



Web Application Development II

Week 7: Vue.js Basics



Agenda

Introduction and installation

Vue Instance

Vue Directives

Data Bindings

Conditional Rendering

Event Handling

Putting them together



What is a Reactive Application?

A reactive web application

- Observe changes in application state
- Propagate change notification throughout the application
- Render views automatically in response to changes in state
- Provide timely feedback for user interactions

Responsive

Resilient

Scalable

Message-driven

What is Vue.js

Vue (pronounced like View) is a progressive framework for building user interfaces.

The core library is focused on the view layer only, and is easy to pick up and integrate with other libraries or existing projects.

On the other hand, Vue is also perfectly capable of powering sophisticated Single-Page Applications when used in combination with modern tooling and supporting libraries.

Example

[Themes](#)

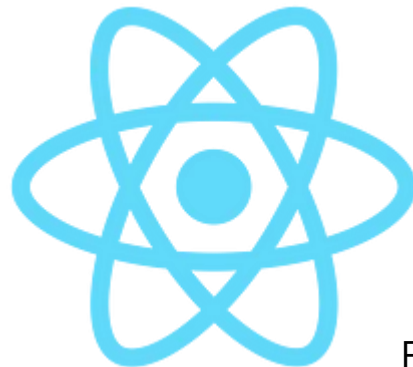
Why Vue.js



Used by
Alibaba

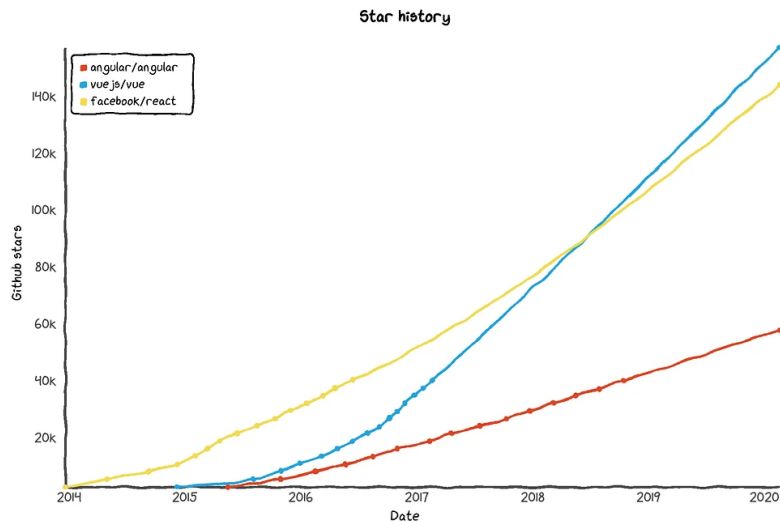


Google



Facebook

Why Vue.js



Popularity: [Read Here](#)

"React has been a leader in popularity among JavaScript frameworks. It takes the first place with the number of 48,718 dependents, whereas, Vue.js is the second most popular JavaScript framework with half as many dependents — 21,575, as reported by Node Package Manager. Though, if we take a look at statistics at the end of 2018, we will see that Vue's dependents had tripled, while React's had doubled."

Vue.js

Extremely small (16KB)

Great runtime performance

Installation is almost trivial

Easy to learn

Examples

```
<html>
<head>
  <script src="vue.js"></script>
</head>
...
</html>
```

Vue.js Features

Data Binding: This feature helps manipulate or assign values to HTML attributes, change the style, assign classes with the help of binding directive called v-bind

Directives: built-in directives such as v-if, v-else, v-on, v-bind, and v-model, which are used to perform various actions on the frontend.

Event Handling: with v-on directive, it can listen to the events

Computed Properties: helps to listen to the changes made to the UI elements and performs necessary calculations

Components: custom elements that can be reused in HTML

Vue.js Hello World

Installation — Vue.js

There are two versions.

- Production version `vue.min.js`
- Developer version `vue.js` (with full warnings and helps debugging)

You can either download the file and install locally or use CDN.

