

**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»**

**(РУТ (МИИТ)**

|  |
| --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Кафедра «Управление и защита информации»

**Отчет по лабораторной работе**

« **циклы, наблюдатели, витки**»

**по дисциплине**

«Web программирование»

Вариант 2

**Выполнил:** студенты

группы ТКИ-541

Порхун Д.Д. , Макаров Д.И.

**Проверил:** доцент кафедры УиЗИ, к.т.н. Сафронов А.И.

**Москва 2024 г.**

Оглавление

[Цель работы 3](#_Toc180347218)

[Формулировки задач 3](#_Toc180347219)

[Спецификация оборудования, на котором выполнялась работа. 4](#_Toc180347220)

[Информация об используемом браузере. 5](#_Toc180347221)

[Технология локального подключения фреймворка Vue.js 5](#_Toc180347222)

[Русифицированные схемы жизненного цикла Vue.js. 5](#_Toc180347223)

[Таблица соответствия переменных и методов, используемых в web-приложениях 7](#_Toc180347224)

[Содержательная часть по виткам жизненного цикла 11](#_Toc180347225)

[Содержательная часть по циклической отрисовке 17](#_Toc180347226)

[Содержательная часть по наблюдателям 23](#_Toc180347227)

[Вывод 30](#_Toc180347228)

# Цель работы

Цель работы заключается в разработке локальных одностраничных web-приложений (LSPWA) под управлением фреймворка Vue.js на языке JavaScript в соответствии с указаниями вариантов индивидуального задания.

# Формулировки задач

– локально (LSPWA), не прибегая к инструментарию Node.js и npm (Node Package Manager);

* **Задание 1:**

Реализовать витки жизненного цикла в файле с именем index-hooks\_Familiya\_I\_O.html  
  
Продумать схему тестирования и демонстрации работы всех витков / хуков / методов жизненного цикла фреймворка Vue.js в формате одностраничного web-приложения, отличающегося от рассмотренного в лекционном материале курса «Web-программирование». Реализовать схему тестирования строго под Vue.js 3.x. Русифицировать схему жизненного цикла Vue.js 3.x (допускается использование готовой реализации из официальной документации).

* **Задание 2:**

Реализовать v-for в файле с именем index-v-for\_Familiya\_I\_O.html;

Сделать статический, нумерованный список (гипертекстовая разметка – <ol>) по тематике «Марки автомобилей». Предоставить возможность оператору для добавления в список элементов – для ввода наименований автомобилей использовать окно prompt. Предоставить возможность оператору для удаления из списка элементов. Если в списке остаётся только один элемент – запретить его удаление. Для удаления / добавления использовать экранные кнопки <button>.

* **Задание 3:**

Реализовать watches в файле с именем index-watches\_Familiya\_I\_O.html;

Размеры обрамлённого элемента для размещения масштабируемой векторной графики <svg> настраиваются двумя элементами <input> в режиме range с единичным шагом в диапазонах 0…640 и 0…480 по ширине и высоте, соответственно. Организовать проверку и запретить превышение совокупных размеров <svg> над, в точности, 0.3 мегапикселями.

# Спецификация оборудования, на котором выполнялась работа.

|  |
| --- |
|  |
| Рисунок 1 – Сведения об операционной системе |

Таблица 1 – Спецификация параметров операционной системы

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Характеристика** | **Сведения** |
| **1** | Тип устройства | Стационарный ПК |
| **2** | Модель | Пользовательская сборка |
| **3** | Операционная система | *Microsoft Windows 10* |
| **4** | Процессор | *Intel Core i7* |
| **5** | Оперативная память | 32 Гб |
| **6** | Объём жёсткого диска | 1 Тб |

# Информация об используемом браузере.

|  |
| --- |
|  |
| Рисунок 2 – Версия браузера |

# Технология локального подключения фреймворка Vue.js

Vue.js подключается локально через ссылку на загруженный файл `vue.global.js`. Этот файл добавляется в `<script>` в HTML-документе**:**

