Czy warto uczyć się Vue w 2022r i dlaczego nie?

Przemysław Beigert
Lead developer @ Omniscopy

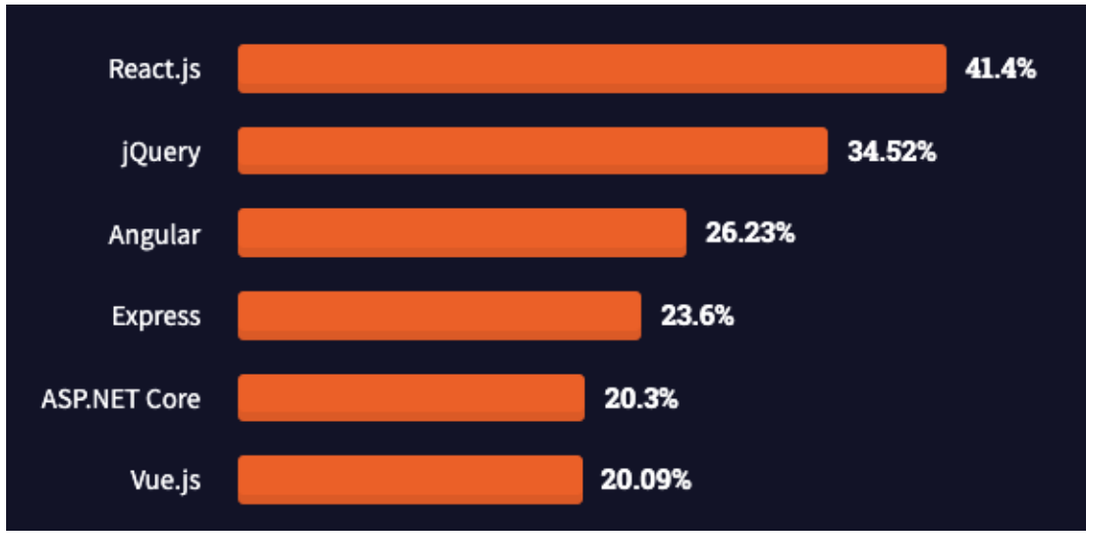
https://github.io/przemyslawjanpietrzak
https://twitter.com/przemyslawjanp
https://dev.to/przemyslawjanpietrzak

Spis treści

Rozdział I

Wprowadzenie do Vue

Stackoverflow survey 2021



```
<template>
  <div>
    <div>Tasks: {{ count }}</div>
    <div v-if="isLoading">spinner</div>
    <ul>
      <li v-for="task in tasks" :key="task.id">
        <todo-item :task="task" />
      </li>
    </ul>
    <button @click="addTodo(todo)">add todo</button>
  </div>
</template>

<script>
import TodoItem from "./TodoItem.vue";

export default {
  name: "TodoApp",
  components: {
    TodoItem,
  },
  props: ['tasks'],
```

https://github.com/vuejs

- Vuex/Pinia - state manger
- Router
- Jest plugin and utils
- Eslint plugin
- CLI
- Types
- Dev Tools
- VSCode plugin

Rozdział II

Porównanie

I/O

I/O React

```
1 import { Child } from "./child";
2
3 export const Component = ({ input, output }) => (
4   <Child onClick={() => output(42)} input={input}>{input}</Child>
5 );
```

I/O React + TS

```tsx
import { Child } from "./child";

interface Props {
  input: string;
  output: (arg: number) -> void;
}

export const Component: React.FC<Props> = ({ input, output }) => (
  <Child onClick={() => output(42)} input={input}>{input}</Child>
);
```

I/O React + JS

```
1  import PropTypes from "prop-types";
2  import { Child } from "./child";
3
4  const Component = ({ input, output }) => (
5    <Child onClick={() => output(42)} input={input}>{input}</Child>
6  );
7
8  Component.propTypes = {
9    input: PropTypes.string,
10   output: PropTypes.func,
11 }
```

```vue
<template>
  <child @click="onEvent($event)" :input="input">{{ input }}</child>
</template>

<script>
import Child from "./Child.vue";

export default {
  components: {
    Child,
  },
  emits: ["output"], // since v3, optional
  props: {
    input: {
      type: String,
      required: true,
      validate(input) {
        return true;
      },
    },
  },
  actions: {
```

I/O Angular

```
1  import { Component, Input } from '@angular/core';
2  import { Observable } from 'rxjs';
3
4  @Component({
5      selector: 'app-todo-item',
6      template: '
7       <child (click)="output.emit(42)" [input]="input">
8        {{ input }} {{ input$ | async }}
9       </child>',
10 })
11 export class AppComponent {
12   @Input() input: string;
13   @Input() input$: Observable<string>;
14   @Output() output = new EventEmitter<number>();
15 }
```

