**Паспорт Образовательной программы**

**«Браузерные интерфейсы прикладного программирования (API)»**

|  |  |
| --- | --- |
| **Версия программы** | **1** |
| **Дата Версии** | **14.10.2020** |

1. **Сведения о Провайдере**

|  |  |  |
| --- | --- | --- |
| 1.1 | Провайдер | **АНО ДПО "СофтЛайн Эдюкейшн"** |
| 1.2 | Логотип образовательной организации |  |
| 1.3 | Провайдер ИНН | **7736228783** |
| 1.4 | Ответственный за программу ФИО | **Максименко Денис Владиславович** |
| 1.5 | Ответственный должность | Руководитель проектного офиса |
| 1.6 | Ответственный Телефон | +7-495-232-0023 доб. 0889 |
| 1.7 | Ответственный Е-mail | Edu2035@softline.com |

1. **Основные Данные**

|  |  |  |
| --- | --- | --- |
| **№** | **Название** | **Описание** |
| 2.1 | Название программы | Браузерные интерфейсы прикладного программирования (API) |
| 2.2 | Ссылка на страницу программы | <https://go.teachbase.ru/course_sessions/brauzernye-interfeysy-prikladnogo-programmirovaniya-api-11-01/apply> - страница программы  <https://go.teachbase.ru/course_sessions/vhodnoe-testirovanie-kurs-brauzernye-interfeysy-prikladnogo-programmirovaniya-api-11-01/apply> - входное тестирование |
| 2.3 | Формат обучения | Электронное обучение |
|  | Подтверждение от ОО наличия возможности реализации образовательной программы с применением электронного обучения и (или) дистанционных образовательных технологий с возможностью передачи данных в форме элементов цифрового следа | Подтверждаем возможность реализации образовательной программы с применением электронного обучения с возможностью передачи данных в форме элементов цифрового следа |
| 2.4 | Уровень сложности | Базовый |
| 2.5 | Количество академических часов | 72 академических часа |
|  | Практико-ориентированный характер образовательной программы: не менее 50 % трудоёмкости учебной деятельности отведено практическим занятиям и (или) выполнению практических заданий в режиме самостоятельной работы | 57 ак. часов отведено практическим занятиям и выполнению практических заданий в режиме самостоятельной работы, что составляет около 79,2 % трудоёмкости учебной деятельности |
| 2.6 | Стоимость обучения одного обучающегося по образовательной программе, а также предоставление ссылок на 3 (три) аналогичные образовательные программы иных организаций, осуществляющих обучение, для оценки объективности стоимости или обоснование уникальности представленной образовательной программы в случае отсутствия аналогичных образовательных программ на рынке образовательных услуг | 29900   1. <https://digital-academy.ru/blog/java-script-kursy>   стоимость: 66 000   1. <https://gekkon-club.ru/programming/full-stack>   Стоимость: 40 000   1. <https://career.habr.com/courses/341>   стоимость: 149 090 |
| 2.7 | Минимальное количество человек на курсе | 1 |
| 2.8 | Максимальное количество человек на курсе | 1000 |
| 2.9 | Данные о количестве слушателей, ранее успешно прошедших обучение по образовательной программе | отсутствует |
| 2.10 | Формы аттестации | **тестирование** |
|  | Указание на область реализации компетенций цифровой экономики, к которой в большей степени относится образовательная программа, в соответствии с Перечнем областей | Программирование и создание ИТ-продуктов |

1. **Аннотация программы**

Браузерные приложения, как и мобильные, позволяют решать задачи пользователей в широком диапазоне от создания списка задач до редактирования изображений и других медиаресурсов онлайн. Они управляются языком JavaScript и используют программные интерфейсы – API. С помощью API осуществляется управление устройствами Интернета вещей, агентами искусственного интеллекта и инструментами анализа больших данных. Чтобы более свободно ориентироваться в цифровой среде и автоматизировать информационные процессы с целью извлечения прибыли и повышения эффективности бизнеса, важно приобрести опыт программирования API, в том числе – создания клиентских сетевых запросов и генерации динамической графики на стороне браузера. Возможность приобрести знание того, как современный JavaScript используется для управления веб-интерфейсами, и развить способность применять это знание для решения практических задач, определяет актуальность данной программы повышения квалификации. Программа направлена на совершенствование профессиональной компетенции «Способен ставить и решать прикладные задачи по написанию программного кода на JavaScript и его размещению в веб-страницах, созданных при верстке информационных ресурсов и использующих Web API». Для успешного освоения программы обучающимся необходимо владеть базовыми компетенциями цифровой грамотности, в том числе навыками владения ПК на уровне среднего или продвинутого пользователя; компетенции в области основ алгоритмизации, в том числе знаниями и умениями в области алгоритмизации (знание основных алгоритмических структур, умения выражать их на псевдокоде и в виде блок-схем) и основ программирования, базовыми знаниями в области языка JavaScript. В процессе обучения по программе обучающимся рекомендуется прослушать все видеолекции, выполнять по каждой теме практические задания с автоматизированной проверкой и проработать самостоятельно дополнительные материалы, после этого необходимо выполнить тесты по каждой теме и для успешного завершения обучения выполнить итоговый тест (70% верных ответов). Обучающийся в результате освоения программы: будут знать синтаксис языков программирования JavaScript, JSON и SVG, особенности выбранной среды программирования (редактор кода), стандартные библиотеки JavaScript для работы с Web API на примерах axios, fabric и других, технологии программирования на языке JavaScript с использованием Web API; будут уметь применять JavaScript для написания программного кода, использующего Web API использовать выбранную среду программирования (редактор кода), использовать возможности имеющейся программной архитектуры веб-приложения на примере MVVM; будут владеть навыками создания программного кода на JavaScript, использующего браузерные прикладные интерфейсы, в соответствии со спецификациями; инструментами размещения программного кода на JavaScript в веб-страницах, созданных при верстке информационных ресурсов и использующих Web API; приемами размещения программного кода в клиентской части информационных ресурсов; навыками разработки мультимедийных элементов в составе веб-страниц. Результаты обучения по программе будут востребованы при решении следующих профессиональных задач: создание браузерных приложений в соответствии с спецификациями и его совершенствование с использованием специализированных программных средств на примере линтеров JavaScript; написание программного кода на языке JavaScript, управляющего браузерными API, определение и манипулирование данными; размещение программного кода в клиентской части веб-ресурса.

**Автономная некоммерческая организация**

**дополнительного профессионального образования**

**«СофтЛайн Эдюкейшн»**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ:  Генеральный директор  АНО ДПО «СофтЛайн Эдюкейшн»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Э. Разуваев  «\_14\_» \_\_Октября\_\_\_ 2020\_ г. |

**Дополнительная профессиональная программа повышения квалификации**

**«Браузерные интерфейсы прикладного программирования (API)»**

72 часа

Москва, 2020

**ОБЩАЯ ХАРАКТЕРИСТИКА ПРОГРАММЫ**

**1.Цель программы**

Формирование профессиональных компетенций, необходимых для решения прикладных задач по созданию и совершенствованию программного кода на JavaScript и его размещению в веб-страницах, созданных при верстке информационных ресурсов и использующих Web API

**2.Планируемые результаты обучения:**

2.1.Знание (осведомленность в областях)

2.1.1 синтаксис языков JavaScript, JSON и SVG;

2.1.2 особенностей выбранной среды программирования (редактора кода);

2.1.3 стандартных библиотек языка программирования JavaScript для работы с Web API на примерах axios, fabric и других;

2.1.4 технологии программирования на языке JavaScript с использованием Web API.

2.2. Умение (способность к деятельности)

2.2.1 применять JavaScript для написания программного кода, использующего Web API;

2.2.2 использовать выбранную среду программирования (редактор кода) для создания сценариев;

2.2.3 использовать возможности имеющейся программной архитектуры веб-приложения на примере MVVM;

2.3.Навыки (использование конкретных инструментов)

2.3.1 создания программного кода на JavaScript, использующего браузерные прикладные интерфейсы, в соответствии с спецификациями;

2.3.2. инструментами размещения программного кода на JavaScript в веб-страницах, созданных при верстке информационных ресурсов и использующих Web API;

2.3.3. приемами размещения программного кода в клиентской части информационных ресурсов;

2.3.4. навыками разработки мультимедийных элементов в составе веб-страниц.

