

VUE.JS



An incrementally adoptable ecosystem that scales between a library and a full-featured framework.

USING VUE.JS

CDN:

```
<head>  
<script  
  
src=https://cdn.jsdelivr.net/npm/vue/dist/vue.js >  
  
</script>  
</head>
```

TEMPLATE SYNTAX

<https://jsfiddle.net/>

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

<script>
var app = new Vue( {
  el: '#app',
  data: {
    message: 'Hello Vue!'
  }
})
</script>
```

BIND ELEMENT ATTRIBUTES

```
<div id="app-2">
  <span v-bind:title="message">
    Hover your mouse over me!
  </span>
</div>

<script>
var app2 = new Vue({
  el: '#app-2',
  data: {
    message: 'You loaded this page on ' + new
Date().toLocaleString()
  }
})
</script>
```

CONDITIONALS

```
<div id="app-3">  
  <span v-if="seen">Now you see me</span>  
</div>
```

```
<script>
```

```
var app3 = new Vue({  
  el: '#app-3',  
  data: {  
    seen: true  
  }  
})
```

```
</script>
```

LOOPS

```
<div id="app-4">
  <ol>
    <li v-for="todo in todos">
      {{ todo.text }}
    </li>
  </ol>
</div>

var app4 = new Vue({
  el: '#app-4',
  data: {
    todos: [
      { text: 'Learn JavaScript' },
      { text: 'Learn Vue' },
      { text: 'Build something awesome' }
    ]
  }
})
```

USER INPUT

```
<div id="app-5">
  <p>{{ message }}</p>
  <button v-on:click="reverseMessage">Reverse
Message</button>
</div>
```

```
var app5 = new Vue({
  el: '#app-5',
  data: {
    message: 'Hello Vue.js!'
  },
  methods: {
    reverseMessage: function () {
      this.message =
this.message.split('').reverse().join('')
    }
  }
})
```

V-MODEL DIRECTIVE

```
<div id="app-6">
  <p>{{ message }}</p>
  <input v-model="message">
</div>
```

```
<script>
var app6 = new Vue( {
  el: '#app-6',
  data: {
    message: 'Hello Vue!'
  }
} )
</script>
```


COMPONENTS #1

```
<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>
```

COMPONENTS #2

```
Vue.component('todo-item', {
  props: ['todo'],
  template: '<li>{{ todo.text }}</li>'
})

var app7 = new Vue({
  el: '#app-7',
  data: {
    groceryList: [
      { id: 0, text: 'Vegetables' },
      { id: 1, text: 'Cheese' },
      { id: 2, text: 'Whatever else humans are
supposed to eat' }
    ]
  }
})
```

INSTANCE LIFECYCLE HOOKS

- **created**
- mounted
- updated
- destroyed

```
new Vue( {  
  data: {  
    a: 1  
  },  
  created: function () {  
    // `this` points to the vm instance  
    console.log('a is: ' + this.a)  
  }  
} )
```

VUE.JS DIRECTIVES

- `v-once` one-time interpolation
- `v-html` output real HTML
- `v-bind` `<a v-bind:href="url"> ... `
- `v-for`
- `v-on` `<a v-on:click="doSomething"> ... `

Conditionals

- `v-if`
- `v-else`
- `v-else-if`
- `v-show`

SHORTHANDS

v-bind Shorthand

```
<!-- full syntax -->
```

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

```
<!-- shorthand -->
```

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

v-on Shorthand

```
<!-- full syntax -->
```

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

```
<!-- shorthand -->
```

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

COMPUTED PROPERTIES

```
<div id="example">
  <p>Original message: "{{ message }}"</p>
  <p>Computed reversed message: "{{ reversedMessage }}"</p>
</div>

var vm = new Vue({
  el: '#example',
  data: {
    message: 'Hello'
  },
  computed: {
    // a computed getter
    reversedMessage: function () {
      return this.message.split('').reverse().join('')
    }
  }
})
```

..AND METHODS

