Министерство транспорта Российской Федерации

Федеральное государственное автономное

образовательное учреждение высшего образования

«Российский университет транспорта (МИИТ)»

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Лабораторная работа №5

по дисциплине:

«Web-программирование»

на тему:  
«Циклы, наблюдатели, витки жизненного цикла»

Выполнил: ст. гр. ТКИ-541

Черкас Н.В., Родин К.П.

Вариант №13

Проверил: к.т.н., доцент Сафронов А.И.

Москва – 2025 г.

# **1. Цель работы**

Целью данной лабораторной работы является приобретение практических навыков разработки одностраничных локальных *web*-приложений (*LSPWA*) под управлением фреймворка *Vue.js* на языке *JavaScript*.

# **2. Формулировка задачи**

Разработать локальные одностраничные *web*-приложение (*LSPWA*) под управлением фреймворка *Vue.js* на языке *JavaScript* в соответствии с указаниями вариантов индивидуального задания.

*Базовая структура отчёта по работе:*

1. Цель работы.

2. Формулировки задач.

2.1. Задача на витки жизненного цикла (*hooks*).

2.2. Задача на циклы (*v-for*).

2.3. Задача с наблюдателями (*watch*).

3. Спецификация оборудования, на котором выполнялась работа.

4. Информация об используемом браузере.

5. Технология локального подключения фреймворка *Vue.js.*

6. Схемы жизненного цикла.

6.1. Русифицированная схема жизненного цикла *Vue.js* из официальной документации.

6.2. Русифицированная схема жизненного цикла *Vue.js* с точки зрения ТП согласно нотации, определённой рабочим коллективом.

7. Таблица соответствия переменных и методов, используемых в *web*-приложении.

8. Содержательная часть

8.1. Витки жизненного цикла (код *web*-приложения + отображение в браузере + схема ТП согласно нотации, определённой рабочим коллективом).

8.2. Циклическая отрисовка (код *web*-приложения + отображение в браузере + схема ТП согласно нотации, определённой рабочим коллективом).

8.3. Наблюдатели (код *web*-приложения + отображение в браузере + схема ТП согласно нотации, определённой рабочим коллективом).

9. Вывод.

***Реализовать:***

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

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

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

– витки жизненного цикла в файле с именем *index-hooks\_Familiya\_I\_O.html.*

***Комплектность отчёта:***

– *Word*-документ согласно «Базовой структуре отчёта»;

– перечень *web*-страниц и приложений к ним согласно разделу «Реализовать».

**Вариант для приложения 5.1:**

Реализовать схему тестирования и демонстрации работы всех витков / хуков / методов жизненного цикла фреймворка *Vue.js* в формате одностраничного *web*-приложения согласно варианту индивидуального задания по разделу [5.1].

Приветствуется реализация схемы тестирования под управлением *Vue.js 3.x* (**Приложение 1**), однако основной для выполнения лабораторной работы является версия *Vue.js 2.x*, поскольку большая часть лекционных материалов содержит примеры именно для неё.

**Примечание:** если в задании явно указана разработка как на *Vue.js 3.x*, так и на *Vue.js 2.x*, то реализовать обе.

Привести в отчёте русифицированную схему жизненного цикла *Vue.js 3.x / Vue.js 2.x.* Рекомендуется провести пофрагментрую аналогию с готовой схемой из официальной документации.

Близкая реализация *web*-приложения для тестирования под *Vue.js 2.x* приведена в **Приложении 2**.

Привести информацию об используемом браузере по аналогии с иллюстрациями, представленными в **Приложении 3**.

Составить таблицу соответствия переменных и методов, используемых в *web*-приложении. Структура таблицы соответствия представлена в **Приложении 4**.

Заполнить спецификацию оборудования, используемого для выполнения работы **Приложение 5**.

**Не использовать:** *jQuerry*, «чистый» *JavaScript* (**внимание:** в некоторых вариантах**\*** без него не обойтись) (**Приложение 2**).

**Реализовать:** локально, не прибегая к инструментарию *Node.js* и *npm* (*Node Package Manager*) – раскрыть данный вопрос в пункте 5 отчёта.

*Рекомендации:* для отработки отдельных витков / хуков / методов жизненного цикла неизбежно потребуется воспользоваться встроенной браузерной консолью.

Эффективно использовать и продемонстрировать работу обновляющих витков жизненного цикла экземпляра *Vue.js* таким образом, чтобы при попадании в такой виток при отсутствии каких-либо изменений в контролируемом значении обработка в рамках *updated* не производилась, а в случае детектирования любых изменений контролируемого значения – производилась в *updated* незамедлительно.

**Вариант для приложения 5.2:**

Изобразить внутри обрамлённой области масштабируемой векторной графики <*svg*> с фиксированными значениями ширины и высоты функцию вида *y(x) = sin(x)* окружностями <*circle*> настраиваемого радиуса (2 – 10 пикселей с единичным шагом в <*input*> в режиме *range*) на белом фоне без осей. Предоставить оператору возможность масштабирования с единичным шагом функции в фиксированном диапазоне 10%–100% через <*input*> в режиме *range*.

**Вариант для приложения 5.3:**

Создать конвертер валют. Любая вводимая величина влияет на остальные. Отдельных кнопок конвертации не предусмотрено. В процессе конвертации участвуют рубли, доллары, евро, йены, юани, рупии. Взять к рассмотрению курс, зафиксированный на момент: 10 января 2024 года. Оформить конвертер таблично.

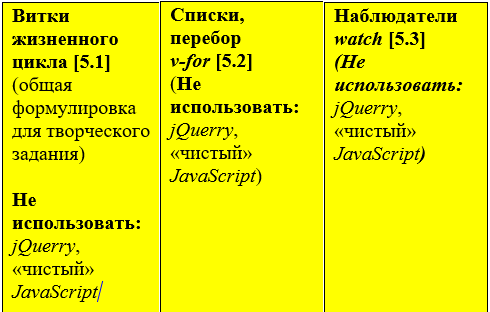
****

Рисунок 1 – Ограничения в жёлтых клетках таблиц

# **3. Спецификация оборудования, на котором выполнялась работа*.***

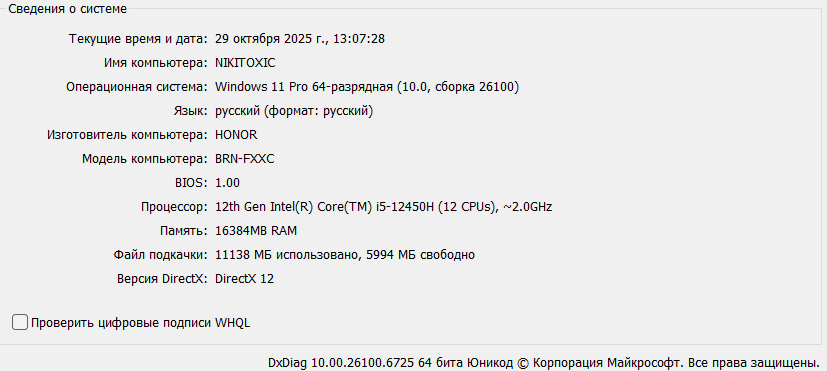


Рисунок 2 – Сведения об ОС и оборудовании, на котором выполнялась работа

# **4. Информация об используемом браузере*.***

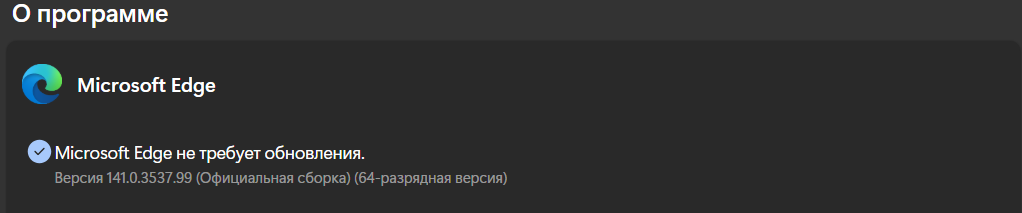


Рисунок 3 – Информация об используемом браузере

# **5. Технология локального (*LSPWA*) подключения фреймворка *Vue.js.***

# 

Рисунок 4 – Технология локального (*LSPWA*) подключения фреймворка *Vue.js.*

**6. Схемы жизненного цикла.**

**6.1 Русифицированная схема жизненного цикла *Vue.js* из официальной документации.**

См. **Приложение 2**

**6.2 Русифицированная схема жизненного цикла *Vue.js* с точки зрения ТП согласно нотации, определённой рабочим коллективом (Сеть Петри).**

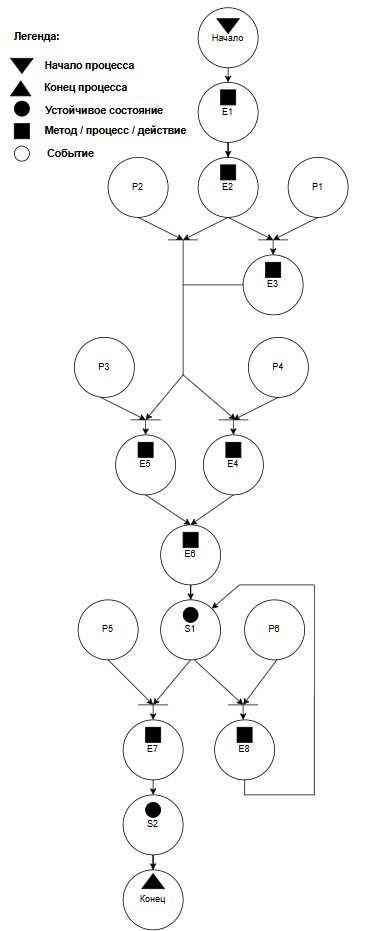
****

Рисунок 5 – Русифицированная схема жизненного цикла *Vue.js* с точки зрения ТП согласно нотации, определённой рабочим коллективом (Сеть Петри)