**3.Категория слушателей** (возможно заполнение не всех полей)

3.1. Образование: среднее профессиональное, высшее

3.2. Квалификация: уровень квалификации не ниже 4

3.3. Наличие опыта профессиональной деятельности: опыт разработки программного обеспечения, электронных образовательных ресурсов и информационных систем

3.4. Предварительное освоение иных дисциплин/курсов /модулей

**4.Учебный план программы «Браузерные интерфейсы прикладного программирования (API)»**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Модуль** | **Всего, час** | **Виды учебных занятий** | | |
| **лекции** | **практические занятия** | **самостоятельная работа** |
|  | Модуль 1. Виды браузерных API | **4** | 1 | 2 | 1 |
|  | Модуль 2. Управление Document Object Model (DOM) и History API | **10** | 1 | 7 | 2 |
|  | Модуль 3. Управление обработкой событий и промисы | **17** | 3 | 10 | 4 |
|  | Модуль 4.Работа с изображениями и пользовательским интерфейсом | **19** | 3 | 13 | 3 |
|  | Модуль 5. Управление воспроизведением аудио и видео | **5** | 1 | 3 | 1 |
|  | Модуль 6. Выполнение параллельного кода в веб-воркерах | **6** | 2 | 3 | 1 |
|  | Модуль 7. Организация хранения данных в Web Storage | **5** | 2 | 2 | 1 |
|  | Модуль 8. Измерение производительности кода с помощью Performance API | **5** | 2 | 2 | 1 |
| **Итоговая аттестация** | | **1** | **Тест, зачет** | | **1** |
| **ИТОГО:** | | **72** | **15** | **42** | **15** |

**5.Календарный план-график реализации образовательной программы**

(дата начала обучения – дата завершения обучения) в текущем календарном году, указания на периодичность набора групп (не менее 1 группы в месяц)

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Наименование учебных модулей** | **Трудоёмкость (час)** | **Сроки обучения** |
| **1** | Модуль 1. Виды браузерных API | 4 | **01.11** |
| **2** | Модуль 2. Управление Document Object Model (DOM) и History API | 10 | **02.11-03.11** |
| **3** | Модуль 3. Управление обработкой событий и промисы | 17 | **04.11-07.11** |
| **4** | Модуль 4.Работа с изображениями и пользовательским интерфейсом | 19 | **07.11-10.11** |
| **5** | Модуль 5. Управление воспроизведением аудио и видео | 5 | **11.11** |
| **6** | Модуль 6. Выполнение параллельного кода в веб-воркерах | 6 | **12.11** |
| **7** | Модуль 7. Организация хранения данных в Web Storage | 5 | **13.11** |
| **8** | Модуль 8. Измерение производительности кода с помощью Performance API | 5 | **14.11** |
| **11** | Итоговая аттестация | 1 | **15.11** |
| **Всего:** | | **72** | **15 календарных дней** |

**6.Учебно-тематический план программы «Браузерные интерфейсы прикладного программирования (API)»**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Модуль / Тема** | **Всего, час** | **Виды учебных занятий** | | | **Формы контроля** |
| **лекции** | **практические занятия** | **самостоятельная работа** |
| 1 | **Модуль 1.** Виды браузерных API | **4** | **1** | **2** | **1** |  |
| 1.1 | Тема 1. Виды браузерных API | 4 | 1 | 2 | 1 | тест |
| 2 | **Модуль 2. Управление Document Object Model (DOM) и History API** | **10** | **1** | **7** | **2** |  |
| 2.1 | Тема 2.1. Управление Document Object Model (DOM) и History API | 10 | 1 | 7 | 2 | тест |
| 3 | **Модуль 3. Управление обработкой событий и промисы** | **17** | **3** | **10** | **4** |  |
| 3.1 | Тема 3.1. Управление обработкой событий | 5 | 1 | 3 | 1 | тест |
| 3.2 | Тема 3.2. Цепочки промисов | 6 | 1 | 4 | 1 | тест |
| 3.3 | Тема 3.3. Извлечение данных с помощью Fetch API | 6 | 1 | 3 | 2 | тест |
| 4 | **Модуль 4.Работа с изображениями и пользовательским интерфейсом** | **19** | **3** | **13** | **3** |  |
| 4.1 | Тема 4.1. Построение изображений с помощью SVG и Canvas | 6 | 1 | 4 | 1 | тест |
| 4.2 | Тема 4.2. Реализация перетаскивания | 6 | 1 | 4 | 1 | тест |
| 4.3 | Тема 4.3. Веб-компоненты | 7 | 1 | 5 | 1 | тест |
| 5 | **Модуль 5. Управление воспроизведением аудио и видео** | **5** | **1** | **3** | **1** |  |
| 5.1 | Тема 5.1. Управление воспроизведением аудио и видео | 5 | 1 | 3 | 1 | тест |
| **6** | **Модуль 6. Выполнение параллельного кода в веб-воркерах** | **6** | **2** | **3** | **1** |  |
| 6.1 | Тема 6.1. Выполнение параллельного кода в веб-воркерах | 6 | 2 | 3 | 1 | тест |
| **7** | **Модуль 7. Организация хранения данных в Web Storage** | **5** | **2** | **2** | **1** |  |
| 7.1 | Тема 7.1. Организация хранения данных в Web Storage | 5 | 2 | 2 | 1 | тест |
|  | **Модуль 8. Измерение производительности кода с помощью Performance API** | **5** | **2** | **2** | **1** |  |
|  | Тема 8.1. Измерение производительности кода с помощью Performance API | 5 | 2 | 2 | 1 | тест |
|  | **Итоговая**  **аттестация** | **1** |  |  | **1** | **Итоговый тест** |
|  |  | **72** | 15 | 42 | **15** |  |

**7. Учебная (рабочая) программа повышения квалификации «Браузерные интерфейсы прикладного программирования (API)»**

**Модуль 1.** Виды браузерных API **(4 час.)**

Тема 1. Виды браузерных API (4 час.)

Интерфейсы прикладного программирования; Web API; RESTful API

**Модуль 2. Управление Document Object Model (DOM) и History API (10 час.)**

Тема 2.1. Управление Document Object Model (DOM) и History API (10 час.)

Иерархическая модель документа. Объект window. Метод querySelector / querySelectorAll. Типы данных и структуры DOM. Объекты location и history

**Модуль 3. Управление обработкой событий и промисы (17 час.)**

Тема 3.1. Управление обработкой событий (5 час.)

Слушатели событий и коллбэки. Добавление и удаление слушателей. Возбуждение событий. Кастомные события

Тема 3.2. Цепочки промисов (6 час.)

Понятие ада коллбэков и промисификации. Цикл событий JavaScript и очередь. Использование async/await

Тема 3.3. Извлечение данных с помощью Fetch API (6 час.)

Технология AJAX. Объект XMLHttpRequest. Метод fetch. Выполнение асинхронных заданий

**Модуль 4. Работа с изображениями и пользовательским интерфейсом (19 час.)**

Тема 4.1. Построение изображений с помощью SVG и Canvas (6 час.)

Статическая и динамическая графика на веб-страницах. Декларативный и императивный подход к построению графических изображений. ХолстТема

4.2. Реализация перетаскивания (5 час.)

Интерфейсы, основанные на технологии drag and drop. Технологии перетаскивания в HTML5. События перетаскивания. Использование сторонних библиотекТема

4.3. Веб-компоненты (8 час.)

Модуляризация и компонентизация. Повторное использование разметки. Кастомные элементы HTML5

**Модуль 5. Управление воспроизведением аудио и видео (5 час)**

Тема 5.1. Управление воспроизведением аудио и видео (5 час.)

HTML5 Audio. HTML5 Video. WebAudio API

**Модуль 6. Выполнение параллельного кода в веб-воркерах (6 час.)**

Тема 6.1. Выполнение параллельного кода в веб-воркерах (6час.)

Параллелизм, однопоточность и многопоточность в JavaScript. Создание веб-воркеров

**Модуль 7. Организация хранения данных в Web Storage (5час.)**

Тема 7.1. Организация хранения данных в Web Storage (5 час.)

Использование LocalStorage. Использование IndexedDB. Использование PouchDB

**Модуль 8. Измерение производительности кода с помощью Performance API (5 час.)**

Тема 8.1. Измерение производительности кода с помощью Performance API (5 час.)

