免费电子书

学习

Vue.js

Free unaffiliated eBook created from
Stack Overflow contributors.

#vue.js

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: vue-js

1: Vue.js

Vue.jsJavaScriptAngular.js Reactive.jsRivets.js ◦

MVVM Model-View View-Model ◦ JS◦

2.4.1	2017713
2.3.4	201768
2.3.3	201759
2.2.6	2017326
2.0.0	2016102
1.0.26	2016628
1.0.0	20151026
0.12.0	2015612
0.11.0	2014116

Examples

“”

Vue.js HTML◦ HTML◦

```
<script src="https://npmcdn.com/vue/dist/vue.js"></script>
```

HTML

```
<div id="app">
  {{ message }}
</div>
```

JavaScript

```
new Vue({
  el: '#app',
  data: {
```

```
    message: 'Hello Vue.js!'
  }
})
```

。

Vue.js“Hello World” 。

Vue 2Hello WorldJSX

JSX。 Javascript。 JSXbabel `babel-plugin-transform-vue-JSX`

```
npm install babel-plugin-syntax-jsx babel-plugin-transform-vue-jsx babel-helper-vue-jsx-merge-
props --save-dev
```

`.babelrc`

```
{
  "presets": ["es2015"],
  "plugins": ["transform-vue-jsx"]
}
```

VUE JSX

```
import Vue from 'vue'
import App from './App.vue'

new Vue({
  el: '#app',
  methods: {
    handleClick () {
      alert('Hello!')
    }
  },
  render (h) {
    return (
      <div>
        <h1 on-click={this.handleClick}>Hello from JSX</h1>
        <p> Hello World </p>
      </div>
    )
  }
})
```

JSXJavaScriptHTML / XML。

:)

VueJSv-model。

HTML

```
<script src="https://unpkg.com/vue/dist/vue.js"></script>
<div id="app">
    {{message}}
<input v-model="message">
</div>
```

JS

```
new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue.js!'
  }
})
```

`v-model`VueJS。

。

Vue.js https://riptutorial.com/zh-CN/vue-js/topic/1057/vue-js

2: Polyfill“webpack”

	`npm i -save babel-polyfill`
karma.conf.js	`files: ['../../node_modules/babel-polyfill/dist/polyfill.js','./index.js'],`
webpack.base.conf.js	`app: ['babel-polyfill', './src/main.js']`

“Internet Explorer” `npm test`。

Examples

polyfillfind

```
<template>
  <div class="hello">
    <p>{{ filtered() }}</p>
  </div>
</template>

<script>
export default {
  name: 'hello',
  data () {
    return {
      list: ['toto', 'titi', 'tata', 'tete']
    }
  },
  methods: {
    filtered () {
      return this.list.find((el) => el === 'tata')
    }
  }
}
</script>
```

Polyfill“webpack” https://riptutorial.com/zh-CN/vue-js/topic/9174/polyfill-webpack-

3: VueJS + ReduxVua-Redux

Examples

Vua-Redux

NPMVua Redux

```
npm i vua-redux --save
```

==========

// main.js

```
import Vue from 'vue';
import { reduxStorePlugin } from 'vua-redux';
import AppStore from './AppStore';
import App from './Component/App';

// install vua-redux
Vue.use(reduxStorePlugin);

new Vue({
    store: AppStore,
    render(h) {
        return <App />
    }
});
```

// AppStore.js

```
import { createStore } from 'redux';

const initialState = {
  todos: []
};

const reducer = (state = initialState, action) => {
  switch(action.type){
    case 'ADD_TODO':
      return {
        ...state,
        todos: [...state.todos, action.data.todo]
      }

    default:
      return state;
    }
}

const AppStore = createStore(reducer);

export default AppStore;
```

// components / App.js

```
import { connect } from 'vua-redux';

const App = {
    props: ['some-prop', 'another-prop'],

    /**
     * everything you do with vue component props
     * you can do inside collect key
     */
    collect: {
        todos: {
            type: Array,
        },
        addTodo: {
            type: Function,
        },
    },

    methods: {
        handleAddTodo() {
            const todo = this.$refs.input.value;
            this.addTodo(todo);
        }
    },

    render(h) {
        return <div>
            <ul>
                {this.todos.map(todo => <li>{todo}</li>)}
            </ul>

            <div>
                <input type="text" ref="input" />
                <button on-click={this.handleAddTodo}>add todo</button>
            </div>
        </div>
    }
};

function mapStateAsProps(state) {
    return {
        todos: state.todos
    };
}

function mapActionsAsProps(dispatch) {
    return {
        addTodo(todo) {
            dispatch({
                type: 'ADD_TODO',
                data: { todo }
            })
        }
    }
}

export default connect(mapStateAsProps, mapActionsAsProps)(App);
```

VueJS + ReduxVua-Redux https://riptutorial.com/zh-CN/vue-js/topic/7396/vuejs-plus-reduxvua-redux--