**Описание сети Петри**

- состояния (*states*)

s1 – экземпляр смонтирован

s2 – экземпляр уничтожен

- действия (*effects*)

e1 – инициализация событий и жизненного цикла

e2 – инициализация инъекций и системы реактивности

e3 – ожидание vm.$mount(el)

e4 – компиляция шаблона в рендер функцию

e5 – компиляция outerHTML из el как шаблона

e6 – создание vm.$el и замена им содержимого «el»

e7 – удаление наблюдателей, дочерних компонентов слушателей событий

e8 – обновление виртуального DOM и применение изменений

- события (*prompts*)

p1 – опция «el» есть

p2 – опции «el» нет

p3 – опция «template» есть

p4 – опции «template» нет

p5 – вызван vm.$destroy()

p6 – изменены данные

**7. Таблица соответствия переменных и методов, используемых в *web*-приложениях.**

В ходе выполнения работы были использованы «говорящие» переменные.

# **8. Содержательная часть.**

**8.1 Витки жизненного цикла**

# **8.1.1 Составление сети Петри технологического процесса внутри веб-страницы приложения 1.**

# 

Рисунок 6 – Сеть Петри технологического процесса веб-страницы приложения 1

**Описание сети Петри**

- состояния (*states*)

s1 – форма ожидает действий пользователя

- действия (*effects*)

e1 – Произведение обработки в рамках updated

e2 – закрытие формы

- события (*prompts*)

p1 – Изменение текущего значения счётчика

p2 – Нажата кнопка закрытия веб-страницы

# **8.1.2 Код веб-страницы приложения 1.**

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<title>Витки жизненного цикла – Черкас Н.В. | Родин К.П.</title>

<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>

<style>

body { font-family: Arial, sans-serif; margin: 20px; }

button { margin-top: 10px; }

.log { margin-top: 20px; background: #f0f0f0; padding: 10px; }

</style>

</head>

<body>

<h2>Витки жизненного цикла Vue.js (вариант 13) — Черкас Н.В. | Родин К.П.</h2>

<div id="app">

<p>Текущее значение счётчика: <strong>{{ count }}</strong></p>

<input type="number" v-model.number="count">

<button @click="increment">Увеличить</button>

<div class="log">

<h4>Журнал событий:</h4>

<ul>

<li v-for="msg in logs" :key="msg">{{ msg }}</li>

</ul>

</div>

</div>

<script>

new Vue({

el: '#app',

data() {

return {

count: 0,

lastValue: 0,

logs: []

};

},

methods: {

increment() {

this.count++;

}

},

beforeUpdate() {

console.log('beforeUpdate — проверка изменений');

},

updated() {

if (this.count !== this.lastValue) {

this.logs.push(`updated: count изменился с ${this.lastValue} на ${this.count}`);

console.log(`updated — count изменился с ${this.lastValue} на ${this.count}`);

this.lastValue = this.count;

} else {

console.log('updated вызван, но изменений нет — обработка пропущена');

}

},

created() {

this.logs.push('created — экземпляр создан');

},

mounted() {

this.logs.push('mounted — экземпляр смонтирован');

this.lastValue = this.count;

}

});

</script>

</body>

</html>

# **8.1.3 Результат отображения веб-страницы приложения 1 в браузере.**

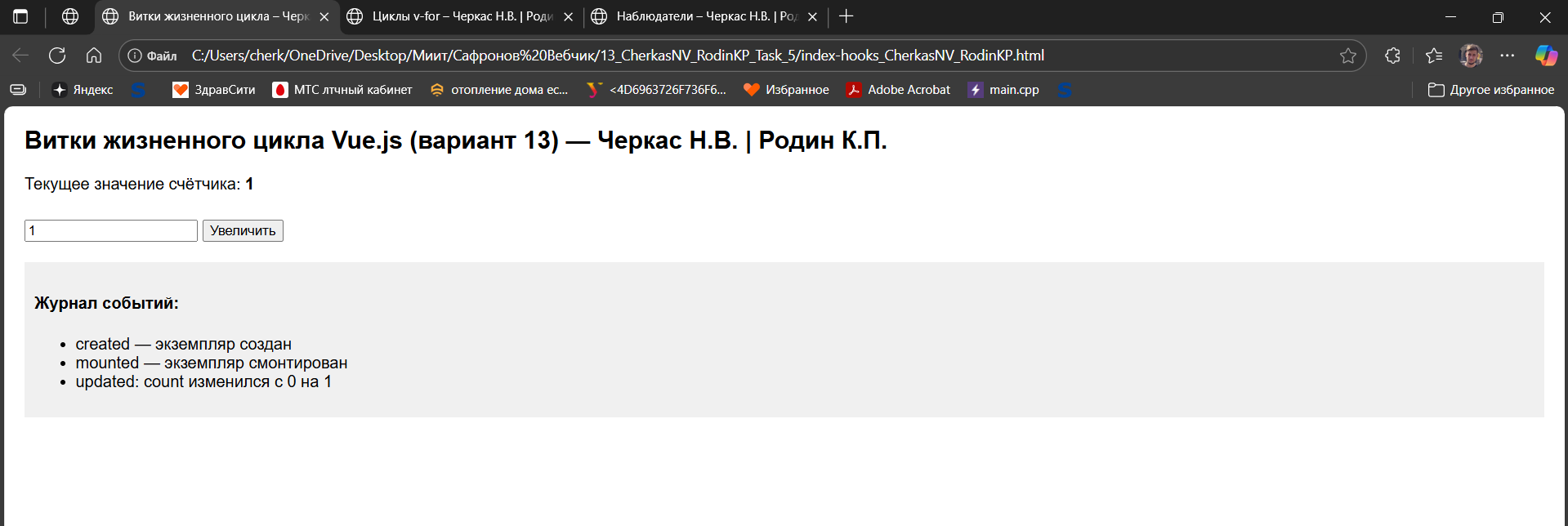


Рисунок 7 – Результат отображения в браузере веб-страницы приложения 1

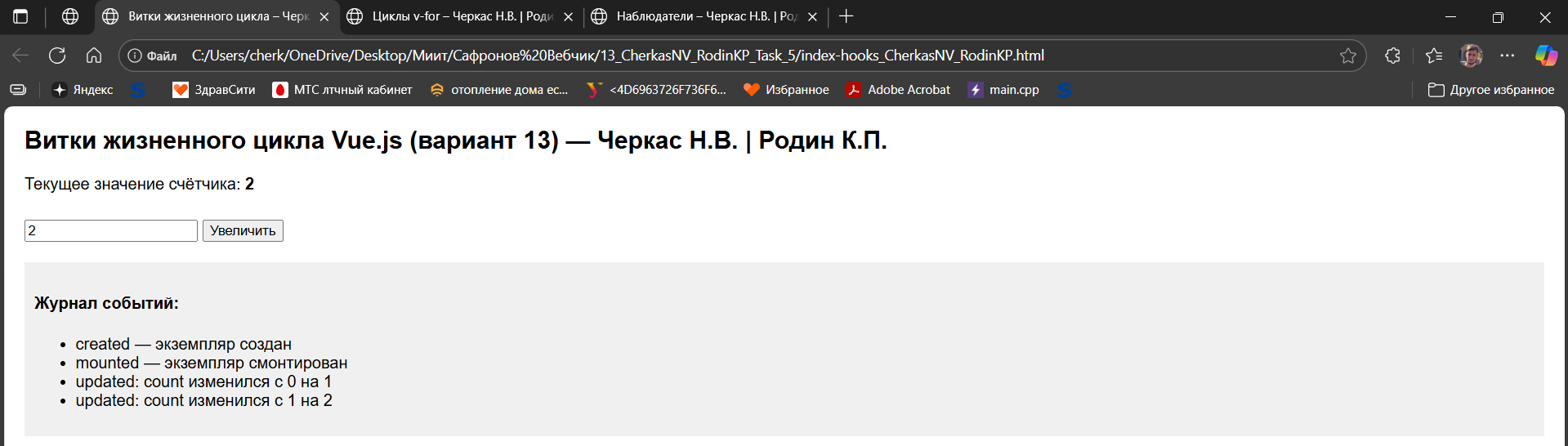


Рисунок 8 – Результат двойного нажатия на кнопку «Увеличить» (увеличение счётчика на 2)

