Введение

**[Что такое Vue.js?]**

**[**[**Начало работы**](https://ru.vuejs.org/v2/guide/index.html#%D0%9D%D0%B0%D1%87%D0%B0%D0%BB%D0%BE-%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%8B)**]**

Установка, Node.js, cmd (Командный строка) vue-cli

**[**[**Декларативная отрисовка**](https://ru.vuejs.org/v2/guide/index.html#%D0%94%D0%B5%D0%BA%D0%BB%D0%B0%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F-%D0%BE%D1%82%D1%80%D0%B8%D1%81%D0%BE%D0%B2%D0%BA%D0%B0)**]**

* <div id="app">
* {{ message }}
* </div>
* data() {
* return{
* message: 'Привет, Vue!'
* }}

**Кроме интерполяции текста, можно также связывать атрибуты элементов:**

* <div id="app-2">
* <span v-bind:title="message">
* Наведи на меня курсор на пару секунд,
* чтобы увидеть динамически связанное значение title!
* </span>
* </div>
* data: {
* message: 'Вы загрузили эту страницу: ' + new Date().toLocaleString()
* }

**[**[**Условия**](https://ru.vuejs.org/v2/guide/index.html#%D0%A3%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-%D0%B8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B)**]**

* <div id="app-3">
* <span v-if="seen">Сейчас меня видно</span>
* </div>
* data() {
* return{
* seen: true
* }}

**[Циклы]**

* <div id="app-4">
* <ol>
* <li v-for="todo in todos">
* {{ todo.text }}
* </li>
* </ol>
* </div>
* data() {
* return{
* todos: [
* { text: 'Изучить JavaScript' },
* { text: 'Изучить Vue' },
* { text: 'Создать что-нибудь классное' }
* ]
* }}

todos.push ({ text: 'Profit' })

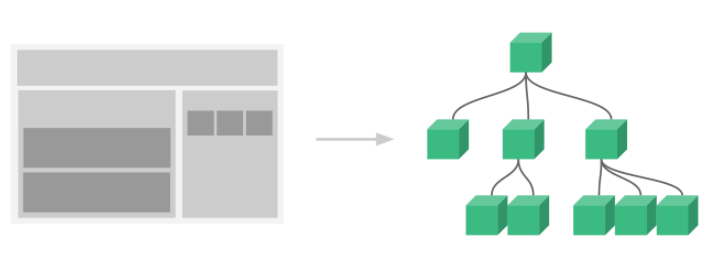
**[**[**Работа с пользовательским вводом**](https://ru.vuejs.org/v2/guide/index.html#%D0%A0%D0%B0%D0%B1%D0%BE%D1%82%D0%B0-%D1%81-%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D0%BA%D0%B8%D0%BC-%D0%B2%D0%B2%D0%BE%D0%B4%D0%BE%D0%BC) **(**v-on**)]**

* <div id="app-5">
* <p>{{ message }}</p>
* <button v-on:click="reverseMessage">Перевернуть сообщение</button>
* </div>
* data() {
* return{
* message: 'Привет, Vue.js!'
* }},
* methods: {
* reverseMessage: function () {
* this.message = this.message.split('').reverse().join('')
* } }

**[Vue также предоставляет директиву v-model]**

* <div id="app-6">
* <p>{{ message }}</p>
* <input v-model="message">
* </div>
* data() {
* return{
* message: 'Привет, Vue!'
* }}

**[**[**Разбиение приложения на компоненты**](https://ru.vuejs.org/v2/guide/index.html#%D0%A0%D0%B0%D0%B7%D0%B1%D0%B8%D0%B5%D0%BD%D0%B8%D0%B5-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D1%8B)**]**

****

* <div id="app-7">
* <ol>
* <todo-item
* v-for="item in groceryList"
* v-bind:todo="item"
* v-bind:key="item.id" >
* </todo-item>
* </ol>
* </div>
* data() {
* return{
* groceryList: [
* { id: 0, text: 'Овощи' },
* { id: 1, text: 'Сыр' },
* { id: 2, text: 'Что там ещё люди едят?' }
* ]
* }},
* components:{
* todo-item,
* }

-------------------

* data() {
* return{
* props: ['todo'],
* }}

---------------------

**Бірнеше рет қолдануға болады**

* <div id="app">
* <app-nav></app-nav>
* <app-view>
* <app-sidebar></app-sidebar>
* <app-content></app-content>
* </app-view>
* </div>

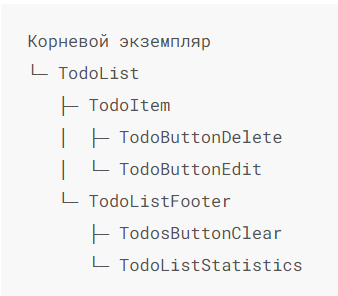
**[**[**Отношение к пользовательским элементам Web Components**](https://ru.vuejs.org/v2/guide/index.html#%D0%9E%D1%82%D0%BD%D0%BE%D1%88%D0%B5%D0%BD%D0%B8%D0%B5-%D0%BA-%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D0%BA%D0%B8%D0%BC-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D0%BC-Web-Components)**]**

Vue реализуют [**API слотов**](https://github.com/w3c/webcomponents/blob/gh-pages/proposals/Slots-Proposal.md) и специальный атрибут is

Экземпляр Vue

**[**[**Создание экземпляра Vue**](https://ru.vuejs.org/v2/guide/instance.html#%D0%A1%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-%D1%8D%D0%BA%D0%B7%D0%B5%D0%BC%D0%BF%D0%BB%D1%8F%D1%80%D0%B0-Vue)**]**

* var vm = new Vue({
* // опции
* })



**[**[**Данные и методы**](https://ru.vuejs.org/v2/guide/instance.html#%D0%94%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5-%D0%B8-%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D1%8B)**]**

* data: {
* newTodoText: '',
* visitCount: 0,
* hideCompletedTodos: false,
* todos: [],
* error: null
* }

---------------------

* var obj = {
* foo: 'bar'
* }
* Object.freeze(obj) // Өзгермейтіндей қылу

---------------------

* new Vue({
* el: '#app',
* data: obj
* })

--------------------------

* <div id="app">
* <p>{{ foo }}</p>
* <!-- мы теперь не можем обновить `foo`! -->
* <button v-on:click="foo = 'baz'">Изменить</button>
* </div>

**[**[**Хуки жизненного цикла экземпляра**](https://ru.vuejs.org/v2/guide/instance.html#%D0%A5%D1%83%D0%BA%D0%B8-%D0%B6%D0%B8%D0%B7%D0%BD%D0%B5%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE-%D1%86%D0%B8%D0%BA%D0%BB%D0%B0-%D1%8D%D0%BA%D0%B7%D0%B5%D0%BC%D0%BF%D0%BB%D1%8F%D1%80%D0%B0)**]**