4: Vuex

VuexVue.js+。 。 Vue/。

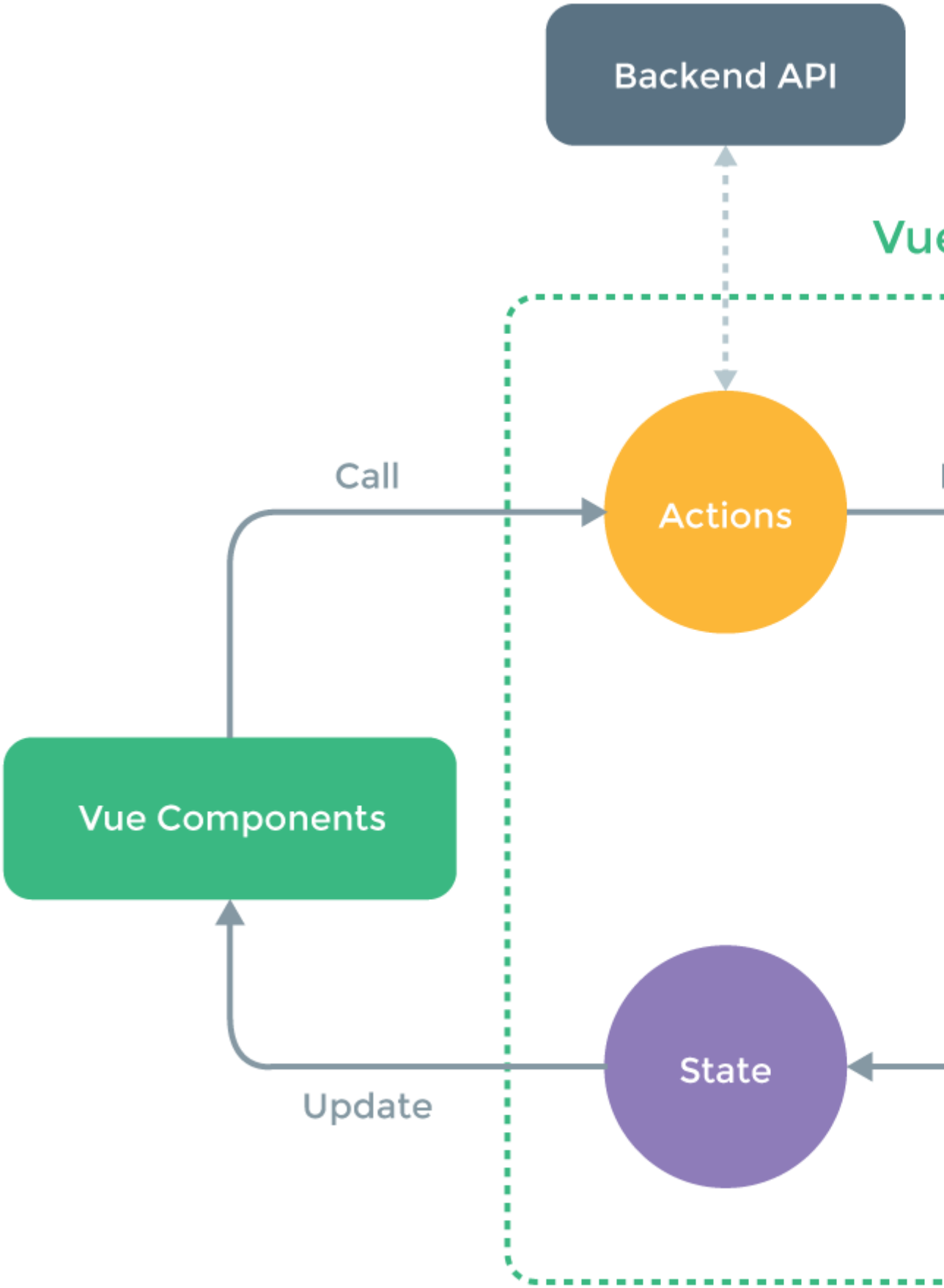
Examples

Vuex

VuexVue.js。 Flux。

Vuex 。 。

Vuex。


Backend API
Call
Actions
Vue Components
State
Update

Vuex GitHub 。

Vue.js**getter getter**。

。 。

getter。

---

vuevuex。

```
const state = {
    lastClickTime: null
}

const mutations = {
    updateLastClickTime: (state, payload) => {
      state.lastClickTime = payload
  }
}

const getters = {
  getLastClickTime: state => {
    return new Date(state.lastClickTime)
  }
}

const actions = {
    syncUpdateTime: ({ commit }, payload) => {
    commit("updateLastClickTime", payload)
  },
  asyncUpdateTime: ({ commit }, payload) => {
    setTimeout(() => {
      commit("updateLastClickTime", payload)
    }, Math.random() * 5000)
  }
}

const store = new Vuex.Store({
  state,
  getters,
  mutations,
  actions
})

const { mapActions, mapGetters } = Vuex;

// Vue
const vm = new Vue({
    el: '#container',
  store,
  computed: {
      ...mapGetters([
        'getLastClickTime'
    ])
  },
  methods: {
      ...mapActions([
        'syncUpdateTime',
```

```
      'asyncUpdateTime'
    ]),
    updateTimeSyncTest () {
        this.syncUpdateTime(Date.now())
    },
    updateTimeAsyncTest () {
        this.asyncUpdateTime(Date.now())
    }
  }
})
```

HTML

```
<div id="container">
  <p>{{ getLastClickTime || "No time selected yet" }}</p>
  <button @click="updateTimeSyncTest">Sync Action test</button>
  <button @click="updateTimeAsyncTest">Async Action test</button>
</div>
```

1. null**lastClickTime**。 。 getter 。
2. getterstate。
3. 。
4. ActionAPIsetTimeout。

Vuex

SPA。 。

。 。

VueJS。 vuex`emiton`。

1

```
this.$emit('eventWithDataObject', dataObject)
```

2

```
this.$on('eventWithDataObject', function (dataObject) {
    console.log(dataObject)
})
```

vuex。

```
this.$store.state.myData
```

mutatorgetter。

Getter。

```
this.$store.getters.myGetter
```

。

```
this.$store.dispatch('myAction', myDataObject)
```

vuex。

```
this.$store.commit('myMutation', myDataObject)
```

Vuex

```
state: {
  myData: {
    key: 'val'
  }
},
getters: {
  myGetter: state => {
     return state.myData.key.length
  }
},
actions: {
  myAction ({ commit }, myDataObject) {
    setTimeout(() => {
      commit('myMutation', myDataObject)
    }, 2000)
  }
},
mutations: {
  myMutation (state, myDataObject) {
    state.myData = myDataObject
  }
}
```

Vuex

VuexWebpackBrowserifyVueify。

VuexNPM。 Vuex。

```
 npm install --save vuex
```

`require('vue')`VuexVue。

```
Vue.use(require('vuex'))
```

VuexCDN;cdnjs。

vuex

notifications.js

vuex

```
//Vuex store previously configured on other side
import _store from 'path/to/store';

//Notification default duration in milliseconds
const defaultDuration = 8000;

//Valid mutation names
const NOTIFICATION_ADDED = 'NOTIFICATION_ADDED';
const NOTIFICATION_DISMISSED = 'NOTIFICATION_DISMISSED';
```

```
const state = {
  Notifications: []
}
```

getters

```
const getters = {
  //All notifications, we are returning only the raw notification objects
  Notifications: state => state.Notifications.map(n => n.Raw)
}
```

```
const actions = {
  //On actions we receive a context object which exposes the
  //same set of methods/properties on the store instance
  //{commit} is a shorthand for context.commit, this is an
  //ES2015 feature called argument destructuring
  Add({ commit }, notification) {
    //Get notification duration or use default duration
    let duration = notification.duration || defaultDuration

    //Create a timeout to dismiss notification
    var timeOut = setTimeout(function () {
      //On timeout mutate state to dismiss notification
      commit(NOTIFICATION_DISMISSED, notification);
    }, duration);

    //Mutate state to add new notification, we create a new object
    //for save original raw notification object and timeout reference
    commit(NOTIFICATION_ADDED, {
      Raw: notification,
      TimeOut: timeOut
    })
  },
  //Here we are using context object directly
  Dismiss(context, notification) {
    //Just pass payload
    context.commit(NOTIFICATION_DISMISSED, notification);
  }
}
```

```
const mutations = {
  //On mutations we receive current state and a payload
  [NOTIFICATION_ADDED](state, notification) {
    state.Notifications.push(notification);
  },
  //remember, current state and payload
  [NOTIFICATION_DISMISSED](state, rawNotification) {
    var i = state.Notifications.map(n => n.Raw).indexOf(rawNotification);
    if (i == -1) {
      return;
    }

    clearTimeout(state.Notifications[i].TimeOut);
    state.Notifications.splice(i, 1);
  }
}
```

getteractionsmutation

```
_store.registerModule('notifications', {
  state,
  getters,
  actions,
  mutations
});
```

componentA.vue

。

```
<template>
<transition-group name="notification-list" tag="div" class="top-right">
  <div v-for="alert in alerts" v-bind:key="alert" class="notification alert alert-dismissible"
v-bind:class="'alert-'+alert.type">
    <button v-on:click="dismiss(alert)" type="button" class="close" aria-label="Close"><span
aria-hidden="true">&times;</span></button>
    <div>
      <div>
        <strong>{{alert.title}}</strong>
      </div>
      <div>
        {{alert.text}}
      </div>
    </div>
  </div>
</transition-group>
</template>

<script>
export default {
  name: 'arc-notifications',
  computed: {
    alerts() {
      //Get all notifications from store
      return this.$store.getters.Notifications;
    }
  },
  methods: {
```

```
    //Manually dismiss a notification
    dismiss(alert) {
      this.$store.dispatch('Dismiss', alert);
    }
  }
}
</script>
<style lang="scss" scoped>
$margin: 15px;

.top-right {
    top: $margin;
    right: $margin;
    left: auto;
    width: 300px;
    //height: 600px;
    position: absolute;
    opacity: 0.95;
    z-index: 100;
    display: flex;
    flex-wrap: wrap;
    //background-color: red;
}
.notification {
    transition: all 0.8s;
    display: flex;
    width: 100%;
    position: relative;
    margin-bottom: 10px;
    .close {
        position: absolute;
        right: 10px;
        top: 5px;
    }

    > div {
        position: relative;
        display: inline;
    }
}
.notification:last-child {
    margin-bottom: 0;
}
.notification-list-enter,
.notification-list-leave-active {
    opacity: 0;
    transform: translateX(-90px);
}
.notification-list-leave-active {
    position: absolute;
}
</style>
```

```
//payload could be anything, this example content matches with componentA.vue
this.$store.dispatch('Add', {
  title = 'Hello',
  text = 'World',
  type = 'info',
  duration = 15000
});
```

Vuex https://riptutorial.com/zh-CN/vue-js/topic/3430/vuex

5: Vue

.vue。

。

Examples

.vue

```
<template>
  <div class="nice">Component {{title}}</div>
</template>

<script>
export default {
    data() {
        return {
            title: "awesome!"
        };
    }
}
</script>

<style>
.nice {
    background-color: red;
    font-size: 48px;
}
</style>
```

