Programação Web I

M01 - Introduction to Vue.js

ESMAD | TSIW | 2020-21

M01 - Introduction to Vue.js

Index

M01 - Introduction to Vue.js

Index

M01 - Introduction to Vue.js

1. Introduction to Vue.js

- JavaScript progressive framework
- Created by Evan You
- History:
  - Started in 2013
  - Last version: 3.0 (October, 2020)
- Links:
  - Site: https://vuejs.org
  - Repository: https://github.com/vuejs/vue-next
- License: MIT

M01 - Introduction to Vue.js

1. Introduction to Vue.js

- Javascript Framework
  - To organize and simplify the frontend development
  - To develop interactive Web interfaces
- Main advantages:
  - Small (33kb - production version)
  - Easy to install
  - Easy to learn (small learning curve)
  - Easy to integrate with other projects and libraries

M01 - Introduction to Vue.js

1. Introduction to Vue.js

- Main libraries/tools
  - Vue-router
  - Vuex
  - Vue-loader
  - Vue-devtools
  - Vue-cli
  - Vue-test-utils
  - Vuetify
  - Bootstrap Vue

M01 - Introduction to Vue.js

Index

M01 - Introduction to Vue.js

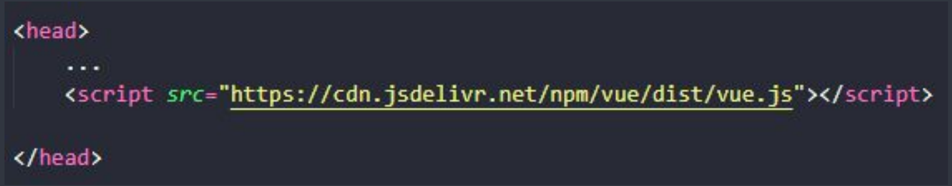
2. Installation

- 3 ways to install Vue.js
  - Using CDN
  - Using NPM
  - Using Vue CLI

M01 - Introduction to Vue.js

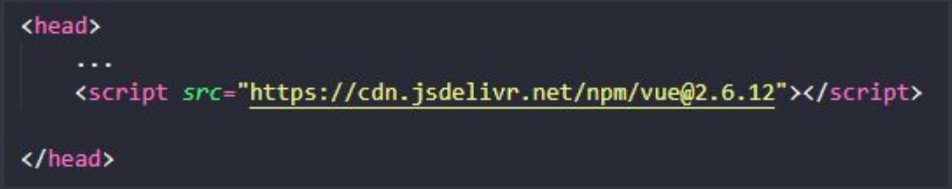
2. Installation

- CDN
  - For prototyping or learning purposes, use the latest version:



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

  - For production, link to a specific version number and build to avoid unexpected breakage from newer versions

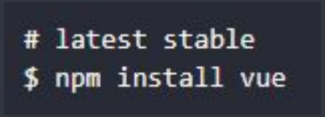


```
<head>
    ...
    <script src="https://cdn.jsdelivr.net/npm/vue@2.6.12"></script>
</head>
```

M01 - Introduction to Vue.js

2. Installation

- Node Package Manager (NPM)
  - NPM is the recommended installation method when building large scale applications with Vue.

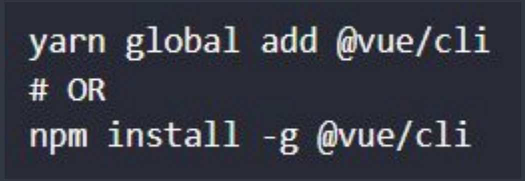


```
# latest stable
$ npm install vue
```

M01 - Introduction to Vue.js

2. Installation

- Vue CLI
  - Vue provides an official CLI for quickly scaffolding Single Page Applications (SPA)
  - It provides build setups for a modern frontend workflow, hot-reload, lint-on-save, etc.
  - For Vue 3, use Vue CLI v4.5 available on npm as @vue/cli

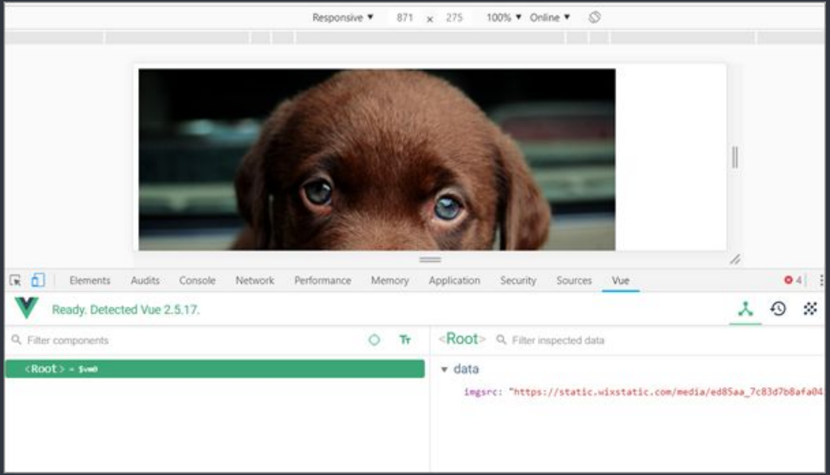


```
yarn global add @vue/cli
# OR
npm install -g @vue/cli
```

M01 - Introduction to Vue.js

2. Installation

- Complementary installations
  - Visual Studio Code
  - Vue DevTools - Browser extension to vue.js apps debug
  - Vetur extension (syntax highlighting, snippets, etc.)