Styles

Styles React

```
1 import styled from "styled-components";
2
3 export const Element = styled.div`
4   color: white;
5   &:last-child {
6     margin-bottom: 0;
7   }
8
9   ${(props) => `width: ${props.width}%`}
```

Styles Vue

```
<template>
  <div>42</div>
  <child />
</template>

<styles scoped lang="scss">
@import "colors.scss";
div {
  color: $blue;
}
:v-deep span {
  color: red;
}
</styles>
```

Styles Angular

```
1  import { Component, ViewEncapsulation } from '@angular/core';
2
3  @Component({
4      selector: 'app-component',
5      stylesUrl: './app-component.scss',
6      template: '<div>42</div>',
7      encapsulation: ViewEncapsulation.None | ViewEncapsulation.Emulated | ViewEncapsulation.ShadowDom,
8      animations: [],
9  })
10 export class AppComponent {}
```

Forms

Form React

```
1 import { useState } from "react";
2
3 export const Form = () => {
4   const [input, setInput] = useState("");
5   return <input value={input} onChange={(e) => setInput(e.target.value)} />;
6 };
```

```jsx
import { useForm } from "react-hook-form";

export const Component = () => {
  const {
    register,
    handleSubmit,
    formState: { errors },
  } = useForm();
  const onSubmit = (data) => {};

  return (
    <form onSubmit={handleSubmit(onSubmit)}>
      <input defaultValue="Jan" {...register("firstName")} />

      <input {...register("lastName", { required: true })} />
      {errors.required && <span>This field is required</span>}

      <input type="submit" />
    </form>
  );
}
```

Form Vue

```
<template><input v-model="name" /> {{ name }}</template>

<script>
export default {
  name: "Component",
  data() {
    return {
      name: "",
    };
  },
};
</script>
```

Form Angular

```
1 import { NgModule } from '@angular/core';
2 import { ReactiveFormsModule } from '@angular/forms';
3
4 @NgModule({
5   imports: [
6     ReactiveFormsModule
7   ],
8 })
9 export class AppModule {}
```

```
@Component({
  selector: 'form-component',
  template: `
  <form [formGroup]="form">
    <input
      id="firstName"
      type="text"
      [class.is-invalid]="formValidator.isDirty(form, 'firstName')"
      formControlName="firstName"
    >
    <button type="submit" (submit)="onSubmit()">Submit</button>
  </form>
  `,
})
export class FormComponent {
  form: FormGroup;
  constructor(private fb: FormBuilder) {
    this.form = this.fb.group({
      firstName: [null, [Validators.required, Validators.minLength(3), Validators.maxLength(42)]],
      lastName: [null, [Validators.required, Validators.minLength(3), Validators.maxLength(42)]],
    });
  }
```

Update

Update React

```
1 import { useEffect } from "react";
2
3 export const Component = ({ attr }) => {
4   useEffect(() => {
5     console.log(attr);
6   }, [attr]);
7
8   return <div>{input}</div>;
9 };
```

Update Vue

```
<template><div>{{ field }}</div></template>

<script>
import TodoItem from "./TodoItem.vue";

export default {
  methods: {
    increase() {
      this.field++;
    },
  },
  watch: {
    field(newValue, oldValue) {},
    'some.nested.key'(newValue, oldValue) {}
  },
};
</script>
```

Update Angular

```
1  import {Component, OnChanges, SimpleChanges} from '@angular/core';
2
3  @Component({
4    selector: 'app-component',
5    templateUrl: './app.component.html'
6  })
7  export class AppComponent implements OnChanges {
8    attr: string;
9    ngOnChanges({ attr }: SimpleChanges) {
10      if (attr.oldValue !== attr.newValue) {
11          // HERE
12      }
13   }
14 }
```

Update Angular

```
1  import {Component, Input} from '@angular/core';
2
3  @Component({
4    selector: 'app-component',
5    templateUrl: './app.component.html'
6  })
7  export class AppComponent {
8    private _attr: string;
9
10   get attr() {
11     return this._attr;
12   }
13   @Input('attr') set attr(value: string) {
14     this._attr = value;
15     // HERE
16   }
17 }
```

Ale ...

Store

Redux

```
1  import produce from "immer";
2
3  export const reducer = (state = initialState, action) => {
4    switch (action.type) {
5      case "counter/incremented":
6        return { ...state, value: state.value + 1 };
7      case "counter/decrement":
8        return produce(state, (draft) => {
9          draft.value--;
10       });
11     default:
12       return state;
13   }
14 };
```

Redux

```
1   import { createAction } from "@reduxjs/toolkit";
2
3  export const increment = createAction("counter/increment");
4
5  export const increment = (amount) => {
6    return {
7      type: INCREMENT,
8      payload: amount,
9    };
10 };
```

Redux

```
1  import { createDraftSafeSelector, createSelector } from "@reduxjs/toolkit";
2
3  export const selectSelf = (state) => state;
4  export const unsafeSelector = createSelector(
5    selectSelf,
6    (state) => state.value
7  );
8  export const draftSafeSelector = createDraftSafeSelector(
9    selectSelf,
10   (state) => state.value
11 );
```