Измерение производительности JavaScript-кода. Использование методов консоли. Использование Performance API

**Итоговая аттестация (1 час.)**

**Описание практико-ориентированных заданий и кейсов**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Номер темы/модуля** | **Наименование практического занятия** | **Описание** |
| **1.1** | **Модуль 1. Тема 1.** | 1.Использование встроенного API для получения данных | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умений создания переменной в JavaScript.  *Пример задания (по видеолекции).*  Перейдите по адресу  <https://node-server.online/r> и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – apiintro-001\_ (выбрать из выпадающего списка) – и пароль (nodetasker).  3. В последнем-нижнем поле напишите число – код ответа веб-сервера по адресу kodaktor.ru/hi, если отправить его с помощью fetch методом POST. Пример оформления:    // здесь ваше решение из одной строки |
| **2.1** | **Модуль 2. Тема 2.1** | 1.Отбор элементов веб-страницы по селектору   1. Создание экземпляра элемента типа span 2. Создание экземпляра элемента типа ul 3. Создание экземпляра элемента типа ol | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой*. Практические задания на отработку умения использовать методы выборки элементов из поддерева DOM. *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке массив элементов, выбранных из родительского элемента по селектору x.  *Пример задания.* Практические задания на отработку умения создавать новый элемент в структуре DOM. *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания– и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке созданный с помощью метода createElement новый DOM-элемент типа span у которого значением атрибута id является x.*.*  *Пример задания.* Практические задания на отработку умения создавать новый элемент в структуре DOM. *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке созданный с помощью метода createElement новый DOM-элемент типа ul у которого имеется x пустых элементов-потомков li  *Пример задания.* Практические задания на отработку умения создавать новый элемент в структуре DOM. *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке созданный с помощью метода createElement новый DOM-элемент типа ol у которого имеется x пустых элементов-потомков li |
| **3.1** | **Модуль 3. Тема 3.1** | 1.Создание экземпляра пользовательского события | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  *Пример задания.*  Практическое задание на отработку умения создавать пользовательское событие. Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке новое пользовательское событие класса CustomEvent. |
| **3.2** | **Модуль 3. Тема 3.2** | 1.Создание экземпляра отвергающего промиса  2. Создание экземпляра разрешённого промиса | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.* Практические задания на отработку умения использовать методы выборки элементов из поддерева DOM.  *Пример задания*. Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания prom\_001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке промис, разрешённый в x.  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке промис, отвергающий с причиной x.  *Пример задания*. Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке промис, разрешённый в x.  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке промис, разрешающийся в значение x. |
| **3.3** | **Модуль 3. Тема 3.3** | 1.Выполнение запроса с помощью Fetch API с использованием метода GET  2. Выполнение запроса с помощью Fetch API с использованием метода POST | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку использовать метод fetch для получения данных в формате JSON.  *Пример задания.*  Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания fetch-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), (аннотированной словом async), который обращается к адресу https://kodaktor.ru/api/m/x с помощью метода fetch и передаёт заголовок "Content-Type: application/json". Ваш код получит JSON-объект со свойством result. Если x – число, то и result будет числом. Вы можете в качестве примера выполнить curl -SL 'https://kodaktor.ru/api/m/5' -H 'Content-Type: application/json' чтобы посмотреть, какой ответ присылает сервер. Функция task должна возвращать этот result.  Практические задания на отработку умения использовать метод fetch для получения данных методом POST.  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), (аннотированной словом async), который обращается к адресу https://kodaktor.ru/api/p/x с помощью метода fetch методом POST и возвращает то, что возвратил этот адрес. |
| **4.1** | **Модуль 4. Тема 4.1** | 1.Создание нового экземпляра элемента canvas  2. Создание нового экземпляра элемента svg  3. Автоматизация получения информации об элементе canvas | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.* Практические задания на отработку умения создавать новый элемент canvas.  *Пример задания* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), который возвращает новый элемент класса CanvasElement.  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), который возвращает новый элемент svg.    *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), который по идентификатору элемента canvas возвращает его ширину. |
| **4.2** | **Модуль 4. Тема 4.2** | 1.Создание экземпляра события DragEvent  2. Программная установка атрибута draggable | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.* Практические задания на отработку умения создавать событие перетаскивания.  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке новое пользовательское событие класса DragEvent.  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке выражение, которое устанавливает элементу с идентификатором x атрибут draggable в истинное значение. |
| **4.3** | **Модуль 4. Тема 4.3** | 1.Создание экземпляра HTML-элемента  2. Создание нового экземпляра элемента заголовка с помощью метода библиотеки React  3. Создание нового экземпляра элемента, имеющего атрибут, с помощью API библиотеки React | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.* Практические задания на отработку умения создавать кастомные элементы HTML5.  *Пример задания*. Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке новый экземпляр тега x.  Практическое задание на отработку умения работать с элементами React.  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания react-001 (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей созданный с помощью функции createElement объекта React (он будет доступен в функции и используется версия 16.13.1) новый элемент h1 без свойств и с единственным потомком в виде текста, заданного аргументом функции x.    *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания– и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающей созданный с помощью функции createElement объекта React (он будет доступен в функции и используется версия 16.13.1) новый элемент div со свойством id, имеющим значение текста, заданного аргументом функции x. |
| **5.1** | 1. Модуль 5. Тема 5.1 | 1. Создание экземпляра элемента audio 2. Создание экземпляра класса video | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умений создавать медиаэлементы HTML5. *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке новый экземпляр тега audio со значением атрибута id, равным x.  Практические задание на отработку умения создавать медиаэлементы HTML5. *Пример заданий.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке новый экземпляр тега video со значением атрибута id, равным x. |
| **6.1** | **Модуль 6. Тема 6.1** | 1.Получение экземпляра класса WebWorker  2.Создание примера блокирующего кода ворвера | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения создавать веб-воркеры.  *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке новый экземпляр класса WebWorker.    *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке код воркера, работающего в течение x секунд в блокирующем режиме. |
| **7.1** | **Модуль 7. Тема 7.1** | 1.Получение значения из локального хранилища | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения обращаться к объекту LocalStorage. *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), возвращающий по переданной в качестве аргумента x строке значение из LocalStorage по ключу x. |
| **8.1** | **Модуль 8. Тема 8.1** | 1. Создание кода, занимающего определённое время для выполнения интерпрератором | *Видеоролик с демонстрацией выполнения заданий. Самостоятельное выполнение обязательных практических заданий с автоматизированной проверкой.*  Практические задания на отработку умения оценивать производительность кода. *Пример задания.* Перейдите по адресу https://node-server.online/r и заполните поля: ваши Фамилия-Имя-Отчество, ключ (код) задания (выбрать из выпадающего списка) – и пароль (nodetasker). В последнем-нижнем поле напишите код функции task(x), выполняющийся не менее 2 секунд и не более 3 секунд и блокирующий цикл событий. |

