**Масштабирование**

# - Роутинг

## [Официальный роутер](https://ru.vuejs.org/v2/guide/routing.html#%D0%9E%D1%84%D0%B8%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9-%D1%80%D0%BE%D1%83%D1%82%D0%B5%D1%80)

Роутер туралы толық ақпарат

## [Простой роутер с нуля](https://ru.vuejs.org/v2/guide/routing.html#%D0%9F%D1%80%D0%BE%D1%81%D1%82%D0%BE%D0%B9-%D1%80%D0%BE%D1%83%D1%82%D0%B5%D1%80-%D1%81-%D0%BD%D1%83%D0%BB%D1%8F)

Егер сіз үшін қарапайым маршрутизатор жеткілікті болса және сіз толыққанды сыртқы кітапхананы пайдаланғыңыз келмесе, бет деңгейіндегі компонентті динамикалық түрде көрсету жеткілікті болуы мүмкін:

const NotFound = { template: '<p>Страница не найдена</p>' }

const Home = { template: '<p>главная</p>' }

const About = { template: '<p>о нас</p>' }

const routes = {

'/': Home,

'/about': About

}

new Vue({

el: '#app',

data: {

currentRoute: window.location.pathname

},

computed: {

ViewComponent () {

return routes[this.currentRoute] || NotFound

}

},

render (h) { return h(this.ViewComponent) }

})

## [Интеграция сторонних роутеров](https://ru.vuejs.org/v2/guide/routing.html#%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B0%D1%86%D0%B8%D1%8F-%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D1%85-%D1%80%D0%BE%D1%83%D1%82%D0%B5%D1%80%D0%BE%D0%B2)

Page.js немесе Director сияқты үшінші тарап маршрутизаторын пайдаланғыңыз келсе, біріктіру де өте оңай.

# -Управление состоянием приложения

## [Официальная Flux-подобная библиотека](https://ru.vuejs.org/v2/guide/state-management.html#%D0%9E%D1%84%D0%B8%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F-Flux-%D0%BF%D0%BE%D0%B4%D0%BE%D0%B1%D0%BD%D0%B0%D1%8F-%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0)

Сложность больших приложений ... компонентам и связям между ними. Для решения этой проблемы, Vue предлагает **[Vuex](https://github.com/vuejs/vuex" \t "_blank)**

### #[Информация для React-разработчиков](https://ru.vuejs.org/v2/guide/state-management.html#%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D1%8F-%D0%B4%D0%BB%D1%8F-React-%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%87%D0%B8%D0%BA%D0%BE%D0%B2) Redux ???

Егер сіз React көмегімен Vue-ге ауысатын болсаңыз, сіз React экожүйесіндегі Flux-тің ең танымал іске асырылуы Vuex және Redux қалай байланысты екендігі туралы сұрақ туындауы мүмкін.

Redux - бұл агностикалық көру қабаты, сондықтан оны Vue көмегімен қарапайым байланыстыру арқылы тікелей пайдалануға болады.

Vuex, керісінше, жақсы интеграцияға, интуитивті API интерфейсіне мүмкіндік беретін Vue қолданбасымен жұмыс істейтінін біледі және нәтижесінде дамуды жағымды етеді.

## [Простой контейнер состояния с нуля](https://ru.vuejs.org/v2/guide/state-management.html#%D0%9F%D1%80%D0%BE%D1%81%D1%82%D0%BE%D0%B9-%D0%BA%D0%BE%D0%BD%D1%82%D0%B5%D0%B9%D0%BD%D0%B5%D1%80-%D1%81%D0%BE%D1%81%D1%82%D0%BE%D1%8F%D0%BD%D0%B8%D1%8F-%D1%81-%D0%BD%D1%83%D0%BB%D1%8F)

var sourceOfTruth = {};

var vmA = new Vue({

data: sourceOfTruth

});

var vmB = new Vue({

data: sourceOfTruth

});

 доступ — используя this.$root.$data

var store = {

debug: true,

state: {

message: "Привет!"

