ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»  
(РУТ (МИИТ))

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

ОТЧЁТ  
ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

По дисциплине «Web-программирование»

ВАРИАНТ 4

Выполнили: ст. гр. ТКИ-541

Степанов Дмитрий

Кузнецов Кирилл

Проверил: к.т.н., доц. Сафронов А.И.

Москва 2024

Оглавление

[1. Цель работы](#_ln31jucd3b65) 3

[2. Формулировка задачи](#_flks7avmiw2v) 3

[3. Спецификация оборудования, на котором выполнялась работа](#_kwx72bcthqce) 4

[4. Информация об используемом браузере](#_dw50tsy7bpkc) 4

[5. Технология локального подключения Vue.js](#_cdi652kefye2) 4

[6. Русифицированная схема жизненного цикла Vue.js.](#_njky1gf62jtm) 6

[7. Таблица соответствия переменных и методов, используемых в web-приложении](#_nf8qv1fjkg6f) 7

[8. Содержательная часть по виткам жизненного цикла](#_764y0uyi9bxj) 9

[8.1 Витки жизненного цикла](#_h0h4f1rifmry) 9

[8.1.1 Код приложения](#_v2fs01ubzrwc) 9

[8.1.2 Отображение в браузере](#_30vk9yhlmin) 13

[8.1.3 Сеть Петри](#_h4fapx7dc0xi) 14

[8.2 Циклическая отрисовка](#_6x3cwgb9yppg) 15

[8.2.1 Код приложения](#_codg6bck94b9) 15

[8.2.2 Отображение в браузере](#_3ke67ki7ksy4) 19

[8.2.3 Сеть Петри](#_z94pvbajvnli) 20

[8.3 Наблюдатели](#_e9g811xvjp4p) 20

[8.3.1 Код приложения](#_qwtqsmk5uyf5) 20

[8.3.2 Отображение в браузере](#_hcnpjents1nh) 24

[8.3.3 Сеть Петри](#_hmyaqxidfmp0) 25

[9. Вывод](#_bvxhn9lmmxta)25

# Цель работы

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

# Формулировка задачи

**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;*

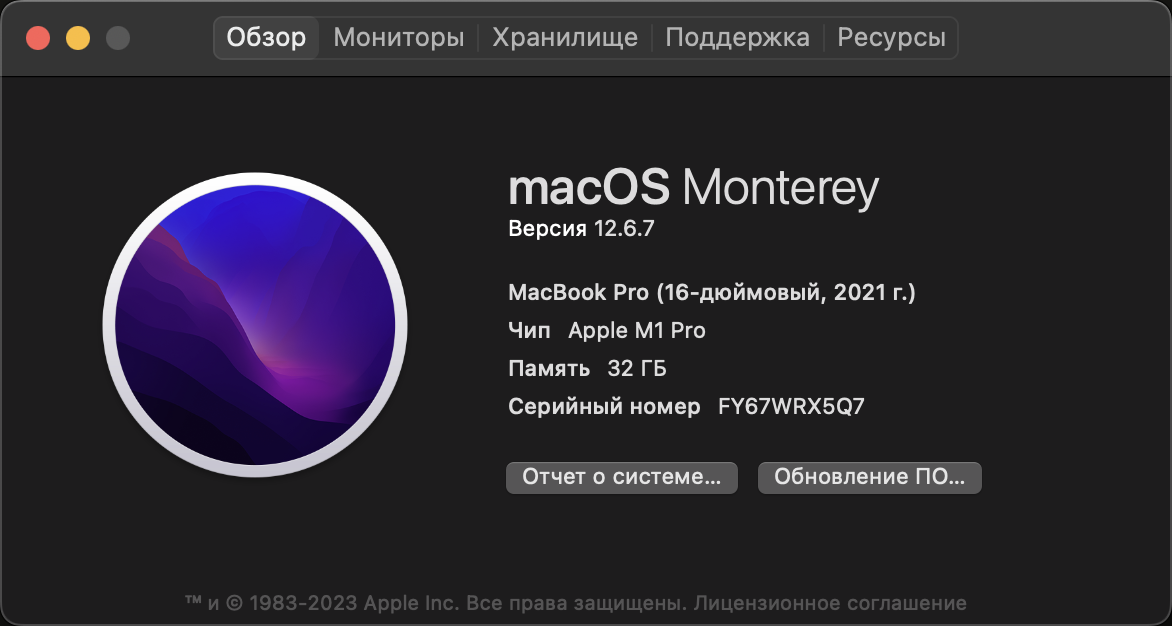
– *watches* в файле с именем *index-watches\_Familiya\_I\_O.html;*