**8.Оценочные материалы по образовательной программе**

**8.1. Вопросы тестирования по модулям**

|  |  |  |  |
| --- | --- | --- | --- |
| **№ модуля** | **Вопросы входного тестирования** | **Вопросы промежуточного тестирования** | **Вопросы итогового тестирования** |
| **0 (Входное тестирование)** | 1. Как называется иерархическая модель, позволяющая сценариям на JavaScript работать с элементами веб-страницы как с объектами?   1. DOM 2. HTTP 3. WWW 4. URL   2. Как называется технология, позволяющая пользователю перетаскивать и оставлять элементы интерфейса?   1. Drag and drop 2. Take and leave 3. AJAX 4. CORS   3. Как называется сценарий в отдельном файле, обеспечивающий изоляцию от других сценариев и позволяющий точно указать, что он использует и чем позволяет пользоваться?   1. модуль 2. скрипт 3. объект 4. блок   4. Каково официальное название формата MP3?   1. MPEG-1 Audio Layer III 2. MPEG-3 Layer Audio 1-3 3. MPG3-1 4. Audio MPEG 3   5. В каком контейнере обычно помещаются звуковые файлы, закодированные с помощью Vorbis?   1. OGG 2. AVI 3. WAV 4. SWF   6. Назовите открытый формат контейнера для мультимедиа файлов, представленный компанией Google в 2010 году, не требующий лицензионных отчислений и основанный на открытых видеокодеках VP8 и VP9, а также аудиокодеках Vorbis и Opus.   1. WebM 2. MP4 3. AVI 4. SWF   7. Назовите элемент, который в языке HTML5 используется для внедрения в веб-страницу звуковых файлов?   1. audio 2. sound 3. song 4. mp3   8. Как называется движок, положенный компанией Google в основу браузера Chrome?   1. V8 2. Trident 3. G-Engine 4. JS Engine 8   9. Как называются файлы, в которых браузер хранит информацию на компьютере пользователя?   1. cookies 2. cakes 3. candies 4. sweets   10. Выберите нереляционную базу данных:   1. MongoDB 2. MySQL 3. PostgreSQL 4. MariaDB   11. Какому объекту принадлежат методы log, time и timeEnd?   1. console 2. window 3. document 4. history   12. Как называется объект, которому принадлежит метод alert?   1. window 2. img 3. title 4. message   13. Какая инструкция не блокирует исполнение основного потока JavaScript?   1. for(;;); 2. while(1); 3. setTimeout(()=>{}); 4. do{}while({});   14. Как называется событие, возникающее при наведении указателя мыши на элемент веб-страницы?   1. mouseover 2. mousehover 3. mousearrived 4. mousehere   15. Как называется функция, позволяющая отложить действие на указанное количество миллисекунд, одинаково называющаяся и на браузерной платформе, и на платформе Node.js?   1. setTimeout 2. makeTime 3. postponeAction 4. delayDoing   16. Как называется функция, передаваемая другой функции в качестве функционального аргумента с целью дальнейшего вызова?   1. коллбэк 2. рекурсия 3. таймер 4. конструктор   17. Какое свойство окна позволяет присоединить функцию, реагирующую на событие, которое возникает, когда завершается загрузка веб-страницы?   1. onload 2. onfinish 3. loaddone 4. pageLoaded   18. Как называется технология, позволяющая осуществлять запросы веб-страницы к внешним ресурсам, не перезагружая страницу целиком?   1. AJAX 2. ACHIllES 3. KLEENEX 4. POLLUX   19. Как называется язык, позволяющий описывать векторные изображения с помощью тегов, подобных тегам языка HTML?   1. SVG 2. JSON 3. ActiveX 4. ActionScript   20. Выберите формат растровой графики:   1. PNG 2. VML 3. SWF 4. TTF | **-** | **-** |
| **1.1** | **-** | 1. Совокупность функций и инструментов, позволяющих программисту создавать приложения (программы), работающие в некоторой среде – это: - API  - XML - JSON - HTTP 2. Canvas относится к: - HTML5 API - REST API - Win32 Sound API - xAPI 3. ClipBoard относится к: - Web API - OpenGL API - Win32 Sound API - TinCan API 4. Элементы веб-страницы в DOM, создаваемые тегами, относятся к классу: - HTMLElement - TagElement - HTMLTag - HTMLAPI 5. Какой метод DOM позволяет получить первый из элементов, удовлетворяющих указанному CSS-селектору? - document.querySelector  - document.firstElement - window.selector - browser.getElementBySelector 6. Для привязки слушателей событий в DOM используется метод.. - addEventListener - addCustomListener  - attachEventObserver - listenToEvent 7. Как называется браузерное API, позволяющие осуществлять асинхронные сетевые запросы? - Fetch API - Net API - WebRequest API - HTMLPage API 8. Какое браузерное API позволяет обрабатывать операции над аудио с помощью специального аудио контекста? - WebAudio API - BrowserSounds API - ChromeMusic API - PlayTunes API | **-** |
| **2.1** |  | 1. Среда, которую предоставляет браузер, и с которой взаимодействует ядро JavaScript, называется? - DOM  - JSON - XML - HTTP 2. Объектная модель документа представляет собой: - иерархию - линейный список - рекурсивную функцию - кодировку 3. Объект history является свойством: -объекта window - объекта document - объекта global - объекта location 4. Как верно записывается инструкция JavaScript для объявления переменной без присваивания? - let - LET - Let - const 5. Какой метод DOM позволяет получить коллекцию из элементов, удовлетворяющих указанному CSS-селектору? - document.querySelectorAll  - document.CSSElements - window.selector - browser.getAllElementsBySelector 6. Свойство nextSibling является частью спецификации - DOM Level 1 - DOM Level 0 - HTML5 - JSON 7. Какой метод позволяет указать место вставки HTML-разметки как ' beforebegin ' либо ' afterbegin ' либо ' beforeend ' либо ' afterend '? - insertAdjacentHTML - appendChild - pasteElementInto - .incorporateElement 8. Свойство childNodes из интерфейса Node возвращает? - live-коллекцию - статическую коллекцию - функцию - числовую константу |  |
| **3.1** | **-** | 1. Назовите класс DOM/ JavaScript для создания собственных пользовательских событий (сообщений, обрабатываемых слушателями событий типа 'click')? - CustomEvent  - HTML5Event - NewEvent - UserEvent 2. Какой метод используется для добавления к элементу DOM слушателя события в виде коллбэка? -addEventListener - attachEventListener - addNewListener - appendEventListener 3. Какое событие возникает при щелчке по кнопке? - click - buttonclick  - btPress  - pushbutton 4. Код слушателей событий исполняется: - в очереди - всегда синхронно - в отдельном процессе - в новом окне 5. За выполнение кода, сбора и обработки событий и выполнения под-задач из очереди отвечает? - Цикл Событий - Менеджер параллельных потоков - Стек событий - Системный таймер 6. При асинхронном исполнении кода, изменяющего DOM, интерфейс пользователя - не блокируется - блокируется - отключается - перезагружается 7. Аналогом метода addEventListener в Node.js является -addListener - attachListener - appendListener - addNodeListener 8. Предположим, в элементе script объявлена функция function f(){ alert(1); } и далее следует элемент button, при щелчке по которому она должна быть вызвана. Как указать связь между щелчком и вызовом функции в открывающем теге <button>? -onclick="f()" - onclick="f" - onclick="[f]" - onclick=(f) | **-** |
| **3.2** | **-** | 1. Промис – это структура, которую возвращает - fetch  - alert - addEventListener - setTimeout 2. Каково число состояний, в которых может находиться промис? - 3 - 4 - 5 - бесконечно много 3. Что возвращает функция, аннотированная как async? - Promise - Function - Array - Map 4. Какой метод класса Promise позволяет получить отвергнутый промис? - Reject - Refuse - Decline - Recline 5. На что следует заменить xxx в выражении void Promise.xxx(50).then(x => console.log(x + 50)), чтобы получить число 100 в консоли? - resolve - reject - run - approve 6. Что возвращает метод fetch? - Promise - Function - Set - document 7. Для того, чтобы в теле функции могло быть использовано слово await, каким словом должна быть аннотирована эта функция? -async - awaited - future - promise 8. В каком состоянии будет находиться промис, созданный как new Promise(r => r())? - resolved - undefined - rejected - pending | **-** |
| **3.3** | **-** | 1. Какой API является предшественником Fetch API? - XMLHttpRequest  - HTMLAJAX API - BrowserRequest API - Prefetch API 2. Вызов метода fetch возвращает объект с методом -then - after - later  - future 3. При вызове fetch('//kodaktor.ru/hi').then(x => x.text()).then(y => alert(y)) объект какого класса помещается в x? -Response - Reply  - Promise - String 4. При вызове fetch('//kodaktor.ru/hi').then(x => x.text()).then(y => alert(y)) объект какого типа помещается в y? - string - object - array - function 5. Что происходит при передаче методу fetch несуществующего URL? - TypeError - обращение к URL по умолчанию - прекращение работы сценария - перезагрузка страницы 6. Что происходит при попытке преобразовать в JSON данные, которые не соответствуют требованиям JSON, как в случае fetch('//kodaktor.ru/hi').then(x => x.json())? - SyntaxError - TypeError - ReferenceError - VarError 7. Что не может находиться на месте пропуска fetch('//kodaktor.ru/hi').then(x => x.\_\_\_\_()) по причине отсутствия соответствующего метода у объекта Response? - html - text - json - blob 8. Какой метод позволяет обработать ошибку при передаче методу fetch несуществующего URL? - catch - try - again - log | **-** |
| **4.1** | **-** | 1. Провайдером какого объекта является canvas (холст)? - CanvasRenderingContext2D  - CanvasHTMLContext - CanvasRenderingContextProvider - CanvasContext2D 2. Как направлены оси системы координат canvas? - вправо и вниз - влево и вниз - влево и вверх  - вправо и вверх 3. Где располагается начало системы координат canvas? - в левом верхнем углу  - в правом верхнем углу - в центре  - в нижнем углу 4. Экземпляром какого класса является canvas? - HTMLCanvasElement - HTMLCanvas2DElement - HTMLCanvasTag - HTML5Canvas 5. Какой метод у canvas позволяет создать контекст? - getContext - createContext - getNewContext  - attachContext 6. Каким методом задаётся начальная точка рисования? - moveTo  - drawFrom - startPoint - initialize 7. Каким методом создаётся контур? - beginPath - startContour - newPath - createFrame 8. Какой длины отрезок получается при рисовании командами ctx.moveTo(20,20); ctx.lineTo(20,100); (где ctx – контекст у canvas) - 80 - 180  - 100 - 20 | **-** |
| **4.2** | **-** | 1. Какой метод отменяет встроенную в браузер реакцию на событие? - preventDefault  - cancelDefaultReaction - removeEventDefaultListener - revokeDefaultListener 2. Какое API позволяет работать с перетаскиванием программно? - HTML Drag and Drop API - HTML5 Dragging API - ECMAScript Drag API - JavaScript Drop API 3. Какое событие возникает, когда пользователь начинает новое выделение? - selectstart - beginselect - newselection - createselection 4. Какой атрибут помечает элемент как доступный для перетаскивания? - draggable - isdroppable - available  - movable 5. Какие элементы или части веб-страницы недоступны для перетаскивания по умолчанию? - кнопки - выделения текста - изображения - ссылки 6. Какой объект, который может быть получен из свойства всех событий перетаскивания, используется для хранения данных, перетаскиваемых мышью во время операции drag and drop? - DataTransfer - DataStore - LocalStorage - SessionStorage 7. Какое событие вызывается , когда элемент или выделенный текст перетаскивается над доступной «целью бросания»? + dragover - dragon - dragarys - dragqueen 8. Какое событие вызывается при «бросании» перетаскиваемого элемента на предназначенную для этого цель? - drop - throw - finishdragging - draganddrop | **-** |