},

setMessageAction(newValue) {

if (this.debug) console.log("setMessageAction вызвано с ", newValue);

this.state.message = newValue;

},

clearMessageAction() {

if (this.debug) console.log("clearMessageAction вызвано");

this.state.message = "";

}

};

**шақырылған кезде ішіндегі данныйларды өзгерте алу үшін**

var vmA = new Vue({

data: {

privateState: {},

sharedState: store.state

}

});

var vmB = new Vue({

data: {

privateState: {},

sharedState: store.state

}

});

Каждый экземпляр/компонент... может иметь собственное

Осы кодтарды жетілдіріп барсақ ақырында Vuex-ке барамыз

Бұл бөлімнің нәтижесі Vuex - ті меңгеру

# - SSR. Отрисовка на стороне сервера

Сервер жағынан көрсету

## [Полное руководство по SSR](https://ru.vuejs.org/v2/guide/ssr.html#%D0%9F%D0%BE%D0%BB%D0%BD%D0%BE%D0%B5-%D1%80%D1%83%D0%BA%D0%BE%D0%B2%D0%BE%D0%B4%D1%81%D1%82%D0%B2%D0%BE-%D0%BF%D0%BE-SSR)

Біз серверлік рендерингпен Vue қолданбаларын жасау үшін жеке оқу құралын жасадық. Нұсқаулық ssr.vuejs.org сайтында орналасқан.

## Что такое серверный рендеринг (SSR)?

Әдепкі бойынша, Vue құрамдастары браузерде DOM жасайды және басқарады. Дегенмен, сервердегі HTML жолдарына бірдей құрамдастарды көрсетуге, оларды браузерге жіберуге және ақырында статикалық белгілеуді клиенттегі толық интерактивті қолданбаға «гидрлеуге» болады.

Серверде тұрып DOM-ды басқаруға, сол арқылы SEO-ны жақсартуға көмектеседі,яғни кленттерге барынша қолайлы қылу тәсілдері

Бұл нәрсені істеуде серверде тұрып жаңа HTML-код жасауға мәжбүрлегеннен көрі, алдын ала frod-end кезінде статикалық (яғный жан-жақты) HTML файл жасау жақсырақ, бұл frod-end-ті толығымен статикалық ұстауға мүмкіндңк береді

руковотства demo - https://vue-hn.herokuapp.com/top

[**Что такое SEO**](https://promo.ingate.ru/seo-wikipedia/seo/)**. ???**

Search Engine Optimization

Іздеу жүйесін оңтайландыру - бұл трафикті алу үшін іздеу нәтижелерінде веб-сайтты жылжыту технологиясы.

## [Nuxt.js](https://ru.vuejs.org/v2/guide/ssr.html#Nuxt-js)

Nuxt.js — жылдам, жан-жақты Vue қолданбасын әзірлеу тәжірибесін қамтамасыз ету үшін Vue экожүйесінің үстіне салынған жоғары деңгейлі құрылым.

SSR мен Nuxt бірдей қызмет атқаратын екі тіл

## [Quasar Framework SSR + PWA](https://ru.vuejs.org/v2/guide/ssr.html#Quasar-Framework-SSR-PWA)

# Quasar Framework сіздің идеяңызды әзірлеу мен жүзеге асыруды қарапайым және оңай ету үшін класстағы ең жақсы құрастыру жүйесін, ақылға қонымды әдепкі конфигурацияны және әзірлеушінің оңай кеңейтілуін пайдаланатын SSR (қосымша PWA) қолданбасын жасауға мүмкіндік береді.

Жүзден астам Material Design 2.0 үйлесімді құрамдастары арқылы серверде қайсысы іске қосылатынын, қайсысы браузерде қолжетімді екенін шеше аласыз және тіпті сайтыңыздың <meta> тегтерін басқара аласыз.

Quasar — ​​SPA, PWA, SSR, Electron, Capacitor және Cordova қолданбаларының дамуын жылдамдататын және оңтайландырады, node.js және веб-пакет негізіндегі әзірлеу ортасы - барлығы бір код базасынан.

# -Безопасность ???

## [Сообщение об уязвимости](https://ru.vuejs.org/v2/guide/security.html#%D0%A1%D0%BE%D0%BE%D0%B1%D1%89%D0%B5%D0%BD%D0%B8%D0%B5-%D0%BE%D0%B1-%D1%83%D1%8F%D0%B7%D0%B2%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8)

Осалдық туралы есеп

Егер Vue бойынша кемшіліктер болса біздің почтаға жіберіңіз, ешқандай кемшілік кездеспеседе сақтың үшін Vue қосымшасның соңғы үлгісін қолданыңыз.