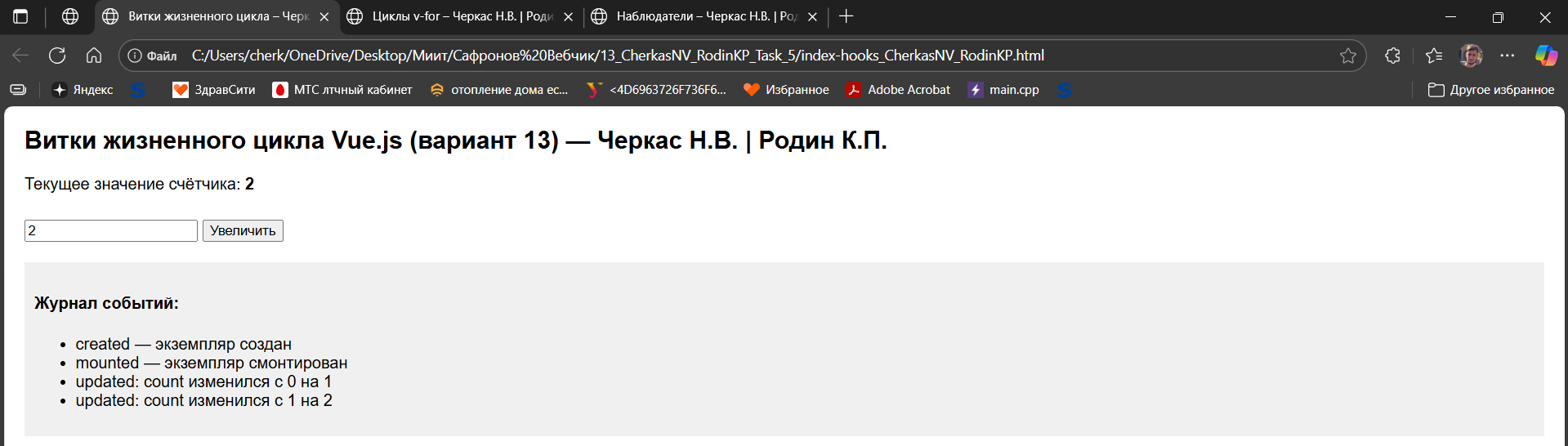


Рисунок 9 – Изменение текущего значения счётчика на то же значение

# **8.2. Методы и события.**

# **8.2.1 Составление сети Петри технологического процесса внутри веб-страницы приложения 2.**

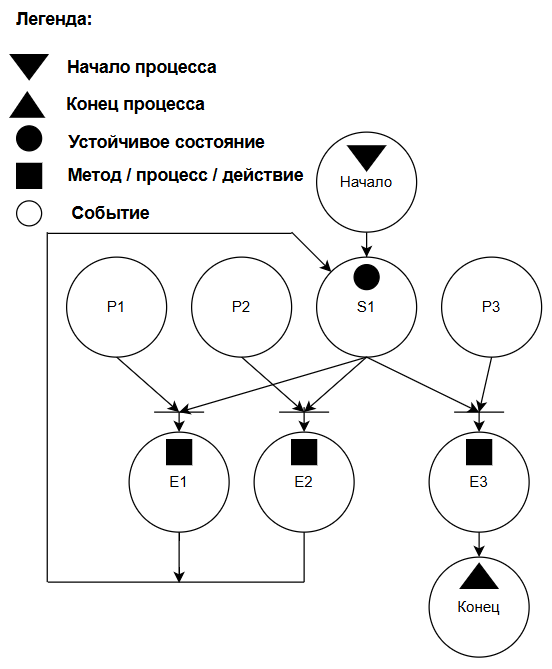


Рисунок 10 – Сеть Петри технологического процесса веб-страницы приложения 2

**Описание сети Петри**

- состояния (*states*)

s1 – форма ожидает действий пользователя

- действия (*effects*)

e1 – изменение радиуса точек согласно новому значению радиуса точек

e2 – изменение масштаба функции согласно новому значению масштаба

e3 – закрытие формы

- события (*prompts*)

p1 – изменён радиус точек

p2 – изменен масштаб функции

p3 – нажата кнопка закрытия веб-страницы

**8.2.2 Код веб-страницы приложения 2.**

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<title>Циклы v-for – Черкас Н.В. | Родин К.П.</title>

<script src="https://unpkg.com/vue@3/dist/vue.global.js"></script>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

background: #f5f5f5;

}

.container {

max-width: 900px;

margin: 0 auto;

background: white;

padding: 20px;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0,0,0,0.1);

}

.controls {

margin: 20px 0;

padding: 15px;

background: #f8f9fa;

border-radius: 5px;

}

.control-group {

margin: 10px 0;

}

label {

display: inline-block;

width: 150px;

font-weight: bold;

}

input[type="range"] {

width: 300px;

margin: 0 10px;

}

.value-display {

display: inline-block;

width: 50px;

text-align: center;

font-weight: bold;

}

.svg-container {

border: 2px solid #333;

border-radius: 5px;

background: white;

padding: 10px;

margin: 20px 0;

}

h2 {

text-align: center;

color: #2c3e50;

}

</style>

</head>

<body>

<div class="container">

<h2>Циклическая отрисовка функции y(x) = sin(x) — Черкас Н.В. | Родин К.П.</h2>

<div id="app">

<div class="controls">

<div class="control-group">

<label>Радиус точек:</label>

<input type="range" min="2" max="10" step="1" v-model.number="radius">

<span class="value-display">{{ radius }} px</span>

</div>

<div class="control-group">

<label>Масштаб функции:</label>

<input type="range" min="10" max="100" step="1" v-model.number="scale">

<span class="value-display">{{ scale }}%</span>

</div>

</div>

<div class="svg-container">

<svg :width="width" :height="height">

<!-- Отрисовка функции sin(x) окружностями -->

<circle

v-for="(point, index) in points"

:key="index"

:cx="point.x"

:cy="point.y"

:r="radius"

fill="#3498db"

:opacity="0.7"

/>

</svg>

</div>

<div style="text-align: center; margin-top: 10px;">

<small>Функция: y(x) = sin(x) | Диапазон: x ∈ [-2π, 2π]</small>

</div>

</div>

</div>

<script>

const { createApp, ref, computed } = Vue;

createApp({

setup() {

const width = ref(800);

const height = ref(400);

const radius = ref(5); // начальный радиус 5px

const scale = ref(50); // начальный масштаб 50%

const points = computed(() => {

const pointsArray = [];

const centerY = height.value / 2;

const amplitude = (height.value / 2 - 20) \* (scale.value / 100);

// Отрисовываем 2 периода синусоиды (-2π до 2π)

const xMin = -2 \* Math.PI;

const xMax = 2 \* Math.PI;

const step = (xMax - xMin) / 200; // 200 точек для гладкого отображения

for (let i = 0; i <= 200; i++) {

const xValue = xMin + i \* step;

const yValue = Math.sin(xValue);

// Преобразуем математические координаты в SVG координаты

const x = ((xValue - xMin) / (xMax - xMin)) \* (width.value - 40) + 20;

const y = centerY - yValue \* amplitude;

pointsArray.push({ x, y });

}

return pointsArray;

});

return {

width,

height,

radius,

scale,

points

};

}

}).mount('#app');