<script src="vue.global.js"></script>

# Русифицированные схемы жизненного цикла Vue.js.

|  |
| --- |
|  |
| Рисунок 3 – Русифицированная схема жизненного цикла Vue.js. |

# Таблица соответствия переменных и методов, используемых в web-приложениях

**Задание 1:**

|  |  |  |
| --- | --- | --- |
| Переменная/Хук | Тип | Описание |
| messages | ref | Переменная, содержащая массив сообщений, отображаемых на странице. |
| flgUpdated | ref | Флаг, указывающий, было ли выполнено обновление (обнуляется при каждом клике на кнопку "Обновить"). |
| flgBefore | ref | Флаг, указывающий, было ли выполнено действие перед обновлением (обнуляется при каждом клике). |
| toUpdate | method | Метод, добавляющий сообщение о клике на кнопку "Обновить" и сбрасывающий флаги flgUpdated и flgBefore. |
| toDestroy | method | Метод, размонтирующий приложение Vue (appInstance.unmount()). |
| onBeforeMount | hook | Хук, вызываемый перед монтированием компонента. Добавляет соответствующее сообщение в messages. |
| onMounted | hook | Хук, вызываемый после монтирования компонента. Добавляет соответствующее сообщение в messages. |
| onBeforeUpdate | hook | Хук, вызываемый перед обновлением компонента. Добавляет соответствующее сообщение в messages, если флаг flgBefore не установлен. |
| onUpdated | hook | Хук, вызываемый после обновления компонента. Добавляет соответствующее сообщение в messages, если флаг flgUpdated не установлен. |
| onBeforeUnmount | hook | Хук, вызываемый перед размонтированием компонента. Добавляет соответствующее сообщение в messages и выводит его в консоль. |
| onUnmounted | hook | Хук, вызываемый после размонтирования компонента. Выводит соответствующее сообщение в консоль. |

**Задание 2:**

|  |  |  |
| --- | --- | --- |
| Переменная/Метод | Тип | Описание |
| cars | ref | Переменная, содержащая массив строк, представляющих список автомобилей. |
| newCar | ref | Переменная, содержащая строку с названием нового автомобиля, вводимого пользователем. |
| message | ref | Переменная, содержащая сообщение об ошибке или другой информации для пользователя. |
| addCar | method | Метод для добавления нового автомобиля в массив cars. Проверяет, чтобы название автомобиля не было пустым. |
| removeCar | method | Метод для удаления автомобиля из массива cars. Проверяет, чтобы в массиве оставался хотя бы один элемент. |

**Задание 3:**

|  |  |  |
| --- | --- | --- |
| Переменная/Метод | Тип | Описание |
| width | ref | Переменная, хранящая текущую ширину SVG элемента в пикселях. |
| height | ref | Переменная, хранящая текущую высоту SVG элемента в пикселях. |
| errorMessage | ref | Переменная, хранящая сообщение об ошибке (если размеры SVG превышают допустимый лимит). |
| MAX\_PIXELS | constant | Константа, задающая максимальное количество пикселей (0.3 мегапикселя). |
| updateSVG | method | Метод для обновления размеров SVG элемента и проверки на превышение максимального количества пикселей. Если превышение обнаружено, сбрасывает размеры и устанавливает сообщение об ошибке. |
| resetValues | method | Метод для сброса значений ширины и высоты SVG элемента до нуля и очистки сообщения об ошибке. |

