React vs Vue vs Stimulus

Каков наш выбор?

•••

Автор: Гринцалев А. В.

План

- Описание;
- Плюсы / минусы;
- Сравнение в парадигме использования в RoR;
- Итоги;
- QA;
- ~~Анекдот;~~
- ~~Овации.~~

В чем разница между библиотекой и фреймворком?

Библиотека

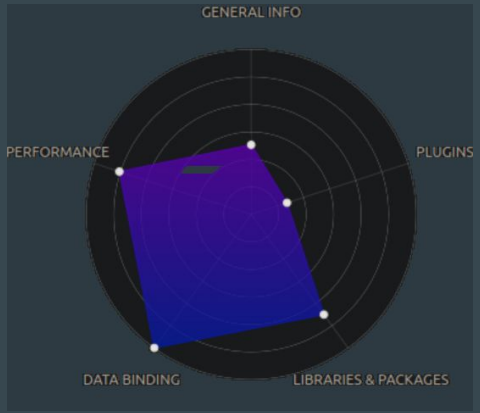
— это набор функций и компонентов, которые помогают в определенных задачах, но не навязывают общую архитектуру приложения. Разработчик имеет большую свободу выбора остальных инструментов и архитектурных решений.

Фреймворк

— это комплексное решение, которое предоставляет набор инструментов, библиотек и правил для разработки приложений. Фреймворк часто определяет структуру приложения, а разработчику предоставляется меньше свободы в выборе инструментов и архитектурных решений.

React

- Библиотека;
- Open-source;
- Component-based;
- Cross-platform



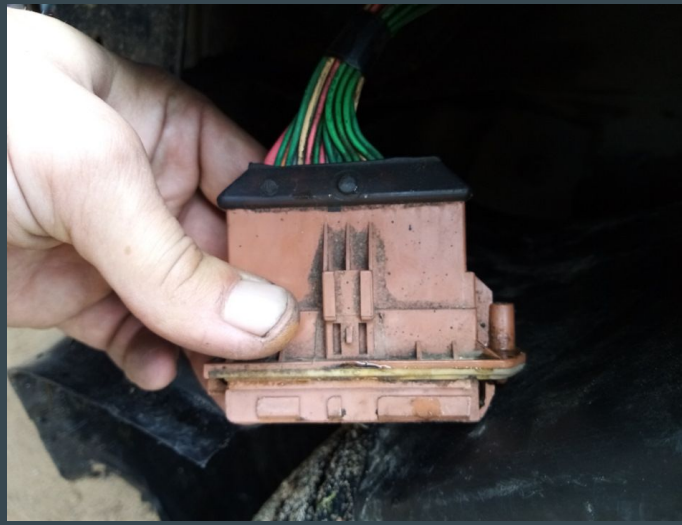
Цифры

- Самый популярный front-end framework 2023;
- 370k мемберов в Reddit;
- 216k звездочек на GitHub;
- 3230 вакансий HeadHunter;
- ~2530$ средняя ЗП по России;

Hello, World!

```
1  import ReactDOM from 'react-dom/client';
2
3  const root = ReactDOM.createRoot(document.getElementById('root'));
4  root.render(
5    <h1>Hello, world!</h1>
6  );
7
```