[**created**](https://ru.vuejs.org/v2/api/#created) -01

new Vue({

data: {a: 1 },

created: function () {

// `this` указывает на экземпляр vm

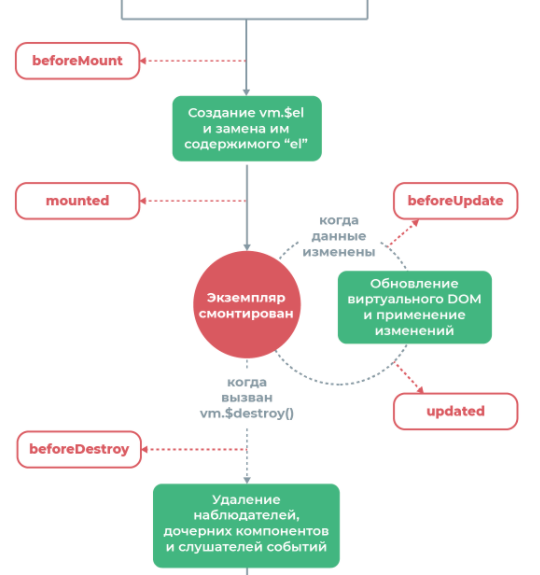
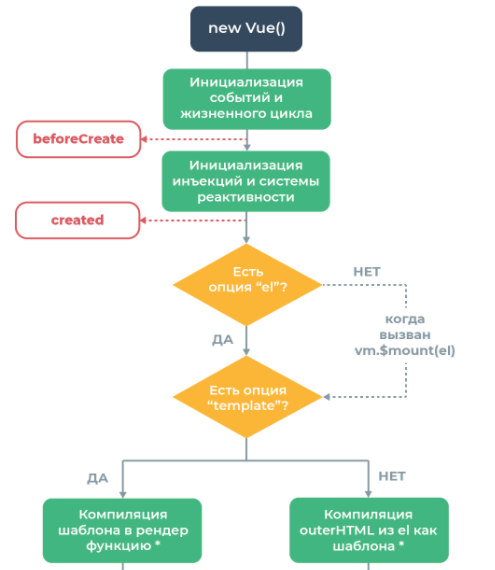
console.log('Значение a: ' + this.a)

}})

[**mounted**](https://ru.vuejs.org/v2/api/#mounted)(){ қандай да бір оқиға жазылады чтобы HTML - ден кейін орындалатын }

*html-02,* [**mounted**](https://ru.vuejs.org/v2/api/#mounted) - 03 , [**updated**](https://ru.vuejs.org/v2/api/#updated) - 04  и [**destroyed**](https://ru.vuejs.org/v2/api/#destroyed) - 05

**[**[**Диаграмма жизненного цикла**](https://ru.vuejs.org/v2/guide/instance.html#%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0-%D0%B6%D0%B8%D0%B7%D0%BD%D0%B5%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE-%D1%86%D0%B8%D0%BA%D0%BB%D0%B0)**]**





Синтаксис шаблонов

**[**[**Интерполяции**](https://ru.vuejs.org/v2/guide/syntax.html#%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D0%BE%D0%BB%D1%8F%D1%86%D0%B8%D0%B8)**]** *(render)*

**#**[**Текст**](https://ru.vuejs.org/v2/guide/syntax.html#%D0%A2%D0%B5%D0%BA%D1%81%D1%82)

<span>Сообщение: {{ msg }}</span>

<span v-once>Это сообщение никогда не изменится: {{ msg }}</span>

**#**[**Сырой HTML**](https://ru.vuejs.org/v2/guide/syntax.html#%D0%A1%D1%8B%D1%80%D0%BE%D0%B9-HTML)

<p>Двойные фигурные скобки: {{ rawHtml }}</p>

<p>Директива v-html: <span v-html="rawHtml"></span></p>

**#**[**Атрибуты**](https://ru.vuejs.org/v2/guide/syntax.html#%D0%90%D1%82%D1%80%D0%B8%D0%B1%D1%83%D1%82%D1%8B)

<div v-bind:id="dynamicId"></div>

<button v-bind:disabled="isButtonDisabled">Кнопка</button>

 null, undefined или false, то атрибут disabled

#[**Использование выражений JavaScript**](https://ru.vuejs.org/v2/guide/syntax.html#%D0%98%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%B2%D1%8B%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9-JavaScript) **(**Math и Date)

{{ number + 1 }}

{{ ok ? 'YES' : 'NO' }}

{{ message.split('').reverse().join('') }}

<div v-bind:id="'list-' + id"></div>

{{ if (ok) { return message } }}

[[Директивы](https://ru.vuejs.org/v2/guide/syntax.html#%D0%94%D0%B8%D1%80%D0%B5%D0%BA%D1%82%D0%B8%D0%B2%D1%8B)]

<p v-if="seen">Сейчас меня видно</p>

#[**Аргументы**](https://ru.vuejs.org/v2/guide/syntax.html#%D0%90%D1%80%D0%B3%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D1%8B)

<a v-bind:href="url"> ... </a>

<a v-on:click="doSomething"> ... </a>

**#**[**Динамические аргументы**](https://ru.vuejs.org/v2/guide/syntax.html#%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5-%D0%B0%D1%80%D0%B3%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D1%8B)

<a v-bind:[attributeName]="url"> ... </a>

<a v-on:[eventName]="doSomething"> ... </a>

v-on:focus.

<a v-bind:['foo' + bar]="value"> ... </a>

<a v-bind:[someAttr]="value"> ... </a>

**#**[**Модификаторы**](https://ru.vuejs.org/v2/guide/syntax.html#%D0%9C%D0%BE%D0%B4%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80%D1%8B)

<form v-on:submit.prevent="onSubmit"> ... </form>

*event.preventDefault()*

[**v-on**](https://ru.vuejs.org/v2/guide/events.html#%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5-%D0%BC%D0%BE%D0%B4%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80%D1%8B-%D1%81%D0%BE%D0%B1%D1%8B%D1%82%D0%B8%D0%B9) и [**v-model**](https://ru.vuejs.org/v2/guide/forms.html#%D0%9C%D0%BE%D0%B4%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80%D1%8B).

[[Сокращения](https://ru.vuejs.org/v2/guide/syntax.html#%D0%A1%D0%BE%D0%BA%D1%80%D0%B0%D1%89%D0%B5%D0%BD%D0%B8%D1%8F)]

### #[Сокращение v-bind](https://ru.vuejs.org/v2/guide/syntax.html#%D0%A1%D0%BE%D0%BA%D1%80%D0%B0%D1%89%D0%B5%D0%BD%D0%B8%D0%B5-v-bind)

<!-- полный синтаксис -->

<a v-bind:href="url"> ... </a>

<!-- сокращённая запись -->

<a : href="url"> ... </a>

<!-- сокращённая запись с динамическим именем аргумента (2.6.0+) -->

<a : [key]="url"> ... </a>

### # [Сокращение v-on](https://ru.vuejs.org/v2/guide/syntax.html" \l "%D0%A1%D0%BE%D0%BA%D1%80%D0%B0%D1%89%D0%B5%D0%BD%D0%B8%D0%B5-v-on" \o "Сокращение v-on)