Ready. Detected Vue 2.5.17.

<Root> = $vm0

<Root>

▾ data

```
imgsrc: "https://static.wixstatic.com/media/ed85aa_7c83d7b8afa04
```

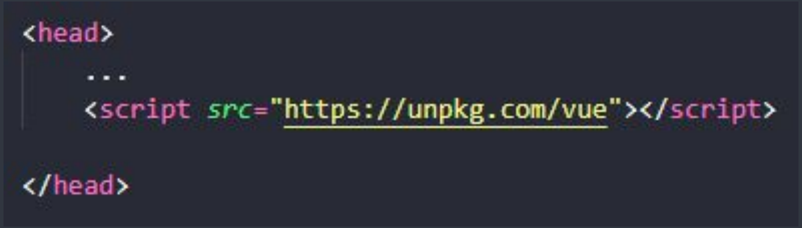
M01 - Introduction to Vue.js

Index

M01 - Introduction to Vue.js

3. My first application

1. Create a folder HelloVue
2. Open the folder in VSC
3. Create a file index.html and add the initial skeleton
4. Create a reference in the html file to the vue.js file

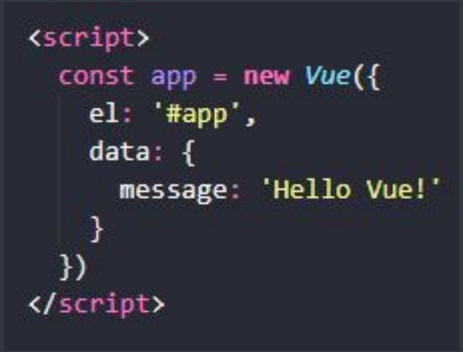


```html
<head>
    ...
    <script src="https://unpkg.com/vue"></script>

</head>
```

M01 - Introduction to Vue.js

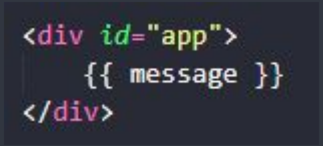
3. My first application

5. Add a new tag <script> to create a new Vue instance



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

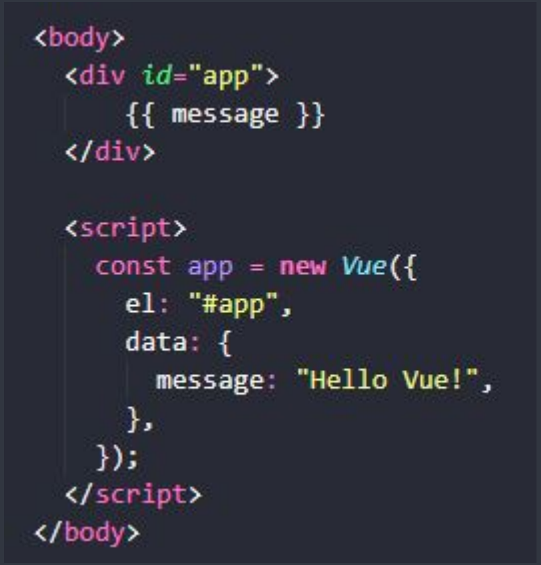
6. Add a <div> tag inside the <body> tag



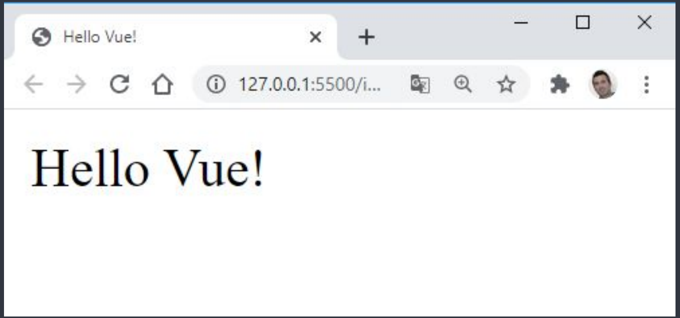
```
<div id="app">
    {{ message }}
</div>
```

