

**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**

«**Линейный и разветвляющийся вычислительные процессы**»

**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](#_Toc180318033)

[Постановка задачи 3](#_Toc180318034)

[Технология локального подключения фреймворка Vue.js 4](#_Toc180318035)

[Таблица соответствия переменных и методов, используемых в web-приложениях 5](#_Toc180318036)

[Содержательная часть по параметрической вставке 6](#_Toc180318037)

[Содержательная часть по параметрической связке 9](#_Toc180318038)

[Содержательная часть по методам и событиям 12](#_Toc180318039)

[Содержательная часть по условной отрисовке 16](#_Toc180318040)

[Вывод 24](#_Toc180318041)

# Цель работы

Цель работы заключается в создании локальных web-приложений на Vue.js с использованием параметрических вставок, связок, методов и событий, а также условной отрисовки. Разработка будет выполнена без использования Node.js и npm, с акцентом на локальное подключение фреймворка Vue.js.

# Постановка задачи

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

* **Задание 1:**

в файле с именем index01\_Familiya\_I\_O1\_Familiya\_I\_O2.html параметрическую вставку;  
  
Настроить при помощи параметрической подстановки тег таблицы <table></table>. В таблице содержатся две строки <tr></tr> и два столбца <td></td>. В первом столбце ячейки заполняются с подстановкой в «усатых скобках» по шаблону: «Ширина: [значение подставляемого параметра]», «Внеш. отступ: [значение подставляемого параметра]». Во втором столбце ячейки заполняются с подстановкой в «усатых скобках» по шаблону: «Высота: [значение подставляемого параметра]», «Внутр. отступ: [значение подставляемого параметра]». Параметры тега, принимающие подстановки, height, width, cellspacing и cellpadding принять равными, соответственно: 100, 400, 10, 20 пикселей. Ширина рамки таблицы задаётся так же, параметрически, толщиной в 1 пиксель.

* **Задание 2:**

в файле с именем index02\_Familiya\_I\_O1\_Familiya\_I\_O2.html параметрическую связку;

Разместить на web-странице элемент ввода данных <input></input> с начальным состоянием режима, подставляемого по директиве «v-bind» – им должен быть «text» вне зависимости, что он задаётся для элемента по умолчанию. Рядом разместить экранную кнопку <button></button>, которая последовательно, в любом порядке по нажатии по ней кнопкой мыши реализует перебор всех режимов элемента ввода данных <input> («password», «image», «button», «checkbox» и т.д, в том числе новые, появившиеся в рамках стандарта html5 режимы, например, range). По достижении последнего из известных режимов состояние элемента для ввода данных вновь возвращается в исходное – «text».

* **Задание 3:**

в файле с именем index03\_Familiya\_I\_O1\_Familiya\_I\_O2.html работу с методами и событиями;

В обрамлённой пунктиром области масштабируемой векторной графики <svg></svg> размером 640 на 480 пикселей по ширине и по высоте, соответственно, в псевдослучайно сгенерированной позиции появляется изображение квадрата размером 5 на 5 пикселей. Стрелками курсора на клавиатуре позволить оператору менять с шагом в 5 пикселей положение квадрата в соответствующем стрелке курсора направлении. Не позволять квадрату подступать к границам области <svg></svg> ближе, чем на 2 пикселя даже при инициализации его начального положения.

* **Задание 4:**

в файле с именем index04\_Familiya\_I\_O1\_Familiya\_I\_O2.html условную отрисовку.

Смоделировать работу автомобильного светофора. На протяжении некоторого времени горит «зелёный» сигнал, затем непродолжительное фиксированное время горит «жёлтый» сигнал и, на протяжении некоторого третьего времени, горит «красный» сигнал. В разворачиваемых по галочке (флагу) настройках иметь возможность настраивать времена горения «зелёного» и «красного» сигналов светофора. Предусмотреть режимы управления «включён» / «выключен», активируемых двумя радиокнопками.