Фишки



Универсальность

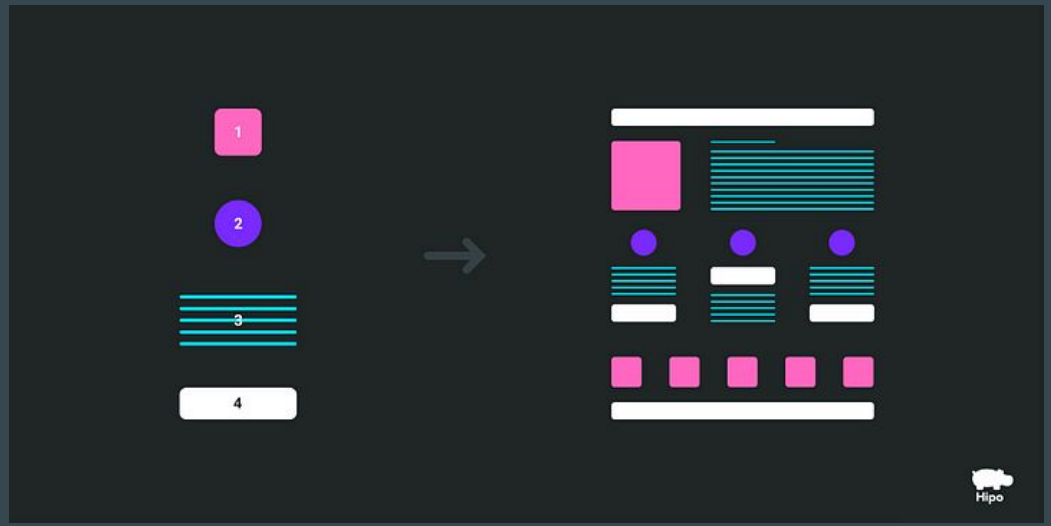
- Библиотека может быть использована на сервере, на мобильных платформах с помощью React Native.
  Learn Once, Write Anywhere.

Декларативность

- С помощью React разработчик описывает, как компоненты интерфейса выглядят в разных состояниях. Декларативный подход сокращает код и делает его понятным.

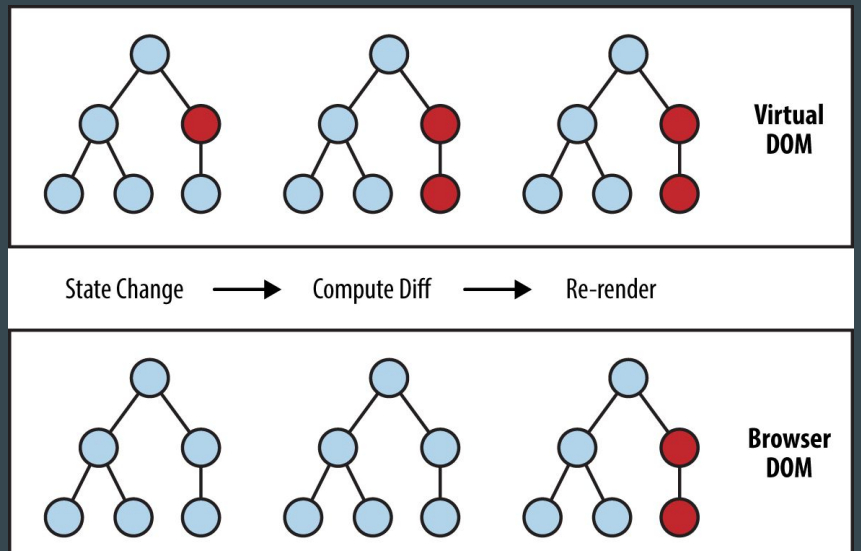
Component-based

- React основан на компонентах, это ещё одна ключевая особенность библиотеки. Каждый компонент возвращает часть пользовательского интерфейса со своим состоянием. Объединяя компоненты, программист создаёт завершённый интерфейс веб-приложения.



Виртуальный DOM

- Облегченное представление в памяти реального DOM. Позволяет React обновлять только те части пользовательского интерфейса, которые изменились, что делает его более эффективным по сравнению с другими библиотеками, которые проводят ререндеринг всей страницы.