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

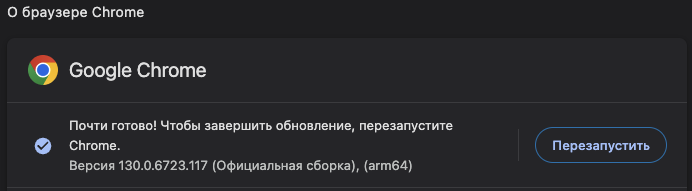
**Задания по варианту:**

* 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*
* Изобразить в области масштабируемой векторной графики <svg> размерности 1000 на 300 пикселей по ширине и высоте, соответственно, горизонтальную координатную ось со стрелкой, засечками, числовыми значениями под засечками и особой, более жирной засечкой в ноле (присутствует всегда, когда указан диапазон с различными знаками (sign) значений (values) в начале и конце, даже в ситуациях, когда шаг отрисовки засечек в точности не попадает в ноль), разграничивающей области отрицательных и положительных значений. Ось отстаёт от рамок области <svg> на 100 пикселей как сверху, так и по бокам. Стрелка оси всегда фиксированных размеров, устанавливаемых разработчиком (оператор не управляет настройками вида этого элемента). Оператор меняет шаг отрисовки засечек, указывает начало и конец диапазона, указывает значение округления для текста маркеров под засечками при использовании элементов <input>. В качестве основных элементов внутри <svg> использовать <line>, для маркеров, соответственно, использовать <text>.
* Заполняется статический маркированный список <ul>. Ввод новых значений предусмотрен через <input> по нажатии сочетания «Ctrl» + «Enter». Как только в поле <input> возникает значение-команда «#редактировать#» статический список меняется на функциональный <select>, в котором можно выбирать желаемую для редактирования позицию, её текст передаётся в <input>, где можно его отредактировать и по «Ctrl» + «Enter» изменить в перечне. Значение-команда «#вводить#» переводит список обратно в состояние статического, маркированного.