# 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>

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

**Задание 1:**

|  |  |
| --- | --- |
| **Переменные и методы** | **Описание** |
| width | Ширина таблицы в пикселях |
| height | Высота таблицы в пикселях |
| cellspacing | Внешний отступ таблицы в пикселях |
| cellpadding | Внутренний отступ таблицы в пикселях |
| tableStyle | Стиль таблицы в виде строки CSS |
| createApp | Функция, создающая новое приложение Vue |
| ref | Функция, создающая реактивную ссылку на значение |
| setup | Функция, вызываемая при создании приложения, в которой определяются переменные и методы |
| mount | Метод, монтирующий приложение в элемент DOM |

**Задание 2:**

|  |  |
| --- | --- |
| **Переменные и методы** | **Описание** |
| inputType | Тип поля ввода |
| inputTypes | Массив типов полей ввода |
| currentIndex | Индекс текущего типа поля ввода |
| changeInputType | Функция, переключающая режим поля ввода |
| createApp | Функция, создающая новое приложение Vue |
| ref | Функция, создающая реактивную ссылку на значение |
| setup | Функция, вызываемая при создании приложения, в которой определяются переменные и методы |
| mount | Метод, монтирующий приложение в элемент DOM |

**Задание 3:**

|  |  |
| --- | --- |
| **Переменные и методы** | **Описание** |
| x | Координата X квадрата в SVG |
| y | Координата Y квадрата в SVG |
| step | Шаг перемещения квадрата |
| moveSquare | Функция, перемещающая квадрат по нажатию клавиш стрелок |
| createApp | Функция, создающая новое приложение Vue |
| ref | Функция, создающая реактивную ссылку на значение |
| onMounted | Функция, вызываемая после монтирования приложения |
| setup | Функция, вызываемая при создании приложения, в которой определяются переменные и методы |
| mount | Метод, монтирующий приложение в элемент DOM |

**Задание 4:**

|  |  |
| --- | --- |
| **Переменные и методы** | **Описание** |
| currentLight | Текущий цвет светофора |
| greenTime | Время, в течение которого горит зеленый цвет |
| yellowTime | Время, в течение которого горит желтый цвет |
| redTime | Время, в течение которого горит красный цвет |
| power | Состояние светофора (включен/выключен) |
| intervalId | ID интервала, по которому происходит смена цвета |
| showSettings | Флаг, показывающий, отображаются ли настройки |
| lightSequence | Последовательность цветов светофора |
| sequenceIndex | Индекс текущего цвета в последовательности |
| togglePower | Метод, переключающий состояние светофора |
| startTrafficLight | Метод, запускающий светофор |
| stopTrafficLight | Метод, останавливающий светофор |
| changeLight | Метод, меняющий цвет светофора |
| toggleSettings | Метод, переключающий отображение настроек |

# Содержательная часть по параметрической вставке

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>index01\_Familiya\_I\_O1\_Familiya\_I\_O2</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;  }  table {  border-collapse: collapse;  margin-bottom: 20px;  }  td {  padding: 10px;  border: 1px solid #ccc;  }  input {  padding: 10px;  margin: 5px;  border: 1px solid #ccc;  border-radius: 4px;  width: 100px;  }  button {  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;  }  </style>  </head>  <body>  <div id="app">  <table :style="tableStyle">  <tr>  <td>Ширина: {{ width }}px</td>  <td>Высота: {{ height }}px</td>  </tr>  <tr>  <td>Внеш. отступ: {{ cellspacing }}px</td>  <td>Внутр. отступ: {{ cellpadding }}px</td>  </tr>  </table>  <div>  <label for="widthInput">Ширина:</label>  <input id="widthInput" type="number" v-model="width" @input="updateTableStyle" />  <label for="heightInput">Высота:</label>  <input id="heightInput" type="number" v-model="height" @input="updateTableStyle" />  <label for="cellspacingInput">Внеш. отступ:</label>  <input id="cellspacingInput" type="number" v-model="cellspacing" @input="updateTableStyle" />  <label for="cellpaddingInput">Внутр. отступ:</label>  <input id="cellpaddingInput" type="number" v-model="cellpadding" @input="updateTableStyle" />  <button @click="resetValues">Сбросить значения</button>  </div>  </div>  <script>  const { createApp, ref, watch } = Vue;  createApp({  setup() {  const width = ref(200);  const height = ref(100);  const cellspacing = ref(20);  const cellpadding = ref(20);  const tableStyle = ref(`width: ${width.value}px; height: ${height.value}px; border-spacing: ${cellspacing.value}px; padding: ${cellpadding.value}px;`);  const updateTableStyle = () => {  tableStyle.value = `width: ${width.value}px; height: ${height.value}px; border-spacing: ${cellspacing.value}px; padding: ${cellpadding.value}px;`;  };  const resetValues = () => {  width.value = 200;  height.value = 100;  cellspacing.value = 20;  cellpadding.value = 20;  updateTableStyle();  };  return {  width,  height,  cellspacing,  cellpadding,  tableStyle,  updateTableStyle,  resetValues  };  }  }).mount('#app');  </script>  </body>  </html> |
| Код программы Задание 1 |

|  |
| --- |
|  |
| Рисунок 1 – Отображение в браузере задание 1 |
|  |
| Рисунок 2 – Отображение в браузере задания 1 в изменённом состоянии |