Vue https://riptutorial.com/zh-CN/vue-js/topic/10118/vue

6: VUE

vue-routervue.js。

- `<router-link to="/path">Link Text</router-link> <!-- Creates a link to the route that matches the path -->`
- `<router-view></router-view> <!-- Outlet for the currently matched route. It's component will be rendered here. -->`

Examples

vue-routerCDN。

HTML

```
<script src="https://unpkg.com/vue/dist/vue.js"></script>
<script src="https://unpkg.com/vue-router/dist/vue-router.js"></script>

<div id="router-example">
    <router-link to="/foo">Link to Foo route</router-link>
    <router-view></router-view>
</div>
```

JavaScriptES2015

```
const Foo = { template: <div>This is the component for the Foo route</div> }

const router = new VueRouter({
    routes: [
       { path: '/foo', component: Foo}
    ]
})

const routerExample = new Vue({
    router
}).$mount('#router-example')
```

VUE https://riptutorial.com/zh-CN/vue-js/topic/9654/vue

7: v-model

/◦ vuex/

◦

v-model ◦

1. "
2. input◦

```
<component v-model='something'></component>
```

```
<component
    :value="something"
    @input="something = $event.target.value"
>
</component>
```

Examples

v

counterdemov-model◦

```
// child component
Vue.component('counter', {
  template: `<div><button @click='add'>+1</button>
  <button @click='sub'>-1</button>
  <div>this is inside the child component: {{ result }}</div></div>`,
  data () {
    return {
      result: 0
    }
  },
  props: ['value'],
  methods: {
    emitResult () {
      this.$emit('input', this.result)
    },
    add () {
      this.result += 1
      this.emitResult()
    },
    sub () {
      this.result -= 1
      this.emitResult()
    }
  }
})
```

sub()add()result ◦

```
// parent component
new Vue({
  el: '#demo',
  data () {
    return {
      resultFromChild: null
    }
  }
})

// parent template
<div id='demo'>
  <counter v-model='resultFromChild'></counter>
  This is in parent component {{ resultFromChild }}
</div>
```

v-modelvalueprop counter◦

v-model https://riptutorial.com/zh-CN/vue-js/topic/9353/v-model

8:

`event.preventDefault()event.stopPropagation()` ◦ DOM◦

Examples

Vue`v-on`◦

- `.stop`
- `.prevent`
- `.capture`
- `.self`
- `.once`

```
<!-- the click event's propagation will be stopped -->
<a v-on:click.stop="doThis"></a>

<!-- the submit event will no longer reload the page -->
<form v-on:submit.prevent="onSubmit"></form>

<!-- use capture mode when adding the event listener -->
<div v-on:click.capture="doThis">...</div>

<!-- only trigger handler if event.target is the element itself -->
<!-- i.e. not from a child element -->
<div v-on:click.self="doThat">...</div>
```

◦ keyCodesVue

- `.enter`
- `.tab`
- `.delete` “”“”
- `.esc`
- `.space`
- `.up`
- `.down`
- `.left`
- `.right`

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

- `.trim`

`trim`v-model

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

- `.number`

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

- `.lazy`

`v-modellazy`

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

https://riptutorial.com/zh-CN/vue-js/topic/8612/

9:

Examples

v-for◦ ◦ items item◦

HTML

```
<div id="app">
  <h1>My List</h1>
  <table>
    <tr v-for="item in items">
      <td>{{item}}</td>
    </tr>
  </table>
</div>
```

```
new Vue({
  el: '#app',
  data: {
    items: ['item 1', 'item 2', 'item 3']
  }
})
```

◦

HTML


```
<ul id="render-sample">
  <li v-for="n in 5">
    Hello Loop
  </li>
</ul>
```

```
<ul>
  <li v-for="n in 10">{{11 - n}} pigs are tanning at the beach. One got fried, and
</ul>
```

https://jsfiddle.net/gurghet/3jeyka22/

v-for

HTML

```
<div v-for="(value, key) in object">
  {{ key }} : {{ value }}
</div>
```

```
new Vue({
  el: '#repeat-object',
  data: {
    object: {
      FirstName: 'John',
      LastName: 'Doe',
      Age: 30
    }
  }
})
```

https://riptutorial.com/zh-CN/vue-js/topic/1972/

10:

<component>◦

v-bind◦ *v*-Vue◦

Examples

<component>*v-bindis*

Javascript

```
new Vue({
  el: '#app',
  data: {
    currentPage: 'home'
  },
  components: {
    home: {
      template: "<p>Home</p>"
    },
    about: {
      template: "<p>About</p>"
    },
    contact: {
      template: "<p>Contact</p>"
    }
  }
})
```

HTML

```
<div id="app">
   <component v-bind:is="currentPage">
       <!-- component changes when currentPage changes! -->
       <!-- output: Home -->
   </component>
</div>
```

<keep-alive>

Javascript

```
new Vue({
  el: '#app',
  data: {
    currentPage: 'home',
  },
  methods: {
```

```
    switchTo: function(page) {
           this.currentPage = page;
    }
  },
  components: {
    home: {
      template: `<div>
                 <h2>Home</h2>
                 <p>{{ homeData }}</p>
                 </div>`,
      data: function() {
         return {
            homeData: 'My about data'
         }
       }
     },
    about: {
      template: `<div>
                 <h2>About</h2>
                 <p>{{ aboutData }}</p>
                 </div>`,
      data: function() {
         return {
            aboutData: 'My about data'
         }
      }
    },
    contact: {
      template: `<div>
             <h2>Contact</h2>
             <form method="POST" @submit.prevent>
             <label>Your Name:</label>
             <input type="text" v-model="contactData.name" >
             <label>You message: </label>
             <textarea v-model="contactData.message"></textarea>
             <button type="submit">Send</button>
             </form>
             </div>`,
       data: function() {
          return {
            contactData: { name:'', message:'' }
         }
      }
    }
  }
})
```

HTML

```
<div id="app">
  <div class="navigation">
    <ul>
      <li><a href="#home" @click="switchTo('home')">Home</a></li>
      <li><a href="#about" @click="switchTo('about')">About</a></li>
      <li><a href="#contact" @click="switchTo('contact')">Contact</a></li>
    </ul>
  </div>

  <div class="pages">
```

```
    <keep-alive>
      <component :is="currentPage"></component>
    </keep-alive>
  </div>
</div>
```

CSS

```
.navigation {
  margin: 10px 0;
}

.navigation ul {
  margin: 0;
  padding: 0;
}

.navigation ul li {
  display: inline-block;
  margin-right: 20px;
}

input, label, button {
  display: block
}

input, textarea {
  margin-bottom: 10px;
}
```

https://riptutorial.com/zh-CN/vue-js/topic/7702/

11: Vue“this”

StackOverflowVue`this`。 promises`this`。

Examples

Vue“this”。

```
new Vue({
  el:"#app",
  data:{
    foo: "bar"
  },
  methods:{
    doSomethingAsynchronous(){
      setTimeout(function(){
        // This is wrong! Inside this function,
        // "this" refers to the window object.
        this.foo = "baz";
      }, 1000);
    }
  }
})
```

“this”。

```
new Vue({
  el:"#star-wars-people",
  data:{
    people: null
  },
  mounted: function(){
    $.getJSON("http://swapi.co/api/people/", function(data){
      // Again, this is wrong! "this", here, refers to the window.
      this.people = data.results;
    })
  }
})
```

“”

`this`。

```
new Vue({
  el:"#star-wars-people",
  data:{
    people: null
  },
  mounted: function(){
    // Before executing the web service call, save this to a local variable
    var self = this;
    $.getJSON("http://swapi.co/api/people/", function(data){
```

```
      // Inside this call back, because of the closure, self will
      // be accessible and refers to the Vue object.
      self.people = data.results;
    })
  }
})
```

bind。

。

```
new Vue({
  el:"#star-wars-people",
  data:{
    people: null
  },
  mounted:function(){
    $.getJSON("http://swapi.co/api/people/", function(data){
      this.people = data.results;
    }.bind(this));
  }
})
```