| **4.3** | **-** | 1. Какой символ должен присутствовать в имени кастомного (пользовательского) элемента HTML5? - дефис  - тире - двоеточие - тильда 2. Какой из компонентов Web Components API объявлен устаревшим? - HTML Imports - Shadow DOM - HTML Templates - Custom Elements 3. Какой из компонентов Web Components API отвечает за шаблонизацию? - HTML Templates  - Shadow DOM - HTML Imports - Custom Elements 4. Какой метод позволяет создать для элемента теневой DOM, который служат для инкапсуляции его функциональности?  - attachShadow - appendShadow - createNewShadow - pushShadow 5. Какой элемент будучи частью набора технологии Web Components, является заполнителем внутри веб компонента, который можно заполнить собственной разметкой, которая позволяет создавать отдельные деревья DOM и представлять их вместе? - slot - sloth; - facet; - stash; 6. Как называется реестр пользовательских элементов в HTML5, имеющий метод define? - customElements - userElements - customElementsRegistryStore - newCreatedElements 7. От какого класса происходит наследование классов, реализующий кастомные элементы? - HTMLElement - window - document - CustomClass 8. Какой метод создаёт копию узла, но не вставляет его в дерево DOM? - importNode - appendChild - adoptNode - copyAndPasteNode | **-** |
| **5.1** | **-** | 1. Какое браузерное API позволяет обрабатывать операции над аудио с помощью специального аудио контекста? - WebAudio API - BrowserSounds API - ChromeMusic API - PlayTunes API 2. Каково имя открывающего тега элемента, позволяющего вставить в веб-страницу проигрыватель звука в формате ogg vorbis? - audio - webaudio - ogg - vorbis 3. Как называется обрабатывающий звук объект, состоящий из связанных вместе аудио модулей, где каждый является экземпляром класса AudioNode? -AudioContext - SoundContext - AudioContext2D - HTML5AudioContext 4. Каково имя открывающего тега элемента, позволяющего вставить в веб-страницу проигрыватель видеоконтента?  - video - player - clip - videocanvas 5. Экземпляром какого класса является audio? - HTMLAudioElement - HTMLWebAudioElement - HTMLAudioTag - HTMLSoundElement 6. Для проигрывания на веб-странице создаётся 8-минутный стереоаудиофайл без сжатия с частотой дискретизации 16000 кГц, числом уровней квантования сигнала 65536. Если известно, что максимально возможная скорость передачи данных по WiFi-каналу составляет 10 Мбит в секунду, какое целое минимальное количество секунд потребуется для его передачи? - 24   - 23 - 123 - 18   1. Какой MIME-тип следует указать для проигрывания mp3-файла с помощью элемента audio? - audio/mpeg - audio/mp-3 - mp3/audio - mpeg/audio 2. Какой кодек использован для кодирования видео в файл kodaktor.ru/audiovideo.mp4? - H264 - H263 - H265 - HHVM | **-** |
| **6.1** | **-** | 1. К какому масштабированию относится создание копий запущенного приложения? - горизонтальному - вертикальному - круговому - зеркальному 2. В какой версии Node.js была введена многопоточность за экспериментальным флагом --experimental-worker? - 10.5.0 - .1.0.1 - 8.3.3 - 4.4.4 3. Какой класс позволяет создать в браузере новый рабочий поток, не связанный с DOM и служащий для выполнения вычислений в фоне? - Worker - Computer - Calculator - Prole 4. С помощью какого метода происходит отправка сообщения из воркера в основной сценарий?  - postMessage - sendSignal - callMain - doneWorking 5. Сколько целых секунд как минимум займёт выполнение воркера по адресу https://kodaktor.ru/j/worker\_sample3? - 3 - 7 - 10 - 15 6. Выберите корректный вариант создания воркера:  - const worker = new Worker('//kodaktor.ru/j/worker\_sample3'); - const worker = new WebWorker(//kodaktor.ru/j/worker\_sample3) - const worker = new CustomWorker(/kodaktor.ru/j/worker\_sample3/) - const worker = new CustomWorker(/worker/) 7. Если f – функция, а worker – воркер, как выглядит верное присвоение функции f обработчику события завершения работы воркера? - worker.onmessage = f - worker.onmessage(f) - worker.message = f - worker.message[f] 8. Если f – функция, как выглядит верное присвоение функции f обработчику события начала работы воркера внутри него? - onmessage = f - onmessage(f) - .message = f - .message[f] | **-** |
| **7.1** | **-** | 1. Какой механизм веб-хранилища обслуживает область хранения данных для каждого домена, доступное даже в случае, если переоткрыть браузер? - localStorage - windowStorage - documentStorage - locationStorage 2. Какое выражение не вернёт ссылку на локальное хранилище? - document.localStorage - top.localStorage - window.localStorage - localStorage 3. Чтобы можно было прочитать запись из локального хранилища с помощью выражения localStorage.getItem('a'), какой вариант попытки размещения этой записи не сработает? - localStorage.setItem('a'); - window.localStorage.a = 'b'; - window.localStorage['a'] = 'b'; - localStorage.setItem('a', 'z'); 4. Выберите верный вариант помещения в localStorage значения 2020 по ключу this-year - localStorage.setItem('this-year', 2020); - localStorage.this-year = 2020 - localStorage[this-year] = 2020 - localStorage.setItem({'this-year': 2020}); 5. Какое свойство позволяет выяснить количество сохраняемых в localStorage записей? - length - size - number - items 6. Предположим, что выражение localStorage.length возвращает значение 10. Что оно будет возвращать после выполнения инструкции localStorage.length = 9? - 10 - 9  - null  - TypeError 7. Какой оператор позволяет удалить ключ из localStorage? - delete - remove - erase - terminate 8. Как называется низкоуровневое API для клиентского хранилища большого объема структурированных данных, позволяющее совершать асинхронные операции? - IndexedDB - IndexedDB - 'ab' - 'a1' | **-** |
| **8.1** | **-** | 1. В каких единицах измеряется пауза, указываемая при вызове метода setTimeout?  - миллисекунды  - секунды - минуты - часы 2. Какой метод консоли позволяет поставить метку начала измерения времени?  - time - log - table - measure 3. Какой метод консоли позволяет поставить метку завершения измерения времени? - timeEnd - timeStop - finishTime - measureTime 4. Какое целое количество секунд будет выведено в консоль в результате выполнения фрагмента console.time('a');((y = 2 \*\* 13 , x = Date.now()) => { while (Date.now() - x < y) ;})();console.timeEnd('a'); - 8 - 88 - 18 - 23 5. Какой интерфейс позволяет измерять время с точностью до 5 микросекунд и имеет методы now и toJSON?  - Performance API - Fetch - Canvas - WebCrypto 6. Какой метод, доступный из объекта window.performance, позволяет создать отметку для начала измерения? - mark - markup - markdown - remark 7. Какой интерфейс используются для наблюдения за событиями измерения производительности и подписки на уведомления о новых записях производительности, по мере того как они отражаются во временной шкале производительности браузера? - PerformanceObserver - TimeLineWatcher - EventsRecorder - RecordsNotifier 8. Какой метод позволяет отложить выполнение кода (содержащего реализацию анимации) до момента, когда браузер сможет сделать это наиболее эффективно? - requestAnimationFrame - waitForAnimation - queryAnimatedFrame - delayCodeExecution | **-** |
| **Итоговое тестирование** |  |  | Примеры вопросов (тест формируется из базы тестовых вопросов)  1. К какому объекту DOM принадлежит свойство title, отвечающее за заголовок вкладки или окна?   1. document 2. window 3. frame 4. history   2. Какой методэлемента, играющего роль «холста», позволяет получить объект «контекста рисования­­»?   1. getContext 2. makeCanvas 3. createCanvasContext 4. createImage   3. Как называется сценарий в отдельном файле, обеспечивающий изоляцию от других сценариев и позволяющий точно указать, что он использует и чем позволяет пользоваться?   1. модуль 2. скрипт 3. объект 4. блок   4. Какой атрибут делает элемент перетаскиваемым?   1. draggable 2. candrag 3. isdragged 4. unbearable   5. Какой знак должен присутствовать в имени элемента, инстанциирующего веб-компонент?   1. дефис 2. точка с запятой 3. восклицательный знак 4. двоеточие   6. Назовите открытый формат контейнера для мультимедиа файлов, представленный компанией Google в 2010 году, не требующий лицензионных отчислений и основанный на открытых видеокодеках VP8 и VP9, а также аудиокодеках Vorbis и Opus.   1. WebM 2. MP4 3. AVI 4. SWF   7. Какой атрибут открывающего тега элемента video позволяет указать изображение, демонстрируемое вместо видеоролика, пока видеоролик загружается?   1. poster 2. image 3. snapshot 4. loading   8. Какой метод используется для передачи сообщения из воркера в основной сценарий?   1. postMessage 2. fireMessage 3. sendSignal 4. justDoIt   9. Как называются файлы, в которых браузер хранит информацию на компьютере пользователя?   1. cookies 2. cakes 3. candies 4. sweets   10. Выберите хранилище, которое предоставляет возможность осуществлять операции асинхронно:   1. localStorage 2. sessionStorage 3. document.dataSet 4. IndexedDB   11. Какова единица измерения времени, используемая функцией console.time?   1. миллисекунды 2. микросекунды 3. наносекунды 4. петасекунды   12. Выберите метод DOM, который возвращает статическую коллекцию узлов и использует селектор каскадных стилей для отбора элементов.   1. querySelectorAll 2. querySelector 3. getElementById 4. fetchDOMElements   13. Какой метод позволяет установить метку на линейке времени для измерения?   1. performance.mark 2. performance.setMarker 3. performance.timeLabel 4. performance.putBookmark   14. Как называется событие, возникающее при щелчке мышью по элементу веб-страницы?   1. click 2. mouseclick 3. mousepress 4. mousehit   15. Выберите метод DOM, позволяющий, используя такой класс как MouseEvent, возбудить событие в элементе, который снабжён слушателем этого события?   1. dispatchEvent 2. makeEvent 3. postponeEvent 4. delayEvent   16. Как называется функция, передаваемая другой функции в качестве функционального аргумента с целью дальнейшего вызова?   1. коллбэк 2. рекурсия 3. таймер 4. конструктор   17. Какой метод объектов класса Promise позволяет выполнить асинхронное действие в случае перехода в отвергнутое состояние?   1. catch 2. fetch 3. scratch 4. matchError   18. Объект какого класса возвращает метод fetch?   1. Promise 2. Future 3. Contract 4. String   19. Как называется технология современных браузеров – «совместное использование ресурсов между разными источниками», – которая позволяет предоставить веб-странице доступ к ресурсам другого домена?   1. CORS 2. JSON 3. HTTPS 4. SSL   20. Из перечисленных ниже выберите формат векторной графики:   1. PNG 2. SVG 3. GIF 4. JPEG |