Example: *wk7example1.html*

```
<script src="vue.js"></script>
```

Alternatively, use CDN

```
<script src="https://unpkg.com/vue@next"></script>
```

Vue.js: Syntax

Create a new Vue instance by passing in an **options** object.

Options are **optional** but typically contain 2 options: **data** and **methods**

Other common options: **mounted**, **computed**, **created**

Full options list:

<https://v3.vuejs.org/api/options-data.html>

Example: *wk7example2.html*

// create Vue instance containing data and methods

```
const app = Vue.createApp( {  
  data() {  
    return {  
      // name:value pairs  
    }  
  },  
  methods: {  
    // format  
    foo() {  
      return "a JS function";  
    }  
  }  
})
```

// Link (mount) Vue instance to DOM

```
const vm = app.mount("#app")
```

Vue.js: Syntax

Alternatively:

Example: *wk7example2.html*

```
Vue.createApp( {  
    data() {  
        return {  
            // name:value pairs  
        }  
    },  
    methods: {  
        // format  
        foo() {  
            return "a JS function";  
        }  
    }  
})  
).mount('#app')
```

Since options are optional, a Vue instance can be created as `Vue.createApp({ }).mount('#app')`

Data Binding

The most basic form of data binding is text interpolation using the “Mustache” syntax (double curly braces):

`{{ ... }}` tells Vue to take everything inside them and parse it as code.

The value will be updated whenever there is a change in property.

Example: *wk7example2.html*

```
<h1> Name : {{name}} </h1>  
<h1> Phone : {{getPhone()}} </h1>
```

In the above example, *name* will be replaced with the value of the name property of the vue instance.

getPhone() will be replaced with the outcome of the invocation of vue instance’s *getPhone* method.

Data Binding

The most basic form of data binding is text interpolation using the “Mustache” syntax (double curly braces):

`{{ ... }}` tells Vue to take everything inside them and parse it as code.

The value will be updated whenever there is a change in property.

Experiment:

1. Inspect *wk7example2.html* in Chrome
2. Change phone property in the Console
3. Observe that the page is updated automatically

Vue Debugging 101

Use Chrome console to debug

Enter variable names and check if you get expected results

If you run into something you can't figure out, or you find a particularly nasty error, check out <https://forum.vuejs.org/c/help>

Vue extension for Chrome could be useful.

Two common issues

Symptom: "Uncaught SyntaxError: Unexpected identifier"

Illness: usually typo error, missing comma or braces, or hidden strange characters

Symptom: "[Vue warn]: Property or method \"propertyname\" is not defined"

Illness: something was not defined in the options object (property or method)

Vue Template

Vue code can be placed in a separate file and imported.

Separation of concern is a fundamentally good principle

Example: *wk7example3.html*

```
<div id="app">  
  <h1> Name : {{name}} </h1>  
  <h1> Phone : {{getPhone()}} </h1>  
</div>
```

```
<script src="js/wk7example3.js"></script>
```

Name : John

Phone : This is my number 123456

Exercise 1

In ex1-fruits.html, the HTML part of the page is given. Program the script part by creating a Vue instance so that the page looks like the following.

Choose your favorite fruit:

apple ☐ orange ☐ peach ☐

Vue Directives

Directives are special attributes with the v-prefix.

A directive's job is to reactively apply side effects to the DOM when the value of its expression changes.

Directive attribute values are expected to be a single JavaScript expression.

Example: *wk7example4.html*

```
<p v-if="x==1">Now you see me</p>
```

Here, the v-if directive would remove/insert the <p> element based on the truthiness of the expression `x==1`.

Note that x should be a variable of the corresponding Vue instance (i.e., the one which is mounted to this part of the DOM tree).

