|  |  |
| --- | --- |
| Российский университет транспорта (МИИТ)  Институт транспортной техники и систем управления  Кафедра «управление и защита информации» | |
| **Программная инженерия** | |
| Отчет | |
| Фреймворк Vue.js витки жизненного цикла, v-for и наблюдатели | |
|  | |
|  | Выполнил:  Студент группы ТУУ-171  Литвиненкова А.П. |
|  | Проверил:  к.т.н., доц. Сафронов А.И. |
| Москва  2025  Содержание | |

[1. Цель работы 1](#_Toc200126459)

[2. Задание 1](#_Toc200126460)

[3. Содержательная часть 3](#_Toc200126461)

[3.1. Витки жизненного цикла 3](#_Toc200126462)

[3.2. Циклическая отрисовка (v-for) 9](#_Toc200126463)

[3.2.1. html 9](#_Toc200126464)

[3.2.2. скрипт 10](#_Toc200126465)

[3.2.3. Отображение в браузере 11](#_Toc200126466)

[3.3. Наблюдатели (watcher) 14](#_Toc200126467)

[3.3.1. html 14](#_Toc200126468)

[3.3.2. Скрипт 14](#_Toc200126469)

[3.3.1. Отображение в браузере 15](#_Toc200126470)

[4. Вывод 17](#_Toc200126471)

1. Цель работы

– 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..

1. Задание

Сделать статический, маркированный, двухуровневый список (гипертекстовая разметка – <ul>) с маркерами различного типа на разных уровнях по тематике «Транспорт». Верхний уровень – «Категория», например, водный, железнодорожный и так далее; нижний уровень «Разновидность, относящаяся к категории», например, «лодка», «катер», «дрезина», «поезд пригородного сообщения» и т.д. Предоставить оператору возможность добавления в список элементов на каждый уровень списка при помощи окна prompt. Способ разграничения добавления элементов на уровни разработчику необходимо продумать самостоятельно.

Посредством <input> последовательно заполняется статический, нумерованный список по нажатии на «пробел» («Spacebar»). Позиции в списке – семантически значимые. Имеется ограничение на ввод пяти символов. Как только в <input> вводится значение, превышающее 5 символов – содержимое нумерованного списка принудительно опустошается без возможности восстановления.

1. Содержательная часть
   1. Витки жизненного цикла

<html>

<title>Лаба05.1</title>

<script SRC="Vue.js"></script>

</head>

<body>

<div id="appVue">

<p v-if="message[0]">{{ message[0] }}</p>

<p v-if="message[1]">{{ message[1] }}</p>

<p v-if="message[2]">{{ message[2] }}</p>

<p v-if="message[3]">{{ message[3] }}</p>

<p v-if="message[4]">{{ message[4] }}</p>

<input type="button" @click="toUpdate" :value="msg" />

<button @click="toDestroy">Удалить</button>

</div>

<script>

const { createApp } = Vue;

createApp({

data() {

return {

message: Array(5).fill(""),

msg: "Обновить",

flgUpdated: false,

flgBefore: false,

};

},

methods: {

toUpdate() {

this.flgUpdated = false;

this.flgBefore = false;

this.msg += ".";

},

toDestroy() {

this.message = [];

this.destroyed = true;

}

},

beforeMount() {

const now = new Date();

this.message[1] = "Перед монтажом: " + now + " мс: " + now.getMilliseconds();

},

created() {

const now = new Date();

this.message[2] = "Создан: " + now + " мс: " + now.getMilliseconds();

},

mounted() {

const now = new Date();

this.message[0] = "Объект скопирован: " + now + " мс: " + now.getMilliseconds();

},

beforeUpdate() {

if (!this.flgBefore) {

const now = new Date();

this.message[3] = "Перед обновлением: " + now + " мс: " + now.getMilliseconds();

this.flgBefore = true;

}

},

updated() {

if (!this.flgUpdated) {

const now = new Date();

this.message[4] = "Объект обновлен: " + now + " мс: " + now.getMilliseconds();

this.flgUpdated = true;