* 1. **описание показателей и критериев оценивания, шкалы оценивания.**

Шкалы оценивания

* 1. Входное тестирование

**Шкала критериев оценивания по входному тестированию**

|  |  |  |
| --- | --- | --- |
| **Шкала оценивания** | **Критерии** | **Описание** |
| 70%-100% верных ответов на вопросы входного теста | **зачтено** | Потенциальный обучающийся успешно выполнил входной тест и может быть допущен к обучению по программе |
| 69% и менее верных ответов на вопросы входного теста | **не зачтено** | Потенциальный обучающийся не выполнил успешно входной тест и не может быть допущен к обучению по программе |

* 1. Промежуточное тестирование по темам/модулям

**Шкала критериев оценивания по промежуточному тестированию**

|  |  |  |  |
| --- | --- | --- | --- |
| **Шкала** | **Критерии** | **Описание** | **Уровень сформированности компетенции** |
| 80%-100% верных ответов на вопросы промежуточного теста | зачтено | Обучающийся успешно выполнил промежуточный тест по теме/модулю | Продвинутый уровень |
| 70%-79% | зачтено | Обучающийся успешно выполнил промежуточный тест по теме/модулю | Базовый уровень |
| 50%-69% верных ответов на вопросы промежуточного теста | не зачтено | Обучающийся не выполнил успешно промежуточный тест, необходимо повторное назначение теста и консультации преподавателя | Начальный уровень |
| Менее 50% верных ответов на вопросы промежуточного теста | не зачтено | Обучающийся не выполнил успешно промежуточный тест, необходимо повторное назначение теста и консультации преподавателя | Не владеет, компетенция не сформирована |

* 1. Итоговое тестирование по программе

**Шкала критериев оценивания по итоговому тестированию**

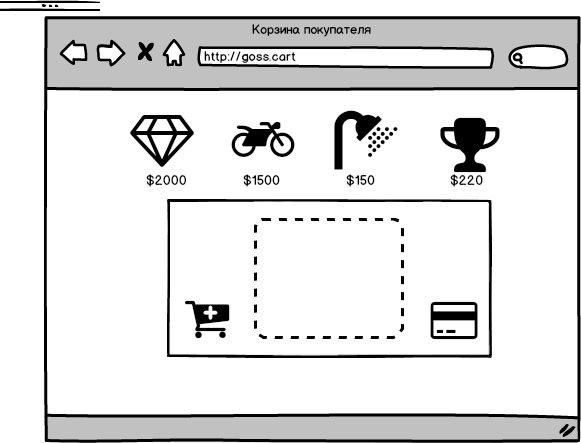
|  |  |  |  |
| --- | --- | --- | --- |
| **Шкала** | **Критерии** | **Описание** | **Уровень сформированности компетенции** |
| 80%-100% верных ответов на вопросы промежуточного теста | зачтено | Обучающийся успешно выполнил итоговый тест по программе | Продвинутый уровень |
| 70%-79% | зачтено | Обучающийся успешно выполнил итоговый тест по программе | Базовый уровень |
| 50%-69% верных ответов на вопросы промежуточного теста | не зачтено | Обучающийся не выполнил успешно итоговый тест по программе | Начальный уровень |
| Менее 50% верных ответов на вопросы промежуточного теста | не зачтено | Обучающийся не выполнил успешно итоговый тест по программе | Не владеет, компетенция не сформирована |

* 1. **примеры контрольных заданий по модулям или всей образовательной программе.**

**Пример контрольного задания по программе**

Создайте на GitHub репозиторий и разместите в виде сайта GitHub Pages  страницу со сценарием, который реализует потребительскую корзину:

* при щелчке по кнопке с изображением товара он добавляется в пространства корзины
* либо то же самое происходит при перетаскивании (можно использовать плагин jQuery для перетаскивания)



У пользователя должна быть возможность зафиксировать бюджет, так чтобы нельзя было выйти за его пределы и добавить товаров больше чем на указанную сумму.