。

```
new Vue({
  el:"#star-wars-people",
  data:{
    people: null
  },
  mounted: function(){
    $.getJSON("http://swapi.co/api/people/", data => this.people = data.results);
  }
})
```

2015ECMAScriptJavaScriptES5。

“this”

```
new Vue({
  el:"#app",
  data:{
    foo: "bar"
  },
  methods:{
    // This is wrong! Arrow functions capture "this" lexically
    // and "this" will refer to the window.
    doSomething: () => this.foo = "baz"
  }
})
```

```
new Vue({
  el:"#app",
```

```
  data:{
    foo: "bar"
  },
  methods:{
    doSomething: function(){
      this.foo = "baz"
    }
  }
})
```

javascriptEcmascript 2015

```
new Vue({
  el:"#app",
  data:{
    foo: "bar"
  },
  methods:{
    doSomething(){
      this.foo = "baz"
    }
  }
})
```

Vue“this” https://riptutorial.com/zh-CN/vue-js/topic/9350/vue-this-

12:

- `Vue.filter(name, function(value){});` //
- `Vue.filter(name, function(value, begin, end){});` //
- `Vue.filter(name, function(value, input){});` //
- `Vue.filter(name, { read: function(value){}, write: function(value){} });` //

String -	
[Callback] Any -	
[Callback] Any -	
[Callback] Any -	
[Callback] Any - Vue	

Examples

two-way filter read writefilterviewmodel ◦

```
//JS
Vue.filter('uppercase', {
    //read : model -> view
    read: function(value) {
        return value.toUpperCase();
    },

    //write : view -> model
    write: function(value) {
        return value.toLowerCase();
    }
});

/*
 * Base value of data: 'example string'
 *
 * In the view : 'EXAMPLE STRING'
 * In the model : 'example string'
 */
```

Vue.jsVue.filter◦

```
//JS
Vue.filter('reverse', function(value) {
    return value.split('').reverse().join('');
});

//HTML
<span>{{ msg | reverse }}</span> //'This is fun!' => '!nuf si sihT'
```

./filters◦ **JS**

```
//JS
Vue.filter('reverse', require('./filters/reverse'));
```

beginend◦

```
//JS
Vue.filter('wrap', function(value, begin, end) {
    return begin + value + end;
});

//HTML
<span>{{ msg | wrap 'The' 'fox' }}</span> //'quick brown' => 'The quick brown fox'
```

https://riptutorial.com/zh-CN/vue-js/topic/1878/

13:

Vuemixins

- MyPlugin.install = functionVueoptions{}

Vue	VueVue

Vue

```
// calls `MyPlugin.install(Vue)`
Vue.use(MyPlugin)
```

```
Vue.use(MyPlugin, { someOption: true })
```

Examples

```
//myLogger.js
export default {

  install(Vue, options) {
     function log(type, title, text) {
       console.log(`[${type}] ${title} - ${text}`);
     }

     Vue.prototype.$log = {
       error(title, text) { log('danger', title, text) },
       success(title, text) { log('success', title, text) },
       log
     }
  }
}
```

Vue

```
//main.js
import Logger from './path/to/myLogger';

Vue.use(Logger);

var vm = new Vue({
  el: '#app',
  template: '<App/>',
  components: { App }
})
```

```
this.$log
```

```
//myComponent.vue
export default {
  data() {
    return {};
  },
  methods: {
    Save() {
      this.$log.success('Transaction saved!');
    }
  }
}
```

https://riptutorial.com/zh-CN/vue-js/topic/8726/

14:

Examples

“Mustache”

```
<span>Message: {{ msg }}</span>
```

mustache`msg` `msg`

```
<span>This will never change: {{* msg }}</span>
```

HTML

HTML HTML

```
<div>{{{ raw_html }}}</div>
```

HTML - partials

MustachesHTML

```
<div id="item-{{ id }}"></div>
```

Vue.js Vue.js

Vue.js“”“”

```
{{ message | capitalize }}
```

`capitalize`“” `message`JavaScript Vue.js

JavaScript;

```
{{ message | filterA | filterB }}
```

```
{{ message | filterA 'arg1' arg2 }}
```

filter `'arg1'arg2`

https://riptutorial.com/zh-CN/vue-js/topic/1213/

15:

- `<element v-if="condition"></element>` // v-if
- `<element v-if="condition"></element><element v-else="condition"></element>` // v-if | V-
- `<template v-if="condition">...</template>` //v-if
- `<element v-show="condition"></element>` // v-show

`v-ifv-show。 v-iftrue`*DOM* `。 v-show` *DOM*`displayfalse`

Examples

Vue.js。

v-if

`true。 false true`DOM。

v-else

`v-iffalse。 v-if。`

v-show

`v-iffalse` DOM。 `false displaynone 。`

v-if / v-else

`Vue.js`

```
var vm = new Vue({
    el: '#example',
    data: {
        a: true,
        b: false
    }
});
```

v-ifhtml;v-iftrue

```
<!-- will render 'The condition is true' into the DOM -->
<div id="example">
    <h1 v-if="a">The condition is true</h1>
</div>
```

`<h1>`“a”。 v-if

```
<div v-if="0 === 1">                  false; won't render</div>
<div v-if="typeof(5) === 'number'">   true; will render</div>
```

`template`

```
<!-- in this case, nothing will be rendered except for the containing 'div' -->
<div id="example">
    <template v-if="b">
        <h1>Heading</h1>
        <p>Paragraph 1</p>
        <p>Paragraph 2</p>
    </template>
</div>
```

`v-if` `v-else`。 v-iffalse。 v-elsev-if。

```
<!-- will render only 'ELSE' -->
<div id="example">
    <h1 v-if="b">IF</h1>
    <h1 v-else="a">ELSE</h1>
</div>
```

v-ifv-else`<template>`html

```
<div v-if="'a' === 'b'"> This will never be rendered. </div>
<template v-else>
    <ul>
      <li> You can also use templates with v-else. </li>
      <li> All of the content within the template </li>
      <li> will be rendered. </li>
    </ul>
</template>
```

v-

`v-show`v-if。 `v-show` `<template>`“”。

```
var vm = new Vue({
    el: '#example',
    data: {
        a: true
    }
});
```

.....

```
<!-- will render 'Condition met' -->
<div id="example">
    <h1 v-show="a">Condition met</h1>
</div>
```

`v-show`v-else“”...

```
<!-- will render 'This is shown' -->
<div id="example">
    <h1 v-show="!a">This is hidden</h1>
```

```
    <h1 v-show="a">This is shown</h1>
</div>
```