DOM Component Lifecycle

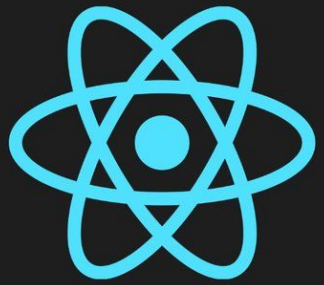
- Initialization;
- Mounting;
- Updation;
- Unmounting

JSX

- Расширение JS, позволяющее описывать UI с помощью собственного синтаксиса, подобного XML.

```
const App = () => {
   return (
     <div>
       <p>Header</p>
       <p>Content</p>
       <p>Footer</p>
     </div>
   );
}
```

```
: yelp_styleguide.ybutton as ybtn

#def render()
    <div class="button-container">
        $ybtn.ybutton(
            size=$ybtn.Size.BIG,
            content='Awesome Button',
        )
    </div>
#end def
```



ecosystem 2023

SSR

```
1  ReactDOMServer.renderToString()
2
3  ReactDOM.hydrate()
```

Webpack

```
rails new myApp --webpack=react
```

```
app/javascript:
  └── packs:
      └── application.js
      └── hello_react.jsx
```

react-rails

- “react_component” helper method;

```
var componentRequireContext = require.context("components", true);
var ReactRailsUJS = require("react_ujs");
ReactRailsUJS.useContext(componentRequireContext);
```

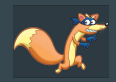
```
<%= react_component("Hello") %>
```

react_on_rails

```
1  import ReactOnRails from 'react-on-rails';
2
3  import HelloWorld from
   '../bundles/HelloWorld/components/HelloWorld';
4
5  // This is how react_on_rails can see the HelloWorld in the
   browser.
6  ReactOnRails.register({
7    HelloWorld,
8  });
```

Плюсы

+ Модульность;
+ Virtual DOM;
+ JSX;
+ Однонаправленный поток данных;
+ Декларативность;
+ Широкое комьюнити;

Минусы 

- Не фреймворк;
- Virtual DOM;
- Проблемы SEO;

Vue

- Framework;
- Open-source;
- Component-based;
- Cross-platform;

Цифры

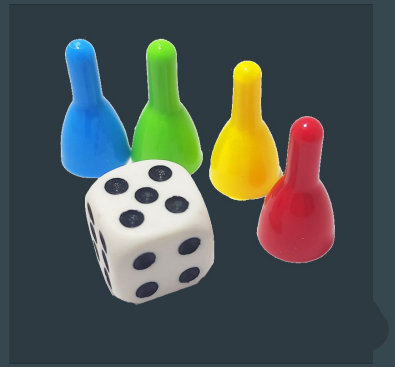
- Один из самых популярных front-end framework’ов 2023;
- 99k в Reddit;
- 206k звездочек на GitHub для 2й версии, 41,7k для 3й;
- 1878 вакансий HeadHunter;
- ~1866$ средняя ЗП по России;

Hello, World!

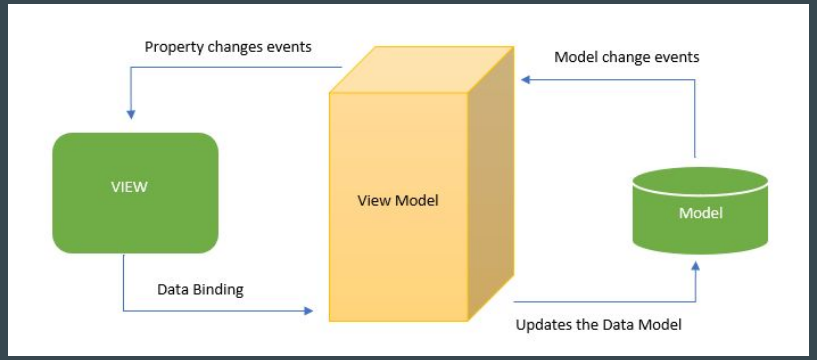
```
import { createApp } from 'vue'
import App from './App.vue'

const app = createApp(App)
app.mount('#app')
```