</script>

</body>

</html>

**8.2.3 Результат отображения веб-страницы приложения 2.**

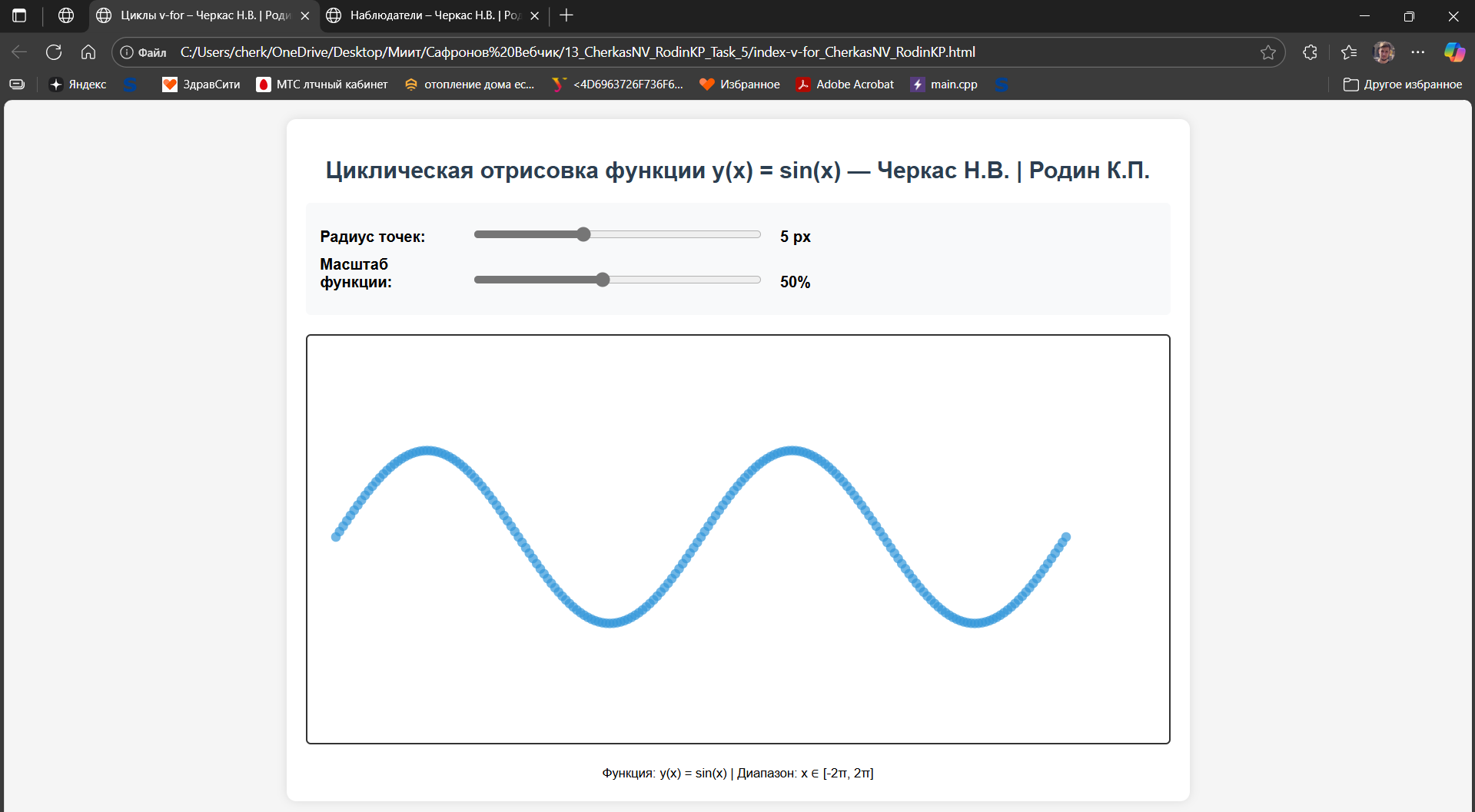


Рисунок 11 – Результат отображения в браузере веб-страницы приложения 2

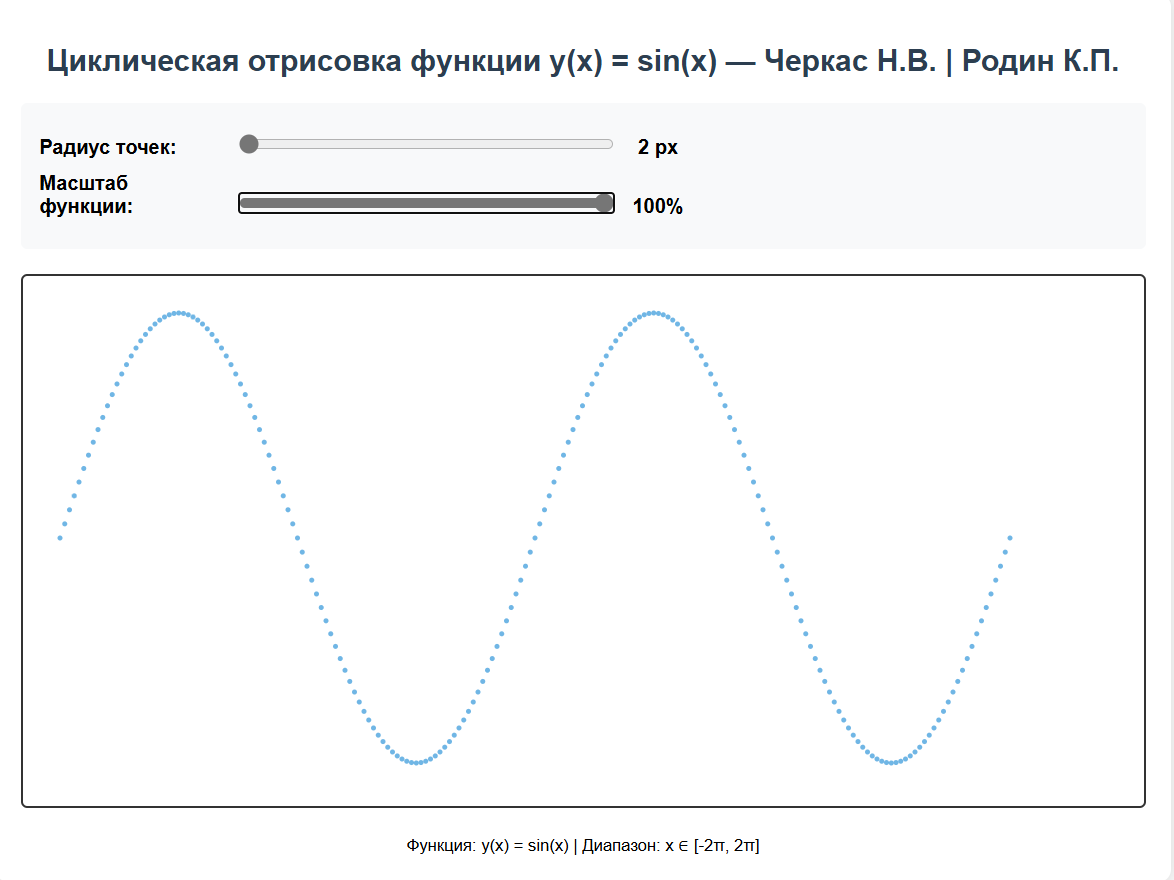


Рисунок 12 – Результат изменения значения радиуса точек и изменения значения масштаба функции

# **8.3 Параметрическая связка.**

# **8.3.1 Составление сети Петри технологического процесса внутри веб-страницы приложения 3.**

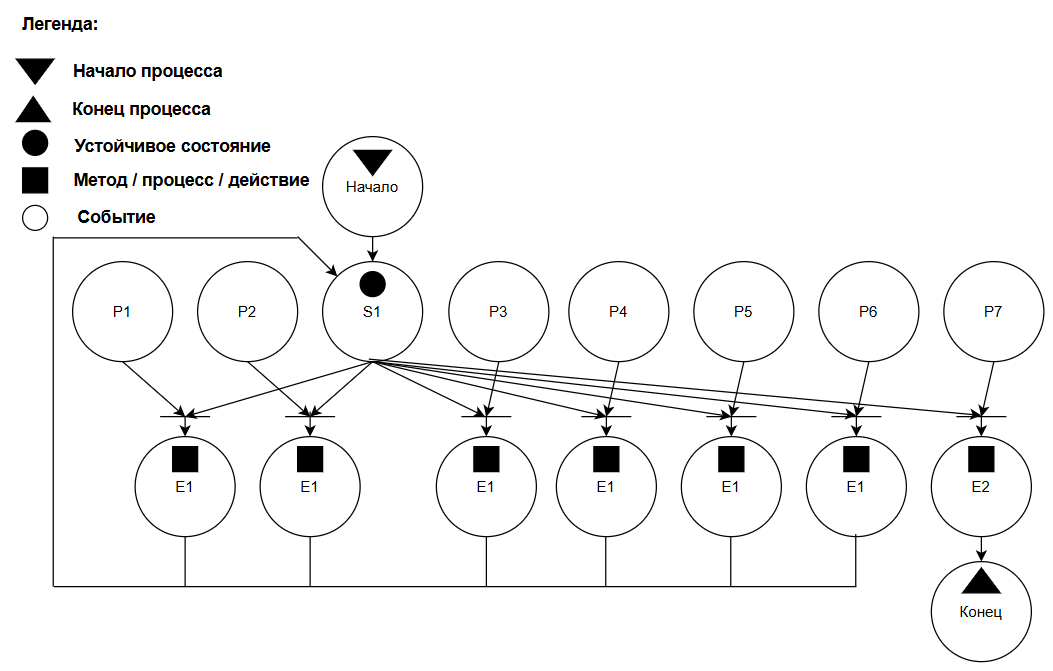


Рисунок 13 – Сеть Петри технологического процесса веб-страницы приложения 3

**Описание сети Петри**

- состояния (*states*)

s1 – форма ожидает действий пользователя

- действия (*effects*)

e1 – перерасчёт конвертаций валют согласно заданному курсу

e2 – закрытие формы

- события (*prompts*)

p1 – изменено значение в поле «Российский рубль»

p2 – изменено значение в поле «Доллар США»

p3 – изменено значение в поле «Евро»

p4 – изменено значение в поле «Японская йена»

p5 – изменено значение в поле «Китайский юань»

p6 – изменено значение в поле «Индийская рупия»

p7 – нажата кнопка закрытия формы

# **8.3.2 Листинг веб-страницы приложения 3.**

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<title>Наблюдатели – Черкас Н.В. | Родин К.П.</title>

<script src="https://unpkg.com/vue@3/dist/vue.global.js"></script>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

background: #f5f5f5;

}

.container {

max-width: 800px;

margin: 0 auto;

background: white;

padding: 20px;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0,0,0,0.1);

}

table {

border-collapse: collapse;

width: 100%;

margin: 20px 0;

box-shadow: 0 2px 5px rgba(0,0,0,0.1);

}

td, th {

border: 1px solid #ddd;

padding: 12px;

text-align: center;

}

th {

background: #2c3e50;

color: white;

font-weight: bold;

}

tr:nth-child(even) {

background: #f9f9f9;

}

tr:hover {

background: #f0f8ff;

}

input {

width: 150px;

text-align: right;

padding: 8px;

border: 1px solid #ccc;

border-radius: 4px;

font-size: 14px;

}

input:focus {

outline: none;

border-color: #3498db;

box-shadow: 0 0 5px rgba(52,152,219,0.3);

}

.header {

text-align: center;

margin-bottom: 20px;