M01 - Introduction to Vue.js

3. My first application



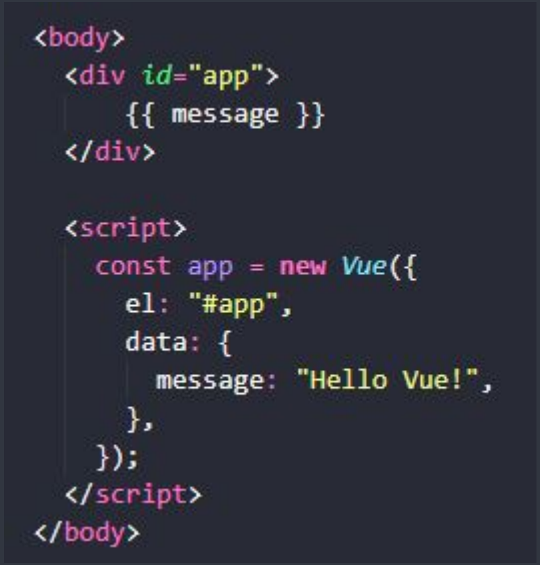
```
<body>
  <div id="app">
    {{ message }}
  </div>

  <script>
    const app = new Vue({
      el: "#app",
      data: {
        message: "Hello Vue!",
      },
    });
  </script>
</body>
```



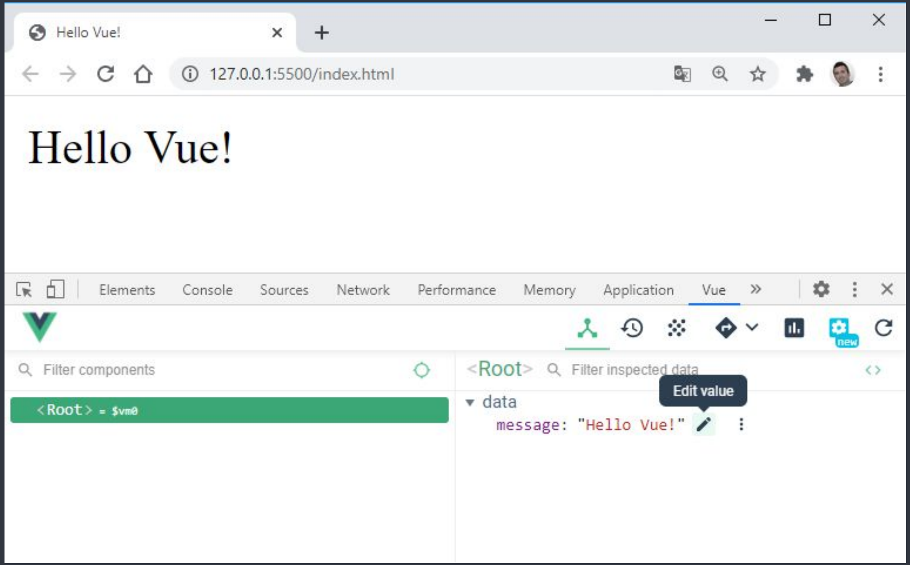
Hello Vue!

M01 - Introduction to Vue.js

3. My first application



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

  <script>
    const app = new Vue({
      el: "#app",
      data: {
        message: "Hello Vue!",
      },
    });
  </script>
</body>
```



Hello Vue!

```
message: "Hello Vue!"
```

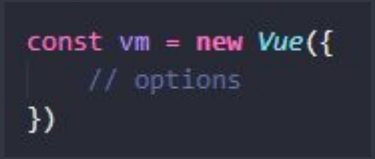
M01 - Introduction to Vue.js

Index

M01 - Introduction to Vue.js

4. The Vue instance

- Every Vue application starts by creating a new Vue instance with the Vue function:

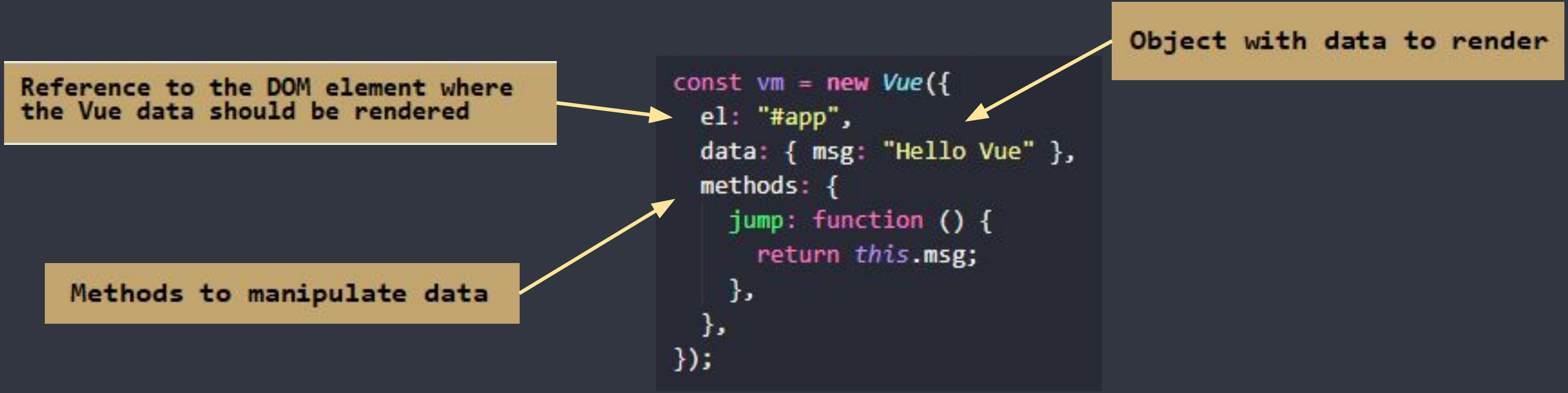