# Содержательная часть по параметрической связке

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>index02\_Familiya\_I\_O1\_Familiya\_I\_O2</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;  }  input {  padding: 10px;  margin-bottom: 20px;  border: 1px solid #ccc;  border-radius: 4px;  width: 200px;  }  button {  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;  }  </style>  </head>  <body>  <div id="app">  <input :type="inputType" placeholder="Введите данные" />  <button @click="changeInputType">Переключить режим</button>  <p>Текущий тип ввода: {{ inputType }}</p>  </div>  <script>  const { createApp, ref } = Vue;  createApp({  setup() {  const inputType = ref("text");  const inputTypes = ["text", "password", "button", "checkbox", "color", "date", "email", "file", "image", "number", "range", "search", "tel", "url"];  let currentIndex = 0;  const changeInputType = () => {  currentIndex = (currentIndex + 1) % inputTypes.length;  inputType.value = inputTypes[currentIndex];  };  return {  inputType,  changeInputType  };  }  }).mount('#app');  </script>  </body>  </html> |
| Код программы Задание 2 |

|  |
| --- |
|  |
| Рисунок 3 – Отображение в браузере задания 2 |
|  |
| Рисунок 4 – Отображение в браузере задания 2 в изменённом состоянии |

# Содержательная часть по методам и событиям

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>index03\_Familiya\_I\_O1\_Familiya\_I\_O2</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 dotted #ccc;  margin-bottom: 20px;  }  .controls {  display: flex;  flex-direction: column;  align-items: center;  }  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;  }  p {  margin: 5px 0;  }  </style>  </head>  <body>  <div id="app" tabindex="0"> <!-- Add tabindex to make the div focusable -->  <svg width="640" height="480">  <rect :x="x" :y="y" width="5" height="5" fill="blue"></rect>  </svg>  <div class="controls">  <p>Используйте стрелки на клавиатуре для перемещения квадрата.</p>  <p>Текущие координаты: X: {{ x }}, Y: {{ y }}</p>  <button @click="resetPosition">Сбросить позицию</button>  </div>  </div>  <script>  const { createApp, ref, onMounted, onBeforeUnmount } = Vue;  createApp({  setup() {  const x = ref(Math.floor(Math.random() \* (640 - 7)) + 2);  const y = ref(Math.floor(Math.random() \* (480 - 7)) + 2);  const step = 5;  const moveSquare = (event) => {  if (event.key === "ArrowLeft" && x.value > 2) {  x.value -= step;  }  if (event.key === "ArrowRight" && x.value < 640 - 7) {  x.value += step;  }  if (event.key === "ArrowUp" && y.value > 2) {  y.value -= step;  }  if (event.key === "ArrowDown" && y.value < 480 - 7) {  y.value += step;  }  };  const resetPosition = () => {  x.value = Math.floor(Math.random() \* (640 - 7)) + 2;  y.value = Math.floor(Math.random() \* (480 - 7)) + 2;  };  onMounted(() => {  const appElement = document.getElementById('app');  appElement.addEventListener('keydown', moveSquare);  appElement.focus(); // Set focus on the app element to capture key events  });  onBeforeUnmount(() => {  const appElement = document.getElementById('app');  appElement.removeEventListener('keydown', moveSquare);  });  return {  x,  y,  resetPosition  };  }  }).mount('#app');  </script>  </body>  </html> |
| Код программы Задание 3 |

|  |
| --- |
|  |
| Рисунок 5 – Отображение в браузере задания 3 |
|  |
| Рисунок 6 – Отображение в браузере задания 3 в изменённом состоянии |
|  |
| Рисунок 7 – Сеть Петри к заданию 3 |