}

},

beforeUnmount() {

const now = new Date();

console.log("Перед уничтожением: " + now + " мс: " + now.getMilliseconds());

},

unmounted() {

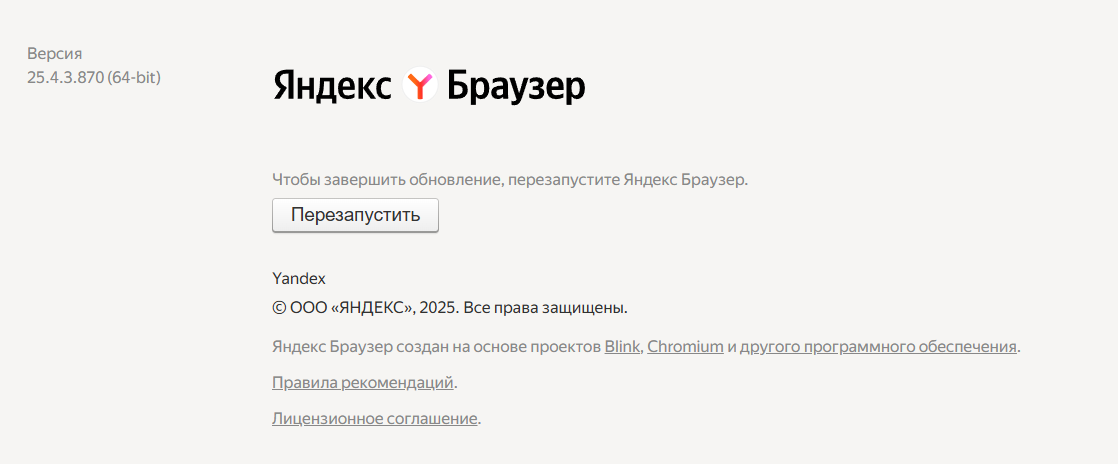
const now = new Date();

console.log("Уничтожено: " + now + " мс: " + now.getMilliseconds());

}

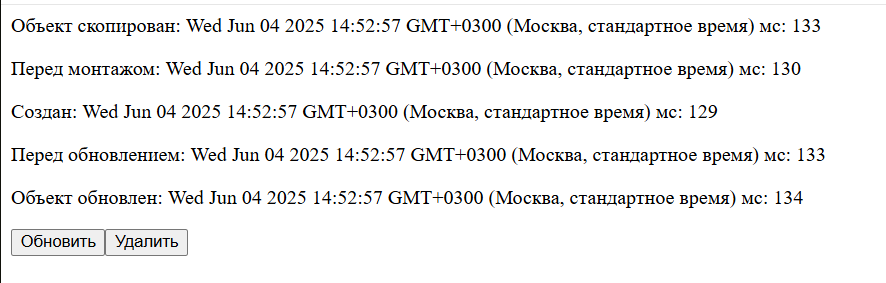
}).mount("#appVue");

</script></body></html>

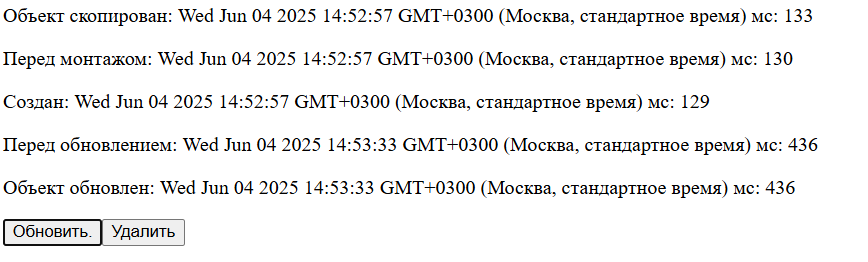


1. Информация о браузере

Отображение в браузере показано на рисунке.