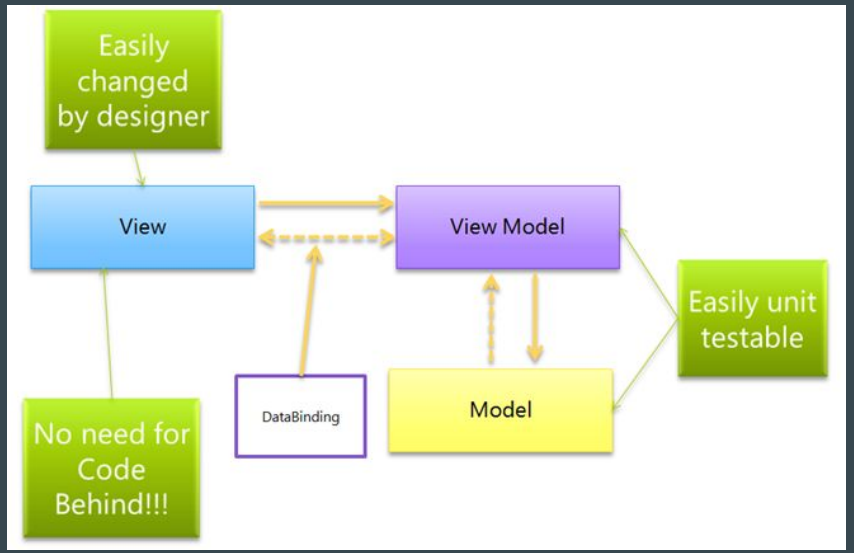
```
<template>
  <h1>Hello, World!</h1>
</template>
```