```
const vm = new Vue({
    // options
})
```

M01 - Introduction to Vue.js

4. The Vue instance

- When we create a vue instance we need to pass a options object
- Main properties of the object:



Reference to the DOM element where the Vue data should be rendered

Object with data to render

Methods to manipulate data

```
const vm = new Vue({
  el: "#app",
  data: { msg: "Hello Vue" },
  methods: {
    jump: function () {
      return this.msg;
    },
  },
});
```

M01 - Introduction to Vue.js

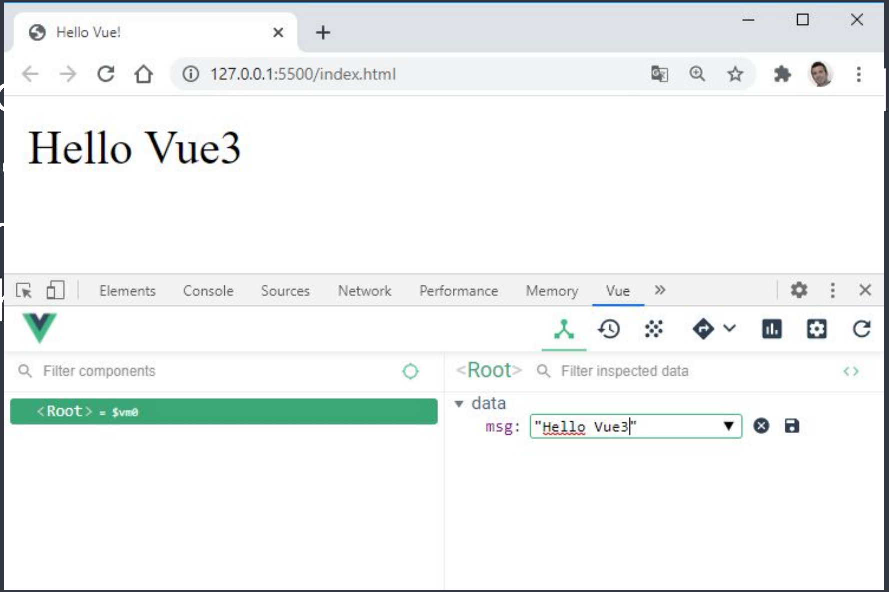
4. The Vue instance

- data property
  - When a Vue instance is created, it adds all the properties found in its data object to Vue's reactivity system.
  - When the values of those properties change, the view will "react", updating to match the new values.

M01 - Introduction to Vue.js

4. The Vue instance

- data property
  - When a Vue instance is crea
    in its data object to Vue's rea
  - When the values of those pr
    "react", updating to match th



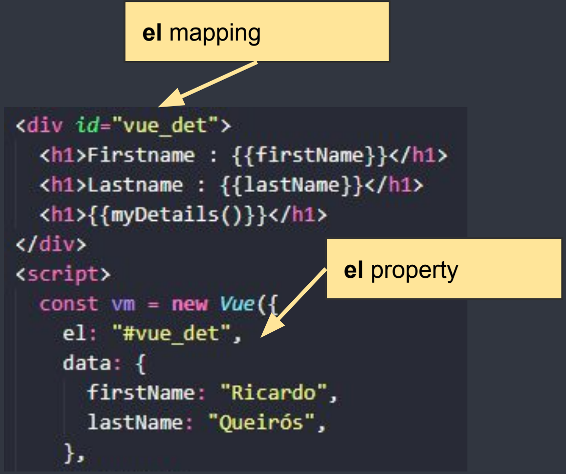
Hello Vue3

