**Регистрация компонентов**

[Именование компонентов](https://ru.vuejs.org/v2/guide/components-registration.html#%D0%98%D0%BC%D0%B5%D0%BD%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)

Vue.component('my-component-name', { /\* ... \*/ })

Vue.component('MyComponentName', { /\* ... \*/ })

[Глобальная регистрация](https://ru.vuejs.org/v2/guide/components-registration.html#%D0%93%D0%BB%D0%BE%D0%B1%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F-%D1%80%D0%B5%D0%B3%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%86%D0%B8%D1%8F)

Vue.component('component-a', { /\* ... \*/ })

Vue.component('component-b', { /\* ... \*/ })

<div id="app">

<component-a></component-a>

<component-b></component-b>

</div>

[**Локальная регистрация**](https://ru.vuejs.org/v2/guide/components-registration.html#%D0%9B%D0%BE%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F-%D1%80%D0%B5%D0%B3%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%86%D0%B8%D1%8F)

var ComponentB = { /\* ... \*/ }

var ComponentC = { /\* ... \*/ }

new Vue({

el: '#app',

components: {

'component-a': ComponentA,

'component-b': ComponentB

}

})

Что локально компоненты **не будут доступны** в дочерних компонентах.

[**Модульные системы**](https://ru.vuejs.org/v2/guide/components-registration.html#%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5-%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B)

Глобальный компоненттер **Base, App или V. сөздерімен басталуы дұрыс, шатаспау үшін**

Пример:

BaseButton

BaseIcon

BaseInput

[Локальная регистрация в модульной системе](https://ru.vuejs.org/v2/guide/components-registration.html#%D0%9B%D0%BE%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F-%D1%80%D0%B5%D0%B3%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%86%D0%B8%D1%8F-%D0%B2-%D0%BC%D0%BE%D0%B4%D1%83%D0%BB%D1%8C%D0%BD%D0%BE%D0%B9-%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B5)

Әрбір компонент жеке файлда тұрса локальный кезде жеке жеке импорт жасайды

import ComponentA from './ComponentA'

import ComponentC from './ComponentC'

export default {

components: {

ComponentA,

ComponentC

},

// ...

}

[Автоматическая глобальная регистрация базовых компонентов](https://ru.vuejs.org/v2/guide/components-registration.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%B0%D1%8F-%D0%B3%D0%BB%D0%BE%D0%B1%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F-%D1%80%D0%B5%D0%B3%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%86%D0%B8%D1%8F-%D0%B1%D0%B0%D0%B7%D0%BE%D0%B2%D1%8B%D1%85)

import BaseButton from './components/BaseButton.vue'

import BaseIcon from './components/BaseIcon.vue'

import BaseInput from './components/BaseInput.vue'

export default {

components: {

BaseButton,

BaseIcon,

BaseInput

}

}

**Осыларды бірден импорт жасау үшін gloval.js пайдаланылады**

import Vue from "vue"; // './components/gloval.js'

//Base деп басталатын компоненттердің қасында тұратын js файл осы компоненттерді бірден глобальный импорт жасау үшін керек

const requireComponent = require.context(".", false, /\_Base-[\w-]+\.vue$/);

requireComponent.keys().forEach((fileName) => {

  const componentConfig = requireComponent(fileName);

  const componentName = fileName

    .replace(/^\.\/\_/, "")

    .replace(/\.\w+$/, "")

    .split("-")

    .map((kebab) => kebab.charAt(0).toUpperCase() + kebab.slice(1))

    .join("");

  Vue.component(componentName, componentConfig.default || componentConfig);

});

# Входные параметры

## [Prop Casing (camelCase vs kebab-case)](https://vuejs.org/v2/guide/components-props.html#Prop-Casing-camelCase-vs-kebab-case)