}

.info {

background: #e8f4f8;

padding: 10px;

border-radius: 5px;

margin: 10px 0;

text-align: center;

}

</style>

</head>

<body>

<div class="container">

<div class="header">

<h2>Конвертер валют (watch) — Черкас Н.В. | Родин К.П.</h2>

</div>

<div id="app">

<table>

<thead>

<tr>

<th>Валюта</th>

<th>Код</th>

<th>Значение</th>

</tr>

</thead>

<tbody>

<tr v-for="currency in currencyList" :key="currency.key">

<td><strong>{{ currency.name }}</strong></td>

<td>{{ currency.key.toUpperCase() }}</td>

<td>

<input

type="number"

step="0.01"

v-model.number="values[currency.key]"

@input="handleInput(currency.key)"

:ref="el => setInputRef(currency.key, el)"

>

</td>

</tr>

</tbody>

</table>

<div class="info">

<p><strong>Курсы зафиксированы на 10 января 2024 года</strong></p>

<p>1 RUB = {{ rates.usd }} USD | 1 RUB = {{ rates.eur }} EUR | 1 RUB = {{ rates.jpy }} JPY</p>

<p>1 RUB = {{ rates.cny }} CNY | 1 RUB = {{ rates.inr }} INR</p>

</div>

</div>

</div>

<script>

const { createApp, ref, watch } = Vue;

createApp({

setup() {

// Курсы валют относительно рубля на 10 января 2024 года

const rates = {

rub: 1,

usd: 0.01124, // 1 RUB = 0.01124 USD

eur: 0.01032, // 1 RUB = 0.01032 EUR

jpy: 1.64, // 1 RUB = 1.64 JPY

cny: 0.0801, // 1 RUB = 0.0801 CNY

inr: 0.931 // 1 RUB = 0.931 INR

};

// Список валют для отображения

const currencyList = [

{ key: 'rub', name: 'Российский рубль' },

{ key: 'usd', name: 'Доллар США' },

{ key: 'eur', name: 'Евро' },

{ key: 'jpy', name: 'Японская йена' },

{ key: 'cny', name: 'Китайский юань' },

{ key: 'inr', name: 'Индийская рупия' }

];

// Реактивные значения валют

const values = ref({

rub: 1000,

usd: 0,

eur: 0,

jpy: 0,

cny: 0,

inr: 0

});

// Референсы на input элементы

const inputRefs = ref({});

// Флаг для предотвращения рекурсивных обновлений

let isUpdating = false;

let lastChangedCurrency = null;

// Функция для установки референсов

const setInputRef = (key, el) => {

if (el) {

inputRefs.value[key] = el;

}

};

// Обработчик ввода

const handleInput = (changedKey) => {

if (isUpdating) return;

lastChangedCurrency = changedKey;

convertAllCurrencies(changedKey);

};

// Функция конвертации всех валют

const convertAllCurrencies = (baseKey) => {

if (isUpdating) return;

isUpdating = true;

const baseValue = values.value[baseKey];

if (baseValue === '' || baseValue === null || isNaN(baseValue)) {

isUpdating = false;

return;

}

// Конвертируем через рубль для единообразия

const baseInRub = baseValue / rates[baseKey];

// Обновляем все остальные валюты

currencyList.forEach(currency => {

if (currency.key !== baseKey) {

values.value[currency.key] = parseFloat((baseInRub \* rates[currency.key]).toFixed(4));

}

});

// Небольшая задержка для сброса флага

setTimeout(() => {

isUpdating = false;

}, 10);

};

// Наблюдатели для каждой валюты

currencyList.forEach(currency => {

watch(

() => values.value[currency.key],

(newValue, oldValue) => {

// Пропускаем если значение не изменилось или это обновление от системы

if (newValue === oldValue || isUpdating) return;

// Пропускаем если это та же валюта, что сейчас обрабатывается

if (currency.key === lastChangedCurrency) return;

handleInput(currency.key);

}

);

});

// Инициализация начальных значений

const initialize = () => {

convertAllCurrencies('rub');

};

// Вызываем инициализацию после монтирования

setTimeout(initialize, 100);

return {

rates,

currencyList,

values,

inputRefs,

setInputRef,

handleInput

};

}

}).mount('#app');