<!-- полный синтаксис -->

<a v-on:click="doSomething"> ... </a>

<!-- сокращённая запись -->

<a @click="doSomething"> ... </a>

<!-- сокращённая запись с динамическим именем события (2.6.0+) -->

<a @[event]="doSomething"> ... </a>

**Вычисляемые свойства и слежение**

[[Вычисляемые свойства](https://ru.vuejs.org/v2/guide/computed.html" \l "%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D0%B5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0" \o "Вычисляемые свойства)]

data(), methods(), computed(),

watch() - *(*$watch) ????? өзгерісті анықтап ажырата алады

***Мысалы:***

*<select v-model="functionName">*

*<option value="all"*

*watch (){*

*functionName (value){*

*console.log(value) // select-тің value-сінің өзгерісін анықтайды*

*}*

*}*

filters() -ішіне қандайда бір функция жазып оны шақыратын кезде

todo.title | functionName белгісі арқылы шақырады

<div id="example">

{{ message.split('').reverse().join('') }}

</div>

**#**[**Простой пример**](https://ru.vuejs.org/v2/guide/computed.html#%D0%9F%D1%80%D0%BE%D1%81%D1%82%D0%BE%D0%B9-%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D1%80)

<div id="example">

<p>Изначальное сообщение: «{{ message }}»</p>

<p>Сообщение задом наперёд: «{{ reversedMessage }}»</p>

</div>

var vm = new Vue({

el: '#example',

data: {

message: 'Привет'

},

computed: {

// геттер вычисляемого значения

reversedMessage: function () {

// `this` указывает на экземпляр vm

return this.message.split('').reverse().join('')

}

}

})

**#[Кэширование вычисляемых свойств](https://ru.vuejs.org/v2/guide/computed.html" \l "%D0%9A%D1%8D%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D1%85-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2" \o "Кэширование вычисляемых свойств)**

computed: {

now: function () {

return Date.now()

}}

**#**[**Вычисляемые свойства и слежение**](https://ru.vuejs.org/v2/guide/computed.html#%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D0%B5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-%D0%B8-%D1%81%D0%BB%D0%B5%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) **(**watch)

<div id="demo">{{ fullName }}</div>

var vm = new Vue({

el: '#demo',

data: {

firstName: 'Foo', lastName: 'Bar', fullName: 'Foo Bar'

},

watch: {

firstName: function (val) {

this.fullName = val + ' ' + this.lastName

},

lastName: function (val) {

this.fullName = this.firstName + ' ' + val

}

}

})

**#[Сеттеры вычисляемых свойств](https://ru.vuejs.org/v2/guide/computed.html" \l "%D0%A1%D0%B5%D1%82%D1%82%D0%B5%D1%80%D1%8B-%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D1%85-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2" \o "Сеттеры вычисляемых свойств)**

computed: {

fullName: {

// геттер:

get: function () {

return this.firstName + ' ' + this.lastName

},

// сеттер:

set: function (newValue) {

var names = newValue.split(' ')

this.firstName = names[0]

this.lastName = names[names.length - 1]

}}}

**[**[**Методы-наблюдатели**](https://ru.vuejs.org/v2/guide/computed.html#%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D1%8B-%D0%BD%D0%B0%D0%B1%D0%BB%D1%8E%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D0%B8) **(**watch[функцияны ажырата, талқылай алу]**)]**

<div id="watch-example">

<p>

Задайте вопрос, на который можно ответить «да» или «нет»:

<input v-model="question">

</p>

<p>{{ answer }}</p>

</div>

<script src="https://cdn.jsdelivr.net/npm/axios@0.12.0/dist/axios.min.js">

</script>

<script src="https://cdn.jsdelivr.net/npm/lodash@4.13.1/lodash.min.js">

</script>

<script>

var watchExampleVM = new Vue({

el: '#watch-example',

data: {

question: '',

answer: 'Пока вы не зададите вопрос, я не могу ответить!'

},

watch: {

question: function (newQuestion, oldQuestion) {

this.answer = 'Ожидаю, когда вы закончите печатать...'

this.debouncedGetAnswer()

}

},

created: function () {

this.debouncedGetAnswer = \_.debounce(this.getAnswer, 500)

},

methods: {

getAnswer: function () {

if (this.question.indexOf('?') === -1) {

this.answer = 'Вопросы обычно заканчиваются вопросительным знаком. ;-)'

return

}

this.answer = 'Думаю...'

var vm = this

axios.get('https://yesno.wtf/api')

.then(function (response) {

vm.answer = \_.capitalize(response.data.answer)

})

.catch(function (error) {

vm.answer = 'Ошибка! Не могу связаться с API. ' + error

})

}

}

})

</script>

Работа с классами и стилями

[Связывание CSS-классов](https://ru.vuejs.org/v2/guide/class-and-style.html#%D0%A1%D0%B2%D1%8F%D0%B7%D1%8B%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-CSS-%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%BE%D0%B2)

**#**[**Использование объектов**](https://ru.vuejs.org/v2/guide/class-and-style.html#%D0%98%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BE%D0%B2) v-bind:class:

<div v-bind:class="{ active: isActive }"></div>

<div

class="static"

v-bind:class="{ active: isActive, 'text-danger': hasError }">

</div>

data: {

isActive: true,

hasError: false

}

**результат**

<div class="static active"></div>

**Используемый объект необязательно указывать прямо в шаблоне:**

<div v-bind:class="classObject"></div>

data: {

classObject: {

active: true,

'text-danger': false

}}

**функция возвращают объект**

<div v-bind:class="classObject"></div>

data: {

isActive: true,

error: null

},

computed: {

classObject: function () {

return {

active: this.isActive && !this.error,

'text-danger': this.error && this.error.type === 'fatal'

}}}

**#**[**Использование массивов**](https://ru.vuejs.org/v2/guide/class-and-style.html#%D0%98%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2)

В  можно передать и массив v-bind:class

<div v-bind:class="[activeClass, errorClass]"></div>

data: {

activeClass: 'active',

errorClass: 'text-danger'

}

**результат:** <div class="active text-danger"></div>

**можно использовать условный**

<div v-bind:class="[isActive ? activeClass : '', errorClass]"></div>

<div v-bind:class="[{ active: isActive }, errorClass]"></div>

**#** [**Использование с компонентами**](https://ru.vuejs.org/v2/guide/class-and-style.html#%D0%98%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D1%81-%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D0%B0%D0%BC%D0%B8)

Vue.component('my-component', {

template: '<p class="foo bar">Привет</p>'

})

<my-component class="baz boo"></my-component>

**В результате :** <p class="foo bar baz boo">Привет</p>

<my-component v-bind:class="{ active: isActive }"></my-component>

**В результате :** <p class="foo bar active">Привет</p>

[**Связывание inline-стилей**](https://ru.vuejs.org/v2/guide/class-and-style.html#%D0%A1%D0%B2%D1%8F%D0%B7%D1%8B%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-inline-%D1%81%D1%82%D0%B8%D0%BB%D0%B5%D0%B9)

#[Использование объектов](https://ru.vuejs.org/v2/guide/class-and-style.html#%D0%98%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BE%D0%B2-1)