Vue.component('blog-post', {

// - postTitle осылай Name беру camelCase деп аталады

props: ['postTitle'],

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

# })

// - post-title осылай Name беру kebab-case деп аталады

# <blog-post post-title="hello!"></blog-post>

## [Prop Types](https://vuejs.org/v2/guide/components-props.html#Prop-Types)

# props: ['title', 'likes', 'isPublished', 'commentIds', 'author']

# props -ты массив емес обьект түрінде шақырған дұрысрақ

props: {

title: String,

likes: Number,

isPublished: Boolean,

commentIds: Array,

author: Object,

callback: Function,

contactsPromise: Promise // или любой другой конструктор

# }

## [Passing Static or Dynamic Props](https://vuejs.org/v2/guide/components-props.html#Passing-Static-or-Dynamic-Props)

# <blog-post v-bind:title="post.title"></blog-post>

### #[Passing a Number](https://vuejs.org/v2/guide/components-props.html#Passing-a-Number)

# <blog-post v-bind:likes="42"></blog-post>

### #[Passing a Boolean](https://vuejs.org/v2/guide/components-props.html#Passing-a-Boolean)

# <blog-post v-bind:is-published="false"></blog-post>

### #[Passing an Array](https://vuejs.org/v2/guide/components-props.html#Passing-an-Array)

# <blog-post v-bind:comment-ids="[234, 266, 273]"></blog-post>

### #[Passing an Object](https://vuejs.org/v2/guide/components-props.html#Passing-an-Object)

<blog-post

v-bind:author="{

name: 'Veronica',

company: 'Veridian Dynamics'

}"

# ></blog-post>

# // <blog-post v-bind:author="post.author"></blog-post>

### #[Passing the Properties of an Object](https://vuejs.org/v2/guide/components-props.html#Passing-the-Properties-of-an-Object)

post: {

id: 1,

title: 'My Journey with Vue'

# }

<blog-post v-bind="post"></blog-post>

<blog-post

v-bind:id="post.id"

v-bind:title="post.title"

# > </blog-post>

## [One-Way Data Flow](https://vuejs.org/v2/guide/components-props.html#One-Way-Data-Flow)

props: ['initialCounter'],

data: function () {

return {

counter: this.initialCounter

}

# }

props: ['size'],

computed: {

normalizedSize: function () {

return this.size.trim().toLowerCase()

}

# }

## [Prop Validation](https://vuejs.org/v2/guide/components-props.html#Prop-Validation)

  <my-component

        prop\_one="ass"

        v-bind:prop\_two="12"

        prop\_three="12"

        prop\_sex="danger"

      >

      </my-component>

# 

Vue.component("my-component", {

  props: {

    // Просто проверка типа (`null` и `undefined` проходят проверку для любого типа)

    prop\_one: String,

    // Несколько допустимых типов

    prop\_two: [Number, Boolean],

    // // Обязательное значение строкового (String) типа

    prop\_three: {

      type: String,

      required: false,

    },

    // // Число со значением по умолчанию

    prop\_foo: {

      type: Number,

      default: 100,

    },

    // // Объект

     // data: {

    //   type: Object,

    //   default: () => {},

    //   required: true,

    // },

    // // Объект со значением по умолчанию

    prop\_five: {

      type: Object,

      // Для объектов или массивов значения по умолчанию

      // должны возвращаться из функции

      default: function () {

        return { message: "hello" };

      },

    },

    // // Пользовательская функция для валидации

    prop\_sex: {

      required: false,

      validator: function (value) {

        // Значение должно соответствовать одной из этих строк

      return ["success", "warning", "danger"].indexOf(value) !== -1;

      },

  // prop\_seven: {

    //   validator: Person,

    // },

    },

  },

  template:

    "<h1>{{ prop\_one }} | {{ prop\_two }} | {{ prop\_three }} | {{prop\_foo}} | {{prop\_five.message}} |{{prop\_sex}}</h1>",

});

// function Person(value) {

//   return ["success", "warning", "danger"].indexOf(value) !== -1;

// }

## ??? [Non-Prop Attributes](https://vuejs.org/v2/guide/components-props.html#Non-Prop-Attributes)

который требует указания атрибута data-date-picker на элементе input. Мы можем добавить этот атрибут к нашему экземпляру компонента:

<bootstrap-date-input data-date-picker="activated"></bootstrap-date-input>

### ??? [Replacing/Merging with Existing Attributes](https://vuejs.org/v2/guide/components-props.html#Replacing-Merging-with-Existing-Attributes)

### ??? [Disabling Attribute Inheritance](https://vuejs.org/v2/guide/components-props.html#Disabling-Attribute-Inheritance)