</script>

</body>

</html>

**8.3.3 Результат отображения веб-страницы приложения 3.**

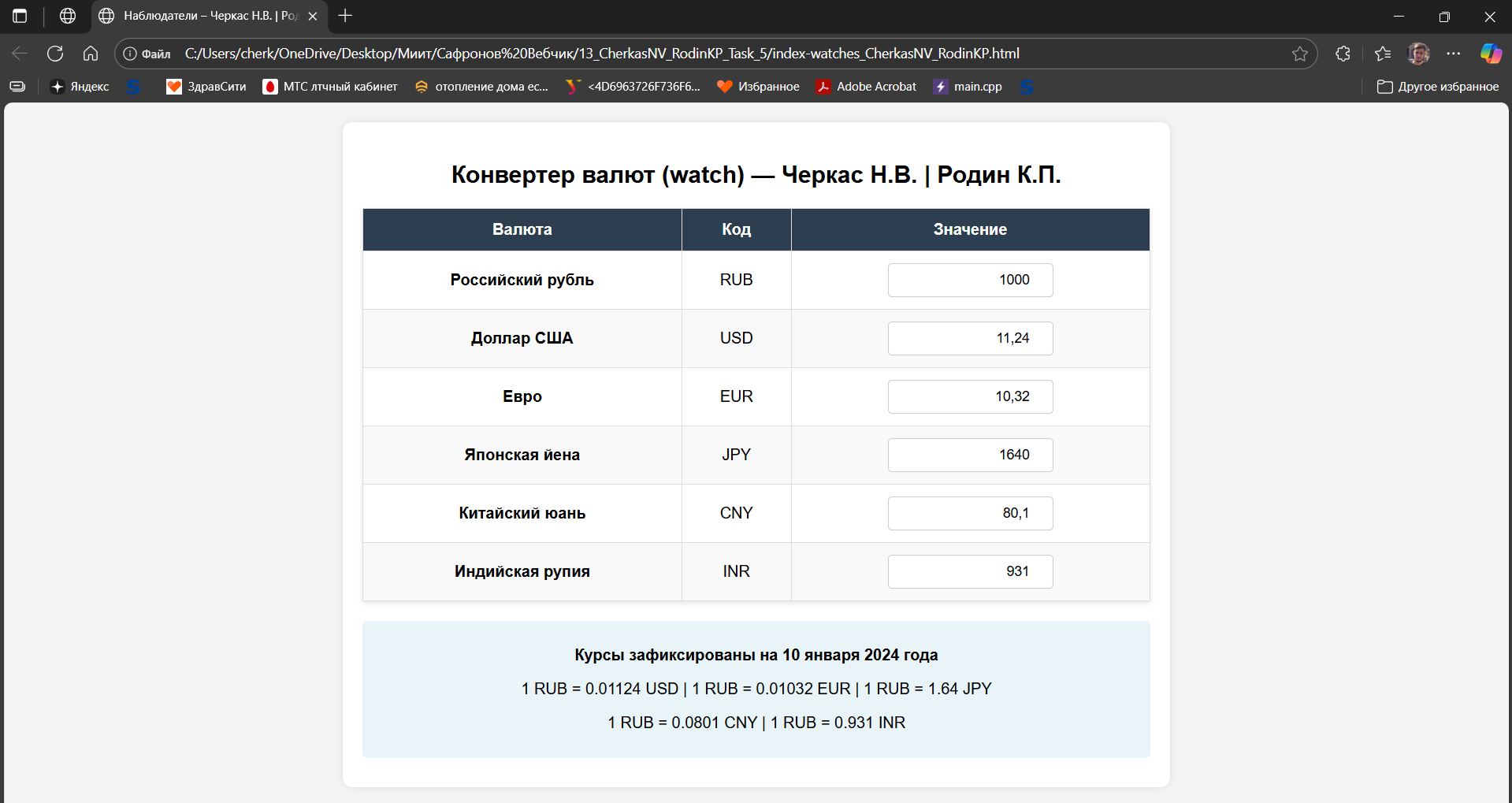


Рисунок 14 – Отображение в браузере веб-страницы приложения 3

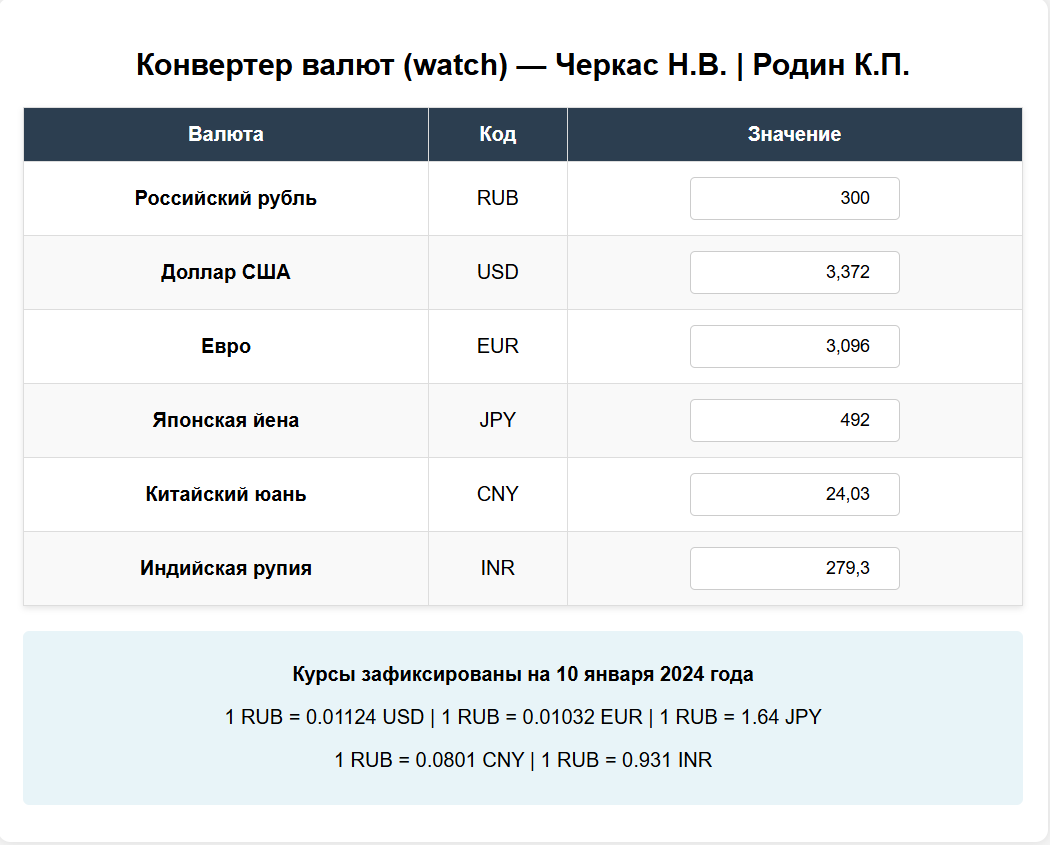
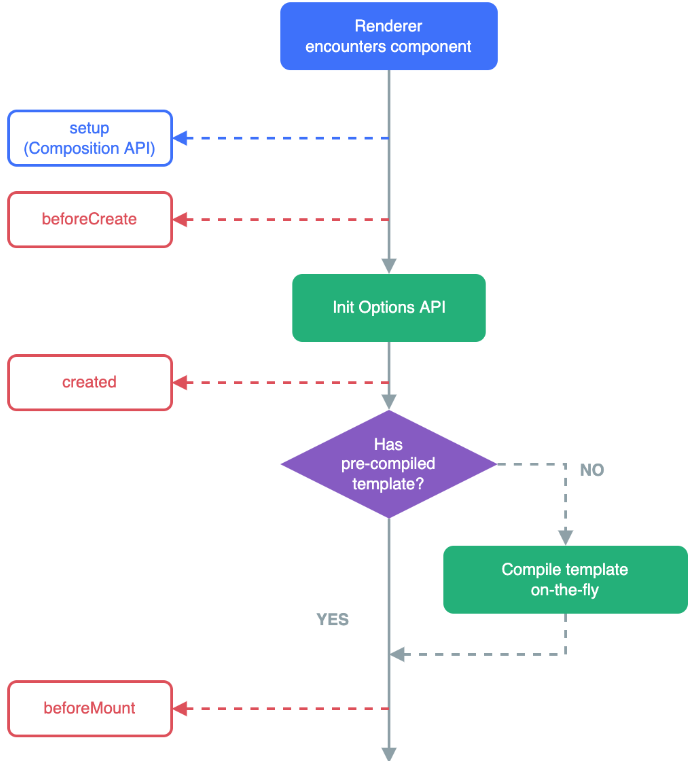


Рисунок 15 – Изменение значения в поле «Российский рубль»

# **9. Формулировку вывода о проделанной работе**

В рамках данной работы были получены навыки разработки одностраничных локальных *web*-приложений (*LSPWA*) под управлением фреймворка *Vue.js* на языке *JavaScript*.

**Приложение 1. Схема жизненного цикла (на английском языке) одностраничного *web*-приложения под управлением *Vue.js 3.x.***



**1**

Рисунок 16 – Витки жизненного цикла экземпляра *Vue 3.x* (начало)

**1**

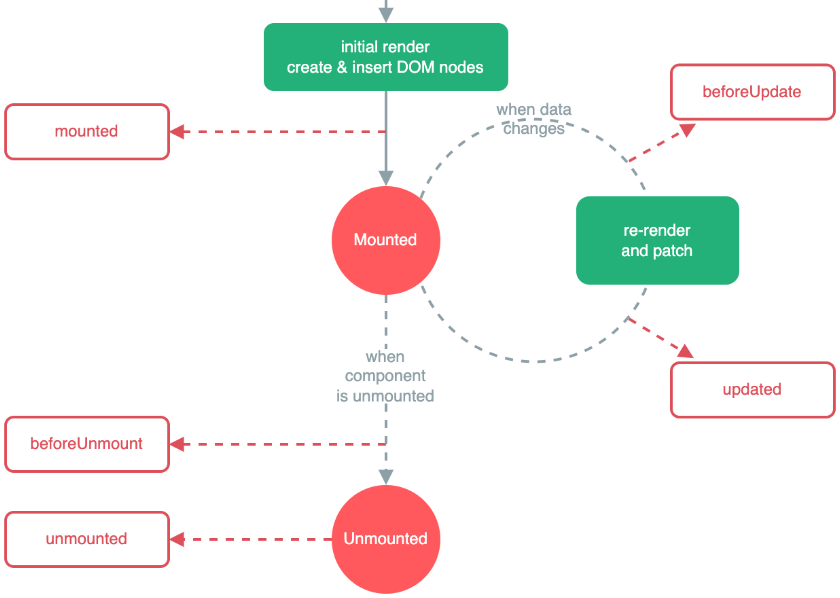
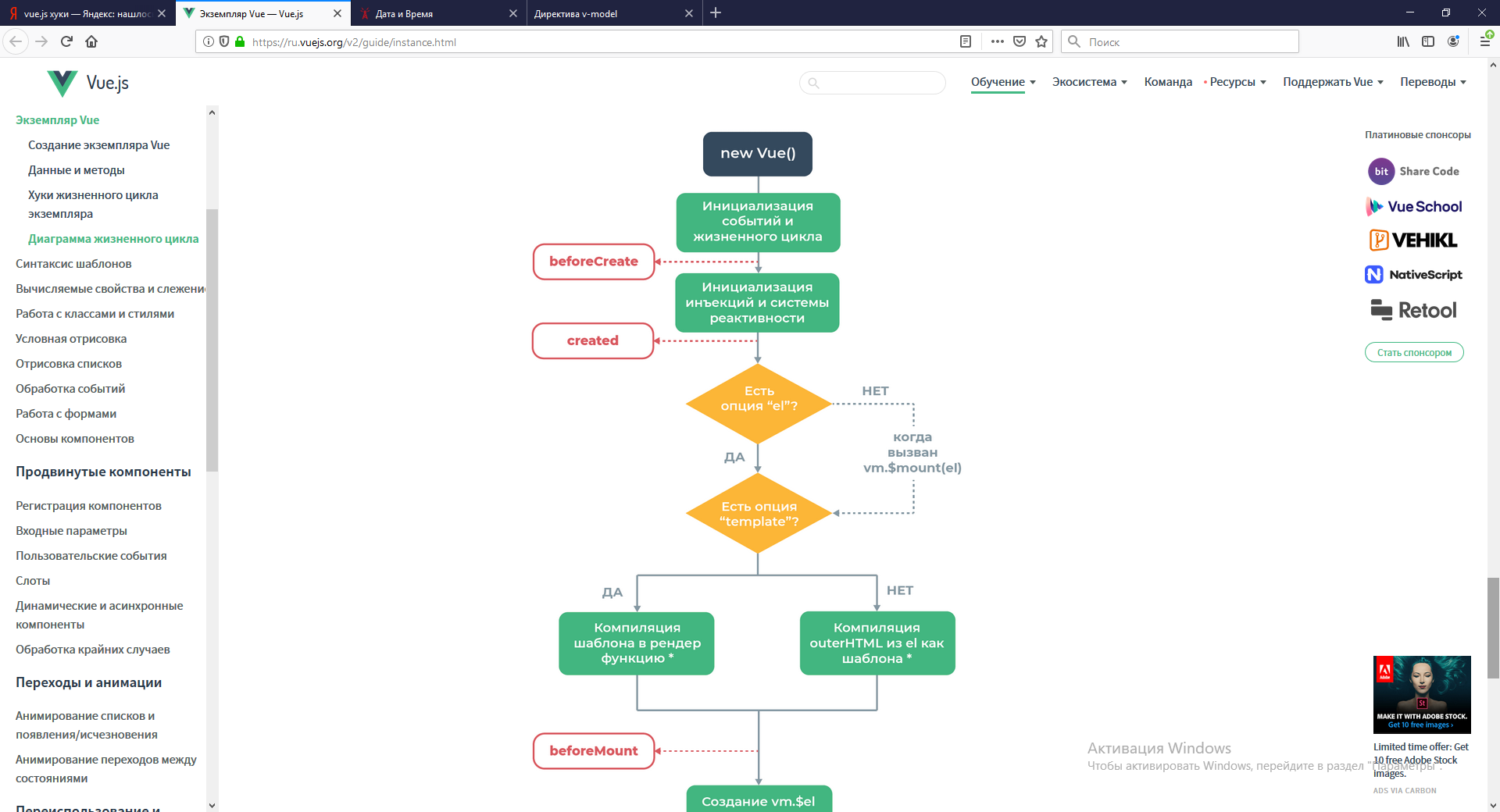


