейс к элементу, который позволяет этому элементу ждать и «прослушивать» указанное событие [10].

Есть три способа назначить события элементам:

* встроенные обработчики событий;
* свойства обработчика событий;
* слушатели событий.

Ниже на рисунке 6 представлен вариант использования встроенного обработчика событий. В примере при нажатии на элемент button изменяется текстовое содержимое элемента p.

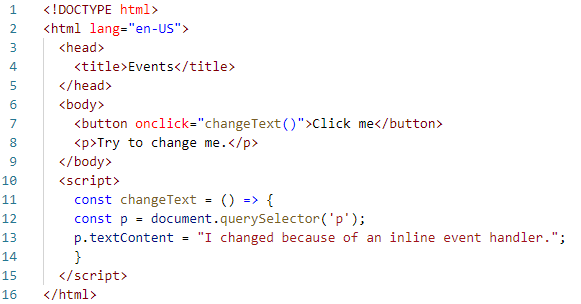


Рисунок 6 – Встроенный обработчик событий

Встроенные обработчики событий не используются вне тестирования, более целесообразно поддерживать отдельный файл сценария JavaScript, чем добавлять обработчики для каждого элемента.

Свойства обработчика событий работают очень похоже на встроенный обработчик, событие onclick элемента в данном случае устанавливается в JavaScript, вместо атрибута в HTML (рисунок 7).



Рисунок 7 – Свойства обработчика событий

Свойство обработчика событий также имеет недостатки, установка нескольких отдельных onclick свойств приведет к перезаписи всех, кроме последнего, поэтому наиболее предпочтительным является способ с применением слушателей событий.

Для прослушивания событий на элементе используется метод addEventListener(). Он принимает два обязательных параметра - событие, которое он должен прослушивать и функцию обратного вызова слушателя. HTML-код для такого прослушивателя событий будет таким же, как в предыдущем примере. В JavaScript вместо события onclick к кнопке прикрепляется метод addEventListener(), как показано на рисунке 8.



Рисунок 8 – Слушатель событий addEventListener()

Для одного элемента может быть установлено несколько слушателей событий. Слушатели событий являются наиболее распространенным и предпочтительным способом обработки событий в JavaScript.

В JavaScript существует также множество других событий. Наиболее распространенные из них: события мыши, события формы и события клавиатуры.

События мыши являются одними из наиболее часто используемых событий. К ним относятся нажатие кнопок мыши или наведение и перемещение указателя мыши. Эти события также соответствуют эквивалентным действиям на сенсорном устройстве. События мыши представлены в таблице 1.

Таблица 1 – События мыши

|  |  |
| --- | --- |
| Событие | Описание |
| click | Запускается при нажатии и отпускании мыши на элементе |
| dblclick | Запускается при двойном нажатии на элемент |
| mouseenter | Запускается, когда указатель входит в элемент |
| mouseleave | Запускается, когда указатель покидает элемент |
| mousemove | Срабатывает, когда указатель перемещается внутри элемента |

Событие click является составным событием, которое состоит из комбинированных mousedown и mouseup событий, которые запускаются, когда кнопка мыши нажимается или отпускается, соответственно.

Использование mouseenter и mouseleave позволяет создать эффект наведения, который сохраняется до тех пор, пока указатель мыши находится на элементе.

События формы — это действия, позволяющие осуществить проверку и отправку формы. События формы представлены в таблице 2.

Таблица 2 – События формы

|  |  |
| --- | --- |
| Событие | Описание |
| submit | Запускается при отправке формы |
| focus | Запускается, когда элемент получает фокус |
| blur | Срабатывает, когда элемент теряет фокус |

Событие focus достигается при выборе элемента, например, с помощью щелчка мыши или перехода к нему с помощью клавиши TAB.

Преимущество использования JavaScript для отправки форм заключается в том, что отправка не требует перезагрузки страницы, а также может быть осуществлена проверка необходимых полей ввода.

События клавиатуры используются для обработки действий пользователя с клавиатуры, таких как нажатие, удержание и отпускание клавиши. События клавиатуры представлены в таблице 3.