# Спецификация оборудования, на котором выполнялась работа



# Информация об используемом браузере

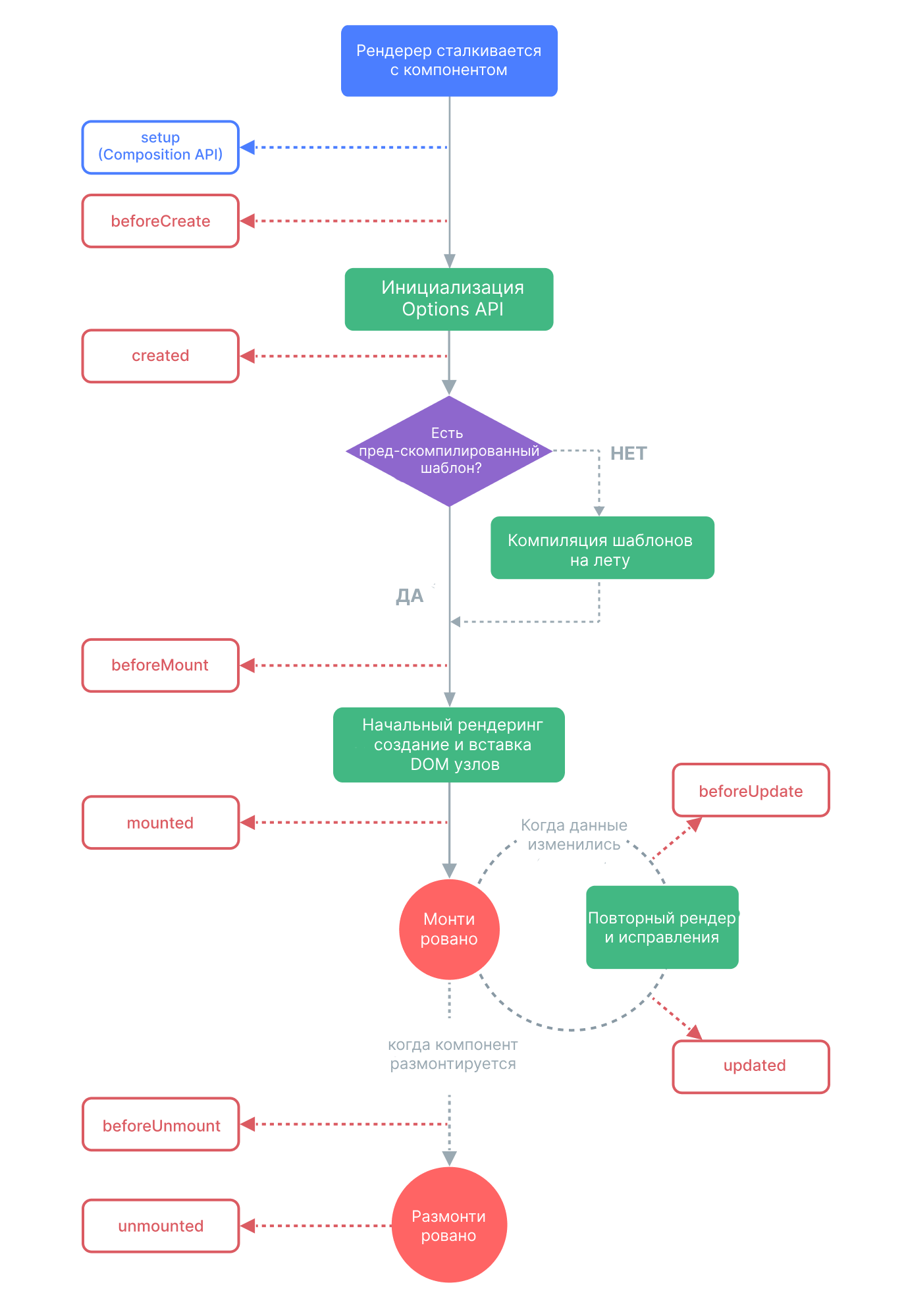


# Технология локального подключения Vue.js

В работе была использована vue.js версии 3.5.12.

К каждому файлу задания была подключена сборка vue.js, установленная с источника <https://unpkg.com/vue@3.5.12/dist/vue.global.js>.