```
msg: "Hello Vue3"
```

M01 - Introduction to Vue.js

4. The Vue instance

- el property
  - References an id of a DOM element
  - All reactivity on data object properties will only be done on this element
  - Values in data object are presented in the element pointed by el with interpolation using the mustache syntax {{...}}

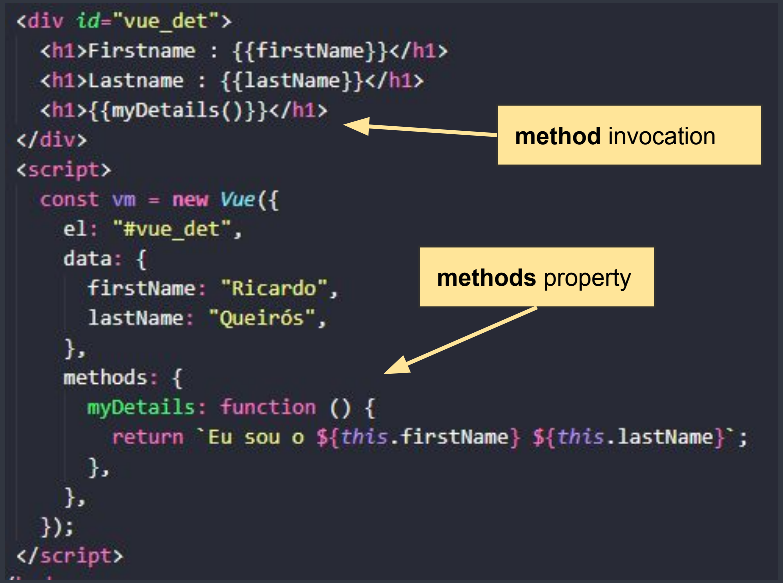


```
<div id="vue_det">
  <h1>Firstname : {{firstName}}</h1>
  <h1>Lastname : {{lastName}}</h1>
  <h1>{{myDetails()}}</h1>
</div>
<script>
  const vm = new Vue({
    el: "#vue_det",
    data: {
      firstName: "Ricardo",
      lastName: "Queirós",
    },
```

M01 - Introduction to Vue.js

4. The Vue instance

- methods property
  - We can also add methods to the instance through the methods object



```
<div id="vue_det">
  <h1>Firstname : {{firstName}}</h1>
  <h1>Lastname : {{lastName}}</h1>
  <h1>{{myDetails()}}</h1>
</div>
<script>
  const vm = new Vue({
    el: "#vue_det",
    data: {
      firstName: "Ricardo",
      lastName: "Queirós",
    },
    methods: {
      myDetails: function () {
        return `Eu sou o ${this.firstName} ${this.lastName}`;
      },
    },
  });
</script>
```

M01 - Introduction to Vue.js

4. The Vue instance

- Other options: props, computed, etc.
- Built-in properties: $attrs and $emit. These properties all have a $ prefix to avoid conflicting with user-defined property names.

M01 - Introduction to Vue.js

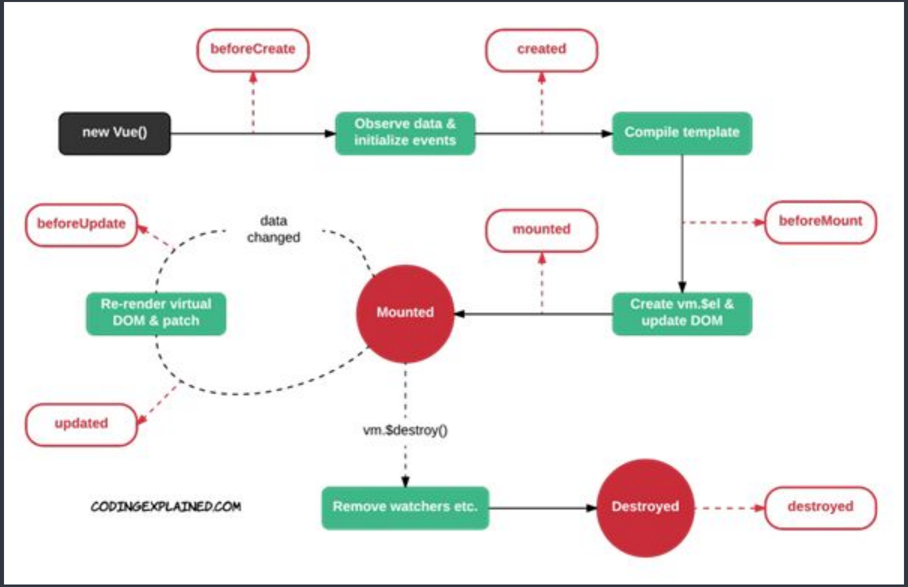
4. The Vue instance

- Life cycle
  - Each instance of Vue goes through a series of startup steps when it is created
  - For example, Vue needs to configure data observation, compile the template, mount the instance in DOM, and update the DOM when data changes
  - Along the way, it also execute functions called lifecycle functions, giving users the opportunity to add code at specific stages of the cycle

M01 - Introduction to Vue.js

4. The Vue instance

- Life cycle



M01 - Introduction to Vue.js

4. The Vue instance

- Four phases with two functions each:

1. Creation (initialization) - beforeCreate and created
2. Mounting (DOM Insertion) - beforeMount and mounted
3. Update (differentiate and render again) - beforeUpdate and updated
4. Teardown - beforeDestroy and destroyed

M01 - Introduction to Vue.js

4. The Vue instance

- Creation (initialization)
  - beforeCreate: fired before instance is initialized
  - created: Fired after the instance was initialized, but before being added to the DOM (good time to get data from external sources)

- Mounting (DOM Insertion)
  - beforeMount: fired after the element is ready to be added to the DOM, but before that
  - mounted: fired after the element has been created (but not necessarily added to DOM: use nextTick for this)

M01 - Introduction to Vue.js

4. The Vue instance

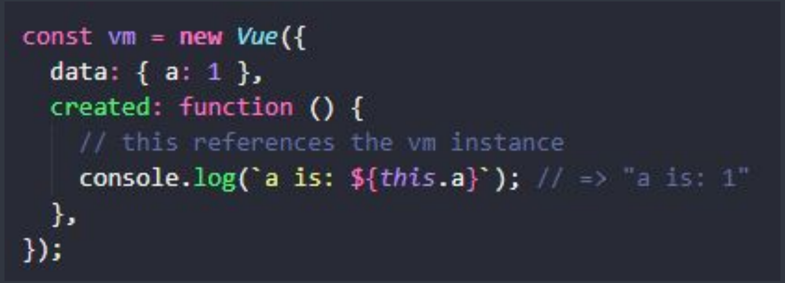
- Update (differentiate and render again)
  - beforeUpdate: fired when there are changes to make to DOM
  - updated: fired after changes are written to DOM

- Teardown
  - beforeDestroy: fired when component is about to be destroyed and removed from DOM
  - destroyed: fires after component has been destroyed

M01 - Introduction to Vue.js

4. The Vue instance

- For instance, the created function can be used to execute code after creating an instance:



```
const vm = new Vue({
  data: { a: 1 },
  created: function () {
    // this references the vm instance
    console.log(`a is: ${this.a}`); // => "a is: 1"
  },
});
```

M01 - Introduction to Vue.js

Index

M01 - Introduction to Vue.js

5. Template syntax

- Vue.js uses an HTML-based template syntax that allows you to declaratively bind the rendered DOM to the Vue instance's data.

- Vue compiles the templates into Virtual DOM render functions.

- Using the reactivity system, Vue finds the minimal number of components to re-render and apply DOM manipulations when the app state changes.

M01 - Introduction to Vue.js

5. Template syntax

- Main concepts in template syntax:
  - Interpolations
  - Directives
  - Shortands

M01 - Introduction to Vue.js

5. Template syntax

- Interpolations
  - Vue.js uses an HTML-based template syntax that lets you declaratively link rendered DOM to underlying Vue instance data
  - Interpolation Types:
    - Text
    - Html
    - Attributes
    - Javascript expressions

M01 - Introduction to Vue.js

5. Template syntax

- Interpolations > Text
    - For text interpolation use the “Mustache” syntax (double curly braces):

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

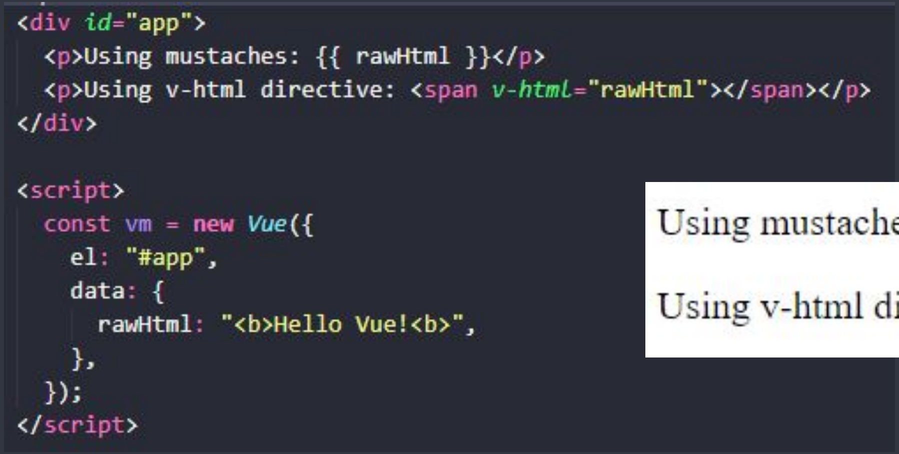
    - The mustache tag will be replaced with the value of the msg property on the corresponding data object. It will also be updated whenever the data object’s msg property changes.
    -
    - You can also perform one-time interpolations that do not update on data change by using the v-once directive:

```html
<span v-once>This will never change: {{ msg }}</span>
```

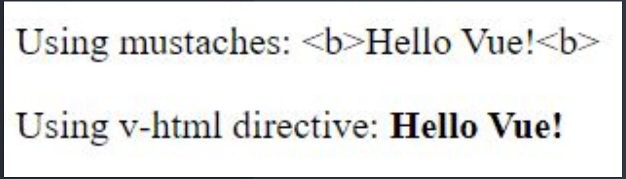
M01 - Introduction to Vue.js

5. Template syntax

- Interpolations > Html
    - The mustache tag interprets data as plain text, not HTML
    - To produce real HTML, you must use the v-html directive:



```
<div id="app">
  <p>Using mustaches: {{ rawHtml }}</p>
  <p>Using v-html directive: <span v-html="rawHtml"></span></p>
</div>

<script>
  const vm = new Vue({
    el: "#app",
    data: {
      rawHtml: "<b>Hello Vue!<b>",
    },
  });
</script>
```



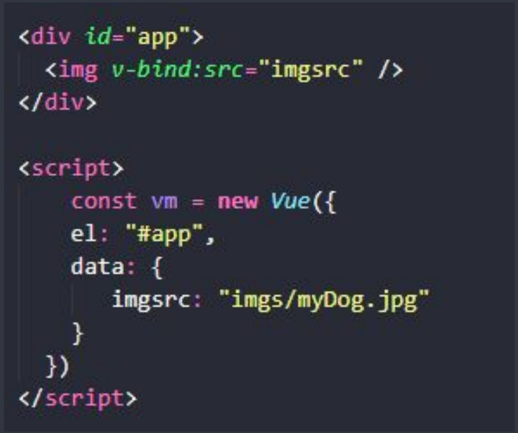
Using mustaches: <b>Hello Vue!<b>

Using v-html directive: **Hello Vue!**

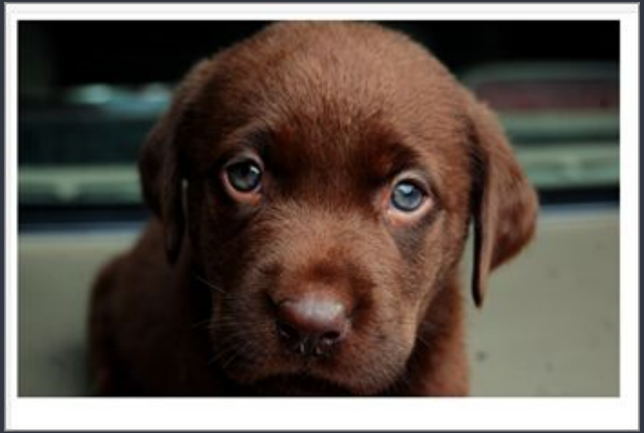
M01 - Introduction to Vue.js

5. Template syntax

- Interpolations > Attributes
  - Mustache tag cannot be used inside html attributes
  - For this you must use the v-bind directive:



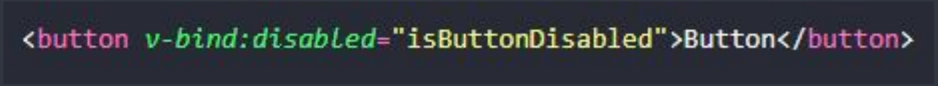
```
<div id="app">
  <img v-bind:src="imgsrc" />
</div>

<script>
    const vm = new Vue({
    el: "#app",
    data: {
        imgsrc: "imgs/myDog.jpg"
    }
  })
</script>
```



M01 - Introduction to Vue.js

5. Template syntax

- Interpolations > Attributes
  - For boolean attributes (true or false), v-bind works differently:



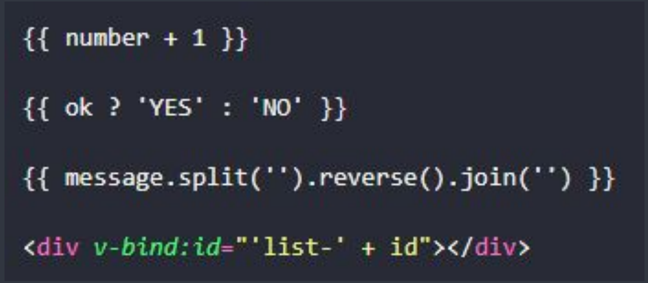
```
<button v-bind:disabled="isButtonDisabled">Button</button>
```

  - If isButtonDisabled has a value of null, undefined, or false, then the disabled attribute is not included in the rendered <button> element

M01 - Introduction to Vue.js

5. Template syntax

- Interpolations > JS expressions
  - Vue.js supports JavaScript expressions within all data bindings



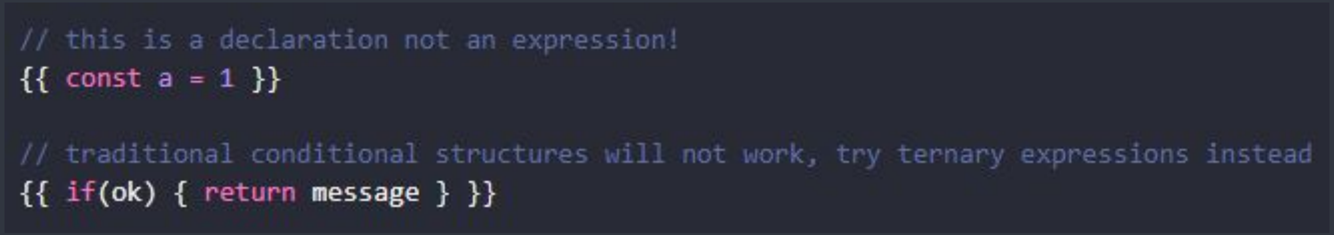
```
{{ number + 1 }}