<div v-bind:style="{color: activeColor, fontSize: fontSize + 'px' }"></div>

data: {

activeColor: 'red',

fontSize: 30

}

**Можно выносить объект стилей из шаблона**

<div v-bind:style="styleObject"></div>

data: {

styleObject: {

color: 'red',

fontSize: '13px'

}}

#[Использование массивов](https://ru.vuejs.org/v2/guide/class-and-style.html#%D0%98%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-1)

Запись v-bind:style **с массивом** позволяет

<div v-bind:style="[baseStyles, overridingStyles]"></div>

**#**[Автоматические префиксы](https://ru.vuejs.org/v2/guide/class-and-style.html" \l "%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5-%D0%BF%D1%80%D0%B5%D1%84%D0%B8%D0%BA%D1%81%D1%8B" \o "Автоматические префиксы) ???? Vue.js - браузерге қарап стилдерді өзі өзгертіп отырады

#[Множественные значения](https://ru.vuejs.org/v2/guide/class-and-style.html#%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5-%D0%B7%D0%BD%D0%B0%D1%87%D0%B5%D0%BD%D0%B8%D1%8F)

**массив из нескольких (префиксных) значений для поддерживаемого браузером.**

<div v-bind:style="{ display: ['-webkit-box', '-ms-flexbox', 'flex'] }"></div>

**В этом примере он будет отображать display: flex для браузеров,**

Условная отрисовка

[v-if](https://ru.vuejs.org/v2/guide/conditional.html#v-if)

<h1 v-if="awesome">Vue восхитителен!</h1>

<h1 v-else>О, нет 😢</h1>

### #[Условные группы с использованием v-if и <template>](https://ru.vuejs.org/v2/guide/conditional.html#%D0%A3%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D1%8B%D0%B5-%D0%B3%D1%80%D1%83%D0%BF%D0%BF%D1%8B-%D1%81-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%D0%BC-v-if-%D0%B8-lt-template-gt)

<template v-if="ok">

<h1>Заголовок</h1>

<p>Абзац 1</p>

<p>Абзац 2</p>

</template>

#[v-else](https://ru.vuejs.org/v2/guide/conditional.html" \l "v-else" \o "v-else)

<div v-if="Math.random() > 0.5"> Сейчас меня видно </div>

<div v-else> А теперь — нет </div>

#[v-else-if](https://ru.vuejs.org/v2/guide/conditional.html" \l "v-else-if" \o "v-else-if)

<div v-if="type === 'A'"> A </div>

<div v-else-if="type === 'B'"> B </div>

<div v-else-if="type === 'C'"> C </div>

<div v-else> Не A/B/C </div>

#[Управление повторным использованием элементов при помощи key](https://ru.vuejs.org/v2/guide/conditional.html#%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5-%D0%BF%D0%BE%D0%B2%D1%82%D0%BE%D1%80%D0%BD%D1%8B%D0%BC-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%D0%BC-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D0%BE%D0%B2)

<template v-if="loginType === 'username'">

<label>Имя пользователя</label>

<input placeholder="Введите имя пользователя" key="username-input">

</template>

<template v-else>

<label>Email</label>

<input placeholder="Введите адрес email" key="email-input">

</template>

[v-show](https://ru.vuejs.org/v2/guide/conditional.html#v-show)

<h1 v-show="ok">Привет!</h1> изменяться будет лишь свойство display

[v-if в сравнении с v-show](https://ru.vuejs.org/v2/guide/conditional.html#v-if-%D0%B2-%D1%81%D1%80%D0%B0%D0%B2%D0%BD%D0%B5%D0%BD%D0%B8%D0%B8-%D1%81-v-show) ???? v-if  өщіріп қайтып қосады көп күш кетеді

v-show көп орын, памит алады

[v-if  вместе с  v-for](https://ru.vuejs.org/v2/guide/conditional.html#v-if-%D0%B2%D0%BC%D0%B5%D1%81%D1%82%D0%B5-%D1%81-v-for) қолданбау керек

Отрисовка списков

[**Отображение массива элементов с помощью v-for**](https://ru.vuejs.org/v2/guide/list.html#%D0%9E%D1%82%D0%BE%D0%B1%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%B0-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D0%BE%D0%B2-%D1%81-%D0%BF%D0%BE%D0%BC%D0%BE%D1%89%D1%8C%D1%8E-v-for)

slice(0,-1)

split("+")

<ul id="example-1">

<li v-for="item in items" :key="item.message">

{{ item.message }}

</li>

</ul>

var example1 = new Vue({

el: '#example-1',

data: {

items: [

{ message: 'Foo' },

{ message: 'Bar' }

]

}})

**с индексом текущего элемента.**

<ul id="example-2">

<li v-for="(item, index) in items">

{{ parentMessage }} - {{ index }} - {{ item.message }}

</li>

</ul>

var example2 = new Vue({

el: '#example-2',

data: {

parentMessage: 'Родитель',

items: [

{ message: 'Foo' },

{ message: 'Bar' }

]

}})

**вместо in разделителем можно использовать of**

### # [v-for для объекта](https://ru.vuejs.org/v2/guide/list.html" \l "v-for-%D0%B4%D0%BB%D1%8F-%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%B0" \o "v-for для объекта)

<ul id="v-for-object" class="demo">

<li v-for="value in object">

{{ value }}

</li>

</ul>

new Vue({

el: '#v-for-object',

data: {

object: {

title: 'How to do lists in Vue',

author: 'Jane Doe',

publishedAt: '2016-04-10'

}

}

})

**Можно использовать второй аргумент для получения имени свойства (ключа объекта):**

<div v-for="(value, name) in object">

{{ name }}: {{ value }}

</div>

<div v-for="(value, name, index) in object">

{{ index }}. {{ name }}: {{ value }}

</div>

[Сохранение состояния](https://ru.vuejs.org/v2/guide/list.html" \l "%D0%A1%D0%BE%D1%85%D1%80%D0%B0%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5-%D1%81%D0%BE%D1%81%D1%82%D0%BE%D1%8F%D0%BD%D0%B8%D1%8F" \o "Сохранение состояния)

<div v-for="item in items" v-bind:key="item.id">

<!-- содержимое -->

</div>

[**Отслеживание изменений в массивах**](https://ru.vuejs.org/v2/guide/list.html#%D0%9E%D1%82%D1%81%D0%BB%D0%B5%D0%B6%D0%B8%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%B8%D0%B7%D0%BC%D0%B5%D0%BD%D0%B5%D0%BD%D0%B8%D0%B9-%D0%B2-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%B0%D1%85)

#[Методы внесения изменений](https://ru.vuejs.org/v2/guide/list.html" \l "%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D1%8B-%D0%B2%D0%BD%D0%B5%D1%81%D0%B5%D0%BD%D0%B8%D1%8F-%D0%B8%D0%B7%D0%BC%D0%B5%D0%BD%D0%B5%D0%BD%D0%B8%D0%B9" \o "Методы внесения изменений)

push(), pop(), shift(), unshift(), splice(), sort(), reverse()

example1.items.push({ message: 'Baz' }).