По адресу <https://kodaktor.ru/cart_data.json> находятся цены для трёх товаров — используйте их для инициализации «витрины»

* 1. **тесты и обучающие задачи (кейсы), иные практико-ориентированные формы заданий.**

Примеры тестов представлены в таблице 8.1.

В программе используются преимущественно задания с автоматизированной проверкой, которые являются обязательными для выполнения по каждой теме. Дополнительно для организации самостоятельной работы используются задания со свободным ответом (в том числе проблемные), которые слушатели выполняют самостоятельно.

**ПРИМЕРЫ ЗАДАНИЙ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ**

**Модуль 1. Тема 1. Вводное занятие**

Самостоятельная работа 1 час

Задание 1. Разработайте и разместите в виде документа на kodaktor.ru сценарий, считывающий значение элемента title веб-страницы и выводящий его в консоль

**Модуль 2. Тема 2.1. Управление Document Object Model (DOM) и History API**

Самостоятельная работа 2 часа

Задание 1. Разработайте и разместите в виде документа на kodaktor.ru либо другом подобном сервисе сценарий, создающий после загрузки содержимого окна (window.onload) новый элемент h1, повторяющий содержимое элемента title в разделе head.

Задание 2. Разработайте и разместите в виде документа на kodaktor.ru либо другом подобном сервисе сценарий, меняющий содержимое URL окна на адрес сайта Yandex.ru.

Задание 3. Разработайте и разместите в виде документа на kodaktor.ru либо другом подобном сервисе сценарий, меняющий содержимое URL окна на адрес, указанный пользователем в содержимом элемента title.

**Модуль 3.**

**Тема 3.1. Управление обработкой событий**

Самостоятельная работа 1 час

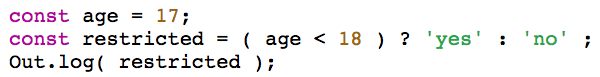
Задание 1. Разработайте и разместите в виде документа на kodaktor.ru либо другом подобном сервисе сценарий, меняющий содержимое URL окна на адрес, указанный пользователем в элементе input при нажатии по кнопку, размещённой рядом с этим элементом.

**Тема 3.2. Цепочки промисов**

Самостоятельная работа 1 час

Задание 1

Рассмотрите код в строках 11-13 на странице https://kodaktor.ru/ternary



И доработайте его так, чтобы ответ yes или no возвращался в виде разрешённого или, соответственно, отвергнутого промиса (restricted должно стать функцией, возвращающей новый промис, а вывод в консоль out должен происходить по результатам, соответственно, работы метода then или catch).

**Тема 3.3. Извлечение данных с помощью Fetch API**

Самостоятельная работа 2 часа

Задание 1 Обращение по адресу https://kodaktor.ru/api/rv/abc

превращает строку abc в cba. Обращение по адресу https://kodaktor.ru/api/rp/abc превращает строку abc

в abcabc. Т.е. первый адрес обращает строку, а второй удваивает.

Используя async/await, напишите код, который по щелчку по кнопке отправляет введённую в текстовое

поле строку на первый адрес, получает результат, отправляет этот результат на второй адрес и выдаёт

окончательный результат

**Модуль 4.**

**Тема 4.1. Построение изображений с помощью SVG и Canvas**

Самостоятельная работа 1 час

Задание 1

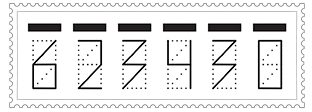
1. Создайте веб-страницу (борд) на Кодактор.ру или ином онлайн-редакторе

2. Вставьте элемент script и подключите сценарий <https://kodaktor.ru/j/canvas_0b> с функцией makeCanvas

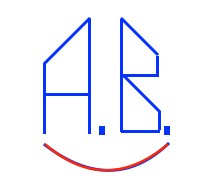
1. Вызовите функцию для создания переменных, содержащих холст и контекст   
   const { canvas, ctx } = makeCanvas(300, 120);
2. Добавьте холст в дерево DOM

document.body.appendChild(canvas);

1. Используя методы контекста, изобразите свои инициалы, использовав в качестве образца написание в стиле почтового индекса:



1. Используя кривую Безье, нарисуйте под инициалами дугу другого цвета:



**Тема 4.2. Реализация перетаскивания**

Самостоятельная работа 1 час

Задание 1

С помощью Drag and Drop API реализуйте простой тест-капчу, проверяющий умение собрать флаг РФ из трёх полос в правильном порядке.

**Тема 4.3. Веб-компоненты**

Самостоятельная работа 1 час

Задание 1

Используя инструкцию из видеоурока, создайте аналогичный компонент, позволяющий добавлять и удалять «звёзды» или «лайки» фреймворкам. Компонент должен помнить своё состояние, иметь кнопки для добавления и удаления, и показывать число лайков не в виде одного числа, а в виде ряда звёзд / сердец / лайков. Первоначальная информация  о количестве голосов, отданных за фреймворки, находится по адресу <https://kodaktor.ru/j/react5b_6cbf2>

**Модуль 5. Тема 5.1. Управление воспроизведением аудио и видео**

Самостоятельная работа 1 час

Задание 1

Ознакомившись с реализацией сценария проигрывания звука по нажатию кнопки https://kodaktor.ru/g/audiobutton реализуйте аналогичную кнопку, но с триггером (конечным автоматом с состояниями «вкл» и «выкл») - при нажатии на кнопку первый раз начинает играть музыка, второй раз - прекращает; при это циклически меняется цвет и надпись.

**Модуль 6. Тема 6.1. Выполнение параллельного кода в веб-воркерах**

Задание 1

Создайте воркер, вычисляющий по передаваемому в него значению n число ряда Фибоначчи с соответствующим номером.

**Модуль 7. Тема 7.1. Организация хранения данных в Web Storage**

Самостоятельная работа 1 час

Задание 1

Модифицируйте простейшее приложение с использованием React, управляющее списком заданий так, чтобы задания сохранялись в LocalStorage и считывались из LocalStorage при посещении страницы.

**Модуль 8. Тема 8.1. Измерение производительности кода с помощью Performance API**

Самостоятельная работа 1 час

Задание 1

Создайте сценарий на основе <https://kodaktor.ru/pauses> который с помощью Performance API измеряет время, затрачиваемое на сетевые запросы, и выдаёт суммарное затраченное время.

* 1. **описание процедуры оценивания результатов обучения.**

В процессе обучения слушатели должны выполнить практические задания с автоматизированной проверкой (в рамках практических занятий) и пройти промежуточную аттестацию по каждой теме – электронное тестирование. Промежуточное тестирование включает 8-10 вопросов по теме. Промежуточное тестирование считается успешно завершённым, если слушателем даны верные ответы на не менее чем 70% вопросов теста. Каждый слушатель имеет три попытки на выполнение промежуточного теста, время тестирования – 0,5 часа.

К итоговой аттестации допускаются слушатели, **выполнившие все практические задания с автоматизированной проверкой (в рамках практических занятий), задания самостоятельной работы и промежуточные тесты по темам (не менее 70% верных ответов по каждому тесту по теме)**, предусмотренные учебной программной. В ходе итогового электронного тестирования слушателю предлагается ответить на 20 вопросов. Тестирование считается успешно завершённым, если слушателем даны верные ответы на не менее чем 70% вопросов теста. Каждый слушатель имеет одну попытку на выполнение итогового теста, время тестирования – 1 час.

