ОАНО ДПО «СКАЕНГ»

Утверждаю

«__» _____ 20__ г.

Соловьев Г.М.

Дополнительная общеобразовательная (общеразвивающая) программа

«JavaScript. Архитектура клиентских приложений»

для взрослых учащихся

Составитель:

ФИО, должность, ученая степень (если есть)

Москва

2021 г.

Пояснительная записка

Дополнительная общеобразовательная (общеразвивающая) программа) «JavaScript. Архитектура клиентских приложений» предназначена для совершенствования профессиональных компетенций широкого круга лиц, владеющих азами программирования на JavaScript и желающих освоить создание сложных приложений и направлена на создание и обеспечение необходимых условий для личностного развития обучающихся и удовлетворения их образовательных потребностей и интересов.

В основе программы лежат следующие **федеральные нормативные акты**:

- Федеральный закон от 29 декабря 2012 г. №273-ФЗ «Об образовании в Российской Федерации»;
- Закон Российской Федерации от 07 февраля 1992 г. № 2300-1 «О защите прав потребителей»;
- Постановление Правительства РФ от 15.04.2014 N295 (ред. от 31.03.2017)
- Порядок организации и осуществления образовательной деятельности по дополнительным профессиональным программам. Приказ МОиН РФ от 01.07.2013 №499;99;
- Порядок организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам. Приказ Министерства просвещения РФ от 9 ноября 2018 г. N 196
- Показатели, характеризующие общие критерии оценки качества условий осуществления образовательной деятельности организациями, осуществляющими образовательную деятельность по дополнительным профессиональным программам. Приказ Министерства науки и высшего образования РФ от 15 апреля 2019 г. N 31н

Актуальность программы определяется содержанием обучения, отвечающим образовательным потребностям обучающихся, желающих начать работу в качестве react-разработчика и создавать жизнеспособные клиентские интерфейсы.

Новизна программы и ее отличительная особенность заключается в использовании наставничества для практического освоения лекционного материала. **Педагогическая целесообразность программы** обусловлена удовлетворением персональных образовательных потребностей обучающихся.

Цель и задачи программы

Цель: овладеть навыками создания клиентских приложений с помощью JavaScript.

Задачи курса:

- научиться собирать модули ECMAScript;
- научиться применять объектно-ориентированный подход к программированию;
- научиться писать полноценные приложения без сервера;
- научиться создавать объекты в JavaScript с помощью классов;
- научиться работать с сетью.

Форма обучения – заочная (дистанционная).

Программа рассчитана на **129 часов** и состоит из 10 модулей.

Характер занятий: занятия включают теоретические и практические.

Режим занятий по календарному плану 2 раза в неделю.

Планируемые результаты:

В результате освоения программы обучающиеся научатся:

- собирать модули ECMAScript;
- применять объектно-ориентированный подход к программированию;
- писать полноценные приложения без сервера;
- создавать объекты в JavaScript с помощью классов;
- работать с сетью.

Контроль в каждом модуле осуществляется в виде: текущего контроля (наставник анализирует выполненную обучающимся работу) и итогового контроля – итогового проекта.

Защита итогового проекта состоит из четырех этапов:

1. Начало подготовки к итоговой защите;

2. Первая оценка проекта проверяющим наставником по критериям качества;

3. Улучшение проекта по замечаниям проверяющего наставника и подача на вторую оценку;

4. Внесение финальных правок и получение итоговой оценки.

Материально-техническое обеспечение программы:

1) Персональный компьютер (стационарный компьютер/ноутбук).
2) Микрофона и наушник.
3) Установленная на персональный компьютер программа Skype.
4) Установленный на персональный компьютер браузер Google Chrome.
5) Необходимая скорость интернет-соединения - от 5 МБит/с, ping не выше 50.

Учебно-тематический план:

Модуль	Содержание	Количество часов	
		Аудиторных	Самостоятельная работа
Знакомство с JavaScript	Структура курса и проектов- Язык JavaScript: Спецификация ECMAScript. - Что даёт JavaScript'у браузер. - Строгий режим 'use strict'. - Необязательные точки с запятой. Основы JavaScript: - Базовый синтаксис: круглые и фигурные скобки, операторы, зарезервированные слова. - Переменные. - Функции. - Типы данных: примитивы. - Приведение типов. Практика: - Создание репозитория, форк, клонирование. - Создание ветки, коммита, синхронизация репозиториев. - Создание пулреквеста в Гитхабе. - Тренажеры.	2	8