#[Замены в массиве](https://ru.vuejs.org/v2/guide/list.html#%D0%97%D0%B0%D0%BC%D0%B5%D0%BD%D1%8B-%D0%B2-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%B5)

filter(), concat() и slice()

example1.items = example1.items.filter(function (item) {

return item.message.match(/Foo/)

})

**#**[**Предостережения**](https://ru.vuejs.org/v2/guide/list.html#%D0%9F%D1%80%D0%B5%D0%B4%D0%BE%D1%81%D1%82%D0%B5%D1%80%D0%B5%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F)

[Отображение отфильтрованных/отсортированных результатов](https://ru.vuejs.org/v2/guide/list.html#%D0%9E%D1%82%D0%BE%D0%B1%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5-%D0%BE%D1%82%D1%84%D0%B8%D0%BB%D1%8C%D1%82%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-%D0%BE%D1%82%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-%D1%80)

<li v-for="n in evenNumbers">{{ n }}</li>

data: {

numbers: [ 1, 2, 3, 4, 5 ]

},

computed: {

evenNumbers: function () {

return this.numbers.filter(function (number) {

return number % 2 === 0

})

}}

--------------------------

<ul v-for="set in sets">

<li v-for="n in even(set)">{{ n }}</li>

</ul>

data: {

sets: [[ 1, 2, 3, 4, 5 ], [6, 7, 8, 9, 10]]

},

methods: {

even: function (numbers) {

return numbers.filter(function (number) {

return number % 2 === 0

})

}}

**#** [v-for и диапазоны](https://ru.vuejs.org/v2/guide/list.html" \l "v-for-%D0%B8-%D0%B4%D0%B8%D0%B0%D0%BF%D0%B0%D0%B7%D0%BE%D0%BD%D1%8B" \o "v-for и диапазоны)

<div>

<span v-for="n in 10">{{ n }} </span>

</div>

# [v-for и тег template](https://ru.vuejs.org/v2/guide/list.html" \l "v-for-%D0%B8-%D1%82%D0%B5%D0%B3-template" \o "v-for и тег template)

<ul>

<template v-for="item in items">

<li>{{ item.msg }}</li>

<li class="divider" role="presentation"></li>

</template>

</ul>

#[v-for и v-if](https://ru.vuejs.org/v2/guide/list.html" \l "v-for-%D0%B8-v-if" \o "v-for и v-if)

<ul v-if="shouldRenderTodos">

<li v-for="todo in todos">

{{ todo }}

</li>

</ul>

#[Компоненты и v-for](https://ru.vuejs.org/v2/guide/list.html" \l "%D0%9A%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D1%8B-%D0%B8-v-for" \o "Компоненты и v-for)

<my-component v-for="item in items" :key="item.id"></my-component>

<my-component

v-for="(item, index) in items"

v-bind:item="item"

v-bind:index="index"

v-bind:key="item.id"

> </my-component>

------------------------- Todo Items --------------------

<div id="todo-list-example">

<form v-on:submit.prevent="addNewTodo">

<label for="new-todo">Добавить задачу</label>

<input

v-model="newTodoText"

id="new-todo"

placeholder="Например, покормить кота"

>

<button>Добавить</button>

</form>

<ul>

<li

is="todo-item"

v-for="(todo, index) in todos"

v-bind:key="todo.id"

v-bind:title="todo.title"

v-on:remove="todos.splice(index, 1)"

></li>

</ul>

</div>

Vue.component('todo-item', {

template: '\

<li>\

{{ title }}\

<button v-on:click="$emit(\'remove\')">Удалить</button>\

</li>\

',

props: ['title']

})

new Vue({

el: '#todo-list-example',

data: {

newTodoText: '',

todos: [

{

id: 1,

title: 'Помыть посуду'

},

{

id: 2,

title: 'Вынести мусор'

},

{

id: 3,

title: 'Подстричь газон'

}

],

nextTodoId: 4

},

methods: {

addNewTodo: function () {

this.todos.push({

id: this.nextTodoId++,

title: this.newTodoText

})

this.newTodoText = ''

}

}})

**Обработка событий**

[**Подписка на события**](https://ru.vuejs.org/v2/guide/events.html#%D0%9F%D0%BE%D0%B4%D0%BF%D0%B8%D1%81%D0%BA%D0%B0-%D0%BD%D0%B0-%D1%81%D0%BE%D0%B1%D1%8B%D1%82%D0%B8%D1%8F)

<div id="example-1">

<button v-on:click="counter += 1">+1</button>

<p>Кнопка выше была нажата {{ counter }} раз</p>

</div>

var example1 = new Vue({

el: '#example-1',

data: {

counter: 0

}})

**[Обработчики событий](https://ru.vuejs.org/v2/guide/events.html" \l "%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%87%D0%B8%D0%BA%D0%B8-%D1%81%D0%BE%D0%B1%D1%8B%D1%82%D0%B8%D0%B9" \o "Обработчики событий)**

<div id="example-2">

<!-- `greet` — это название метода, определённого ниже -->

<button v-on:click="greet">Поприветствовать</button>

</div>

var example2 = new Vue({

el: '#example-2',

data: {

name: 'Vue.js'

},

// определяйте методы в объекте `methods`

methods: {

greet: function (event) {

// `this` внутри методов указывает на экземпляр Vue

alert('Привет, ' + this.name + '!')

// `event` — нативное событие DOM

if (event) {

alert(event.target.tagName)

}}

}})

// вызывать методы можно и императивно

example2.greet() // => 'Привет, Vue.js!'

**[Методы и inline-обработчики](https://ru.vuejs.org/v2/guide/events.html" \l "%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D1%8B-%D0%B8-inline-%D0%BE%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%87%D0%B8%D0%BA%D0%B8" \o "Методы и inline-обработчики)**

<div id="example-3">

<button v-on:click="say('hi')">Скажи hi</button>

<button v-on:click="say('what')">Скажи what</button>

</div>

new Vue({

el: '#example-3',

methods: {

say: function (message) {

alert(message)

}}

})

**Его можно передать в метод, используя специальную переменную $event:**

<button v-on:click="warn('Форма не может быть отправлена.', $event)">

Отправить

</button>

methods: {

warn: function (message, event) {

// теперь у нас есть доступ к нативному событию

if (event) {

event.preventDefault()

}

alert(message)

}}

**[Модификаторы событий](https://ru.vuejs.org/v2/guide/events.html" \l "%D0%9C%D0%BE%D0%B4%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80%D1%8B-%D1%81%D0%BE%D0%B1%D1%8B%D1%82%D0%B8%D0%B9" \o "Модификаторы событий)**

.stop, .prevent, .capture, .self, .once, .passive (fetch) ???? базадан ақпарат шақыру үшін