## [Правило №1: Никогда не использовать ненадёжные шаблоны](https://ru.vuejs.org/v2/guide/security.html#%D0%9F%D1%80%D0%B0%D0%B2%D0%B8%D0%BB%D0%BE-%E2%84%961-%D0%9D%D0%B8%D0%BA%D0%BE%D0%B3%D0%B4%D0%B0-%D0%BD%D0%B5-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D1%8C-%D0%BD%D0%B5%D0%BD%D0%B0%D0%B4%D1%91%D0%B6%D0%BD%D1%8B%D0%B5-%D1%88%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D1%8B)

Сенімсіз шаблондарды қолданбаңыз

new Vue({

el: '#app',

template: `<div>` + userProvidedString + `</div>` // НИКОГДА ТАК НЕ ДЕЛАЙТЕ

})

## [Что делает Vue для вашей защиты](https://ru.vuejs.org/v2/guide/security.html#%D0%A7%D1%82%D0%BE-%D0%B4%D0%B5%D0%BB%D0%B0%D0%B5%D1%82-Vue-%D0%B4%D0%BB%D1%8F-%D0%B2%D0%B0%D1%88%D0%B5%D0%B9-%D0%B7%D0%B0%D1%89%D0%B8%D1%82%D1%8B) ???

### #[HTML-содержимое](https://ru.vuejs.org/v2/guide/security.html#HTML-%D1%81%D0%BE%D0%B4%D0%B5%D1%80%D0%B6%D0%B8%D0%BC%D0%BE%D0%B5)

1. <h1>{{ userProvidedString }}</h1>
2. '<script>alert("hi")</script>'
3. &lt;script&gt;alert(&quot;hi&quot;)&lt;/script&gt;

### Осыла ретпен орындалады, Vue-де осылайша жазылған данныйда қателің жібермейді

### #[Привязка атрибутов](https://ru.vuejs.org/v2/guide/security.html#%D0%9F%D1%80%D0%B8%D0%B2%D1%8F%D0%B7%D0%BA%D0%B0-%D0%B0%D1%82%D1%80%D0%B8%D0%B1%D1%83%D1%82%D0%BE%D0%B2)

1. <h1 v-bind:title="userProvidedString">

hello

</h1>

1. '" onclick="alert(\'hi\')'
2. &quot; onclick=&quot;alert('hi')

данный Динамикалық жолмен берілседе Осылай қауіпсіз қылынады

## [Потенциальные опасности](https://ru.vuejs.org/v2/guide/security.html#%D0%9F%D0%BE%D1%82%D0%B5%D0%BD%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5-%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D0%B8) ???

Ықтимал қауіптер

HTML, CSS, или JavaScript  -терді тексерусіз қолдану ықтималдық қателер беруі мүмкін

### #[Внедрение HTML](https://ru.vuejs.org/v2/guide/security.html#%D0%92%D0%BD%D0%B5%D0%B4%D1%80%D0%B5%D0%BD%D0%B8%D0%B5-HTML) ???

### Жоғарыда айтылғандай, Vue HTML мазмұнын автоматты түрде алып тастап, қолданбаға кездейсоқ HTML енгізуді болдырмайды. Дегенмен, HTML қауіпсіз екеніне сенімді болсаңыз, HTML мазмұнын оның өңделмеген түрінде көрсете аласыз:

Используя шаблон:

<div v-html="userProvidedHtml"></div>

Используя render-функцию:

h('div', {

domProps: {

innerHTML: this.userProvidedHtml

}

})

Используя render-функцию с JSX:

<div domPropsInnerHTML={this.userProvidedHtml}></div>

Есіңізде болсын, егер ол оқшауланған iframe ішінде немесе HTML жазған пайдаланушы ғана оған қол жеткізе алатын қолданбаның бөлігінде болмаса, пайдаланушы қамтамасыз ететін HTML ешқашан 100% қауіпсіз болмайды. Сондай-ақ, пайдаланушыларға өздерінің Vue үлгілерін жазуға рұқсат беру ұқсас қауіптерді тудыруы мүмкін.

### #[Внедрение URL](https://ru.vuejs.org/v2/guide/security.html#%D0%92%D0%BD%D0%B5%D0%B4%D1%80%D0%B5%D0%BD%D0%B8%D0%B5-URL) ???