# Русифицированная схема жизненного цикла *Vue.js*.



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

| Задание | Переменная/метод | Описание |
| --- | --- | --- |
| 1 | logHook() | Регистрация событий жизненного цикла |
| 1 | beforeCreate() | Хук жизненного цикла, вызывается до инициализации данных компонента |
| 1 | created() | Хук жизненного цикла, вызывается после инициализации данных компонента |
| 1 | beforeMount() | Хук жизненного цикла, вызывается перед монтированием компонента в DOM |
| 1 | mounted() | Хук жизненного цикла, вызывается после того, как компонент смонтирован в DOM |
| 1 | beforeUpdate() | Хук жизненного цикла, вызывается перед обновлением компонента |
| 1 | updated() | Хук жизненного цикла, вызывается после обновления компонента в DOM |
| 1 | beforeUnmount() | Хук жизненного цикла, вызывается перед размонтированием компонента |
| 1 | unmounted() | Хук жизненного цикла, вызывается после удаления компонента из DOM |
| 2 | ticks | Одномерный массив содержащий отметки на оси абсцисс. |
| 2 | tick | Объект, содержащий информацию об отметке на оси абсцисс. Содержит координату х (в пикселях)и значение отметки value (безразмерная величина). |
| 2 | start | Начало диапазона координатной оси |
| 2 | end | Конец диапазона координатной оси |
| 2 | step | Размер шага по координатной оси |
| 2 | precision | Размер шага округления при передвижении по координатной оси |
| 2 | ticks() | Генерирует массив ticks для последующей отрисовки координатной оси |
| 2 | inputStep() | Генерирует размер шага округления (параметр step для тега input) у окон ввода начала, конца и размера шага координатной оси |
| 3 | inputValue | Значение вводимое внутрь тега input |
| 3 | items | Коллекция, содержащая значения элементов списка |
| 3 | isEditing | Флаг, сигнализирующий о включении режима редактирования значения элемента списка |
| 3 | selectedIndex | Индекс выбранного пользователем элемента списка |
| 3 | handleKeydown() | Метод-обработчик события “Нажатие сочетания клавиш ctrl+enter” |
| 3 | inputValue() | Наблюдатель за значением, вводимым внутрь тега input |
| 3 | selectedIndex() | Наблюдатель за значением, хранящим индекс выбранного пользователем элемента списка |

# Содержательная часть по виткам жизненного цикла

## Витки жизненного цикла

### Код приложения

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Vue 3 Lifecycle Hooks Demo</title>

<script src="https://unpkg.com/vue@3.5.12/dist/vue.global.js"></script>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

}

.message {

margin-top: 20px;

padding: 5px;

background-color: #f4f4f4;

border-radius: 4px;

}

button {

margin-right: 10px;

padding: 5px 10px;

font-size: 14px;

}

.log-container {

margin-top: 20px;

margin-bottom: 20px;

padding: 10px;

background-color: #eaeaea;

border-radius: 4px;

max-height: 200px;

overflow-y: auto;

white-space: pre-wrap;

}

</style>

</head>

<body>

<div id="app">

<h1>Vue 3 Lifecycle Hooks Demo</h1>

<div class="log-container">

<pre>{{ logMessages }}</pre>

</div>

<button @click="toggleChild">Toggle Child Component</button>

<button @click="updateMessage">Update Child Message</button>

<child-component v-if="showChild" :message="childMessage" />

</div>

<script>

const { createApp, defineComponent, ref, onBeforeMount, onMounted, onBeforeUpdate, onUpdated, onBeforeUnmount, onUnmounted, provide, inject } = Vue;

const ChildComponent = defineComponent({

props: ['message'],

template: `

<div class="message">

<h3>Child Component</h3>

<p>Message: {{ message }}</p>

</div>

`,

setup(props) {

return { logHook: inject('logHook') };

},

beforeCreate() { this.logHook('Child', 'beforeCreate'); },

created() { this.logHook('Child', 'created'); },

beforeMount() { this.logHook('Child', 'beforeMount'); },

mounted() { this.logHook('Child', 'mounted'); },

beforeUpdate() { this.logHook('Child', 'beforeUpdate'); },

updated() { this.logHook('Child', 'updated'); },

beforeUnmount() { this.logHook('Child', 'beforeUnmount'); },

unmounted() { this.logHook('Child', 'unmounted'); },

});

const app = createApp({

components: { ChildComponent },

data() {

return {

showChild: true,

childMessage: 'Initial message',

logMessages: ''

};

},

methods: {

toggleChild() {

this.showChild = !this.showChild;

},

updateMessage() {

this.childMessage = `Updated message at ${new Date().toLocaleTimeString()}`;

},

logHook(entity, hook) {

const message = `${entity}: ${hook} performed at ${new Date().toISOString()}`;

console.log(message);

this.logMessages += message + '\n';

}

},

provide() { return { logHook: this.logHook }; }

});

app.mount('#app');