<!-- событие click не будет всплывать дальше -->

<a v-on:click.stop="doThis"></a>

<!-- событие submit больше не будет перезагружать страницу -->

<form v-on:submit.prevent="onSubmit"></form>

<!-- модификаторы можно объединять в цепочки -->

<a v-on:click.stop.prevent="doThat"></a>

<!-- и использовать без указания метода-обработчика -->

<form v-on:submit.prevent></form>

<!-- можно отслеживать события в режиме capture, т.е. событие, нацеленное -->

<!-- на внутренний элемент, обрабатывается здесь до обработки этим элементом -->

<div v-on:click.capture="doThis">...</div>

<!-- вызов обработчика только в случае наступления события непосредственно -->

<!-- на данном элементе (то есть не на дочернем компоненте) -->

<div v-on:click.self="doThat">...</div>

v-on:click.prevent.self   v-on:click.self.prevent ???? қайта-қайта обновление болмауы үшін

<!-- Событие click сработает только 1 раз -->

<a v-on:click.once="doThis"></a>

.passive и .prevent ???? көп қолданылмайды

<div v-on:scroll.passive="onScroll">...</div>

[Модификаторы клавиш](https://ru.vuejs.org/v2/guide/events.html#%D0%9C%D0%BE%D0%B4%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80%D1%8B-%D0%BA%D0%BB%D0%B0%D0%B2%D0%B8%D1%88)

<!-- вызвать `vm.submit()` только если `key` будет `Enter` -->

<input v-on:keyup.enter="submit">

<input v-on:keyup.page-down="onPageDown">

**#[Коды клавиш](https://ru.vuejs.org/v2/guide/events.html" \l "%D0%9A%D0%BE%D0%B4%D1%8B-%D0%BA%D0%BB%D0%B0%D0%B2%D0%B8%D1%88" \o "Коды клавиш)**

.enter, .tab, .delete (ловит как «Delete», так и «Backspace»)

.esc, .space, .up, .down, .left, .right ????

config.keyCodes:

[Системные модификаторы клавиш](https://ru.vuejs.org/v2/guide/events.html" \l "%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D1%8B%D0%B5-%D0%BC%D0%BE%D0%B4%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80%D1%8B-%D0%BA%D0%BB%D0%B0%D0%B2%D0%B8%D1%88" \o "Системные модификаторы клавиш)

.ctrl, .alt, .shift, .meta

<!-- Alt + C -->

<input v-on:keyup.alt.67="clear">

<!-- Ctrl + Click -->

<div v-on:click.ctrl="doSomething">Сделать что-нибудь</div>

keyup.ctrl

#[Модификатор  (.exact](https://ru.vuejs.org/v2/guide/events.html#%D0%9C%D0%BE%D0%B4%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80-exact))

<!-- сработает, даже если Alt или Shift также нажаты -->

<button v-on:click.ctrl="onClick">A</button>

<!-- сработает, только когда нажат Ctrl и не нажаты никакие другие клавиши -->

<button v-on:click.ctrl.exact="onCtrlClick">A</button>

<!-- сработает, только когда не нажаты никакие системные модификаторы -->

<button v-on:click.exact="onClick">A</button>

#[Модификаторы клавиш мыши](https://ru.vuejs.org/v2/guide/events.html" \l "%D0%9C%D0%BE%D0%B4%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80%D1%8B-%D0%BA%D0%BB%D0%B0%D0%B2%D0%B8%D1%88-%D0%BC%D1%8B%D1%88%D0%B8" \o "Модификаторы клавиш мыши)

.left, .right, .middle

mouseover

**Работа с формами**

[Типичное использование](https://ru.vuejs.org/v2/guide/forms.html#%D0%A2%D0%B8%D0%BF%D0%B8%D1%87%D0%BD%D0%BE%D0%B5-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)

v-model, value, checked или selected, value, (change-чекбок тандау)

#[Текст](https://ru.vuejs.org/v2/guide/forms.html" \l "%D0%A2%D0%B5%D0%BA%D1%81%D1%82" \o "Текст)

<input v-model="message" placeholder="отредактируй меня">

<p>Введённое сообщение: {{ message }}</p>

#[Многострочный текст](https://ru.vuejs.org/v2/guide/forms.html#%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D1%81%D1%82%D1%80%D0%BE%D1%87%D0%BD%D1%8B%D0%B9-%D1%82%D0%B5%D0%BA%D1%81%D1%82)

<span>Введённое многострочное сообщение:</span>

<p style="white-space: pre-line;">{{ message }}</p>

<br>

<textarea v-model="message" placeholder="введите несколько строчек"></textarea>

**#**[Чекбоксы](https://ru.vuejs.org/v2/guide/forms.html" \l "%D0%A7%D0%B5%D0%BA%D0%B1%D0%BE%D0%BA%D1%81%D1%8B" \o "Чекбоксы)

<input type="checkbox" id="checkbox" v-model="checked">

<label for="checkbox">{{ checked }}</label>

**Список чекбоксов, привязанных к одному массиву:**

<input type="checkbox" id="jack" value="Джек" v-model="checkedNames">

<label for="jack">Джек</label>

<input type="checkbox" id="john" value="Джон" v-model="checkedNames">

<label for="john">Джон</label>

<input type="checkbox" id="mike" value="Майк" v-model="checkedNames">

<label for="mike">Майк</label>

<br>

<span>Отмеченные имена: {{ checkedNames }}</span>

new Vue({

el: '...',

data: {

checkedNames: []

}

})

**#**[Радиокнопки](https://ru.vuejs.org/v2/guide/forms.html" \l "%D0%A0%D0%B0%D0%B4%D0%B8%D0%BE%D0%BA%D0%BD%D0%BE%D0%BF%D0%BA%D0%B8" \o "Радиокнопки)

<input type="radio" id="one" value="Один" v-model="picked">

<label for="one">Один</label>

<br>

<input type="radio" id="two" value="Два" v-model="picked">

<label for="two">Два</label>

<br>

<span>Выбрано: {{ picked }}</span>

#[Выпадающие списки](https://ru.vuejs.org/v2/guide/forms.html" \l "%D0%92%D1%8B%D0%BF%D0%B0%D0%B4%D0%B0%D1%8E%D1%89%D0%B8%D0%B5-%D1%81%D0%BF%D0%B8%D1%81%D0%BA%D0%B8" \o "Выпадающие списки)

<select v-model="selected">

<option disabled value="">Выберите один из вариантов</option>

<option>А</option>

<option>Б</option>

<option>В</option>

</select>

<span>Выбрано: {{ selected }}</span>

new Vue({

el: '...',

data: {

selected: ''

}

})

disabled

**Выбор нескольких вариантов из списка (с привязкой к массиву):**

<select v-model="selected" multiple>

<option>А</option>

<option>Б</option>

<option>В</option>

</select>

<br>

<span>Выбрано: {{ selected }}</span>

**Динамическое отображение списка опций с помощью v-for:**

<select v-model="selected">

<option v-for="option in options" v-bind:value="option.value">

{{ option.text }}

</option>

</select>

<span>Выбрано: {{ selected }}</span>

new Vue({

el: '...',

data: {

selected: 'А',

options: [

{ text: 'Один', value: 'А' },

{ text: 'Два', value: 'Б' },

{ text: 'Три', value: 'В' }

]

}

})

[Связывание значений](https://ru.vuejs.org/v2/guide/forms.html#%D0%A1%D0%B2%D1%8F%D0%B7%D1%8B%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%B7%D0%BD%D0%B0%D1%87%D0%B5%D0%BD%D0%B8%D0%B9)