# Содержательная часть по виткам жизненного цикла

|  |
| --- |
| <!DOCTYPE html>  <html lang="ru">  <head>  <meta charset="UTF-8">  <title>index-hooks\_Familiya\_I\_O</title>  <script src="vue.global.js"></script>  <style>  body {  font-family: Arial, sans-serif;  }  #appVue {  margin: 20px;  }  button {  margin: 5px;  }  </style>  </head>  <body>  <div id="appVue">  <h1>Витки жизненного цикла</h1>  <p v-for="(m, index) in messages" :key="index">{{ m }}</p>  <button @click="toUpdate">Обновить</button>  <button @click="toDestroy">Уничтожить</button>  </div>  <script>  const { createApp, ref, onBeforeMount, onMounted, onBeforeUpdate, onUpdated, onBeforeUnmount, onUnmounted } = Vue;  const App = {  setup() {  const messages = ref([]);  const flgUpdated = ref(false);  const flgBefore = ref(false);  const toUpdate = () => {  flgUpdated.value = false;  flgBefore.value = false;  messages.value.push("Нажата кнопка 'Обновить'");  };  const toDestroy = () => {  appInstance.unmount();  };  onBeforeMount(() => {  const now = new Date();  messages.value.push("Перед монтажом объекта: " + now + " мс: " + now.getMilliseconds());  });  onMounted(() => {  const now = new Date();  messages.value.push("Объект смонтирован: " + now + " мс: " + now.getMilliseconds());  });  onBeforeUpdate(() => {  if (!flgBefore.value) {  const now = new Date();  messages.value.push("Перед обновлением объекта: " + now + " мс: " + now.getMilliseconds());  flgBefore.value = true;  }  });  onUpdated(() => {  if (!flgUpdated.value) {  const now = new Date();  messages.value.push("Объект обновлён: " + now + " мс: " + now.getMilliseconds());  flgUpdated.value = true;  }  });  onBeforeUnmount(() => {  const now = new Date();  console.log("Перед уничтожением объекта: " + now + " мс: " + now.getMilliseconds());  messages.value.push("Перед уничтожением объекта: " + now + " мс: " + now.getMilliseconds());  });  onUnmounted(() => {  const now = new Date();  console.log("Объект уничтожен: " + now + " мс: " + now.getMilliseconds());  });  return {  messages,  toUpdate,  toDestroy  };  }  };  const appInstance = createApp(App);  appInstance.mount('#appVue');  </script>  </body>  </html> |
| Код программы Задание 1 |

|  |
| --- |
|  |
| Рисунок 4 – Задание 1 |
|  |
| Рисунок 5 – Задание 1 изменённое |
|  |
| Рисунок 6 – Сеть Петри Задание 1 |

# Содержательная часть по циклической отрисовке

|  |
| --- |
| <!DOCTYPE html>  <html lang="ru">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>index-v-for\_Familiya\_I\_O</title>  <script src="vue.global.js"></script>  <style>  body {  font-family: Arial, sans-serif;  display: flex;  flex-direction: column;  align-items: center;  justify-content: center;  height: 100vh;  margin: 0;  background-color: #f9f9f9;  }  button {  margin: 5px;  padding: 10px 15px;  border: none;  border-radius: 4px;  background-color: #007BFF;  color: white;  cursor: pointer;  transition: background-color 0.3s;  }  button:hover {  background-color: #0056b3;  }  ol {  margin-top: 20px;  padding: 0;  list-style-type: none;  }  li {  padding: 5px;  background: #e7e7e7;  margin: 5px 0;  border-radius: 4px;  }  input {  padding: 10px;  margin: 5px;  border: 1px solid #ccc;  border-radius: 4px;  width: 200px;  }  </style>  </head>  <body>  <div id="app">  <h1>Список автомобилей</h1>  <input v-model="newCar" placeholder="Введите название автомобиля" />  <div>  <button @click="addCar">Добавить автомобиль</button>  <button @click="removeCar">Удалить автомобиль</button>  </div>  <ol>  <li v-for="(car, index) in cars" :key="index">{{ car }}</li>  </ol>  <div v-if="message" style="color: red;">{{ message }}</div>  </div>  <script>  const { createApp, ref } = Vue;  createApp({  setup() {  const cars = ref([]); // Массив для хранения автомобилей  const newCar = ref(''); // Переменная для нового автомобиля  const message = ref(''); // Переменная для сообщений  // Функция для добавления автомобиля  const addCar = () => {  if (newCar.value.trim()) {  cars.value.push(newCar.value.trim()); // Добавляем автомобиль в массив  newCar.value = ''; // Очищаем поле ввода  message.value = ''; // Убираем сообщение  } else {  message.value = 'Введите название автомобиля!';  }  };  // Функция для удаления автомобиля  const removeCar = () => {  if (cars.value.length > 1) {  const carToRemove = prompt('Введите название автомобиля для удаления:');  if (carToRemove) {  const index = cars.value.indexOf(carToRemove.trim());  if (index >= 0) {  cars.value.splice(index, 1); // Удаляем автомобиль из массива  message.value = ''; // Убираем сообщение  } else {  message.value = 'Автомобиль не найден!';  }  }  }    else {  message.value = 'Запрещено удалять последний элемент!'; // Предупреждение при попытке удалить последний элемент  }  };  return {  cars,  newCar,  message,  addCar,  removeCar  };  }  }).mount('#app');  </script>  </body>  </html> |
| Код программы Задание 2 |