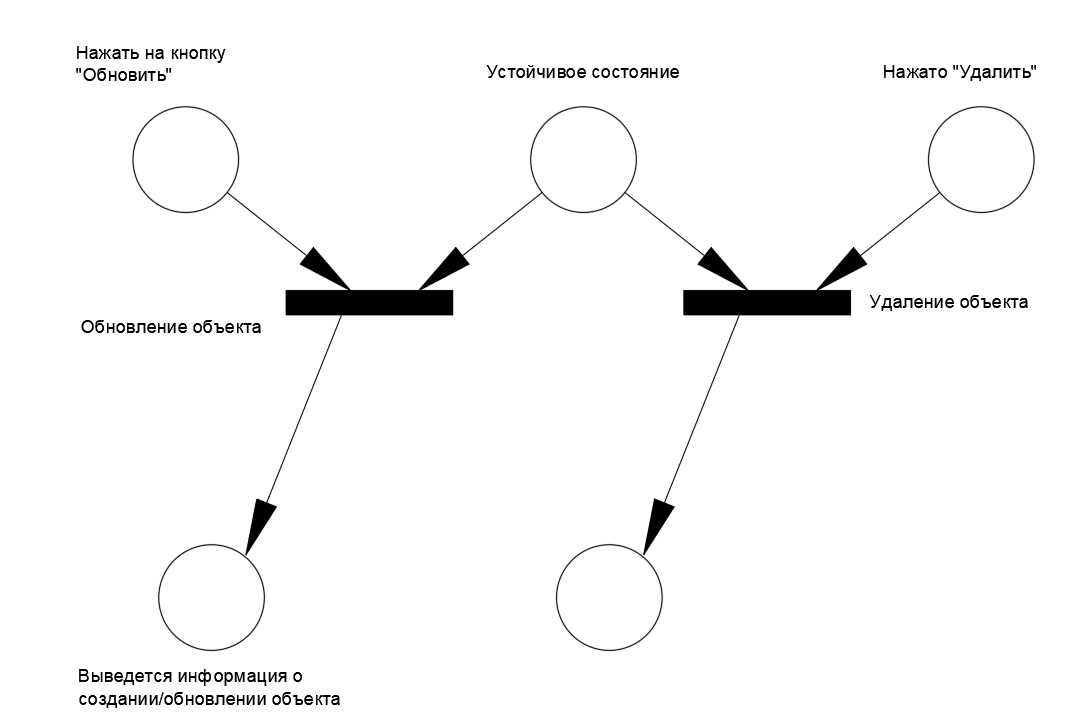
1. Отображение по умолчанию



1. Нажата кнопка «обновить»

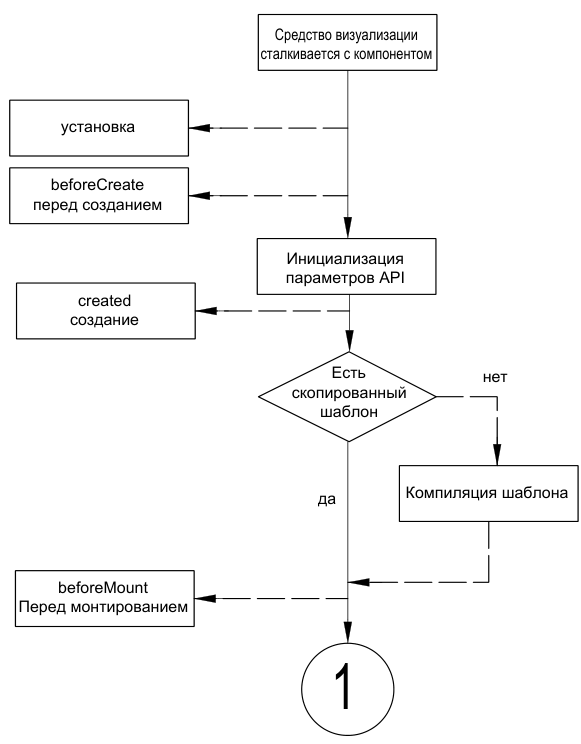


1. Нажата кнопка «удалить»