https://riptutorial.com/zh-CN/vue-js/topic/3465/

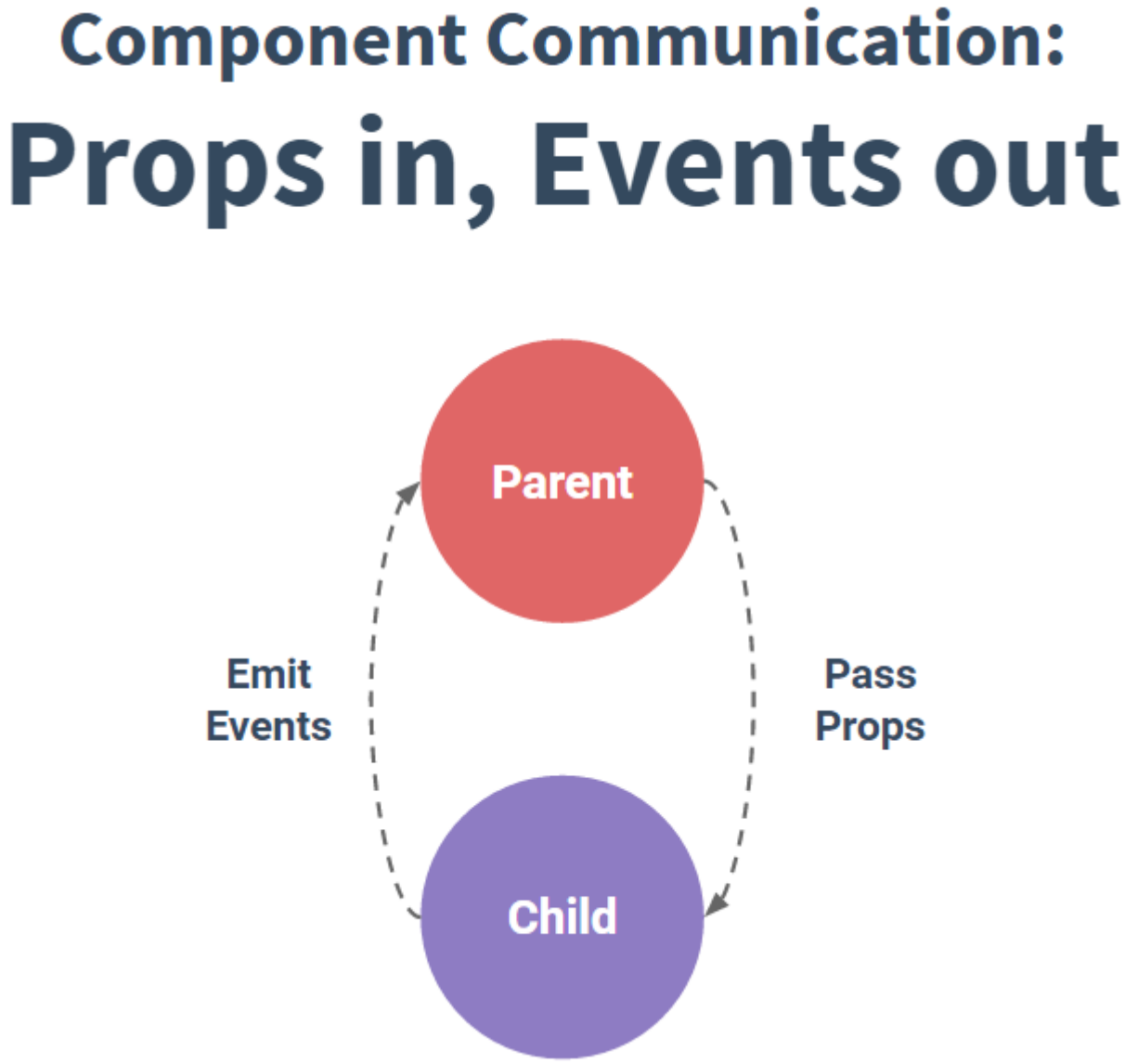
16:

Examples

```
vm.$emit('new-message');

vm.$on('new-message');

vm.$broadcast('new-message');

vm.$dispatch('new-message');
```

Vue2`$broadcast``$dispatch`。 Vue2

。 Evan You VueJS Progressive Framework。



DEMO

HTML

```
<script type="x-template" id="message-box">
    <input type="text" v-model="msg" @keyup="$emit('new-message', msg)" />
</script>

<message-box :msg="message" @new-message="updateMessage"></message-box>
<div>You typed: {{message}}</div>
```

JS

```
var messageBox = {
  template: '#message-box',
  props: ['msg']
};

new Vue({
  el: 'body',
  data: {
    message: ''
  },
  methods: {
    updateMessage: function(msg) {
      this.message = msg;
    }
  },
  components: {
    'message-box': messageBox
  }
});
```

◦ v-model ◦

DEMO Vue1

DEMO Vue2

Vue1<message-box>prop.sync ◦ VueJS◦

◦

HTML Vue1

```
<script type="x-template" id="message-box">
  <input v-model="value" />
</script>

<div id="app">
  <message-box :value.sync="message"></message-box>
  <div>You typed: {{message}}</div>
</div>
```

Vue2'input'$emit ◦ v-model<message-box>◦

HTML Vue2

```
<script type="x-template" id="message-box">
  <input :value="value" @input="$emit('input', $event.target.value)" />
</script>

<div id="app">
  <message-box v-model="message"></message-box>
  <div>You typed: {{message}}</div>
</div>
```

JS Vue 12

```
var messageBox = {
  template: '#message-box',
  props: ['value']
};

new Vue({
  el: '#app',
  data: {
    message: ''
  },
  components: {
    'message-box': messageBox
  }
});
```

。

$ dispatch$ broadcast

$emit。 。

Vue1$dispatch$broadcast Vue2。 。 bus

DEMO

HTML

```
<script type="x-template" id="sender">
  <button @click="bus.$emit('new-event')">Click me to send an event !</button>
</script>

<script type="x-template" id="receiver">
  <div>I received {{numberOfEvents}} event{{numberOfEvents == 1 ? '' : 's'}}</div>
</script>

<sender></sender>
<receiver></receiver>
```

JS

```
var bus = new Vue();

var senderComponent = {
  template: '#sender',
```

```
  data() {
    return {
      bus: bus
    }
  }
};

var receiverComponent = {
  template: '#receiver',
  data() {
    return {
      numberOfEvents: 0
    }
  },
  ready() {
    var self = this;

    bus.$on('new-event', function() {
      ++self.numberOfEvents;
    });
  }
};

new Vue({
  el: 'body',
  components: {
    'sender': senderComponent,
    'receiver': receiverComponent
  }
});
```

`Vue()$emit`catch `$on ◦ Vuebus ◦ bus◦`

https://riptutorial.com/zh-CN/vue-js/topic/5941/

17:

。

vue$emit$on 。

1. Vue

vuex。

Examples

eventBus

```
// setup an event bus, do it in a separate js file
var bus = new Vue()

// imagine a component where you require to pass on a data property
// or a computed property or a method!

Vue.component('card', {
  template: `<div class='card'>
    Name:
      <div class='margin-5'>
          <input v-model='name'>
    </div>
    <div class='margin-5'>
      <button @click='submit'>Save</button>
    </div>
  </div>`,
  data() {
    return {
      name: null
    }
  },
  methods: {
    submit() {
      bus.$emit('name-set', this.name)
    }
  }
})

// In another component that requires the emitted data.
var data = {
  message: 'Hello Vue.js!'
}

var demo = new Vue({
  el: '#demo',
  data: data,
  created() {
    console.log(bus)
    bus.$on('name-set', (name) => {
      this.message = name
```

```
    })
  }
})
```

https://riptutorial.com/zh-CN/vue-js/topic/9498/

18:

Examples

Global Mixin

mixin。 mixin Vue。

```
// inject a handler for `myOption` custom option
Vue.mixin({
  created: function () {
    var myOption = this.$options.myOption
    if (myOption) {
      console.log(myOption)
    }
  }
})

new Vue({
  myOption: 'hello!'
})
// -> "hello!"
```

mixinVue。 。

◦ Vue.config.optionMergeStrategies

```
Vue.config.optionMergeStrategies.myOption = function (toVal, fromVal) {
  // return mergedVal
}
```

methods

```
var strategies = Vue.config.optionMergeStrategies
strategies.myOption = strategies.methods
```

MixinsVue。 mixin。 mixinmixin“”。

```
// define a mixin object
var myMixin = {
  created: function () {
    this.hello()
  },
  methods: {
    hello: function () {
      console.log('hello from mixin!')
    }
  }
}

// define a component that uses this mixin
var Component = Vue.extend({
```

```
  mixins: [myMixin]
})

var component = new Component() // -> "hello from mixin!"
```

mixin“”。 。 mixin

```
var mixin = {
  created: function () {
    console.log('mixin hook called')
  }
}

new Vue({
  mixins: [mixin],
  created: function () {
    console.log('component hook called')
  }
})

// -> "mixin hook called"
// -> "component hook called"
```

methods componentsdirectives。

```
var mixin = {
  methods: {
    foo: function () {
      console.log('foo')
    },
    conflicting: function () {
      console.log('from mixin')
    }
  }
}

var vm = new Vue({
  mixins: [mixin],
  methods: {
    bar: function () {
      console.log('bar')
    },
    conflicting: function () {
      console.log('from self')
    }
  }
})

vm.foo() // -> "foo"
vm.bar() // -> "bar"
vm.conflicting() // -> "from self"
```

Vue.extend()。

https://riptutorial.com/zh-CN/vue-js/topic/2562/

19:

Examples

Vue 1.x

- **init**

  ◦

- **created**

  ◦ $eldata observation computed properties watch/event callbacksmethods ◦

- **beforeCompile**

  Vue◦

- **compiled**

  ◦ directives$el ◦

- **ready**

  $elDOM◦

- **attached**

  $eldirective$appendTo()DOM◦

- **detached**

  DOM/$el◦

- **beforeDestroy**

  Vue◦

- **destroyed**

  ◦ bindingsdirectives◦

Vue.jsfunctions ◦

```
//JS
new Vue({
```

```
    el: '#example',

    data: {
        ...
    },

    methods: {
        ...
    },

    //LIFECYCLE HOOK HANDLING
    created: function() {
        ...
    },

    ready: function() {
        ...
    }

});
```

`ready`DOM

`ready()`DOMJavascript。

VueDOM`ready()`DOM。 。

`$nextTick()`。 tickDOM。

```
module.exports {
  ready: function () {
    $('.cool-input').initiateCoolPlugin() //fails, because element is not in DOM yet.

    this.$nextTick(function() {
      $('.cool-input').initiateCoolPlugin() // this will work because it will be executed
after the DOM update.
    })
  }
}
```

https://riptutorial.com/zh-CN/vue-js/topic/1852/

20:

Examples

Vue。

```
export default {
    data () {
        return {
            watched: 'Hello World'
        }
    },
    watch: {
        'watched' () {
            console.log('The watched property has changed')
        }
    }
}
```

```
export default {
    data () {
        return {
            watched: 'Hello World'
        }
    },
    watch: {
        'watched' (value, oldValue) {
            console.log(oldValue) // Hello World
            console.log(value) // ByeBye World
        }
    },
    mounted () {
        this.watched = 'ByeBye World'
    }
}
```

deep

```
export default {
    data () {
        return {
            someObject: {
                message: 'Hello World'
            }
        }
    },
    watch: {
        'someObject': {
            deep: true,
            handler (value, oldValue) {
                console.log('Something changed in someObject')
            }
        }
    }
}
```

nextTick()

```
export default {
    data() {
        return {
            foo: 'bar',
            message: 'from data'
        }
    },
    methods: {
        action () {
            this.foo = 'changed'
            // If you juste this.message = 'from method' here, the watcher is executed after.
            this.$nextTick(() => {
                this.message = 'from method'
            })
        }
    },
    watch: {
        foo () {
            this.message = 'from watcher'
        }
    }
}
```

https://riptutorial.com/zh-CN/vue-js/topic/7988/

21:

props。

data。 。

events

methods

DOMDOM

readyVue

Examples

HTML

```
<script type="x-template" id="form-template">
    <label>{{inputLabel}} :</label>
  <input type="text" v-model="name" />
</script>

<div id="app">
  <h2>{{appName}}</h2>
  <form-component title="This is a form" v-bind:name="userName"></form-component>
</div>
```

JS

```
// Describe the form component
// Note: props is an array of attribute your component can take in entry.
// Note: With props you can pass static data('title') or dynamic data('userName').
// Note: When modifying 'name' property, you won't modify the parent variable, it is only
descendent.
// Note: On a component, 'data' has to be a function that returns the data.
var formComponent = {
  template: '#form-template',
  props: ['title', 'name'],
  data: function() {
    return {
      inputLabel: 'Name'
    }
  }
};

// This vue has a private component, it is only available on its scope.
// Note: It would work the same if the vue was a component.
// Note: You can build a tree of components, but you have to start the root with a 'new
Vue()'.
var vue = new Vue({
```

```
  el: '#app',
  data: {
    appName: 'Component Demo',
    userName: 'John Doe'
  },
  components: {
    'form-component': formComponent
  }
});
```

Vue。。

Vue/HTML。

- HTML
- CSSHTML
- JavaScript

.vue。

.VUE -

```
<style>
    .hello-world-compoment{
        color:#eeeeee;
        background-color:#555555;
    }
</style>

<template>
    <div class="hello-world-component">
        <p>{{message}}</p>
        <input @keyup.enter="changeName($event)"/>
    </div>
</template>

<script>
    export default{
        props:[ /* to pass any data from the parent here... */ ],
        events:{ /* event listeners go here */},
        ready(){
            this.name= "John";
        },
        data(){
           return{
              name:''
           }
        },
        computed:{
            message(){
                return "Hello from " + this.name;
            }
        },
        methods:{
            // this could be easily achieved by using v-model on the <input> field, but just
to show a method doing it this way.
            changeName(e){
```

```
                this.name = e.target.value;
            }
        }
    }
</script>
```

hello-world.js - JS

```
export default{
    template:require('./hello-world.template.html'),
    props:[ /* to pass any data from the parent here... */ ],
    events:{ /* event listeners go here */ },
    ready(){
        this.name="John";
    },
    data(){
        return{
            name:''
        }
    },
    computed:{
        message(){
            return "Hello World! from " + this.name;
        }
    },
    methods:{
        changeName(e){
            let name = e.target.value;
            this.name = name;
        }
    }
}
```

world.template.html

```
<div class="hello-world-component">
    <p>{{message}}</p>
    <input class="form-control input-sm" @keyup.enter="changeName($event)">
</div>
```

world.css

```
 .hello-world-compoment{
    color:#eeeeee;
    background-color:#555555;
}
```

es2015Babeles5。
BabelBrowserify + vueifyWebpack + vue-loader`hello-world.vue`。

`hello-world`Vue。

`main.js`Vue.component

```
import Vue from 'vue'; // Note that 'vue' in this case is a Node module installed with 'npm
install Vue'
Vue.component('hello-world', require('./hello-world');  // global registeration

new Vue({
    el:'body',

    // Templates can be defined as inline strings, like so:
    template:'<div class="app-container"><hello-world></hello-world></div>'
});
```

```
import Vue from 'vue'; // Note that 'vue' in this case is a Node module installed with 'npm
install Vue'
import HelloWorld from './hello-world.js';

new Vue({
    el:'body',
    template:'<div class="app-container"><hello-world></hello-world></div>",

    components:{HelloWorld}  // local registeration
});
```

Vue。

。

。 。

。

```
var Child = Vue.extend({
    // ...
})

var Parent = Vue.extend({
    template: '...',
    components: {
         'my-component': Child
    }
})
```

Thiw new componentParent。

```
Vue.component('custom-component', {
    template: '<div>A custom component!</div>'
})
```

```
var Parent = Vue.extend({
    components: {
        'custom-component': {
            template: '<div>A custom component!</div>'
        }
    }
})
```

data◦ data ◦

```
var CustomComponent = Vue.extend({
    data: function () {
        return { a: 1 }
    }
})
```

/◦ Vue◦

- $on ◦
- $broadcast ◦
- $dispatch ◦
- $emit self◦

◦

```
var FormComponent = Vue.extend({
  // ...
  components: {
    ButtonComponent
  },
  methods: {
    onSubmit () {
        this.$broadcast('submit-form')
    }
  }
})
```

```
var FormComponent = Vue.extend({
  // ...
  events: {
    'submit-form': function () {
        console.log('I should be hiding');
    }
  }
})
```

- true ◦
- $dispatch()◦
- ◦ this.$broadcast('submit-form', this.formData, this.formStatus)'submit-form': function (formData, formStatus) {}'submit-form': function (formData, formStatus) {}

https://riptutorial.com/zh-CN/vue-js/topic/1775/

22:

◦ ◦

Examples

slot◦ page◦

page

```
<html>
    <head>
        <title>Page Title</title>
    </head>
    <body>
        <slot>
            This will only be displayed if there is no content
            to be distributed.
        </slot>
    </body>
</html>
```

◦

```
<page>
    <p>This content will be displayed within the page component</p>
</page>
```

```
<html>
    <head>
        <title>Page Title</title>
    </head>
    <body>
        <p>This content will be displayed within the page component</p>
    </body>
</html>
```

page<page></page>pageslot◦

```
<html>
    <head>
        <title>Page Title</title>
    </head>
    <body>
        This will only be displayed if there is no content
        to be distributed.
    </body>
</html>
```