Рисунок 17 – Витки жизненного цикла экземпляра *Vue 3.x* (продолжение)

**Приложение 2. Схема жизненного цикла (на русском языке) одностраничного *web*-приложения под управлением *Vue.js 2.x.***



**1**

Рисунок 18 – Витки жизненного цикла экземпляра *Vue 2.x* (начало)

**1**

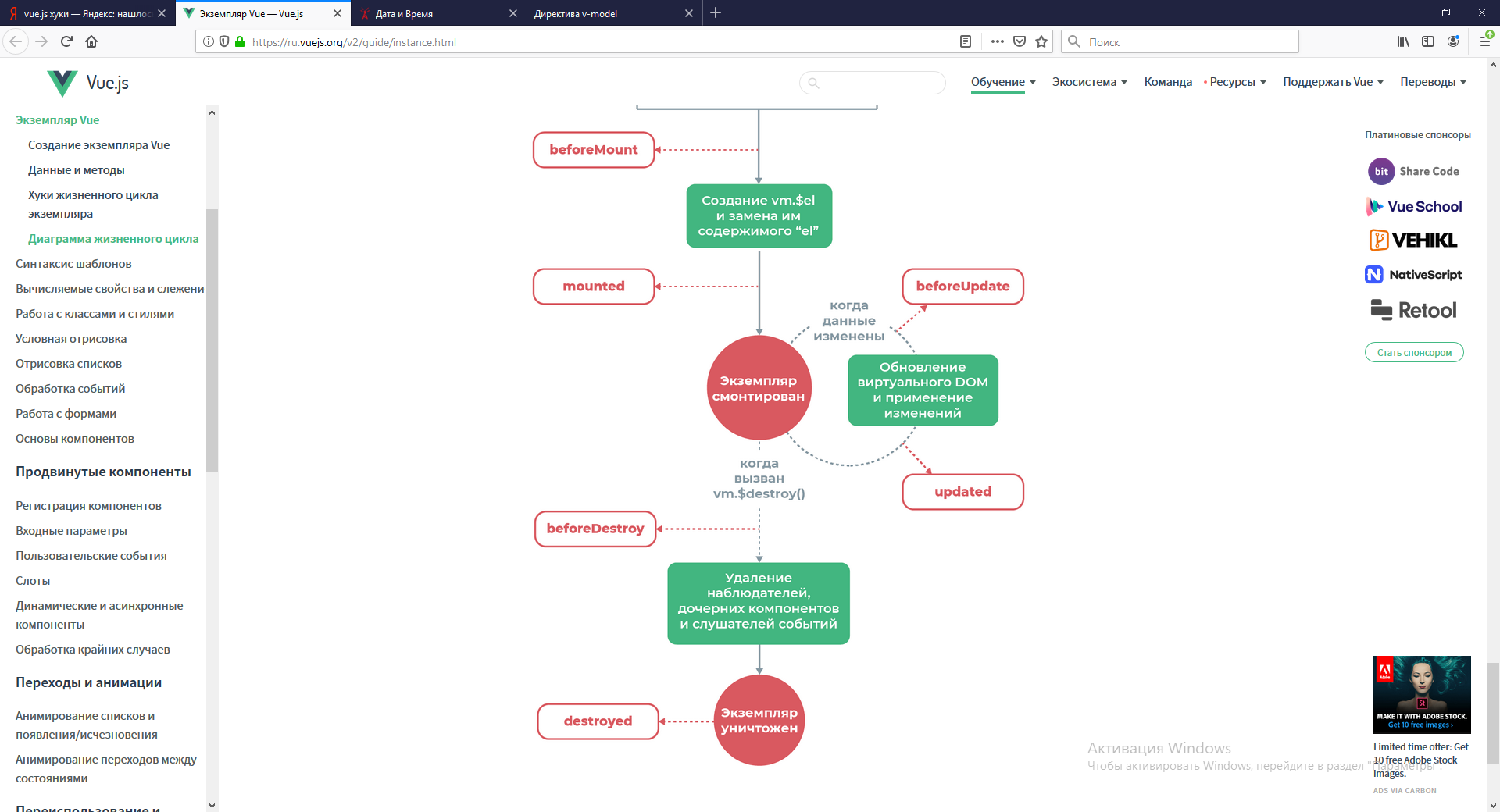
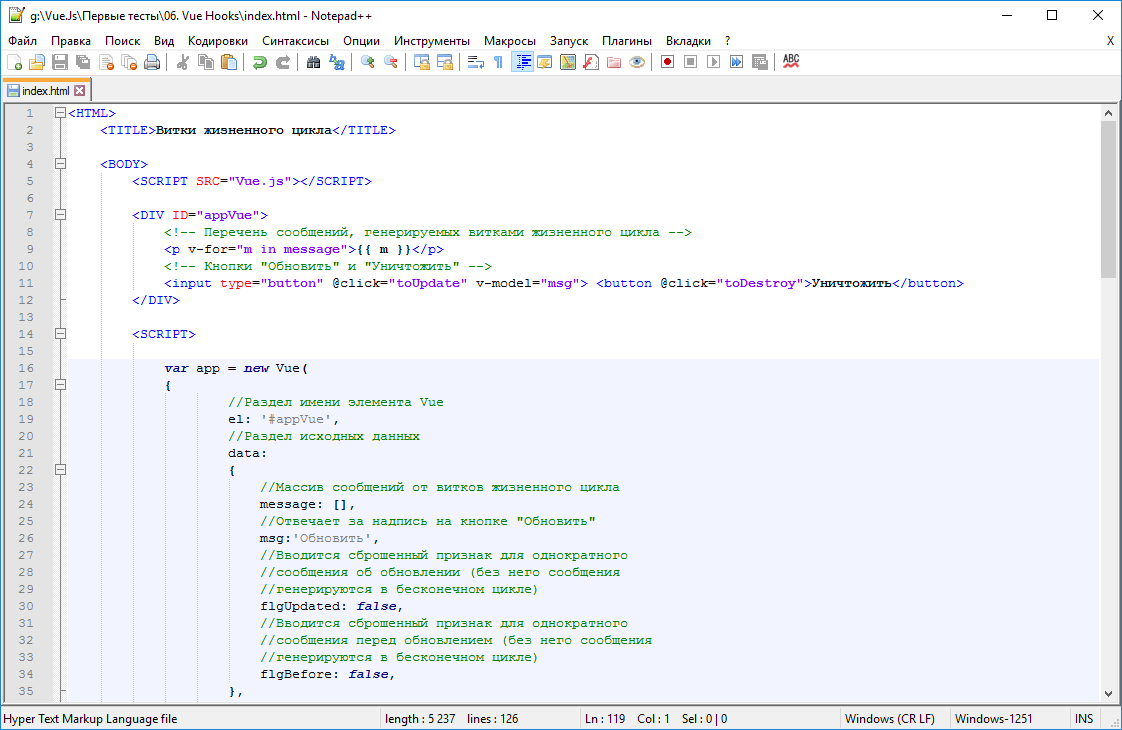


Рисунок 19 – Витки жизненного цикла экземпляра *Vue 2.x* (продолжение)