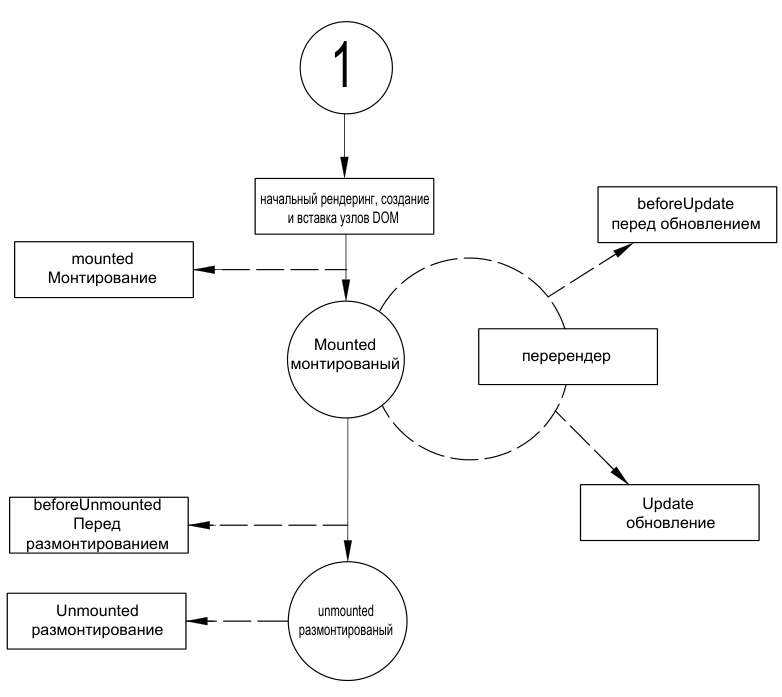


1. Сети Петри
2. Таблица соответствия данных

|  |  |  |  |
| --- | --- | --- | --- |
| **№ 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[]* |



1. Витки жизненного цикла (начало)



1. Витки жизненного цикла (начало)
   1. Циклическая отрисовка (v-for)
      1. html

<div id="appVue">

<div>

<ul id="transportList">

<li v-for="(type, index) in categ" :key="index">

Вид транспорта: {{ type.transport }}<br>

<button @click="addTransport(index)">Добавить транспорт</button>

<ul>

<li v-for="(x, categoryIndex) in type.category" :key="categoryIndex">

{{ x }}

</li></ul></li></ul></div><div>

<p>Литивненкова ТУУ-171 ©2025</p> </div></div>

* + 1. скрипт

var app = new Vue({

el: '#appVue',

data: {

categ: [

{

"transport": "Водный",

"category": [

"Катер",

"Яхта",

"Лайнер",

"Каноэ",

"Теплоход"

]

},

{

"transport": "Рельсовый",

"category": [

"Дрезина",

"Электровоз",

"Телега",

"Трамвай",

"Монорельс"

]

},

{

"transport": "Воздушный",

"category": [

"Самолет",

"Вертолет",

"Истребитель",

"Воздушный шар",

"Параплан"

] },

{

"transport": "Наземный",

"category": [

"Автомобиль",

"Самокат",

"Велосипед",

"Автобус",

"Скуттер"

] }] },

methods: {

addTransport(index) {

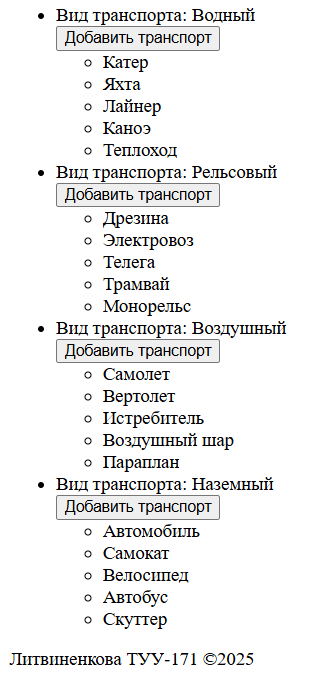
const newTransport = prompt('Введите название нового транспорта:');

if (newTransport) {

this.categ[index].category.push(newTransport); }}}});