<a v-bind:href="userProvidedUrl">

Нажми на меня

### </a>

URL мекенжайы тазартылмаған болса, ықтимал қауіпсіздік мәселесі болуы мүмкін. Бұған көмектесетін sanitize-url сияқты кітапханалар бар, бірақ бұған қарамастан:

Егер сіз Frond-end-тағы URL мекенжайларын тазартқан болсаңыз, сізде қауіпсіздік мәселесі әлдеқашан бар. Пайдаланушы берген URL мекенжайлары дерекқорға сақталмас бұрын әрқашан серверде тазартылуы керек. Содан кейін мәселе сіздің API интерфейсіңіз үшін шешіледі. Сондай-ақ, тазартылған URL мекенжайларын пайдалану олардың қауіпсіз ресурстарға әкелетініне кепілдік бермейтінін есте сақтаңыз.

**#** [**Внедрение стилей**](https://ru.vuejs.org/v2/guide/security.html#%D0%92%D0%BD%D0%B5%D0%B4%D1%80%D0%B5%D0%BD%D0%B8%D0%B5-%D1%81%D1%82%D0%B8%D0%BB%D0%B5%D0%B9) **???**

Стильді енгізу

Сол сияқты, пайдаланушыларға <стиль> элементінің мазмұнын анықтауға рұқсат беру сізге бет мәнерлерін толық басқаруға мүмкіндік беретін одан да үлкен осалдықты тудыратынын елестете аласыз. Сондықтан, Vue үлгілер ішіндегі стиль тегтерін көрсетпеуі керек, мысалы:

<style>{{ userProvidedStyles }}</style>

өзгертуге қауіпсіз нақты сипаттардың мәндерін ғана рұқсат етеміз, мысалы:

<a

v-bind:href="sanitizedUrl"

v-bind:style="{

color: userProvidedColor,

background: userProvidedBackground

}"

>

Нажми на меня

</a>

осылай жазу арқылы қате жазылған <style> себепті қауіп тудырудың алдын алады

### #[Внедрение JavaScript](https://ru.vuejs.org/v2/guide/security.html#%D0%92%D0%BD%D0%B5%D0%B4%D1%80%D0%B5%D0%BD%D0%B8%D0%B5-JavaScript) ???

## [Лучшие практики](https://ru.vuejs.org/v2/guide/security.html#%D0%9B%D1%83%D1%87%D1%88%D0%B8%D0%B5-%D0%BF%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D0%BA%D0%B8)

Жалпы ереже: егер сіз тазартылмаған пайдаланушы жасаған мазмұнды (HTML, JavaScript немесе тіпті CSS сияқты) орындауға рұқсат етсеңіз, сіз шабуылға ашықсыз. Бұл кеңес Vue, басқа фреймворкты пайдалансаңыз да немесе мүлде қолданбасаңыз да жарамды болып қалады.

## [Координация с бэкэндом](https://ru.vuejs.org/v2/guide/security.html#%D0%9A%D0%BE%D0%BE%D1%80%D0%B4%D0%B8%D0%BD%D0%B0%D1%86%D0%B8%D1%8F-%D1%81-%D0%B1%D1%8D%D0%BA%D1%8D%D0%BD%D0%B4%D0%BE%D0%BC)

Cross Site Request Forgery (CSRF / XSRF) немесе Cross Site Scripting Injection (XSSI) сияқты HTTP қауіпсіздік осалдықтары негізінен серверге бағытталған, сондықтан Vue бұл жерде көмектеспейді. Дегенмен, пішіндерді жіберу кезінде CSRF таңбалауыштарын жіберу сияқты API-мен өзара әрекеттесу жолын жақсырақ түсіну үшін серверді әзірлеу тобымен үйлестірген дұрыс.

## [Отрисовка на стороне сервера (SSR)](https://ru.vuejs.org/v2/guide/security.html#%D0%9E%D1%82%D1%80%D0%B8%D1%81%D0%BE%D0%B2%D0%BA%D0%B0-%D0%BD%D0%B0-%D1%81%D1%82%D0%BE%D1%80%D0%BE%D0%BD%D0%B5-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80%D0%B0-SSR)

SSR пайдалану кезінде қосымша қауіпсіздік мәселелері туындауы мүмкін, сондықтан осалдықтарды болдырмау үшін SSR құжаттамасында көрсетілген нұсқауларды орындаңыз.