**Custom Events**

## [Event Names](https://vuejs.org/v2/guide/components-custom-events.html#Event-Names)

 <component-a

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

//enlarge-text -деп жазылса emit- қалай жазылуы керек

    />

    {{ postFontSize }}

**Былай жазылса дұрыс болмайды**

<button v-on:click="$emit('enlargeText', 10)">

      Click

    </button>

**Осылай жазса дұрыс болады, яғни (enlarge-text )**

<button v-on:click="$emit('enlarge-text', 10)">

      Click

    </button>

## [Customizing Component v-model](https://vuejs.org/v2/guide/components-custom-events.html#Customizing-Component-v-model)

Компонент ішінде тұрып v-model -ге настройка жасау үшін, және оның значениесін родительскыйға жіберу

Vue.component('base-checkbox', {

model: {

prop: 'checked', ???

event: 'change'

},

props: {

checked: Boolean й ???

},

template: `

<input

type="checkbox"

v-bind:checked="checked"

v-on:change="$emit('change', $event.target.checked)"

>

`

})

<base-checkbox v-model="lovingVue"></base-checkbox>

## [Binding Native Events to Components](https://vuejs.org/v2/guide/components-custom-events.html#Binding-Native-Events-to-Components)

компонент ішіндегі собитияларының функциясын родительскыйде шақыру

яғни сыртта тұрып іштегі жағдаяттарды анықтап сонымен сырттан функция беру

оны істеу үшін **native** қолданылады

    <base-button @click.native="onFocus()" />

 methods: {

    onFocus() {

      console.log("alert");

    },

  },

<template>

  <div>

    <button>Click Btn</button>

  </div>

</template>

**native**- кей жағдайларда проблема байқатады оны реттеу үшін , қате шығармау үшін, **native** **-** емес басқаны  **$listeners**-ті қолданады

 <base-input v-on:focus="onFocus"></base-input>

  methods: {

    onFocus() {

      console.log("alert");

    },

  },

<template>

  <div>

    <label>

      {{ label }}

      <input v-bind="$attrs" v-bind:value="value" v-on="inputListeners" />

    </label>

  </div>

</template>

<script>

export default {

  inheritAttrs: false,

  props: ["label", "value"],

  computed: {

    inputListeners: function() {

      var vm = this;

      return Object.assign({}, this.$listeners, {

        input: function(event) {

          vm.$emit("input", event.target.value);

        },

      });

    },

  },

};

</script>

## [.sync Modifier](https://vuejs.org/v2/guide/components-custom-events.html#sync-Modifier)

**родительскыйдегі** переменныйдың мәнін **компоненттің** ішіндегі кнопка арқылы өзгерткіміз келсе осы тәсілді қолданамыз

 //родительскый - де

<base-sync :val="number" v-on:sync-text="number=$event"/>

data() {

    return {

      number: 12,

    };

  },

# 

  //components -ішінде

<button @click="$emit('sync-text', val + 12)">Click</button>

props: ["val "],

# егер жоғарыдағы тәсілді қысқа әрі ыңғайлы тәсілде жасау үшін .sync  тәсілімен істеу керек ол төмендегідей

 //родительскый - де

<base-sync v-bind:val.sync="number" />

// v-bind:title.sync="doc.title + '!' -осылай жазса қате болады, өйткені артық нәрселер көбейсе , операция қиындап қате шығарады

data() {

    return {

      number: 12,

    };

  },

//components -ішінде

<button @click="$emit('update:val', val + 12)">Click</button>

props: ["val "],

# .sync  тәсілімен істеген кезде обьект жіберетін болса осылай қысқа қайтаруға болады және общый обьектный данный жібергің келседе осылай істеуге болады v-bind.sync="obj"

  <base-sync v-bind.sync="obj" />

 //v-bind.sync="{ title: doc.title }"' -осылай жазса қате болады, өйткені артық нәрселер көбейсе , операция қиындап қате шығарады

    {{ obj.val }}

data() {

    return {

      obj: { val: 23 },

    };

  },

# 

//components -ішінде

<button @click="$emit('update:val', val + 15)">Click</button>

props: ["val "],

# 