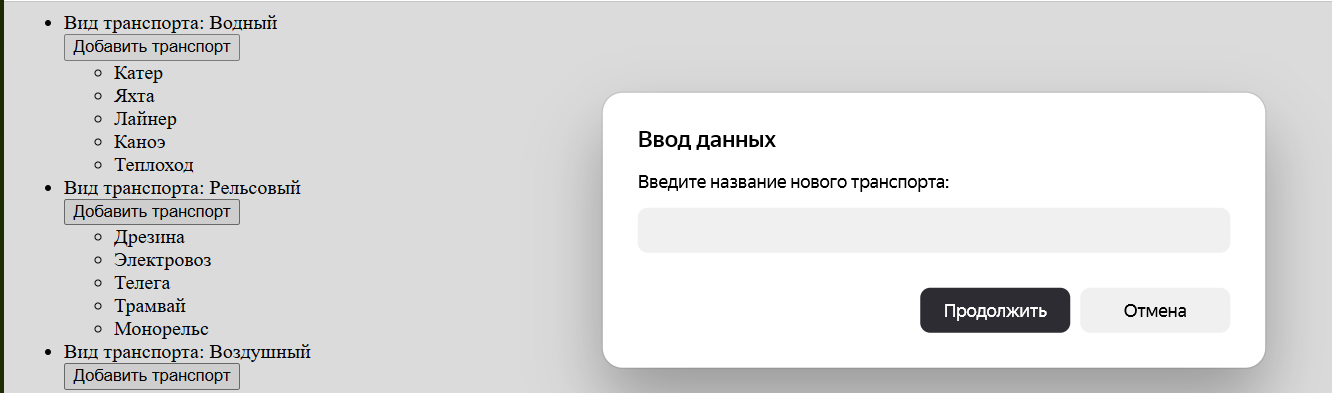
* + 1. Отображение в браузере

При открытии браузера высветится окно



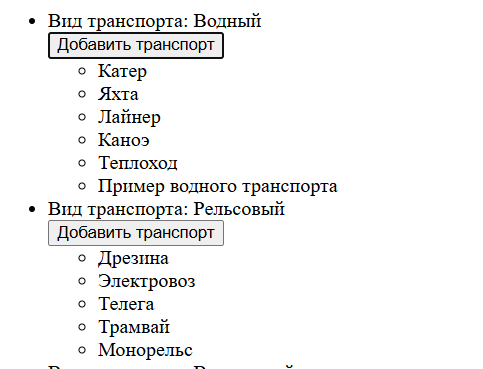
1. Начальный вид

При нажатии на одну из кнопок «добавить транспорт» в соответствующий список будет предложено ввести значение.



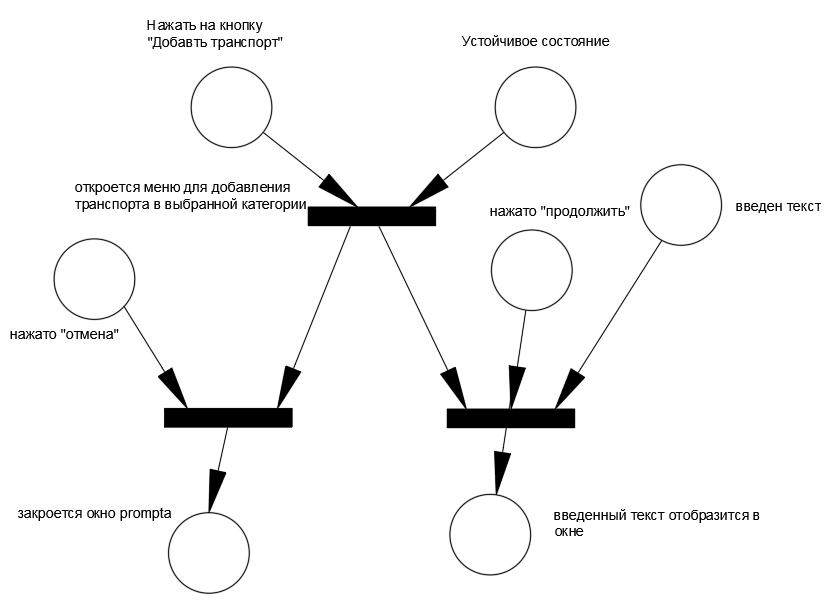
1. Добавление транспорта

Теперь при введем текст «Пример водного транспорта» внутрь окна и нажмем «продолжить».