<!-- `picked` получает строковое значение "a" при клике -->

<input type="radio" v-model="picked" value="a">

<!-- `toggle` может иметь значение true или false -->

<input type="checkbox" v-model="toggle">

<!-- `selected` при выборе первого пункта становится равным строке "abc" -->

<select v-model="selected">

<option value="abc">ABC</option>

</select>

#[Чекбокс](https://ru.vuejs.org/v2/guide/forms.html" \l "%D0%A7%D0%B5%D0%BA%D0%B1%D0%BE%D0%BA%D1%81" \o "Чекбокс)

<input

type="checkbox"

v-model="toggle"

true-value="да"

false-value="нет"

>

// если чекбокс выбран:

vm.toggle === 'да'

// если чекбокс сброшен:

vm.toggle === 'нет'

#[Радиокнопки](https://ru.vuejs.org/v2/guide/forms.html" \l "%D0%A0%D0%B0%D0%B4%D0%B8%D0%BE%D0%BA%D0%BD%D0%BE%D0%BF%D0%BA%D0%B8-1" \o "Радиокнопки)

<input type="radio" v-model="pick" v-bind:value="a">

#[Списки выбора](https://ru.vuejs.org/v2/guide/forms.html" \l "%D0%A1%D0%BF%D0%B8%D1%81%D0%BA%D0%B8-%D0%B2%D1%8B%D0%B1%D0%BE%D1%80%D0%B0" \o "Списки выбора)

<select v-model="selected">

<!-- инлайновый объект с данными -->

<option v-bind:value="{ number: 123 }">123</option>

</select>

// когда выбрано:

typeof vm.selected // => 'object'

vm.selected.number // => 123

[Модификаторы](https://ru.vuejs.org/v2/guide/forms.html" \l "%D0%9C%D0%BE%D0%B4%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80%D1%8B" \o "Модификаторы)

**#** [.lazy](https://ru.vuejs.org/v2/guide/forms.html" \l "lazy" \o ".lazy)

<!-- синхронизируется по событию "change", а не "input" -->

<input v-model.lazy="msg">

**#**[.number](https://ru.vuejs.org/v2/guide/forms.html" \l "number" \o ".number)

<input v-model.number="age" type="number">

**#**[.trim](https://ru.vuejs.org/v2/guide/forms.html" \l "trim" \o ".trim)

<input v-model.trim="msg">

**Основы компонентов**

[Базовый пример](https://ru.vuejs.org/v2/guide/components.html#%D0%91%D0%B0%D0%B7%D0%BE%D0%B2%D1%8B%D0%B9-%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D1%80)

<div id="components-demo">

<button-counter></button-counter>

</div>

new Vue({ el: '#components-demo' }) ????

бұл жердегі el: Vue-нің барлың әрекеттерін белгіленген id-бар блокпен ғана байланыстыру үшін, Бірақ бұл CLI -да қолданылмайды

[Переиспользование компонентов](https://ru.vuejs.org/v2/guide/components.html#%D0%9F%D0%B5%D1%80%D0%B5%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D0%BE%D0%B2)

**Компоненты можно переиспользовать столько раз, сколько захотите:**

<div id="components-demo">

<button-counter></button-counter>

<button-counter></button-counter>

<button-counter></button-counter>

</div>

# [Свойство data должно быть функцией](https://ru.vuejs.org/v2/guide/components.html#%D0%A1%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%BE-data-%D0%B4%D0%BE%D0%BB%D0%B6%D0%BD%D0%BE-%D0%B1%D1%8B%D1%82%D1%8C-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B5%D0%B9)

data: function () {

return {

count: 0

}

}

[Организация компонентов](https://ru.vuejs.org/v2/guide/components.html#%D0%9E%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F-%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D0%BE%D0%B2)

 Есть два типа регистрации компонентов: **глобальная** и **локальная**.

[Передача данных в дочерние компоненты через входные параметры](https://ru.vuejs.org/v2/guide/components.html#%D0%9F%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B0-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-%D0%B2-%D0%B4%D0%BE%D1%87%D0%B5%D1%80%D0%BD%D0%B8%D0%B5-%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D1%8B-%D1%87%D0%B5%D1%80%D0%B5%D0%B7-%D0%B2%D1%85%D0%BE%D0)

<blog-post

v-for="post in posts"

v-bind:key="post.id"

v-bind:title="post.title"

></blog-post>

new Vue({

el: '#blog-post-demo',

data: {

posts: [

{ id: 1, title: 'My journey with Vue' },

{ id: 2, title: 'Blogging with Vue' },

{ id: 3, title: 'Why Vue is so fun' }

]

}

})

-----------------------------

Vue.component('blog-post', {

props: ['title'],

template: '<h3>{{ title }}</h3>'

})

props:{ ????

type:object,

required:true

}

[Один корневой элемент](https://ru.vuejs.org/v2/guide/components.html" \l "%D0%9E%D0%B4%D0%B8%D0%BD-%D0%BA%D0%BE%D1%80%D0%BD%D0%B5%D0%B2%D0%BE%D0%B9-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82" \o "Один корневой элемент)

<blog-post

v-for="post in posts"

v-bind:key="post.id"

v-bind:title="post.title"

v-bind:content="post.content"

v-bind:publishedAt="post.publishedAt"

v-bind:comments="post.comments"

>

</blog-post>

-----------------------------------

<blog-post

v-for="post in posts"

v-bind:key="post.id"

v-bind:post="post"

></blog-post>

Vue.component('blog-post', {

props: ['post'],

template: `

<div class="blog-post">

<h3>{{ post.title }}</h3>

<div v-html="post.content"></div>

</div>

`

[Прослушивание событий из дочерних компонентов в родительских компонентах](https://ru.vuejs.org/v2/guide/components.html#%D0%9F%D1%80%D0%BE%D1%81%D0%BB%D1%83%D1%88%D0%B8%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D1%81%D0%BE%D0%B1%D1%8B%D1%82%D0%B8%D0%B9-%D0%B8%D0%B7-%D0%B4%D0%BE%D1%87%D0%B5%D1%80%D0%BD%D0%B8%D1%85-%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D0%BE%D0%B2-%D0%)

событие с помощью встроенного [**метода $emit**](https://ru.vuejs.org/v2/api/#vm-emit)

<blog-post

...

v-on:enlarge-text="postFontSize += 0.1"

></blog-post>

<button v-on:click="$emit('enlarge-text')">

Увеличить размер текста

</button>

#[Передача данных вместе с событием](https://ru.vuejs.org/v2/guide/components.html#%D0%9F%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B0-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-%D0%B2%D0%BC%D0%B5%D1%81%D1%82%D0%B5-%D1%81-%D1%81%D0%BE%D0%B1%D1%8B%D1%82%D0%B8%D0%B5%D0%BC)

<button v-on:click="$emit('enlarge-text', 0.1)">

Увеличить размер текста

</button>

<blog-post

...

v-on:enlarge-text="postFontSize += $event"

></blog-post>

-------------------------------

**Или, если обработчик события будет методом:**

<blog-post

...