</script>

</body>

</html>

### Отображение в браузере

|  |
| --- |
| 1. – Отображение в браузере |

### Сеть Петри

|  |
| --- |
| 1. – Краткая сеть Петри |

## Циклическая отрисовка

### Код приложения

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Vue 3 SVG Axis</title>

<script src="https://unpkg.com/vue@3.5.12/dist/vue.global.js"></script>

<style>

.axis-line {

stroke: black;

stroke-width: 2;

}

.tick-line {

stroke: black;

stroke-width: 1;

}

.zero-tick-line {

stroke: black;

stroke-width: 3;

}

.arrow {

stroke: black;

stroke-width: 2;

fill: black;

}

.tick-text {

font-size: 12px;

text-anchor: middle;

}

</style>

</head>

<body>

<div id="app">

<svg width="1000" height="300">

<line x1="100" y1="150" x2="900" y2="150" class="axis-line"></line>

<polygon points="900,150 890,145 890,155" class="arrow"></polygon>

<line v-for="tick in ticks" :x1="tick.x" y1="140" :x2="tick.x" y2="160"

:class="tick.value === 0 ? 'zero-tick-line' : 'tick-line'"></line>

<text v-for="tick in ticks" :x="tick.x" y="175" class="tick-text">{{ tick.value.toFixed(precision) }}</text>

</svg>

<div>

<label for="start">Start:</label>

<input type="number" v-model.number="start" :step="inputStep" id="start">

<label for="end">End:</label>

<input type="number" v-model.number="end" :step="inputStep" id="end">

<label for="step">Step:</label>

<input type="number" v-model.number="step" :step="inputStep" id="step">

<label for="precision">Precision:</label>

<input type="number" v-model.number="precision" id="precision">

</div>

</div>

<script>

const { createApp } = Vue;

const offset = 100;

const width = 800;

createApp({

data() {

return {

start: -10,

end: 10,

step: 1,

precision: 0

};

},

computed: {

inputStep() {

return Math.pow(10, -this.precision);

},

ticks() {

if (this.start >= this.end) return;

const ticks = [];

let prev = this.start;

for (let value = this.start; value <= this.end; value += this.step) {

const x = offset + ((value - this.start) / (this.end - this.start)) \* width;

ticks.push({ x: x, value: value });

if (prev < 0 && value >= 0) {

zero = offset + ((0 - this.start) / (this.end - this.start)) \* width;

ticks.push({ x: zero, value: 0 });

}

prev = value;

}

return ticks;

}

}

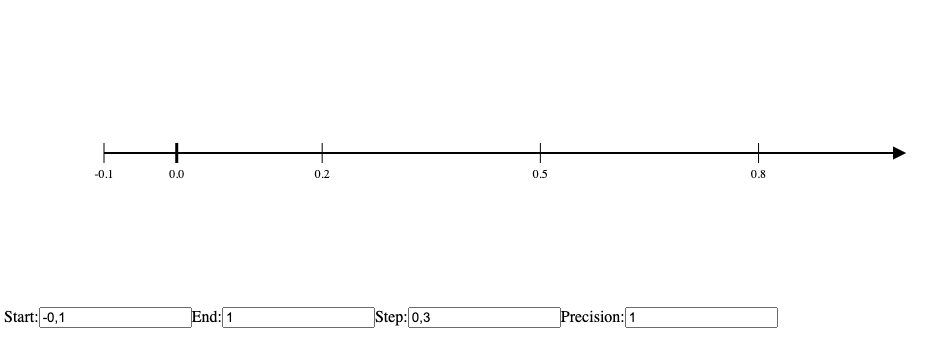
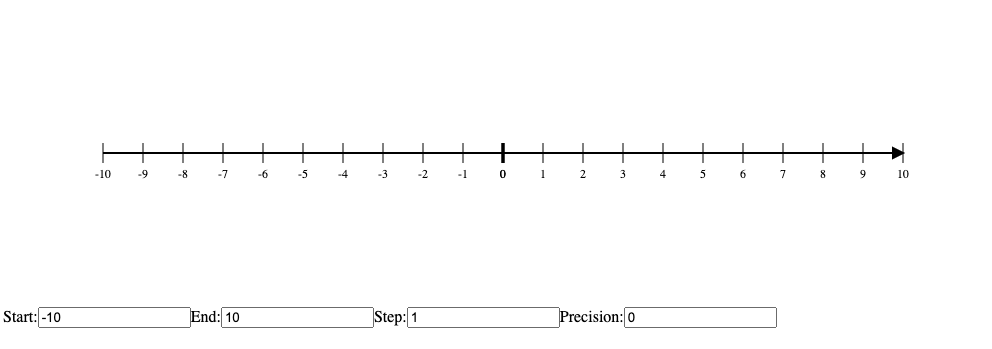
}).mount('#app');