```
<div id="example">
  <p>Original message: "{{ message }}"</p>
  <p>Computed reversed message: "{{ reversedMessage( )
  }}"</p>
</div>

var vm = new Vue({
  el: '#example',
  data: {
    message: 'Hello'
  },
  methods: {
    reverseMessage: function () {
      return this.message.split('').reverse().join('')
    }
  }
})
```

WATCHERS #1

```
<div id="watch-example">
  <p>
    Ask a yes/no question:
    <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>
```


WATCHERS #2

```
<script>
var watchExampleVM = new Vue({
  el: '#watch-example',
  data: {
    question: '',
    answer: 'I cannot give you an answer until you ask a
question!'
  },
  watch: {
    // whenever question changes, this function will run
    question: function (newQuestion, oldQuestion) {
      this.answer = 'Waiting for you to stop typing...'
    }
  }
})
```

WATCHERS #3

```
    this.debounceGetAnswer()
  },
  created: function () {

    this.debounceGetAnswer = _.debounce(this.getAnswer, 500)
  },
  methods: {
    getAnswer: function () {
      if (this.question.indexOf('?') === -1) {
        this.answer = 'Questions usually contain a question
mark. ;-)'
        return
      }
    }
  }
}
```

WATCHERS #4

```
this.answer = 'Thinking...'
var vm = this
axios.get('https://yesno.wtf/api')
  .then(function (response) {
    vm.answer = _.capitalize(response.data.answer)
  })
  .catch(function (error) {
    vm.answer = 'Error! Could not reach the API. ' + error
  })
}
})
</script>
```

AJAX

```
<!-- gives you the freedom to use what you're familiar with.
-->
```

```
<script
```

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

```
    var vm = this
```

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

```
        .then(function (response) {
```

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

```
        })
```

```
        .catch(function (error) {
```

```
            vm.answer = 'Error! Could not reach the API. ' + error
        })
```

CLASS BINDINGS

Object Syntax

```
<div v-bind:class="classObject"></div>
data: {
  classObject: {
    active: true,
    'text-danger': true
  }
}
```

Array Syntax

```
<div v-bind:class="[activeClass, errorClass]"></div>
data: {
  activeClass: 'active',
  errorClass: 'text-danger'
}
```

Output

```
<div class="active text-danger"></div>
```

STYLE BINDINGS

Object Syntax

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

```
data: {  
  styleObject: {  
    color: 'red',  
    fontSize: '13px'  
  }  
}
```

EVENTS

```
<div id="example-3">  
  <button v-on:click="say('hi')">Say hi</button>  
  <button v-on:click="say('what')">Say  
what</button>  
</div>
```

```
new Vue({  
  el: '#example-3',  
  methods: {  
    say: function (message) {  
      alert(message)  
    }  
  }  
})
```

FORM INPUT BINDINGS

```
<div id='example-3'>
  <input type="checkbox" id="jack" value="Jack" v-model="checkedNames">
  <label for="jack">Jack</label>
  <input type="checkbox" id="john" value="John" v-model="checkedNames">
  <label for="john">John</label>
  <input type="checkbox" id="mike" value="Mike" v-model="checkedNames">
  <label for="mike">Mike</label>
  <br>
  <ul>
    <li v-for="name in checkedNames">
      {{ name }}
    </li>
  </ul>
</div>
```

```
new Vue( {
  el: '#example-3',
  data: {
    checkedNames: [ ]
  }
})
```


COMPONENTS

```
<div id="demo">
  <button-counter start="3"></button-counter>
</div>

// Define a new component called button-counter
Vue.component('button-counter', {
  props: ['start'],
  data: function () {
    return {
      count: this.start
    }
  },
  template: '<button v-on:click="count++">You
clicked me {{ count }} times.</button>'
})

new Vue({ el: '#demo' })
```