Таблица 3 – События клавиатуры

|  |  |
| --- | --- |
| Событие | Описание |
| keydown | Срабатывает один раз при нажатии клавиши |
| keyup | Запускается один раз при отпускании ключа |
| keypress | Срабатывает непрерывно, пока нажата клавиша |

1. AJAX

AJAX (асинхронный JavaScript и XML) — это метод передачи HTTP-данных в веб-браузере без обновления страницы. AJAX применяется для форм и кнопок, связанных с элементарными действиями: добавить в корзину, подписаться и др. AJAX позволяет реализовать динамическую подгрузку данных. Например, дерево, которое при раскрытии узла запрашивает данные у сервера [11]. Одним из классических примеров использования AJAX является живой поиск, когда пользователь начинает вводить поисковую фразу, а JavaScript предлагает возможные варианты, получая список самых вероятных дополнений с сервера.

Технически, с помощью AJAX можно обмениваться любыми данными с сервером. Обычно используются форматы: JSON – для отправки и получения структурированных данных и объектов, XML, HTML/текст, бинарные данные, файлы.

JavaScript и XML объединяются, чтобы обеспечить асинхронное обновление посредством использования объекта XMLHttpRequest. Когда пользователь посещает веб-страницу, использующую AJAX, и происходит предписанное событие (пользователь загружает страницу, нажимает кнопку, заполняет форму) JavaScript создает объект XMLHttpRequest, который затем передает данные в формате XML между веб-браузером и веб-сервером (программное или аппаратное обеспечение, на котором хранятся данные веб-сайта). Объект XMLHttpRequest отправляет запрос на обновленные данных страницы на веб-сервер, сервер обрабатывает запрос, ответ создается на стороне сервера и отправляется обратно в браузер, который затем использует JavaScript для обработки ответа и отображения на экране обновленного содержимого. Метод AJAX игнорирует посторонние данные страницы и обрабатывает только запросы на обновление информации и саму обновленную информацию. AJAX позволяет делать веб-сайты и приложения более быстрыми и отзывчивыми для пользователей

1. Императивный и декларативный подходы

Существует два подхода к тому, как JavaScript взаимодействует с DOM: императивный и декларативный. Императивное программирование — это парадигма программирования, которая описывает процесс вычисления в виде инструкций, изменяющих состояние данных. В декларативном программировании задается спецификация решения задачи, то есть описывается, что представляет собой проблема и ожидаемый результат. Декларативные программы не используют состояния, то есть не содержат переменных и операторов присваивания [12].

Существует множество библиотек для манипуляций с DOM, которые предоставляют API-интерфейсы, облегчающие написание кода. Одна из самых популярных библиотек — jQuery, является императивной библиотекой. Сейчас для управления сложными интерфейсами чаще используют декларативные фреймворки и библиотеки наподобие Angular и React.

1. Паттерны проектирования

Паттерны проектирования важны для написания поддерживаемого и многократно используемого кода. Паттерн — это повторно используемое решение, которое может быть применено к часто возникающим проблемам в разработке программного обеспечения [13].

MVC предлагает архитектурные преимущества по сравнению со стандартным JavaScript - он позволяет писать более организованный и понятный код [14]. MVC (представлена на рисунке 9) состоит из трех компонентов:

* модель — это место, где хранятся объекты данных приложения. Модель ничего не знает о представлениях и контроллерах. Когда модель изменяется, обычно она уведомляет своих наблюдателей о том, что произошло изменение;
* представление — это то, что представлено пользователям и как пользователи взаимодействуют с приложением. Представление сделано с HTML, CSS, JavaScript и часто шаблонами и имеет доступ к DOM;
* контроллер является связующим звеном между моделью и представлением, он также добавляет прослушиватели событий в представление и обновляет модель, когда пользователь управляет представлением.

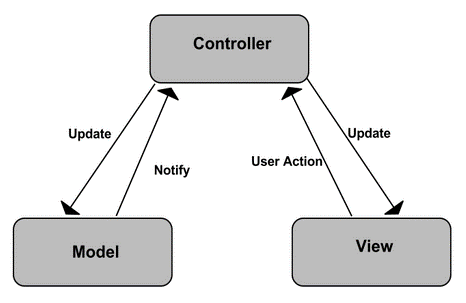


Рисунок 9 – Шаблон MVC

Существует несколько вариантов шаблона проектирования MVC, таких как MVP (Model-View-Presenter) и MVVP (Model-View-ViewModel).