</script>

</body>

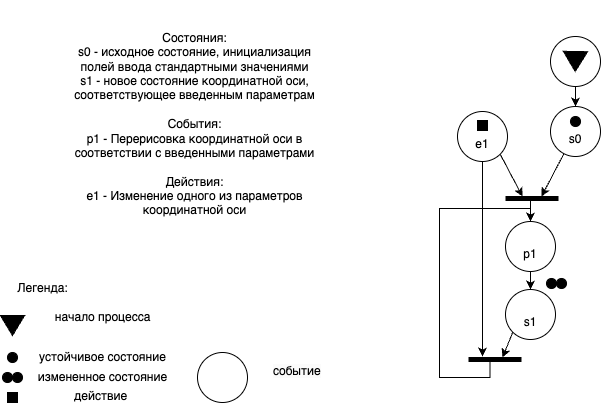
</html>

### Отображение в браузере



1. – Отображение координатной оси в браузере

### Сеть Петри



1. – Краткая сеть Петри

## Наблюдатели

### Код приложения

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Vue 3 List Editing</title>

<script src="https://unpkg.com/vue@3.5.12/dist/vue.global.js"></script>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

}

.input-container {

margin-bottom: 20px;

}

.input-container label {

margin-right: 10px;

}

.input-container input {

padding: 5px;

font-size: 14px;

}

ul, select {

margin-top: 20px;

font-size: 14px;

}

ul li {

margin-bottom: 5px;

}

</style>

</head>

<body>

<div id="app">

<div class="input-container">

<label for="input">Input:</label>

<input type="text" v-model="inputValue" @keydown="handleKeydown" id="input">

</div>

<ul v-if="!isEditing">

<li v-for="(item, index) in items" :key="index">{{ item }}</li>

</ul>

<select v-else v-model="selectedIndex">

<option v-for="(item, index) in items" :key="index" :value="index">{{ item }}</option>

</select>

</div>

<script>

const { createApp, watch } = Vue;

const EDIT\_VALUE = '#редактировать#'

const ENTER\_VALUE = '#вводить'

createApp({

data() {

return {

inputValue: '',

items: [],

isEditing: false,

selectedIndex: null

};

},

watch: {

inputValue(newValue) {

if (newValue === EDIT\_VALUE) {

this.inputValue = '';

this.isEditing = true

} else if (newValue === ENTER\_VALUE) {

this.inputValue = '';

this.isEditing = false

}

},

selectedIndex(newIndex) {

if (newIndex !== null) this.inputValue = this.items[newIndex];

}

},

methods: {

handleKeydown() {

if (!((event.ctrlKey || event.metaKey) && event.key === 'Enter')) return;

if (this.isEditing && this.selectedIndex !== null) {

this.items[this.selectedIndex] = this.inputValue;

} else if (!this.isEditing) {

this.items.push(this.inputValue);

}

this.inputValue = '';

},

}

}).mount('#app');

</script>

</body>

</html>

### Отображение в браузере

### 

|  |
| --- |
| 1. – Различные этапы редактирования списка в браузере |

### Сеть Петри

|  |
| --- |
| 1. – Краткая сеть Петри |

# Вывод

В результате проведенной работы были разработаны локальные одностраничные web-приложения (LSPWA) под управлением фреймворка Vue.js на языке JavaScript в соответствии с указаниями вариантов индивидуального задания.