◦ HTML◦

◦

◦

page◦ ◦

```
<page>
    <article></article>
    <comments></comments>
</page>
```

```
<page>
    <blog-post></blog-post>
    <comments></comments>
</page>
```

```
<page>
    <form></form>
</page>
```

page◦ ◦

“,◦ ”

◦

page

```
<html>
    <head>
        <title>Page Title</title>
    </head>
    <body>
        <aside>
            <slot name="sidebar"></slot>
        </aside>
        <main>
            <slot name="content"></slot>
        </main>
    </body>
</html>
```

pageslotslot

```
<page>
    <p slot="sidebar">This is sidebar content.</p>
    <article slot="content"></article>
</page>
```

```
<html>
    <head>
        <title>Page Title</title>
    </head>
```

```
    <body>
        <aside>
            <p>This is sidebar content.</p>
        </aside>
        <main>
            <article></article>
        </main>
    </body>
</html>
```

`slotnameslot`。

Vue.js。

Vue JSX'babel-plugin-transform-vue-jsx'

VueJS2JSX。 `this.$slots.defaultthis.$slots.default`React JS`this.props.children` 。

Component.js

```
export default {
    render(h) {  //eslint-disable-line
        return (
            <li>
                { this.$slots.default }
            </li>
        );
    }
};
```

ParentComponent.js

```
import Component from './Component';

export default {
    render(h) {  //eslint-disable-line
        return (
            <ul>
                <Component>
                    Hello World
                </Component>
            </ul>
        );
    }
};
```

https://riptutorial.com/zh-CN/vue-js/topic/4484/

23:

- `Vue.directive(id, definition);`
- `Vue.directive(id, update); //when you need only the update function.`

`id`	String - `v-`ID。 `v-`
`definition`	- `bind update``unbind`

Examples

Vue.js。 DOM。

`Vue.directive(id, definition)`ID。 `directives`。

- **bind** 。
- **update** `bind`。 。
- **unbind** 。

```
Vue.directive('my-directive', {
     bind: function () {
       // do preparation work
       // e.g. add event listeners or expensive stuff
       // that needs to be run only once
     },
     update: function (newValue, oldValue) {
       // do something based on the updated value
       // this will also be called for the initial value
     },
     unbind: function () {
       // do clean up work
       // e.g. remove event listeners added in bind()
     }
})
```

Vue.js`v-`

```
<div v-my-directive="someValue"></div>
```

`update`

```
Vue.directive('my-directive', function (value) {
  // this function will be used as update()
})
```

`this`。

- **el** 。

- **vm** ViewModel。
- **expression** 。
- **arg** 。
- **name** 。
- 。
- **descriptor** 。
- **params** param。 。

  。 。

HTML

```
<div id="demo" v-demo:hello.a.b="msg"></div>
```

JavaScript

```
Vue.directive('demo', {
  bind: function () {
    console.log('demo bound!')
  },
  update: function (value) {
    this.el.innerHTML =
      'name - '       + this.name + '<br>' +
      'expression - ' + this.expression + '<br>' +
      'argument - '   + this.arg + '<br>' +
      'modifiers - '  + JSON.stringify(this.modifiers) + '<br>' +
      'value - '      + value
  }
})
var demo = new Vue({
  el: '#demo',
  data: {
    msg: 'hello!'
  }
})
```

```
name - demo
expression - msg
argument - hello
modifiers - {"b":true,"a":true}
value - hello!
```

JavaScript。JavaScript

HTML

```
<div v-demo="{ color: 'white', text: 'hello!' }"></div>
```

JavaScript

```
Vue.directive('demo', function (value) {
  console.log(value.color) // "white"
```

```
  console.log(value.text) // "hello!"
})
```

update◦ update◦

HTML

```
<div v-demo.literal="foo bar baz">
```

JavaScript

```
Vue.directive('demo', function (value) {
  console.log(value) // "foo bar baz"
})
```

https://riptutorial.com/zh-CN/vue-js/topic/2368/

24:

Vuedatacomputed◦

-
-
-
-
-

Examples

```
<div id="example">
  a={{ a }}, b={{ b }}
</div>
```

JavaScript

```
var vm = new Vue({
  el: '#example',
  data: {
    a: 1
  },
  computed: {
    // a computed getter
    b: function () {
      // `this` points to the vm instance
      return this.a + 1
    }
  }
})
```

```
a=1, b=2
```

b ◦ vm.bgetter

```
console.log(vm.b) // -> 2
vm.a = 2
console.log(vm.b) // -> 3
```

vm.bvm.a◦

◦ Vuevm.bvm.a vm.avm.b◦

vs

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

```
var vm = new Vue({
```

```
  el: '#demo',
  data: {
    firstName: 'Foo',
    lastName: 'Bar',
    fullName: 'Foo Bar'
  }
})

vm.$watch('firstName', function (val) {
  this.fullName = val + ' ' + this.lastName
})

vm.$watch('lastName', function (val) {
  this.fullName = this.firstName + ' ' + val
})
```

```
var vm = new Vue({
  el: '#demo',
  data: {
    firstName: 'Foo',
    lastName: 'Bar'
  },
  computed: {
    fullName: function () {
      return this.firstName + ' ' + this.lastName
    }
  }
})
```

◦ Vuesetter

```
<div id="example">
  a={{ a }}, b={{ b }}
</div>
```

Javascript

```
var vm = new Vue({
  el: '#example',
  data: {
    a: 1
  },
  computed: {
    b: {
      // getter
      get: function () {
        return this.a + 1
      },
      // setter
      set: function (newValue) {
        this.a = newValue - 1
      }
    }
  }
```

gettersetter

```
console.log(vm.b)       // -> 2
vm.b = 4                // (setter)
console.log(vm.b)       // -> 4
console.log(vm.a)       // -> 3
```

`vm.b = 4`setter`vm.b = 4`3;vm.b4◦

v

`v-model` ◦ v◦

```
<div id="demo">
  <div class='inline-block card'>
    <div :class='{onlineMarker: true, online: status, offline: !status}'></div>
    <p class='user-state'>User is {{ (status) ? 'online' : 'offline' }}</p>
  </div>

  <div class='margin-5'>
    <input type='checkbox' v-model='status'>Toggle status (This will show you as offline to
others)
  </div>
</div>
```

```
#demo {
  font-family: Helvetica;
  font-size: 12px;
}
.inline-block > * {
  display: inline-block;
}
.card {
  background: #ddd;
  padding:2px 10px;
  border-radius: 3px;
}
.onlineMarker {
  width: 10px;
  height: 10px;
  border-radius: 50%;
  transition: all 0.5s ease-out;
}

.online {
  background-color: #3C3;
}

.offline {
  background-color: #aaa;
}

.user-state {
  text-transform: uppercase;
  letter-spacing: 1px;
}

.margin-5 {
```

```
  margin: 5px;
}
```

```
var demo = new Vue({
    el: '#demo',
    data: {
        statusProxy: null
    },
    computed: {
        status: {
            get () {
                return (this.statusProxy === null) ? true : this.statusProxy
            }
        }
    }
})
```

。

```
var demo = new Vue({
    el: '#demo',
    data: {
        statusProxy: null
    },
    computed: {
        status: {
            get () {
                return (this.statusProxy === null) ? true : this.statusProxy
            },
            set (val) {
                this.statusProxy = val
            }
        }
    }
})
```

/。

https://riptutorial.com/zh-CN/vue-js/topic/2371/

25:

camelCase <=> kebab-case

propsHTML。 prop ...

```
Vue.component('child', {
    props: ['myProp'],
    ...
});
```