В шаблоне MVP (рисунок 10) элемент Presenter берёт на себя функциональность посредника (аналогично контроллеру в MVC) и отвечает за управление событиями пользовательского интерфейса (например, использование мыши) так же, как в других шаблонах обычно отвечает представление. MVP используется чаще всего в приложениях уровня предприятия, где необходимо повторно использовать как можно больше логики представления. Приложения с очень сложными представлениями и большим количеством взаимодействия с пользователем могут обнаружить, что MVC здесь не совсем соответствует требованиям, так как решение этой проблемы может означать большую зависимость от нескольких контроллеров. В MVP вся эта сложная логика может быть заключена в презентатор, что может значительно упростить обслуживание.

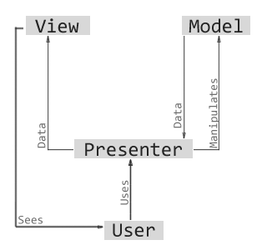


Рисунок 10 – Шаблон MVP

MVVM удобно использовать вместо классического MVC в тех случаях, когда в платформе, на которой ведётся разработка, есть «связывание данных». В шаблонах проектирования MVC/MVP изменения в пользовательском интерфейсе не влияют непосредственно на Модель, а предварительно идут через Контроллер или Presenter.

Шаблон MVVM (рисунок 11) делится на три части:

* модель - так же, как в классической MVC, представляет собой логику работы с данными и описание фундаментальных данных, необходимых для работы приложения;
* представление — графический интерфейс (окна, списки, кнопки). Выступает подписчиком на событие изменения значений свойств или команд, предоставляемых Моделью Представления. В случае, если пользователь воздействует на какой-либо элемент интерфейса, Представление вызывает соответствующую команду, предоставленную Моделью Представления;
* модель представления — с одной стороны, абстракция Представления, а с другой — обёртка данных из Модели, подлежащая связыванию. То есть, она содержит Модель, преобразованную к Представлению, а также команды, которыми может пользоваться Представление, чтобы влиять на Модель.

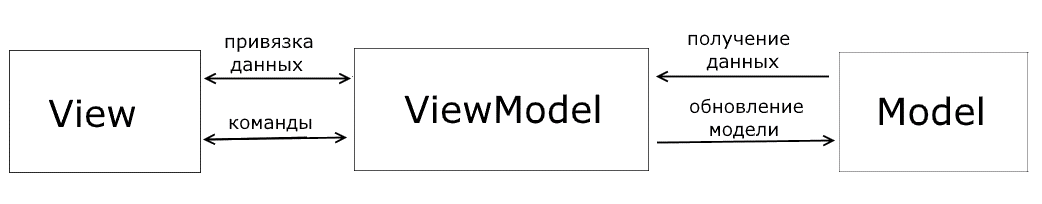


Рисунок 11 – Шаблон MVVM

Большинство JavaScript-фреймворков основано на шаблонах проектирования. AngularJS реализует подход MVC и MVVM, Ember.js основан на шаблоне MVVM, Backbone.js на шаблоне MVP.

1. Фреймворки и библиотеки

JavaScript-фреймворки предоставляют разработчикам набор решений для таких сложных задач, как маршрутизация, оптимизация производительности, управление состояниями. Обычно они используются для создания веб-приложений [15].

AngularJS появился на рынке в октябре 2010 года. Он быстро стал самым популярным JavaScript MVC-фреймворком. Он имеет двусторонний дата-биндинг между моделями и представлениями. Эта привязка данных позволяет проводить автоматическое обновление с обеих сторон всякий раз, когда происходит изменение данных, что позволяет создавать многократно используемые компоненты представления и обеспечивает легкий обмен данными между серверной и клиентской частью.

React — созданная компанией Facebook библиотека с открытым исходным кодом, предназначенная для сборки пользовательских интерфейсов. В паттерне MVC React ближе всего к пользователю. Он отвечает за представление данных, получение и обработку ввода пользователя. React построен на парадигме реактивного программирования, этот декларативный подход предлагает описывать данные в виде набора утверждений или формул. Изменение одного из параметров ведёт за собой автоматический пересчёт всех зависимостей.

Библиотека Vue.js с каждым днем набирает все больше и больше популярность. Технически, Vue.js определена как ViewModel шаблона MVVM. Она соединяет модель и представление в двустороннее связывание данных.