Фишки

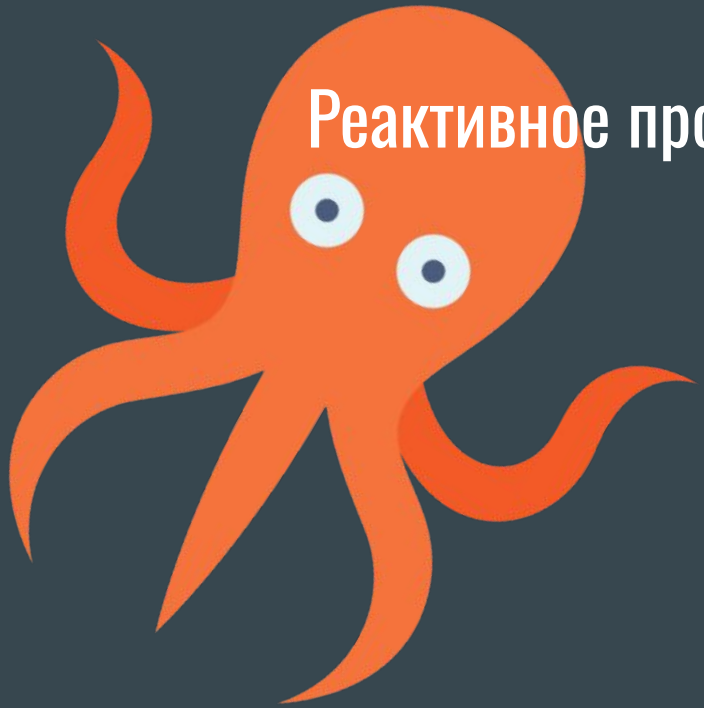


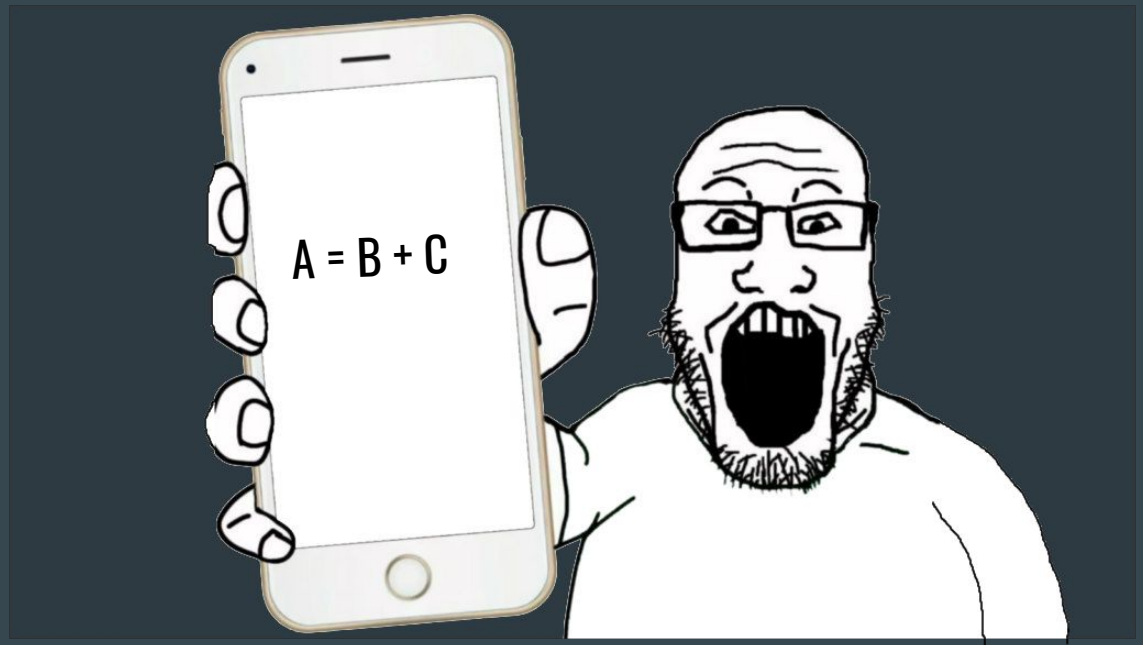
MVVM


Property changes events
Model change events
VIEW
View Model
Model
Data Binding
Updates the Data Model

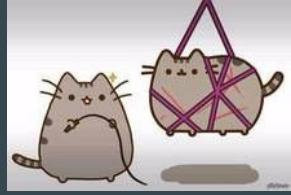

Easily changed by designer
View
View Model
Easily unit testable
DataBinding
Model
No need for Code Behind!!!