...HTMLmy-prop。

Examples

props

Vue.js -。

。/。

props。

usersaddresses
users

	1234 5678 9012	john@dirhard.com
	44777 0007 0077	bond@mi6.com

addresses

Nakatomi		
Mi6 House		

component.js

```
export default{
    template:`<div class="user-component">
                <label for="name" class="form-control">Name: </label>
                <input class="form-control input-sm" name="name" v-model="name">
                <contact-details :phone="phone" :email="email"></contact-details>
              </div>`,
    data(){
        return{
```

```
            name:'',
            phone:'',
            email:''
        }
    },
}
```

details.js

```
import Address from './address';
export default{
    template:`<div class="contact-details-component">
                <h4>Contact Details:</h4>
                <label for="phone" class="form-control">Phone: </label>
                <input class="form-control input-sm" name="phone" v-model="phone">
                <label for="email" class="form-control">Email: </label>
                <input class="form-control input-sm" name="email" v-model="email">

                <h4>Address:</h4>
                <address :address-type="addressType"></address>
                //see camelCase vs kebab-case explanation below
            </div>`,
    props:['phone', 'email'],
    data:(){
        return:{
            addressType:'Office'
        }
    },
    components:{Address}
}
```

address.js

```
export default{
    template:`<div class="address-component">
                <h6>{{addressType}}</h6>
                <label for="block" class="form-control">Block: </label>
                <input class="form-control input-sm" name="block" v-model="block">
                <label for="street" class="form-control">Street: </label>
                <input class="form-control input-sm" name="street" v-model="street">
                <label for="city" class="form-control">City: </label>
                <input class="form-control input-sm" name="city" v-model="city">
             </div>`,
    props:{
        addressType:{
            required:true,
            type:String,
            default:'Office'
        },
    data(){
        return{
            block:'',
            street:'',
            city:''
        }
    }
}
```

main.js

```
import Vue from 'vue';

Vue.component('user-component', require('./user-component');
Vue.component('contact-details', require('./contact-details');

new Vue({
    el:'body'
});
```

index.html

```
...
<body>
    <user-component></user-component>
        ...
</body>
```

phoneemailcontact-detailsuser-component。

templateuser-component.js<contact-details><user-component> <contact-details> - :phone="phone" :email="emailv-bind:phone="phone"v-bind:phone="phone"v-bind:email="email"

-

<user-component><contact-details> 。

-

phone="(44) 777 0007 0077" email="bond@mi6.com"phone="(44) 777 0007 0077" email="bond@mi6.com"。

prop。

<contact-details>bond@mi6.comjamesbond@mi6.com <user-component>bond@mi6.combond@mi6.com 。

<user-component>bond@mi6.comjamesbond@mi6.co <contact-details>jamesbond@mi6.com - 。

:email.sync="email":email="email" 。 prop。

- 。

Vue.js 2.0.sync。 **Vue.js 2.0props** 。

:email.once="email 。

Object**Array**prop:email.sync="email":email="email":email.once="email" prop。

contact-details.jsprops:['phone', 'email']。

-
-

-

address.js。

props。

。

-
-
-
-
-
-
-

http://vuejs.org/guide/components.html#Props

```
Vue.component('example', {
   props: {
        // basic type check (`null` means accept any type)
        propA: Number,
        // multiple possible types (1.0.21+)
        propM: [String, Number],
        // a required string
        propB: {
          type: String,
          required: true
        },
        // a number with default value
        propC: {
           type: Number,
           default: 100
        },
        // object/array defaults should be returned from a
        // factory function
        propD: {
           type: Object,
           default: function () {
              return { msg: 'hello' }
          }
        },
        // indicate this prop expects a two-way binding. will
        // raise a warning if binding type does not match.
        propE: {
           twoWay: true
        },
        // custom validator function
        propF: {
           validator: function (value) {
              return value > 10
           }
        },
        // coerce function (new in 1.0.12)
        // cast the value before setting it on the component
        propG: {
           coerce: function (val) {
             return val + '' // cast the value to string
```

```
            }
        },
        propH: {
            coerce: function (val) {
              return JSON.parse(val) // cast the value to Object
            }
        }
    }
});
```

camelCase vs kebab-case

HTMLaddresstypeaddressType
addressTypeHTMLaddress-type。

v-bindprops。

JS

```
new Vue({
    el: '#example',
    data: {
        msg: 'hello world'
    }
});

Vue.component('child', {
    props: ['myMessage'],
    template: '<span>{{ myMessage }}</span>
});
```

HTML

```
<div id="example">
    <input v-model="msg" />
    <child v-bind:my-message="msg"></child>
    <!-- Shorthand ... <child :my-message="msg"></child> -->
</div>
```

```
hello world
```

Vue JSX

。 'src''src'。

ParentComponent.js

```
import ChildComponent from './ChildComponent';
export default {
```

```
    render(h, {props}) {
        const src = 'https://cdn-images-1.medium.com/max/800/1*AxRXW2j8qmGJixIYg7n6uw.jpeg';
        return (
           <ChildComponent src={src} />
        );
    }
};
```

。 。

ChildComponent.js

```
export default {
    props: ['src'],
    render(h, {props}) {
        return (
            <a href = {props.src} download = "myimage" >
                Click this link
            </a>
        );
    }
};
```

https://riptutorial.com/zh-CN/vue-js/topic/3080/

26:

vue。vueVue。$ setArray.prototype.splice

Examples

Vue。$ set

```
new Vue({
    el: '#app',
    data:{
        myArr : ['apple', 'orange', 'banana', 'grapes']
    },
    methods:{
        changeArrayItem: function(){
            //this will not work
            //myArr[2] = 'strawberry';

            //Vue.$set(array, index, newValue)
            this.$set(this.myArr, 2, 'strawberry');
        }
    }
})
```

Array.prototype.splice

Array`splice()Vue.$set`

```
new Vue({
    el: '#app',
    data:{
        myArr : ['apple', 'orange', 'banana', 'grapes']
    },
    methods:{
        changeArrayItem: function(){
            //this will not work
            //myArr[2] = 'strawberry';

            //Array.splice(index, 1, newValue)
            this.myArr.splice(2, 1, 'strawberry');
        }
    }
})
```

yoi

```
new Vue({
    el: '#app',
    data:{
        myArr : [
            ['apple', 'banana'],
            ['grapes', 'orange']
```

```
        ]
    },
    methods:{
        changeArrayItem: function(){
            this.$set(this.myArr[1], 1, 'strawberry');
        }
    }
})
```

jsfiddle

```
new Vue({
    el: '#app',
    data:{
        myArr : [
            {
                name: 'object-1',
                nestedArr: ['apple', 'banana']
            },
            {
                name: 'object-2',
                nestedArr: ['grapes', 'orange']
            }
        ]
    },
    methods:{
        changeArrayItem: function(){
            this.$set(this.myArr[1].nestedArr, 1, 'strawberry');
        }
    }
})
```

https://riptutorial.com/zh-CN/vue-js/topic/10679/

S. No		Contributors
1	Vue.js	Community, Erick Petrucelli, ironcladgeek, J. Bruni, James, Lambda Ninja, m_callens, MotKohn, rap-2-h, Ru Chern Chong, Sankalp Singha, Shog9, Shuvo Habib, user1012181, user6939352, Yerko Palma
2	Polyfill“webpack”	Stefano Nepa
3	VueJS + ReduxVua-Redux	Aniko Litvanyi, FlatLander, Shuvo Habib, Stefano Nepa
4	Vuex	AldoRomo88, Amresh Venugopal, Daniel Waghorn, Matej Vrzala M4, Ru Chern Chong
5	Vue	jordiburgos, Ru Chern Chong
6	VUE	AJ Gregory
7	v-model	Amresh Venugopal
8		sept08
9		chuanxd, gurghet, Mahmoud, Theo
10		Med, Ru Chern Chong
11	Vue“this”	Bert
12		Finrod, M U, m_callens
13		AldoRomo88
14		gurghet, Jilson Thomas
15		jaredsk, m_callens, Nirazul, user6939352
16		Elfayer
17		Amresh Venugopal
18		Ogie Sado
19		Linus Borg, m_callens, PatrickSteele, xtreak
20		El_Matella
21		Donkarnash, Elfayer, Hector Lorenzo, Jeff, m_callens,

		phaberest, RedRiderX, user6939352
22		Daniel Waghorn, Elfayer, Shuvo Habib, Slava
23		Mat J, Ogie Sado
24		Amresh Venugopal, cl3m, jaredsk, m_callens, Theo, Yerko Palma
25		asemahle, Donkarnash, FlatLander, m_callens, rap-2-h, Shuvo Habib
26		Vamsi Krishna