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

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

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

  - The expressions will be evaluated as JS in the instance data scope

M01 - Introduction to Vue.js

5. Template syntax

- Interpolations > JS expressions
  - One restriction is that each binding can contain only a single expression, so the following will NOT work:

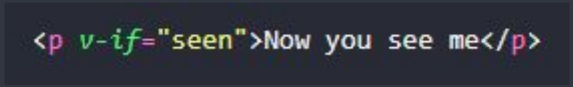


```
// this is a declaration not an expression!
{{ const a = 1 }}

// traditional conditional structures will not work, try ternary expressions instead
{{ if(ok) { return message } }}
```

M01 - Introduction to Vue.js

5. Template syntax

- Directives
  - Directives are special attributes with the prefix v-
  - Attribute values are a single JavaScript expression (except v-for)
  - The job of a directive is to apply side-effects reactively to the DOM when the value of its expression changes
  - Here's an example of a simple directive:



```html
<p v-if="seen">Now you see me</p>
```

  - Here, the v-if directive would remove/insert the <p> element based on the value of the seen expression

M01 - Introduction to Vue.js

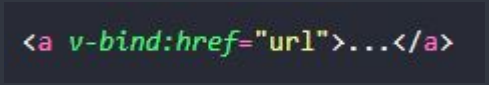
5. Template syntax

- Directives
  - Can have:
    - Arguments
    - Modifiers

M01 - Introduction to Vue.js

5. Template syntax

- Directives with arguments
  - Some directives may be given an "argument", denoted by a colon after the directive name. For instance, the v-bind directive is used to reactively update an HTML attribute:



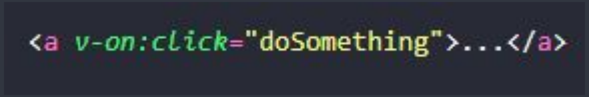
```
<a v-bind:href="url">...</a>
```

  - Here, href is the argument, which tells the v-bind directive to bind the element's href attribute to the value of the url expression

M01 - Introduction to Vue.js

5. Template syntax

- Directives with arguments
  - Another example is the v-on directive, which listens for DOM events:



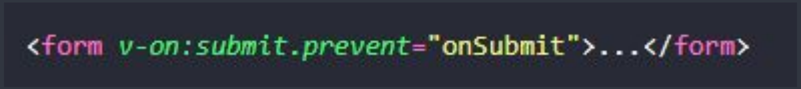
```
<a v-on:click="doSomething">...</a>
```

  - Here the argument is the name of the event to hear

M01 - Introduction to Vue.js

5. Template syntax

- Directives with modifiers
  - Modifiers are special postfixes denoted by a dot, which indicate that a directive must be bound in some special way
  - For instance, the .prevent modifier tells the v-on directive to call event.preventDefault() on the triggered event:

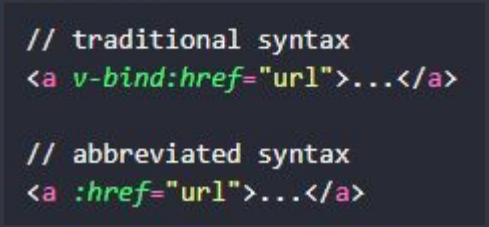


```
<form v-on:submit.prevent="onSubmit">...</form>
```

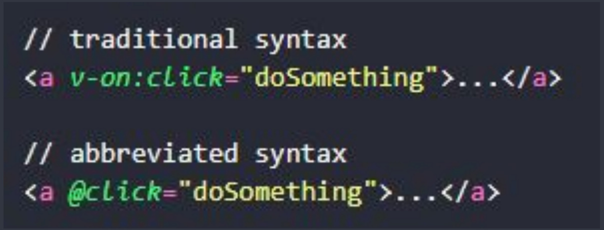
M01 - Introduction to Vue.js

5. Template syntax

- Shorthands
  - The v prefix serves as a cue to identify Vue attributes in your models
  - But it can make the page verbose in case of much use
  - Vue.js provides special shortcuts (: and @) for 2 of the most commonly used directives, v-bind and v-on:



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

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



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

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

M01 - Introduction to Vue.js

Index