1. Окно предупреждения

Как видно список пополнился, при этом не затрагивая другие списки.



1. Сети Петри
   1. Наблюдатели (watcher)
      1. html

<DIV ID=appVue>

<div ><ol><li v-for="(item, index) in items" :key="index">

{{ item }}

</li></ol><input type="text" v-model="message" @keydown="addPoint"/>

</div><DIV><p>Литвиненкова ТУУ-171 ©2025</p></DIV>

* + 1. Скрипт

new Vue({

el: '#appVue',

data: {

message: '',

items: []

},

methods: {

addPoint(event) {

// Проверяем, нажата ли клавиша пробела

if (event.key === ' ') {

event.preventDefault(); // Предотвращаем добавление пробела в input

if (this.message.length > 0 && this.message.length <= 5) {

this.items.push(this.message); // Добавляем значение в список

this.message = ''; // Очищаем input

}}}},

watch: {

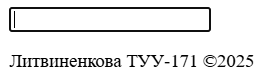
message(newValue) {

// Если длина message больше 5, очищаем список

if (newValue.length > 5) {

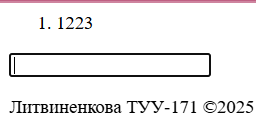
this.items = [];}}}});

* + 1. Отображение в браузере



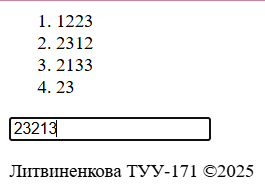
1. Начальное отображение в браузере

Введем текст и нажмем «пробел».



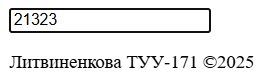
1. Ввод данных

Введем еще несколько элементов.

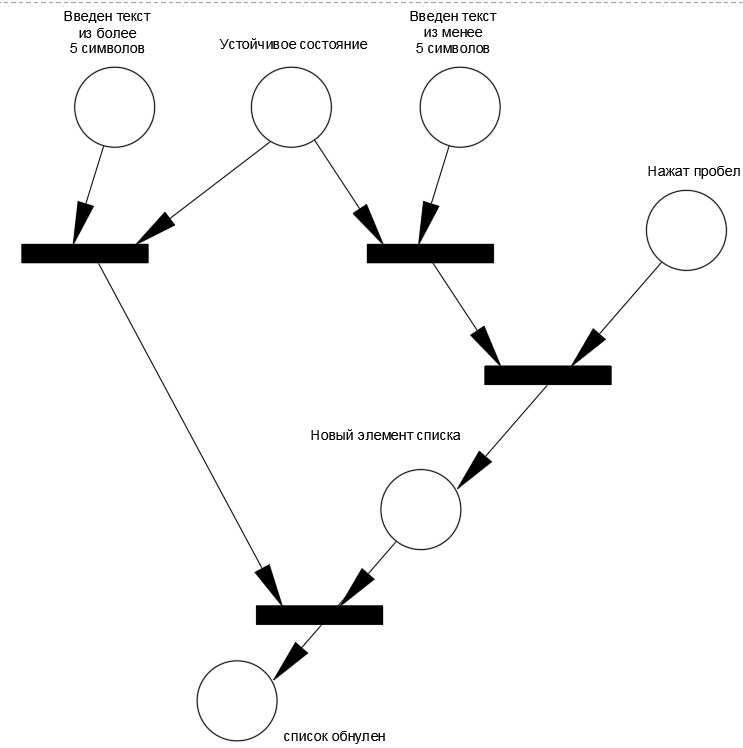


1. Ввод до 6 элементов

При вводе еще одного символа все элементы исчезнут.



1. Ввод 6-го элемента



1. Сети Петри
2. Вывод

Изучены инструменты фреймворка Vue.js такие как: Витки жизненного цикла, v-for и наблюдатели.