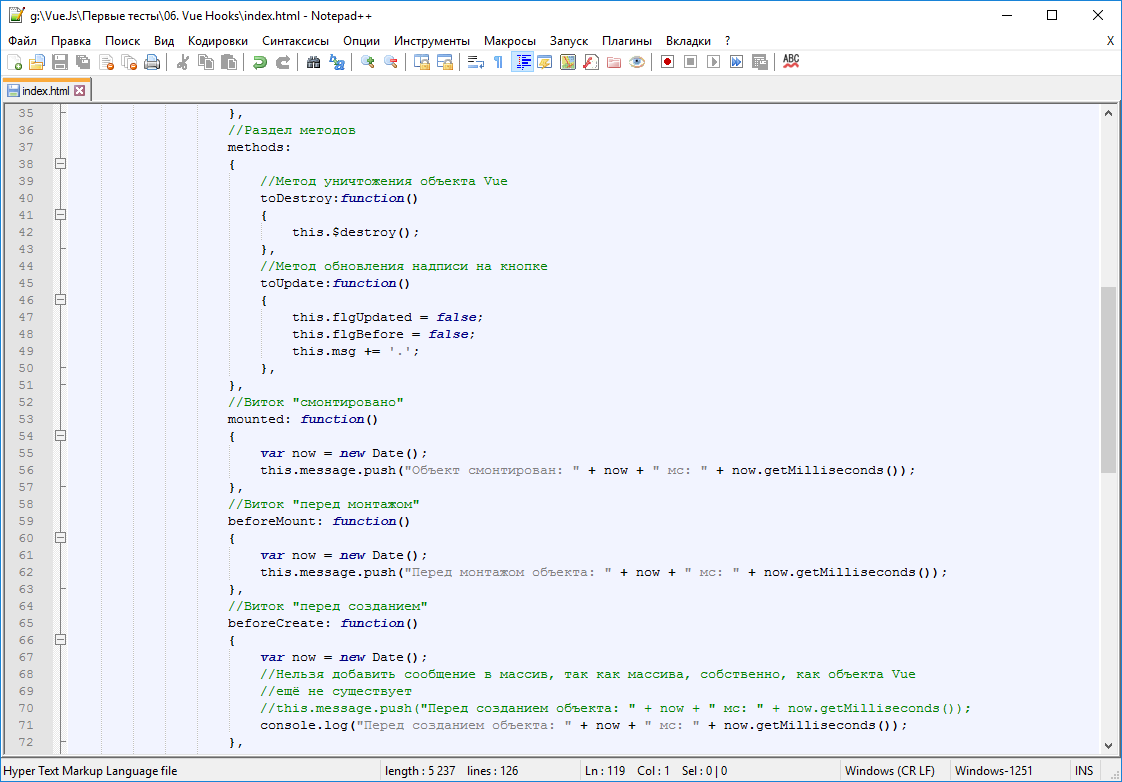
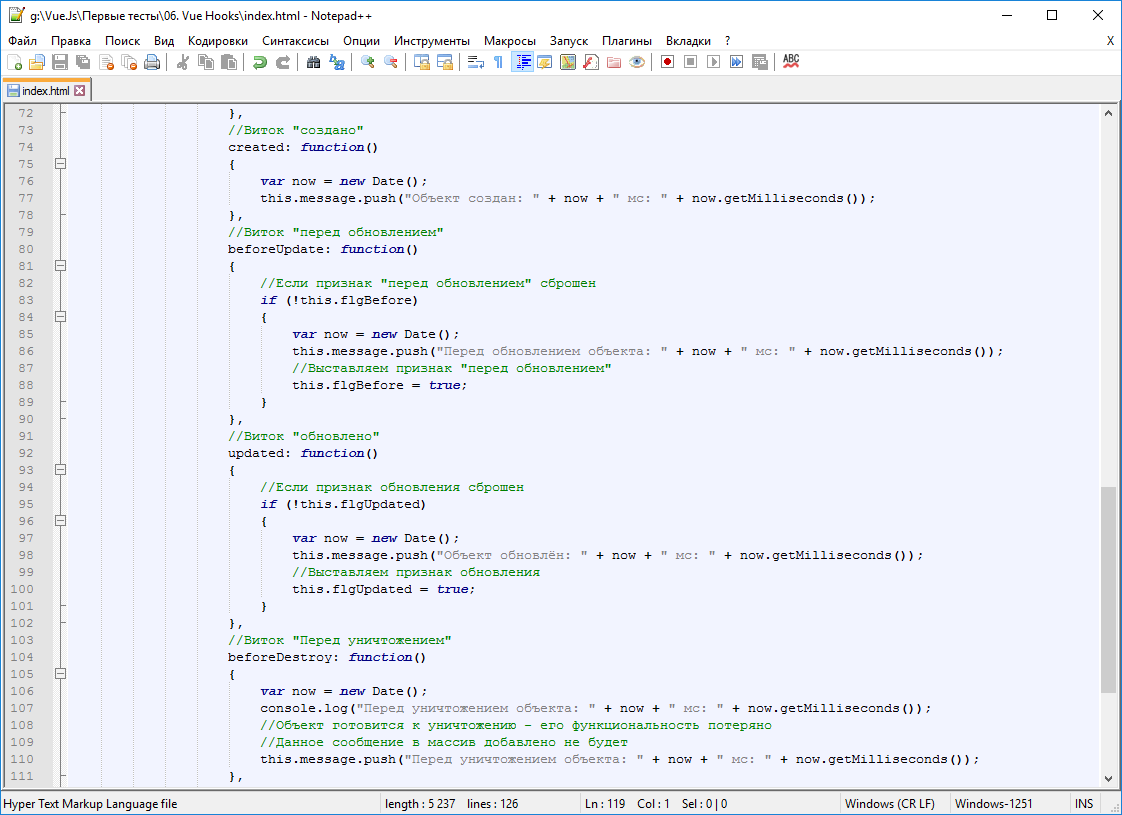


Рисунок 20 – Код примера тестирования работы витков ЖЦ *Vue 2.x* (начало)



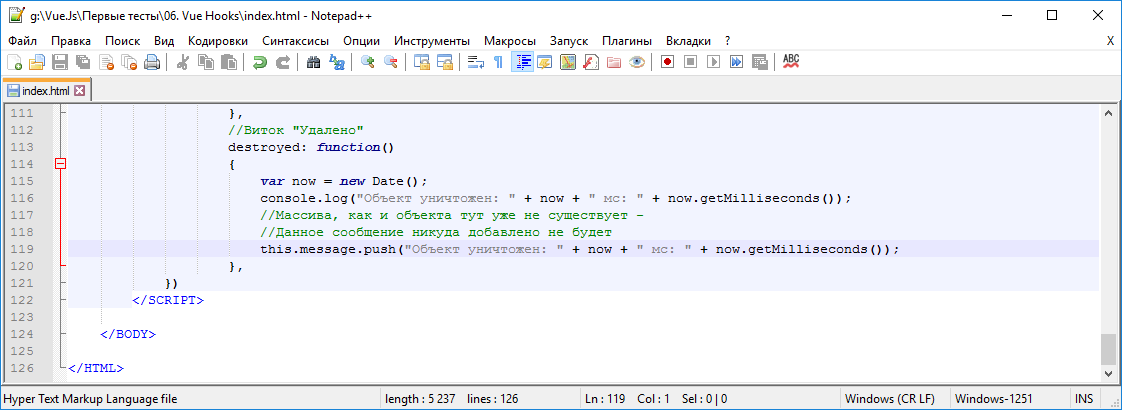


Рисунок 21 – Код примера тестирования работы витков ЖЦ *Vue 2.x* (продолжение)

**Приложение 3. Пример демонстрации версии браузера, используемого для выполнения работы*.***

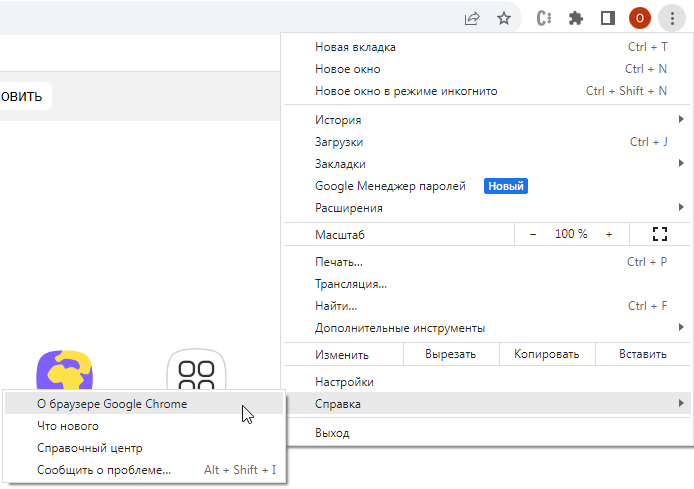


Рисунок 22 – Переход к получению сведений о браузере на примере *Google Chrome*

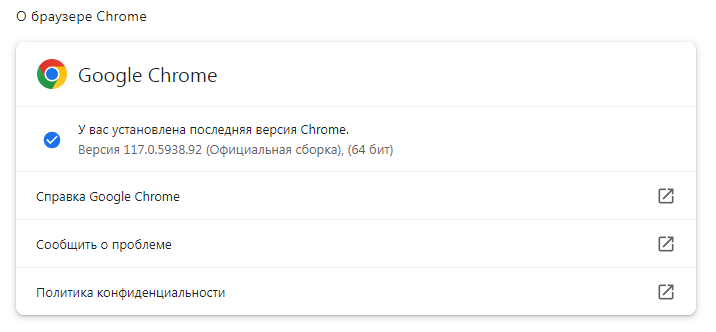


Рисунок 23 – Cведения о браузере *Google Chrome*

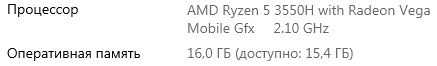
**Приложение 4. Структура таблицы соответствия переменных и методов, входящих в состав одностраничного *web*-приложения*.***

Таблица 1 – Пример таблицы соответствия переменных

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Физический смысл конструкции** | **Наименование конструкции** | **Тип данных** |
| **1** | Метод инициации уничтожения экземпляра прототипа *Vue.js* | *toDestroy* | *function()* |
| **2** | Текущее состояние даты и времени | *now* | *Date()* |
| **3** | Динамическая надпись на экранной кнопке | *msg* | *string* |
| **4** | Список сообщений о событиях (журнал событий) | *message* | *string[]* |

**Приложение 5. Структура таблицы спецификации оборудования*.***

Например, в ОС семейства *Windows* эти сведения можно списать из раздела «Система».



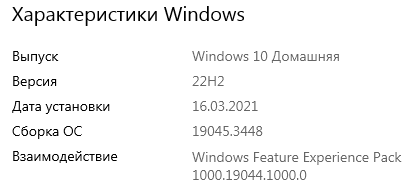


Рисунок 24 – Пример сведений об операционной системе

Таблица 2 – Пример таблицы спецификации 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 | 8 Гб |
| **6** | Объём жёсткого диска | 1 Тб |