|  |
| --- |
|  |
| Рисунок 7 – Задание 2 |
|  |
| Рисунок 8 – Задание 2 изменённое |
|  |
| Рисунок 9 – Сеть Петри к Заданию 2 |

# Содержательная часть по наблюдателям

|  |
| --- |
| <!DOCTYPE html>  <html lang="ru">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>index-watches\_Familiya\_I\_O</title>  <script src="vue.global.js"></script>  <style>  body {  font-family: Arial, sans-serif;  display: flex;  flex-direction: column;  align-items: center;  justify-content: center;  height: 100vh;  margin: 0;  background-color: #f9f9f9;  }  svg {  border: 1px solid black;  margin-top: 20px;  }  label {  display: inline-block;  margin-top: 10px;  }  input[type="range"] {  width: 300px;  }  .error {  color: red;  margin-top: 10px;  }  button {  padding: 10px 15px;  border: none;  border-radius: 4px;  background-color: #007BFF;  color: white;  cursor: pointer;  transition: background-color 0.3s;  margin-top: 10px;  }  button:hover {  background-color: #0056b3;  }  </style>  </head>  <body>  <div id="app">  <h1>Настройка размеров SVG</h1>  <div>  <label for="widthInput">Ширина (0-640): {{ width }} px</label>  <input id="widthInput" type="range" v-model="width" @input="updateSVG" min="0" max="640" />  <label for="heightInput">Высота (0-480): {{ height }} px</label>  <input id="heightInput" type="range" v-model="height" @input="updateSVG" min="0" max="480" />  <button @click="resetValues">Сбросить значения</button>  </div>  <svg :width="width" :height="height"></svg>  <div class="error" v-if="errorMessage">{{ errorMessage }}</div>  </div>  <script>  const { createApp, ref } = Vue;  createApp({  setup() {  const width = ref(0);  const height = ref(0);  const errorMessage = ref('');  const MAX\_PIXELS = 0.3 \* 1024 \* 1024; // 0.3 мегапикселя в пикселях  const updateSVG = () => {  const totalPixels = width.value \* height.value;  if (totalPixels > MAX\_PIXELS) {  errorMessage.value = 'Ошибка: Совокупные размеры не могут превышать 0.3 мегапикселя!';  width.value = 0; // Сброс ширины  height.value = 0; // Сброс высоты  } else {  errorMessage.value = ''; // Убираем сообщение об ошибке  }  };  const resetValues = () => {  width.value = 0;  height.value = 0;  errorMessage.value = '';  };  return {  width,  height,  errorMessage,  updateSVG,  resetValues  };  }  }).mount('#app');  </script>  </body>  </html> |
| Код программы Задание 3 |

|  |
| --- |
|  |
| Рисунок 10 – Задание 3 |
|  |
| Рисунок 11 – Задание 3 изменённое |
|  |
| Рисунок 12 – Сеть Петри к Заданию 3 |

# Вывод

В данной работе были приобретены навыки по созданию локальных веб-приложений на Vue.js, изучены основные понятия, принципы и методы работы с жизненными циклами, наблюдателями, директивами. Было выполнено создание локального приложения без использования Node.js и npm, с акцентом на локальное подключение фреймворка Vue.js. Поставленные задачи выполнены в полном объёме.