**9.Организационно-педагогические условия реализации программы**

**9.1. Кадровое обеспечение программы**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№**  **п/п** | **Фамилия, имя, отчество (при наличии)** | **Место основной работы и должность, ученая степень и ученое звание (при наличии)** | **Ссылки на веб-страницы с портфолио (при наличии)** | **Фото в формате jpeg** | **Отметка о полученном согласии на обработку персональных данных** |
| **1** | **Государев Илья Борисович** | Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО». Доцент факультета программной инженерии и компьютерной техники  Кандидат педагогических наук, доцент | https://kodaktor.ru/g/gosudarev |  | **+** |

**9.2.Учебно-методическое обеспечение и информационное сопровождение**

|  |  |
| --- | --- |
| **Учебно-методические материалы** | |
| Методы, формы и технологии | Методические разработки,  материалы курса, учебная литература |
| Методы: проблемно-развивающие методы; исследовательский, объяснительно-иллюстративный, метод проблемного изложения, частично-поисковый, методы контроля и самоконтроля  Формы: самостоятельная работа по освоению теоретического материала; самоконтроль через ответы на вопросы по видеолекциям и выполнение практических заданий для самоконтроля; промежуточный контроль (тесты по темам для промежуточной аттестации), итоговый контроль (итоговое тестирование по курсу). | 1. Государев, И.Б. Введение в веб-разработку на языке JavaScript [Электронный ресурс]: учебное пособие / И.Б. Государев. — Электрон. дан. — Санкт-Петербург: Лань, 2019. — 144 с. — Режим доступа: <https://e.lanbook.com/book/118648> 2. Современный открытый учебник http://javascript.ru/ |

|  |  |
| --- | --- |
| **Информационное сопровождение** | |
| Электронные  образовательные ресурсы | Электронные  информационные ресурсы |
| [Онлайн-редактор h1 («Кодактор») — авторская разработка; свидетельство о регистрации в Роспатенте (Федеральной службе по интеллектуальной собственности) №2014613612 от 01.04.2014)](https://kodaktor.ru/images/svid1.png)  Авторская система автоматизированной проверки заданий на языке JavaScript  **node-server.online/r** | 1. Выразительный JavaScript https://karmazzin.gitbooks.io/eloquentjavascript\_ru/content/index.html  2. Блог на коллективном ресурсе разработчиков <https://habr.com/ru/hub/javascript/> |

**9.3.Материально-технические условия реализации программы**

|  |  |
| --- | --- |
| Вид занятий | Наименование оборудования,  программного обеспечения |
| Лекционные (видеолекции) | Операционная система семейств Linux Mac Os, Windows.  Браузер Google Chrome  Открытая система управления виртуальными машинами VirtualBox  Рекомендуется скорость канала от 2 Мбит в секунду |
| Практические (видеоролики, практические задания) | Операционная система семейств Linux Mac Os.  Браузер Google Chrome, Canary  Открытая система управления виртуальными машинами VirtualBox  Рекомендуется скорость канала от 1 Мбит в секунду |
| Самостоятельная работа (выполнение практических заданий) | Операционная система семейств Linux Mac Os.  Браузер Google Chrome, Canary  Открытая система управления виртуальными машинами VirtualBox  Рекомендуется скорость канала от 1 Мбит в секунду |

**III.Паспорт компетенций (Приложение 2)**

Автономная некоммерческая организация дополнительного профессионального образования

«СофтЛайн Эдюкейшн»

.

ПАСПОРТ КОМПЕТЕНЦИИ

**Дополнительная профессиональная программа повышения квалификации**

**«Браузерные интерфейсы прикладного программирования (API)»**

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | Наименование компетенции | Способен ставить и решать прикладные задачи по написанию программного кода на JavaScript и его размещению в веб-страницах, созданных при верстке информационных ресурсов и использующих Web API | |
| 2. | Указание типа компетенции | профессиональная | |
| 3. | Определение, содержание и основные сущностные характеристики компетенции | Под компетенцией понимается способность постановки и нахождения путей решения прикладных задач по написанию программного кода на JavaScript и его размещению в веб-страницах, созданных при верстке информационных ресурсов и использующих Web API  Слушатель должен: *знать:*  - синтаксис языков JavaScript, JSON и SVG;  - особенности выбранной среды программирования (редактора кода);  - стандартные библиотеки JavaScript для работы с Web API на примерах axios, fabric и других;  - технологии программирования на JavaScript с использованием Web API;  *уметь:*  - применять JavaScript для написания программного кода, использующего Web API  - использовать выбранную среду программирования или редактор кода  - использовать возможности имеющейся программной архитектуры веб-приложения на примере MVVM;  *владеть*:  - технологиями создания программного кода на JavaScript, использующего браузерные прикладные интерфейсы, в соответствии со спецификациями;  - инструментами размещения программного кода на JavaScript в веб-страницах, созданных при верстке информационных ресурсов и использующих Web API;  - приемами размещения программного кода в клиентской части информационных ресурсов;  - технологиями разработки мультимедийных элементов в составе веб-страниц | |
| 4. | Дескриптор знаний, умений и навыков по уровням | Уровни сформированности компетенции  обучающегося | Индикаторы |
|  | Начальный уровень | *Знает:* синтаксис языков JavaScript, JSON и SVG и отдельные стандартные библиотеки JavaScript для работы с Web API.  *Умеет:* решать простейшие задачи на JavaScript, связанные с управлением API.  *Владеет:* приёмами создания программного кода на JavaScript, использующего браузерные прикладные интерфейсы, в соответствии с требованиями к пользовательскому интерфейсу |
|  | Базовый уровень | *Знает:* особенности выбранной среды программирования (редактора кода) и - стандартные библиотеки JavaScript для работы с Web API на примерах axios, fabric и других.  *Умеет:* использовать выбранную среду программирования или редактор кода.  *Владеет:* инструментами размещения программного кода на JavaScript в веб-страницах, созданных при верстке информационных ресурсов и использующих Web API |
|  | Продвинутый | *Знает:* технологии программирования на JavaScript с использованием Web API.  *Умеет:* использовать возможности имеющейся программной архитектуры веб-приложения на примере MVVM.  *Владеет:* приемами размещения программного кода в клиентской части информационных ресурсов, а также технологиями разработки мультимедийных элементов в составе веб-страниц |
| 5. | Характеристика взаимосвязи данной компетенции с другими компетенциями/ необходимость владения другими компетенциями для формирования данной компетенции | Для формирования данной компетенции слушатель должен владеть:   * компетенциями цифровой грамотности, в том числе навыками владения ПК на уровне среднего или продвинутого пользователя; * компетенции в области основ алгоритмизации (знание основных алгоритмических структур, умения выражать их на псевдокоде и в виде блок-схем) и основ программирования, базовыми знаниями в области языка JavaScript | |
| 6. | Средства и технологии оценки | Тесты, электронное промежуточное (по темам/модулям) и итоговое тестирование | |

**IV.Иная информация о качестве и востребованности образовательной программы** (результаты профессионально-общественной аккредитации образовательной программы, включение в системы рейтингования, призовые места по результатам проведения конкурсов образовательных программ и др.) (при наличии)

-

**V.Рекомендаций к программе от работодателей**: наличие не менее двух писем и/или подтверждения на цифровой платформе Государственной системы предоставления ПЦС от работодателей о рекомендации образовательной программы для реализации в рамках Государственной системы предоставления ПЦС на формирование у трудоспособного населения компетенций цифровой экономики с указанием востребованности результатов освоения программы в сфере деятельности соответствующих компаний и готовности к рассмотрению заявок наиболее успешно освоивших образовательную программу граждан на прохождение стажировки и (или) собеседования на предмет трудоустройства путем проставления отметки в профиле программы

**Загружено на платформу**

**VI.Указание на возможные сценарии профессиональной траектории граждан** по итогам освоения образовательной программы (в соответствии с приложением)

**СЦЕНАРИИ ПРОФЕССИОНАЛЬНОЙ ТРАЕКТОРИИ ГРАЖДАН**

|  |  |
| --- | --- |
| **Цели получения персонального цифрового сертификата** | |
| **текущий статус** | **цель** |
| **Развитие компетенций в текущей сфере занятости** | |
| работающий по найму в организации, на предприятии | сохранение текущего рабочего места |
| работающий по найму в организации, на предприятии | развитие профессиональных качеств |
| работающий по найму в организации, на предприятии | повышение заработной платы |
| работающий по найму в организации, на предприятии | смена работы без изменения сферы профессиональной деятельности |

**VII.Дополнительная информация**

**VIII.Приложенные Скан-копии**

Загружено на платформу:

1. Два рекомендательных письма

2. Скан утвержденной образовательной программы

3. Скан утвержденного паспорта образовательной программы

4. Паспорт образовательной программы в формате word

**Генеральный директор**

**АНО ДПО «СофтЛайн Эдюкейшн» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Э. Разуваев**

М.П.