**SECTION 2: BASICS & CORE CONCEPTS - DOM INTERACTION WITH VUE**

**13. Module Introduction (0:48m)**

**14. Creating and Connecting Vue App Instances + Resources (7:51m)**

Resource: basics-01-starting-code

<!DOCTYPE html>

<html>

  <head>

    <title>Vue Basics</title>

    <!-- import vue package-->

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

    <script src="app.js" defer></script>

  </head>

  <body>

    <header>

      <h1 id="main-heading"></h1>

    </header>

  </body>

</html>

const app = Vue.createApp({

  data() { // data option configuration, must return object.

    return {

      course: 'Learn Vue.js'

    };

  }

});

app.mount('#main-heading'); // mount holds a css selector, prefer using id as they are unique selectors.

Note: If we controll any HTML element with Vue, we’ll also control all child elements of that element.

**15. Interpolation and Data Binding (3:35m)**

<header id="main-heading">

   <h1>{{ course }}</h1> <!—interpolation: outputs values in data properties, can be used only between opening and closing tags-->

<p>{{ 10 + 20}}</p>

<p>{{5 !== 7 ? 'Hi' : 'Hello'}}</p>

</header>

Can use one line expressions within interpolation.

**16. Binding Attributes with the "v-bind" Directive (5:37m)**

<a v-bind:href="url" id="link">click here</a>

const app = Vue.createApp({

  data() {

    return {

url: 'https://vuejs.org/'

    };

  }

});

app.mount('#link');

**17. Understanding "methods" in Vue Apps (5:47m)**

<p>>{{randomGoal()}}</p>

const app = Vue.createApp({

  methods: { // all properties in methods option must be a function

    randomGoal() {

      const randomNumber = Math.random();

      return (randomNumber > 0.5 ? 'Learn Vue.js' : 'Practice Vue.js')

    }

  }

});

**18. Working with Data inside of a Vue App (3:23m)**

      <small>Duration: {{getDuration()}}</small>

const app = Vue.createApp({

  data() {

    return {

      duration: '21 days'

    };

  },

  methods: {

    getDuration() {

      return this.duration;

    }

  }

});

**19. Outputting Raw HTML Content with v-html (3:31m)**

      <p>

        <span v-html="title"></span>

        <span v-html="getDescription()"></span>

      </p>

const app = Vue.createApp({

  data() {

    return {

      title: '<h3>Vue.js<h3>',

    };

  },

  methods: {

    getDescription() {

      return `<p>Vue.js is a progressive JavaScript framework.

      An approachable, performant and versatile framework for building web user interfaces. </p>`

    }

  }

});

**20. A First Summary (4:12m)**

**21. Understanding Event Binding + Resources (7:17m)**

Resource: basics-03-events-starting-code

Note: whenever any data property is changed, Vue updates DOM only for the lines containing the property.

<button v-on:click="counter++">Add</button>

**22. Events & Methods (6:12m)**

**23. Working with Event Arguments (2:59m)**

**24. Using the Native Event Object (8:59m)**

$event : special keyword to pass event object in function.

**25. Exploring Event Modifiers (9:53m)**

[Event Handling | Vue.js (vuejs.org)](https://vuejs.org/guide/essentials/event-handling.html#event-modifiers) – Event, key & mouse button modifiers. Examples: click.right, click.left, etc.

**26. Locking Content with v-once (2:7m)**

**27. Data Binding +** **Event Binding = Two-Way Binding + Resources (6:32m)**

<input type="password" v-model="password" placeholder="Password">

**28. Methods used for Data Binding: How It Works (6:46m)**

Methods must not be used to output data since if there’s a change any data property, it re-renders the output function too as it’s not able to know whether the output function is related to changed property.

**29. Introducing Computed Properties (5:58m)**

Computed properties must be named just like property names.

      <p>Your Name: {{fullName}}</p>

const app = Vue.createApp({

  data() {

      firstName: '',

      lastName: '',

  },

  computed: { // used when output depends on other property

    fullName() {

      if(this.firstName === '' && !this.lastName === '') {

        return '';

      }

      return `${this.firstName} ${this.lastName}`

    }

  },

});

**30. Working with Watchers (10:49m)**

const app = Vue.createApp({

  data() {

    return {

      counter: 0

    };

  },

  watch: { // used to watch property changes

    counter(value) { // function name should be same as data property or computed property

      if(value > 50) {

        const counterThis = this;

        setTimeout(() => {

          counterThis.counter = 0;

        }, 1000);

      }

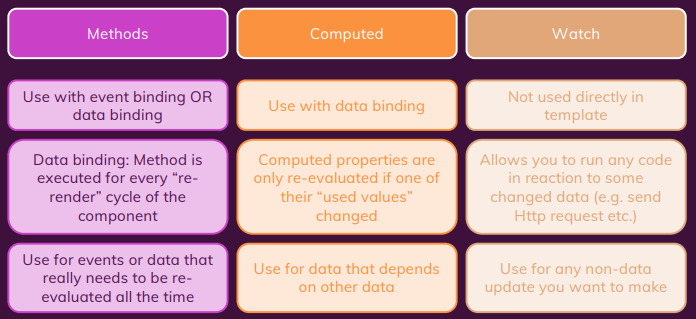
    }

  },

});

Useful for http requests & timers.

**31. Methods vs Computed Properties vs Watchers (2:57m)**



**32. v-bind and v-on Shorthands (1:53m)**

      <!-- <input v-on:input="doSomething" v-bind:value="something"> -->

      <input @input="doSomething" :value="something">

**33. Dynamic Styling with Inline Styles + Resources (7:19m)**

Resource: basics-10-styling-starting-setup

<div class="demo" :style="{'border-color': box1Selected ? 'red' : '#ccc'}" @click="toggleBoxClick(1)"></div>

      <div class="demo" :style="{borderColor: box2Selected ? 'red' : '#ccc'}" @click="toggleBoxClick(2)"></div>

**34. Adding CSS Classes Dynamically (6:31m)**

<div class="demo" :class="{active: box1Selected}" @click="toggleBoxClick(1)"></div>

**35. Classes & Computed Properties (1:57m)**

<div class="demo" :class="box2Classes" @click="toggleBoxClick(2)"></div>

  computed: {

    box2Classes() {

      return {active: this.box2Selected}

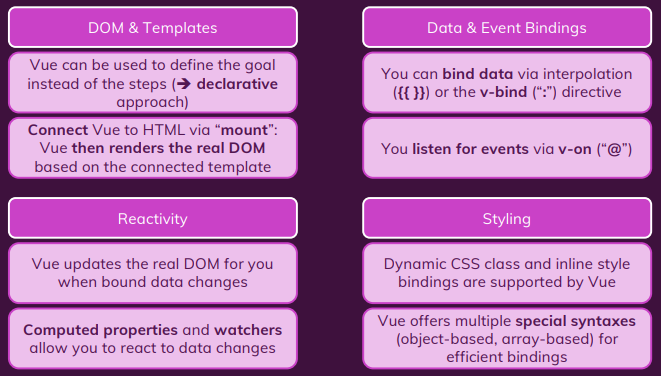
    }

  },

**36. Dynamic Classes: Array Syntax (1:21m)**

<div :class="['demo', {active: box1Selected}]" @click="toggleBoxClick(1)"></div>

**37. Module Summary (4:27m)**



**38. Module Resources + Resources**