# Содержательная часть по условной отрисовке

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>index04\_Familiya\_I\_O1\_Familiya\_I\_O2</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;  }  .traffic-light {  width: 100px;  height: 300px;  background-color: #333;  border-radius: 10px;  padding: 10px;  box-sizing: border-box;  display: flex;  flex-direction: column;  align-items: center;  justify-content: space-around;  }  .light {  width: 80px;  height: 80px;  border-radius: 50%;  background-color: #555;  transition: background-color 0.5s;  }  .light.green {  background-color: green;  }  .light.yellow {  background-color: yellow;  }  .light.red {  background-color: red;  }  .controls {  text-align: center;  margin-top: 20px;  }  .settings {  display: none;  margin-top: 20px;  }  .settings.active {  display: block;  }  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;  }  label {  display: block;  margin: 10px 0;  }  </style>  </head>  <body>  <div id="app">  <div class="traffic-light">  <div :class="['light', currentLight === 'red' ? 'red' : '']"></div>  <div :class="['light', currentLight === 'yellow' ? 'yellow' : '']"></div>  <div :class="['light', currentLight === 'green' ? 'green' : '']"></div>  </div>  <div class="controls">  <label>  <input type="radio" value="on" v-model="power" @change="togglePower"> Включен  </label>  <label>  <input type="radio" value="off" v-model="power" @change="togglePower"> Выключен  </label>  <button @click="toggleSettings">Настройки</button>  </div>  <div class="settings" :class="{active: showSettings}">  <h3>Настройки времени</h3>  <label>Время зеленого (сек): <input type="number" v-model.number="greenTime" min="1"></label>  <label>Время красного (сек): <input type="number" v-model.number="redTime" min="1"></label>  </div>  </div>  <script>  const { createApp, ref, onMounted } = Vue;    createApp({  setup() {  const currentLight = ref(''); // Текущий цвет светофора  const greenTime = ref(5); // Время горения зеленого света  const yellowTime = ref(2); // Время горения желтого света  const redTime = ref(5); // Время горения красного света  const power = ref('off'); // Состояние питания (включен/выключен)  const intervalId = ref(null); // Идентификатор интервала для смены света  const showSettings = ref(false); // Показать/скрыть настройки  const lightSequence = ref(['green', 'yellow', 'red', 'yellow']); // Последовательность цветов  const sequenceIndex = ref(0); // Индекс текущего цвета в последовательности    // Функция для переключения питания светофора  const togglePower = () => {  if (power.value === 'on') {  startTrafficLight();  } else {  stopTrafficLight();  }  };    // Функция для запуска светофора  const startTrafficLight = () => {  sequenceIndex.value = 0;  currentLight.value = lightSequence.value[sequenceIndex.value];  changeLight();  };    // Функция для остановки светофора  const stopTrafficLight = () => {  clearInterval(intervalId.value);  intervalId.value = null;  currentLight.value = ''; // Отключить все огни  };    // Функция для смены цвета светофора  const changeLight = () => {  let time;  switch (currentLight.value) {  case 'green':  time = greenTime.value;  break;  case 'yellow':  time = yellowTime.value;  break;  case 'red':  time = redTime.value;  break;  }    // Удаляем предыдущий интервал, если он существует  if (intervalId.value) {  clearInterval(intervalId.value);  }    // Устанавливаем новый интервал для смены цвета  intervalId.value = setInterval(() => {  // Переход к следующему цвету в последовательности  sequenceIndex.value = (sequenceIndex.value + 1) % lightSequence.value.length;  currentLight.value = lightSequence.value[sequenceIndex.value];  changeLight(); // Обновляем цвет  }, time \* 1000);  };    // Функция для переключения видимости настроек  const toggleSettings = () => {  showSettings.value = !showSettings.value;  };    // Возвращаем все переменные и функции, чтобы они были доступны в шаблоне  return {  currentLight,  greenTime,  yellowTime,  redTime,  power,  intervalId,  showSettings,  togglePower,  toggleSettings  };  }  }).mount('#app');  </script>  </body>  </html> |
| Код программы Задание 4 |

|  |
| --- |
|  |
| Рисунок 8 – Отображение в браузере задания 3 |
|  |
| Рисунок 9 – Отображение в браузере задания 4 в изменённом состоянии |
|  |
| Рисунок 10 – Сеть Петри к заданию 4 |

# Вывод

В данной работе были получены навыки по созданию локальных web-приложений на Vue.js, изучены основные понятия, принципы и методы параметрических вставок, связок, методов и событий, а также условной отрисовки. 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.