Vuex

```
1  import { createStore } from "vuex";
2
3  const moduleA = {
4    state: () => ({}),
5    mutations: {},
6    actions: {},
7    getters: {},
8  };
9
10 export const store = createStore({
11   modules: {
12     a: moduleA,
13   },
14 });
```

Vuex

```
export const actions = {
  actionA({ commit, dispatch, store }, payload) {
    commit("increment", payload);
    dispatch("actionB", store.data);

    return 42;
  },
};
```

Vuex

```
export const mutations = {
  increment(state, payload) {
    state.count += payload;
    Vue.set(state.arr, index, payload);
  },
};
```

Vuex

```
<script>
import { mapActions, mapGetters } from "vuex";

export default {
  computed: {
    ...mapGetters("module2", ["valueB2"])
  }
  methods: {
    ...mapActions("module1", ["actionA1"]),
    method() {
      const actionResult = this.actionA1(this.valueB2);
    },
  },
};
</script>
```

Vuex

```
<script>
import { mapMutations, mapState } from "vuex";

export default {
  computed: {
    ...mapState("module2", "store2")
  }
  methods: {
    ...mapMuttions("module1", ["mutation"]),
    method() {
      this.mutation(this.valueFromGetter, this.store2);
    },
  },
};
</script>
```

NGRX

```
@Injectable()
export class AppEffects {
  constructor(
    private actions$: Actions<AppActions>,
    private apiService: ApiService,
    private store$: Store<RootStore>,
  ) {}

  @Effect()
  data$ = this.actions$.pipe(
    ofType<Action>(AppAcions.Action),
    withLatestFrom(this.store$.select(selector)),
    switchMap(([payload, dataFromSelector]) => {
      return this.apiService
        .fetch()
        .pipe(map((response) => new OtherAction(response)),);
    }),
```

NGRX

```
@Component({
  template: `
    <div *ngIf="isLoading$ | async">loading<div>
    <div *ngElse>{{ data$ | async }}<div>
  `
})
export class AppComponent implements OnInit {
  isLoading$: Observable<boolean>;
  data$: Observable<Date>;

  constructor(private store$: Store<RootStore>) {}

  ngOnInit() {
    this.isLoading$ = this.store$.pipe(select(selectIsLoading));
    this.data$ = this.store$.pipe(select((store) => store.data));

   this.store$.dispatch(new Action());
```

Wnioski

Unit testing

Testy Vue

```
1  import { shallowMount } from "@vue/test-utils";
2  import Component from "./Component.vue";
3
4  describe("Badge", () => {
5    const createComponent = ({ label = "" } = {}) => {
6      return shallowMount(Component, {
7        propsData: {
8          label,
9        },
10       components: {
11         Child,
12       },
13     });
14   };
15
16   it.each(["test42", "wpłynąłem na suchego przestwór oceanu"])(
17     "should render text from label props",
```

Testy React

```
1  import React from "react";
2  import { shallow } from "enzyme";
3
4  import Component from "./Component";
5  import Child from "./Child";
6
7  describe("<MyComponent />", () => {
8    it("renders three <Child /> components", () => {
9      const wrapper = shallow(<Component />);
10     expect(wrapper.find(Child)).to.have.lengthOf(3);
11   });
12
13   it("renders a label, () => {
14     const wrapper = shallow(<MyComponent />);
15     expect(wrapper.find("label")).toContains("label");
16   });
17 });
```

Testy Angular

```
1  import { TestBed, async } from '@angular/core/testing';
2  import { AppComponent } from './app.component';
3
4  describe('AppComponent', () => {
5    beforeEach(async(() => {
6      TestBed.configureTestingModule({
7        declarations: [
8          AppComponent,
9        ],
10     }).compileComponents();
11   }));
12
13   it('should render title in a h1 tag', () => {
14     const fixture = TestBed.createComponent(AppComponent);
15     fixture.detectChanges();
16     const compiled = fixture.debugElement.nativeElement;
17     expect(compiled.querySelector('.label').textContent).toContain('label');
```

TypeScript

Vue + TypeScript

Wewnątrz komponentu	Od 3.0
Pomiędzy komponentemi	Brak wsparcia
HTML	Brak wsparcia
Wewnątrz stora	Niełatwo, ale się da
store <-> component	Tylko przez composition

Rozdział III

Use casy

Use casy

Małe aplikacja

Wiele widoków, ale mało logiki

Niewielki zespół

PoC / Demo

Gdyby...

Vue

React

Angular

Rozdział V

Czy warto uczyć się Vue?

Kiedy się warto uczyć Vue

Junior developer?
Senior developer?
Angular developer?
Backend developer?

Czy warto uczyś się Vue w 2022 roku?

Raczej nie

Dziękuję :*