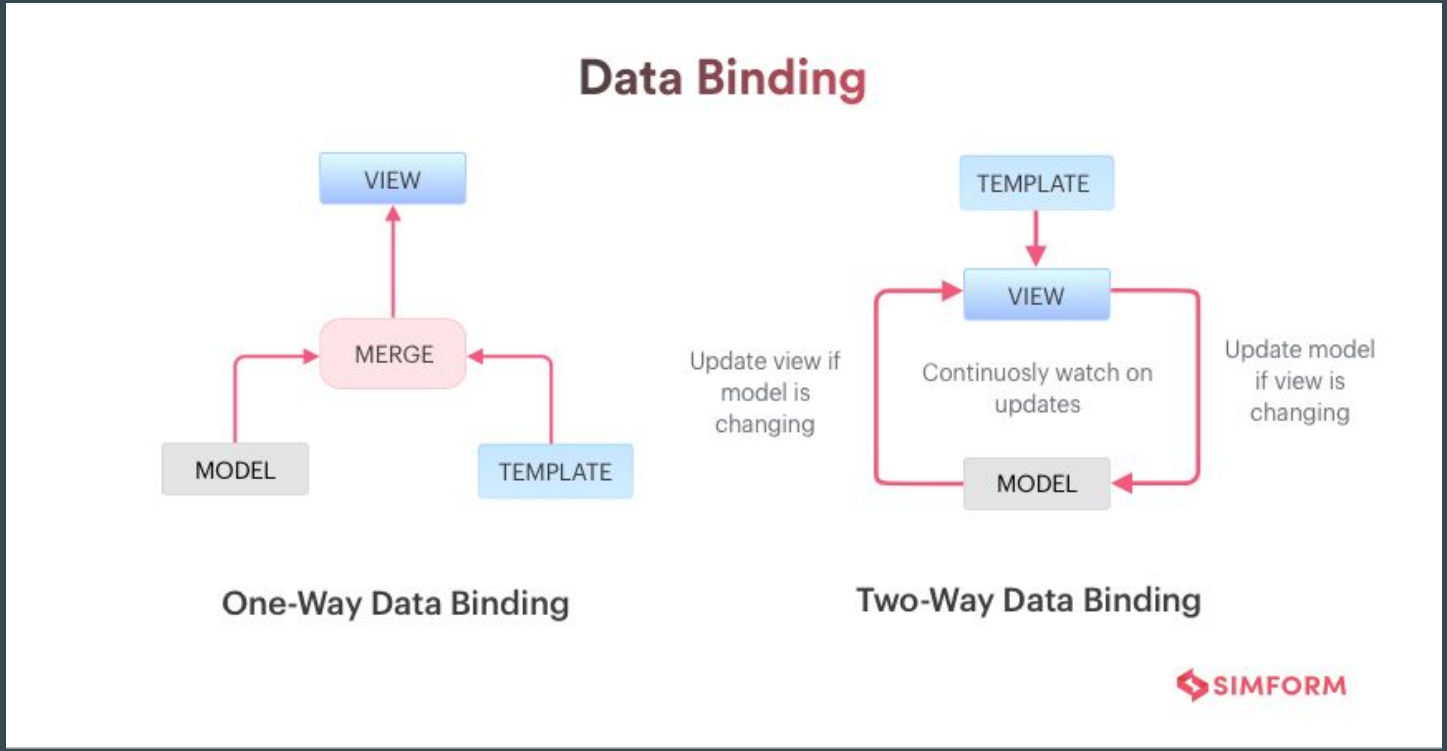
Реактивное программирование




A = B + C

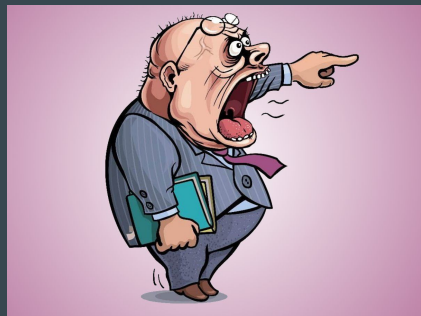
Двустороннее связывание



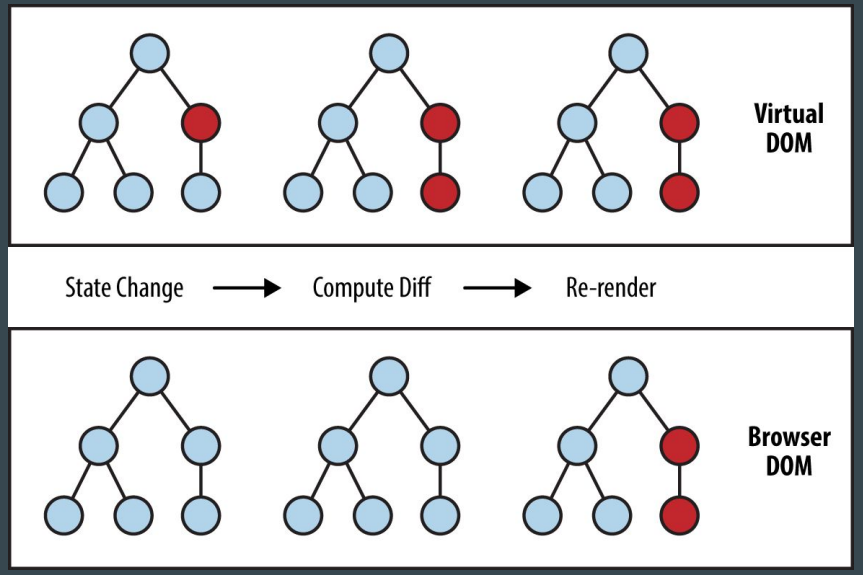


Директивы

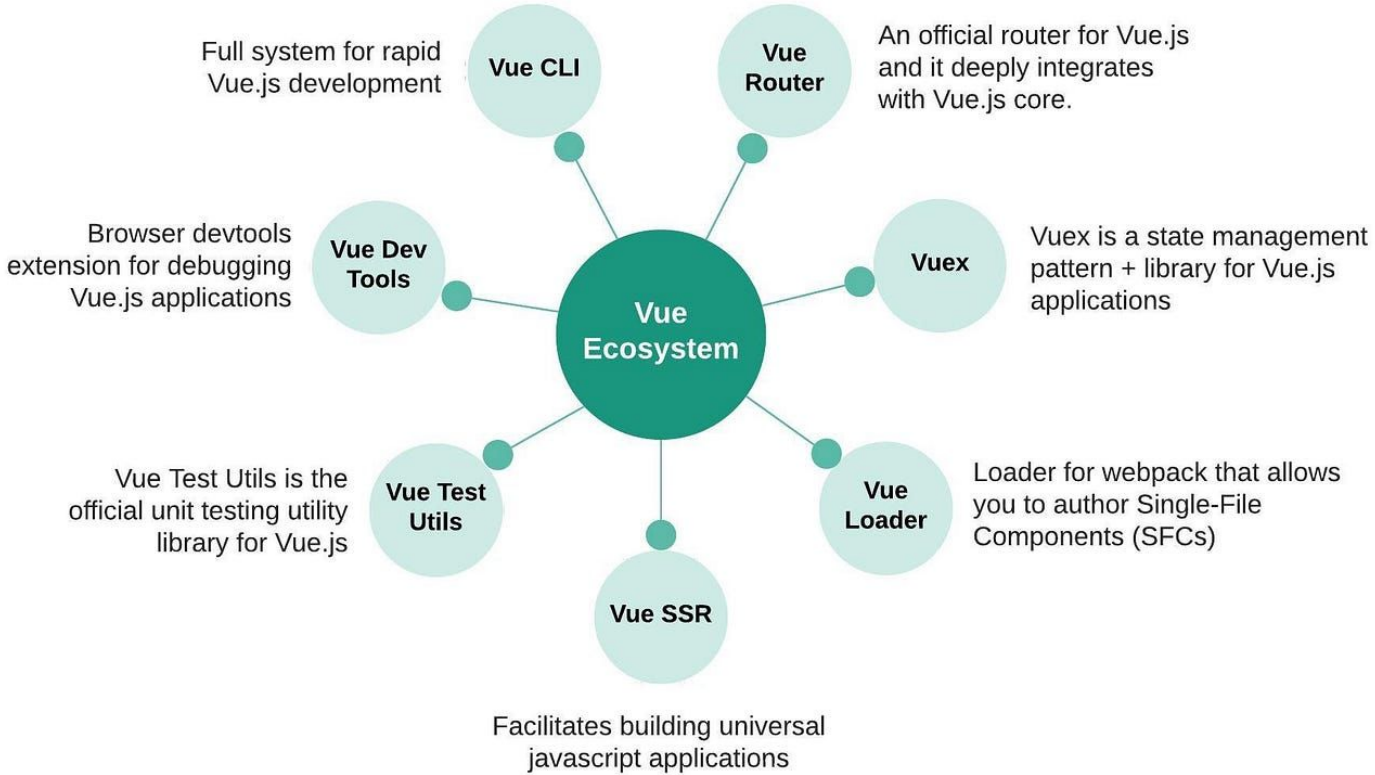
Директивы - специальные атрибуты с префиксом v-. В качестве значения атрибуты принимают одно выражение JavaScript (кроме v-for). Директива реактивно применяет к DOM изменения при обновлении значения этого выражения.



Virtual DOM


Virtual DOM
State Change
Compute Diff
Re-render
Browser DOM

Vue Ecosystem




SSR

Webpacker

```
rails new myApp --webpacker=vue
```

```
rails webpacker:install:erb
```

```
yarn add webpack webpack-cli pnp-webpack-plugin
yarn add turbolinks vue-turbolinks
yarn add rails-ujs activestorage css-loader
yarn add jquery bootstrap popper.js
yarn add axios vue-axios
```