*explore **v-else** on your own

Vue Directives: wk7example4.html

| | | |
|----------------|---|--|
| v-bind | Bind an attribute value with a vue variable | |
| v-html | Print a given vue variable as HTML | <div v-html="message"></div> |
| v-on | Listen to DOM events and run code | <button v-on:click="counter += 1">Add</button> |
| v-for | Loop through items in a Vue parameter – list (items) | <li v-for="item in items"> |
| v-if v-else | Remove/insert an element based on the truth value of Vue parameter | <div v-if="seen">You either see this</div> <div v-else>or this!</div> |
| v-model | Bind Vue parameters to the values of input elements such as text, radio, and checkbox | <input v-model="msg"> |

Directives with Arguments

Some directives can take an “argument”, denoted by a colon after the directive name.

For example, the v-bind directive is used to reactively update an HTML attribute.

Another example is the v-on directive, which listens to DOM events.

Examples

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

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

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

Exercise 2

In ex2-vbind.html, the script part is given. Complete the HTML part such that the image is displayed with data defined in the vue instance.



Use the given 'width' and 'height' properties to set the image width and height

v-if/v-else Revisited

The directive v-if is used to conditionally render a block. The block will only be rendered if the directive's expression returns true.

The directive's expression can also be a JavaScript expression that evaluates to either true or false.

It is also possible to add an "else block" with v-else.

Example:

wk7example5.html

Exercise 3

In ex3-vif.html, a dropdown list is provided to allow users to login with either username or email. Render the page different depending on the selection, i.e., show an input for an email or a username accordingly.

Hint: use v-if and v-else to accomplish the task.

Please select your login option

Username login ▼

Username

Please select your login option

Email login ▼

Email

Class and Style Binding

A common need for data binding is manipulating an element's class and its inline styles, to control CSS

Use v-bind to bind class and style attributes

Syntax:

```
v-bind:class="{className' : isActive}"
```

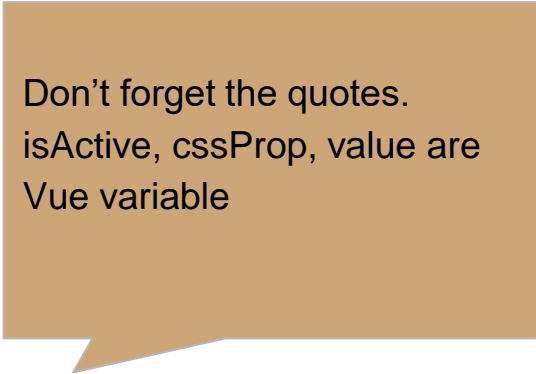
```
v-bind:class = "[cssProp, anotherCssProp]"
```

```
v-bind:style="{property' : value}"
```

Example:

wk7example6.html

See the effect by setting vm.isActive = false



Don't forget the quotes.
isActive, cssProp, value are
Vue variable

v-on Revisited

Use v-on directive to listen to DOM events and run some JavaScript when they're triggered

Syntax

```
<div v-on:eventname="expr"></div>
```

where eventname is any standard DOM event name (e.g. click, submit, change, focus, keyup and onEnter) and “expr” is a Javascript expression or function name.

Example:

wk7example7.html

“expr” is typically a method in the vue instance.

v-on Revisited

When listening for keyboard events, we often need to check for specific keys. Vue allows adding key modifiers for v-on when listening for key events.

Commonly-used key codes: .enter, .tab, .delete (for both “Delete” and “Backspace”), .esc, .space, .up, .down, .left, and .right.

Example: *wk7example8.html*

```
<!-- invoke hello() when `Enter` key is entered -->  
<input v-on:keyup.enter="hello(name)">
```

Exercise 4

Add code in ex4-css.html such that when you click “Change Color”, it changes to red from blue or to blue for red alternately. Use v-on and v-bind.

If time allows, do it in two versions, one with Vue.js and one with vanilla Javascript. Compare the two versions.

div ID : demo1

Change Color

div ID : demo1

Change Color

Take Away Message

With vanilla Javascript, you will constantly go from HTML to Script and back (via DOM, search from the root of the DOM tree).

With Vue.js, you declare the connection between HTML and Script and focus more on the logic in Vue.

Put it simply: no more *document.getElement...*