Backbone основан на шаблоне проектирования MVP. Модели позволяют связать ключ-значение и события для обработки изменения данных. Поддерживает RESTful JSON интерфейс. Backbone лучше всего использовать при построении простых одностраничных приложений.

Ember тоже относится к MVC-фреймворкам. В нём используется шаблонизация и встроенный view engine, который автоматически обновляет данные так же, как и Angular, Backbone и React. Ember поддерживает технологию веб-компонентов, позволяющую расширять HTML с помощью собственных тэгов (как и Angular). Также во фреймворк встроен движок маршрутизации и модели, умеющие работать RESTful API.

Фреймворки, такие, как Angular, React и Ember, на сегодняшний день, являются ключевыми для разработки сложных веб-приложений. Они помогают структурировать код, управлять состоянием приложений и строить сложные интерфейсы, основанные на переиспользуемых компонентах.

# Заключение

В процессе выполнения работы были исследованы технологии, позволяющие разрабатывать интерактивные элементы веб-сайтов на языке JavaScript:

* было определено понятие интерактивности и проанализированы наиболее популярные интерактивные элементы.
* рассмотрены основные аспекты добавления интерактивности на сайт, описана объектная модель документа (DOM) и события, позволяющие создавать интерфейсы, откликающиеся на действия пользователей.
* исследована технология AJAX, которая дает возможность не перезагружать страницу полностью каждый раз, когда произошло изменение какой-либо ее части, что позволяет снизить нагрузку на сервер и обеспечить для пользователя быстрый ответ на взаимодействие.
* также рассмотрены императивный, декларативный подходы и паттерны проектирования, на которых основаны JavaScript-фреймворки и библиотеки, облегчающие разработку интерактивных элементов при повышении сложности проектов.

# Список использованных источников

1. Уровень интерактивности веб-сайта влияет на имидж организации. [Электронный ресурс]. – Электрон. дан. – Режим доступа: https://gtmarket.ru/news/media-advertising-marketing/2008/05/28/1676 (дата обращения: 3.05.2019). – Загл. с экрана.
2. Сизоненко, А.Д. История развития веб-технологий [Электронный ресурс] / А.Д. Сизоненко // Молодежный научный форум: электр. сб. ст. по мат. XVIII междунар. студ. науч.-практ. конф. – 2018. – № 17(18). – с. 37-42 – Электрон. текстовые дан. – Режим доступа: https://nauchforum.ru/archive/ MNF\_interdisciplinarity/17(18).pdf (дата обращения: 30.04.2019). – Загл. с экрана.
3. Front-end Frameworks [Электронный ресурс] – Электрон. дан. – Режим доступа: https://2017.stateofjs.com/2017/front-end/results/ (дата обращения: 5.05.2019). – Загл. с экрана.
4. Интерактивные Веб-документы [Электронный ресурс]. – Электрон. дан. – Режим доступа: http://www.вебnav.ru/books/html4/interactive\_documents/ (дата обращения: 30.04.2019). – Загл. с экрана.
5. Дакетт Д. НТМL и CSS. Разработка и дизайн веб-сайтов [Текст] / Джон Дакетт; [пер. с англ. М. А. Райтмана). – М.: Эксмо, 2013. – 480 с.
6. Зеленко, О. В. Обзор современных веб – технологий [Электронный ресурс] / О. В. Зеленко, Л.Р. Валеева // Вестник технологического университета – 2015. – № 2. – с. 354-356. – Электрон. текстовые дан. – Режим доступа: https://cyberleninka.ru/article/v/obzor-sovremennyh-веб-tehnologiy (дата обращения: 2.05.2019). – Загл. с экрана.
7. Дакетт, Д. Javascript и jQuery. Интерактивная веб-разработка [Текст] / Джон Дакетт; [пер. с англ. М. А. Райтмана]. – Москва: Издательство «Э», 2017. – 640 с.
8. DOM и JavaScript [Электронный ресурс] – Электрон. дан. – Режим доступа: https://developer.mozilla.org/ru/docs/DOM/DOM\_Reference/Введение #DOM\_и\_JavaScript (дата обращения: 16.05.2019). – Загл. с экрана.
9. Маккоу А. Веб-приложения на JavaScript [Электронный ресурс] / А. Маккоу // СПб.: Питер, 2012. – 288 с. – Режим доступа: https://artkiev.com/blog/wp-content/uploads/2012/10/web-programming-JavaScrip t -2012.pdf (дата обращения: 19.05.2019). – Загл. с экрана.
10. Введение в события [Электронный ресурс] – Электрон. дан. – Режим доступа: https://developer.mozilla.org/ru/docs/Learn/JavaScript/Building\_blocks /События (дата обращения: 16.05.2019). – Загл. с экрана.
11. Введение в AJAX и COMET [Электронный ресурс] – Электрон. дан. – Режим доступа: https://learn.javascript.ru/ajax-intro (дата обращения: 16.05.2019). – Загл. с экрана.
12. WebDev → Императивное vs Функциональное программирование [Электронный ресурс] – Электрон. дан. – Режим доступа: http://sauron.org.ua/post/1288 (дата обращения: 16.05.2019). – Загл. с экрана.
13. Обзор паттернов проектирования [Электронный ресурс] – Электрон. дан. – Режим доступа: http://citforum.ru/SE/project/pattern/ (дата обращения: 19.05.2019). – Загл. с экрана.
14. MVC, MVP and MVVM Design Pattern [Электронный ресурс] – Электрон. дан. – Режим доступа: https://medium.com/@ankit.sinhal/mvc-mvp-and-mvvm-design-pattern-6e169567bbad (дата обращения: 19.05.2019). – Загл. с экрана.
15. Анализ шести веб-фреймворков: плюсы, минусы и особенности выбора [Электронный ресурс] – Электрон. дан. – Режим доступа: https://habr.com/company/ruvds/blog/343894/ (дата обращения: 16.05.2019). – Загл. с экрана.