Full Guide

Inertia.js

```
bundle install inertia_rails
```

```
yarn add @inertiajs/inertia @inertiajs/inertia-vue
```

```
import Vue from 'vue'
import { createInertiaApp } from '@inertiajs/inertia-vue'

createInertiaApp({
  resolve: name => require(`../pages/${name}`),
  setup({ el, App, props }) {
    new Vue({
      render: h => h(App, props),
    }).$mount(el)
  },
})
```

```
def index
  render inertia: 'Animals/Index',
         props: { animals: scope.map { |a| a.attributes.slice('id', 'name', 'kind', 'animal_class') } }
end

def show
  animal = Animal.find(params[:id])

  render inertia: 'Animals/Show', props: { animal: animal }
end
```

```vue
<script>
  import Layout from '../Layout'
  import { Link } from '@inertiajs/inertia-vue'

  export default {
    components: {
      Layout,
      Link,
    },
    props: {
      animals: Array,
    }
  }
</script>
```

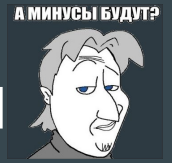
```vue
<template>
  <Layout>
    <ul id="animal-list">
      <li v-for="animal in animals" :key="animal.id">
        <Link :href="`/animals/${animal.id}`">
          {{ animal.name }}
        </Link>
      </li>
    </ul>
  </Layout>
</template>
```

Плюсы

+ Скорость;;
+ Фреймворк;
+ HTML-шаблоны + JSX;
+ Lightweight;

+ Широкое комьюнити;

Минусы



- Чат-сообщество;
- Проблемы SEO;

Stimulus (и Turbo)

- Framework (micro);
- Open-source;
- Minimalistic

Stimulus

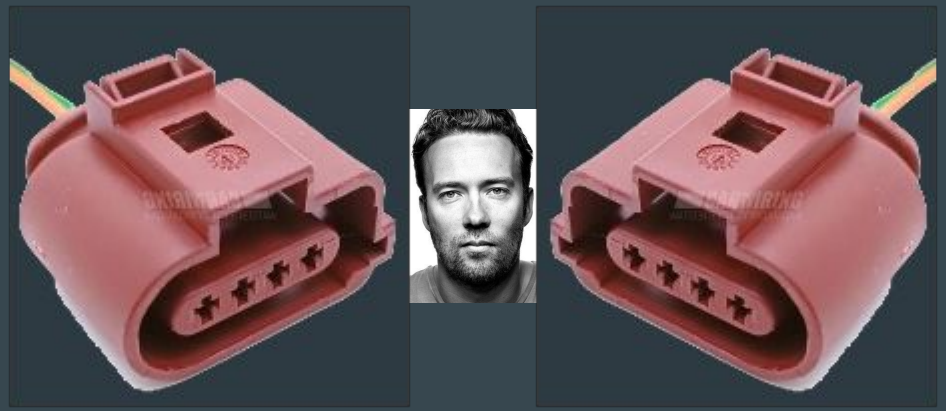
Пример кода Stimulus

Пример кода Turbo

```
<turbo-frame id="new_message">
  <form action="/messages" method="post">
    ...
  </form>
</turbo-frame>
```

```
<turbo-stream action="replace" target="message_1">
  <template>
    <div id="message_1">This changes the existing message!</div>
  </template>
</turbo-stream>
```

Фишки



Controllers

Actions, Targets, Outlets, Values

data-action

data-search-target

this.nameTarget

this.nameTargets

this.hasNameTargets

data-[identifier]-[outlet]-outlet="[selector]"

```
<div>
  <div class="online-user" data-controller="user-status">...</div>
  <div class="online-user" data-controller="user-status">...</div>
  ...
</div>

...

<div data-controller="chat" data-chat-user-status-outlet=".online-user">
  ...
</div>
```

Turbo:
Turbo Drive, Turbo Frames, Turbo Streams, Turbo Native

```
<turbo-frame id="messages" src="/messages">
  <p>This message will be replaced by the response from /messages.</p>
</turbo-frame>
```

```
<turbo-stream action="append" target="messages">
  <template>
    <div id="message_1">My new message!</div>
  </template>
</turbo-stream>
```

Сравнение технологий

Каков наш выбор?

JQuery.

Спасибо за внимание!