v-on:enlarge-text="onEnlargeText"

></blog-post>

methods: {

onEnlargeText: function (enlargeAmount) {

this.postFontSize += enlargeAmount

}

}

#[Использование v-model на компонентах](https://ru.vuejs.org/v2/guide/components.html" \l "%D0%98%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-v-model-%D0%BD%D0%B0-%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D0%B0%D1%85" \o "Использование v-model на компонентах)

<input v-model="searchText">

<input

v-bind:value="searchText"

v-on:input="searchText = $event.target.value"

>

--------------------------------------

Vue.component('custom-input', {

props: ['value'],

template: `

<input

v-bind:value="value"

v-on:input="$emit('input', $event.target.value)"

>

`

})

[Распределение контента слотами](https://ru.vuejs.org/v2/guide/components.html" \l "%D0%A0%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5-%D0%BA%D0%BE%D0%BD%D1%82%D0%B5%D0%BD%D1%82%D0%B0-%D1%81%D0%BB%D0%BE%D1%82%D0%B0%D0%BC%D0%B8" \o "Распределение контента слотами) ????

Родителскый файлда кәдімгідей <alert-box>< /alert-box> компонентті шақырылады,

Енді осы екі тег арасына жазы жасса

***Родителскый файлдағы жазу:***

<alert-box>

Произошло что-то плохое.

</alert-box>

әдетте ештеңе көрінбейді

бірақ енді дочерный элементте <slot></slot> деп жазылса көрінеді,

Родителскый файлдағы жазу, дочерный элементте <slot></slot> деп қай жерге жазылса сол жерден көрінеді

***дочерный файл:***

<div class="demo-alert-box">

<strong>Ошибка!</strong> // <slot></slot> болмаса осы жазуды шығарады

<slot></slot>

</div>

по [**слотам**](https://ru.vuejs.org/v2/guide/components-slots.html)

https://www.youtube.com/watch?v=rve1Tbq-BxE

**[Динамическое переключение компонентов](https://ru.vuejs.org/v2/guide/components.html" \l "%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5-%D0%BF%D0%B5%D1%80%D0%B5%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%B8%D0%B5-%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D0%BE%D0%B2" \o "Динамическое переключение компонентов) ????**

Companents-тің барлығын шақыра бермей is-арқылы условия беріп шақыратың қылуға болады, оған

Мысалы ***is-component*** файлында

<component v-bind:is="currentTabComponent"></component>

В примере выше currentTabComponent может содержать:

[**.prop**](https://ru.vuejs.org/v2/api/#v-bind).

[**Особенности парсинга DOM-шаблона**](https://ru.vuejs.org/v2/guide/components.html#%D0%9E%D1%81%D0%BE%D0%B1%D0%B5%D0%BD%D0%BD%D0%BE%D1%81%D1%82%D0%B8-%D0%BF%D0%B0%D1%80%D1%81%D0%B8%D0%BD%D0%B3%D0%B0-DOM-%D1%88%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D0%B0) ????

Кейбір тег тердің *(<ul>, <ol>, <table> и <select> )* ішіне companents жазғанда ошипка шығуы мүмкін, оны түзету үшін

, алдымен тегтердің ішіне *<li>, <tr>* и *<option>* сәйкесінше жазып components ті **is=" components "** арқылы жазу керек

Мысалы:

ERROR

<table>

<blog-post-row></blog-post-row>

</table>

Дұрыс

<table>

<tr is="blog-post-row"></tr>

</table>

**- Vue.js -50 минут**

router - баска беттер ашу, бір беттен екінші бетке көшу

loader - загруска болганда айналып туратын

**VUE - ТЕОРИЯ**

**beforeCreate()** : data() әлі доступный емес, DOM дерево әлі жоқ

**created()** : data() уже доступ бар , DOM дерево әлі жоқ

(created: back-end - ке запрос жіберетін болса осы дұрыс )

***html***: DOM дерево

**beforeMount()** : DOM дерево бар, бірақ одан данный алуға, оны өзгертуге доступ жоқ

**mounted()** : DOM дерево бар, одан данный алуға, оны өзгертуге доступ бар

(mounted: *innerHTML, quertySelection(#id***) -** сияқтыға доступ**)**

**beforeUpdate, updated** - когда данный изменены , обновление DOM дерево, қосымшаның өзгеруі

кнопканны басқанда немесе қандайда бір жағдайда DOM дерево (html) өзгеруі (өшіп, сосыын қайта өзгеріп барып қосылу емес, өшпей өзгеруі)

*beforeUpdate()* : сол данный өзгергенге дейін

*updated()* : сол данный өзгергеннен кейін

**beforeDestroy()** : удалит дочерный компонент, наблюдател, случатели собития ????

( компонент-тегі DOM дерево өшсе қосылады )

**destroyed():** DOM дерево өшеді, басқа строницаға ауысқанда, немесе данныйлар, кодтар өзгеріп обновление болғанда(өшіп, сосыын қайта өзгеріп барып қосылу) , немесе кнопка басылып обновление болғанда, яғни қашан DOM дерево өшіп қайта қосылғанда орындалады ????

**methods()** - көп қайталанатын функциларда (қайта-қайта қосу не өзгерту кнопкаларында, функцияларында қолданылады) DOM дерево бөлімінде функциядан кейін () *қою жақсы* қоймасада орындалады

**computed()** - бір рет қарна орындалатын функцияларда ( КЕШ - те сақтап қалатын қайта-қайта өзгерту қажет боламамайтын функцияларда ) қолданылады. DOM дерево бөлімінде функциядан кейін () *қоюмау* *жақсы* қойса орындалмайды ????

**watch(),** - бір переменныйдің өзгеруін анықтай алады,сол переменный өзгеру кезіндегі функция,

(бұл жерде бір переменныйді функция ретінде алады)

  watch: {

    success() {

      console.log("seve");

      this.success2 = this.success2 + 10;

    },

  },

filters(), - белгілі бір нақты жерлерде өзгерістер жасау үшін

    <h1>{{ success | capitalize }}</h1>

filters: {

    capitalize(value) {

      // if (!value) return "";

      value = value.toUpperCase();

      return value.charAt(0).toUpperCase() + value.slice(1);

    },

  },

бұл жерде value деген | белгісінен алдынғы данный

бір емес бірнеше filter қатар қолдануға болады ( | filterА | filterB )

**is-component** : компоненттерді қайта-қайта жаза бермеей переменный арқылы функциямен керектісін ғана шығарып отыру

**vue.companent**

локалный элементтер жинағы

new Vue({

  el: "#app",

  data: {

    massege: "Hello Worlad",

  },

  mathods: {

    massegeAdd() {

      this.massege = "KUANYSH";

    },

  },

});

глобальный элементтый

Vue.filter("capatilize", function (value) {

  return value.toUpperCase();

});

глобалный компонент

Vue.component("app-car", {

  data: function () {

    return {

      massege: "Hello Worlad",

    };

  },

  template: "<div><h1>{{massege}}</h1></div>",

});

new Vue({

  el: "#app",

});