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,   
МЕХАНИКИ И ОПТИКИ**

**ОТЗЫВ РУКОВОДИТЕЛЯ**

**о выполнении курсового проекта (работы)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Студент | | Елисеева Екатерина Сергеевна | | | | | | |
| (Фамилия, И., О.) | | | | | | | | |
| Факультет | ПИиКТ | | | | | | Группа | P41622 |
| Направление (специальность) | | | | | | 09.04.02 «Информационные системы и технологии» | | |
| Руководитель | | | | Государев И.Б., Университет ИТМО, доцент, к.пед.н. | | | | |
| (Фамилия, И., О., место работы, должность, ученое звание, степень) | | | | | | | | |
|  | | | | | | | | |
| Дисциплина | | | Проектирование и анализ языков веб-решений | | | | | |
|  | | | | | | | | |
| Наименование темы: | | | | | Исследование интерактивности на основе JavaScript | | | |
| |  | | --- | |  | | | | | | | | | |
|  | | | | | | | | |

**ОЦЕНКА КУРСОВОГО ПРОЕКТА (РАБОТЫ)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Показатели** | **Оценка** | | | |
| **п/п** | **5** | **4** | **3** | **0** |
| 1. | Проект создан обучающимся самостоятельно |  |  |  |  |
| 2. | Предложенные выводы обоснованы |  |  |  |  |
| 3. | Проект технологически грамотный |  |  |  |  |
| 4. | Оформление отвечает требованиям к отчету, отчёт размещён в веб-портфолио |  |  |  |  |
| 5. | Во время защиты обучающийся показал умение кратко, доступно представить результаты работы, умение анализировать, аргументировать свою точку зрения, делать обобщение и выводы, адекватно ответить на поставленные вопросы. |  |  |  |  |
| **ИТОГОВАЯ ОЦЕНКА** | |  | | | |

**Отмеченные достоинства:**

|  |
| --- |
| В отчете обучающегося отражены полученные в ходе выполнения проекта навыки, |
| соответствующие компетенциям по данной тематике. В процессе работы обучающийся |
| подтвердил владение умениями исследования и использования веб-языков |
|  |

**Отмеченные недостатки:**

|  |
| --- |
| Недостаточно полно отражены технологии |
|  |
|  |
|  |
|  |

**Заключение:**

|  |
| --- |
| Обучающийся подтвердил владение компетенциями, предусмотренными учебным |
| планом. |
|  |

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Государев И.Б.   
 (подпись)

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019