Основные возможности JavaScript	Условные операторы: - Тернарный оператор. - Оператор множественного выбора switch. Циклы Функции - Способы объявления: Function Declaration и Function Expression. - Стрелочные функции. - Параметры функций по умолчанию. Встроенные в JavaScript функции, Контекст функций, Знакомство с DevTools (инструментами разработчика), Практика:- Работа с техзаданием: превращение требований в код. - Написание утилитарных функций.	2	8
Структуры данных и встроенные API	Сложные типы данных, Объекты: - Создание объектов через new. - Встроенный объект Object. Массивы: - Встроенный объект Array. - Методы массивов forEach, map, reduce, filter, sort. Встроенные объекты и их методы: - String. - Number. - Boolean. - Date.	2	8

	- Math. Деструктуризация DevTools. Использование Console (консоли Практика: Описание структуры данных проекта.		
Организация кода	Области видимости: - Области видимости. - Глобальная область видимости. - Замыкания. - Потеря окружения. Подвешивание (hoisting) переменных и функций Модульность: - Повторное использование кода, принцип DRY. - Понятие модуля. - Модули ECMAScript, import и export. - Циклические зависимости. - Инкапсуляция. DevTools: отладка кода с помощью Sources (исходников). Практика: - Разделение кода на модули. - Соблюдение принципа DRY и выделение повторяющихся частей кода в функции. - Перенос функций, повторяющихся в разных файлах в отдельные модули.	2	8
DOM и события	Управление DOM-деревом: - DOM-дерево: структура. - Поиск элементов на странице. - Управление атрибутами DOM-элементов. - Перемещение элементов в DOM-дереве.	2	8

	Подходы к созданию DOM-элементов: - Управление разметкой: append, prepend, insertAdjacentHTML, innerHTML, textContent. - Создание DOM-объектов. Шаблонизация: - Строковая шаблонизация (шаблонные строки). - Специальный тег <template>. События: - Обработчики событий. - Объект Event, управление событиями. - Фазы событий и делегирование. - Клавиатурные события и доступность. DevTools: возможности Elements (инспектора) для работы с DOM и событиями. Практика: - Генерация DOM-элементов по шаблону и наполнение их данными. - Обработка пользовательской реакции. - Работа с доступностью.		
Внешние API и сторонние библиотеки	Валидация форм, Валидация с помощью регулярных выражений, Обзор API браузера, Картографические сервисы и их JavaScript API: - OpenStreetMap. - Leaflet. Сторонние библиотеки: - Зачем нужны библиотеки. - Как подключить в проект. Практика: - Валидация форм. - Использование в проекте	2	8

	API карт. - Подключение в проект сторонних библиотек.		
Асинхронность. Работа с сетью	Асинхронность, Колбэки: - setTimeout., Event Loop, Promise, Протокол HTTP и форматы данных: - JSON. - fetch для обращения к серверу. - Обработка ошибок в запросах. DevTools: работа с сетевыми запросами в Network (сети) Практика: - Загрузка данных для шаблонов из интернета. - Добавление реакции на ошибки загрузки.	2	8
Обратная связь и оптимизация	Продвинутая работа с массивами: - Сортировка. - Фильтрация. Оптимизации производительности: - Пропуск кадров — тротлинг (throttle). - Устранение дребезга — дебаунс (debounce) FileReader DevTools: обзор отладчиков Perfomance (производительности) Практика: - Добавление поисковых фильтров на страницу. - Создание функции «устранения дребезга». - Реализация предпросмотра выбранного изображения в форме.	2	8
Сборщики JavaScript	Что такое сборщик, Задачи сборщика, Обзор	2	8

	популярных сборщиков: - Webpack - Rollup. - Parcel. Сервер для разработки: - Browsersync. - Webpack DevServer. Транспайлеры и полифилы: Babel Минификация кода, Легаси: - jQuery. DevTools: отладка кода по source maps (картам исходников). Практика: - Настройка Webpack в проекте.		
Итоговый проект		1	46
Итого		19	110
		129	

Защита проекта состоит из четырех этапов:

1. Начало подготовки к итоговой защите;

2. Первая оценка проекта проверяющим наставником по критериям качества;

3. Улучшение проекта по замечаниям проверяющего наставника и подача на вторую оценку;

4. Внесение финальных правок и получение итоговой оценки.

Кадровое обеспечение образовательного процесса

Квалификация педагогических работников Организации соответствует квалификационным характеристикам, установленным в Едином квалификационном справочнике должностей руководителей, специалистов и служащих, разделе «Квалификационные характеристики должностей руководителей и специалистов высшего профессионального и дополнительного профессионального образования», утвержденного приказом Министерства здравоохранения и социального развития РФ от 11 января 2011 г. N 1н