**Section 24 – Using VueJS To Interact With The DOM**

Chapter 334 – Chapter 340

<html>

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

<div id="app">

    <h1 v-once>{{ title }}</h1>

    <p>{{ sayHello() }} - <a v-bind:href="link">Google</a></p>

    <hr>

    <p v-html="finishedLink"></p>

</div>

</html>

<script>

new Vue({

    el: '#app',

    data: {

        title: 'Hello World!',

        link: 'http://google.com',

        finishedLink: '<a href="http://google.com">Google</a>'

    },

    methods: {

        sayHello: function() {

            this.title = 'Hello !';

            return this.title;

        }

    }

});

</script>

Chapter 342 – Chapter 347

Index.html

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

<div id="app">

    <button v-on:click="increase(2, $event)">Click Me</button>

    <p>{{ counter }}</p>

    <p v-on:mousemove="updateCoordinates">

        Coordinates: {{ x }} / {{ y }}

        - <span v-on:mousemove.stop="">DEAD SPOT</span>

        <!-- - <span v-on:mousemove="dummy">DEAD SPOT</span> -->

    </p>

    <input type="text" v-on:keyup.enter="alertMe">

</div>

<script src="app.js"></script>

App.js

new Vue({

    el: '#app',

    data: {

        counter: 0,

        x: 0,

        y: 0

    },

    methods: {

        increase: function(step, event) {

            this.counter += step;

        },

        updateCoordinates: function (event) {

            this.x = event.clientX;

            this.y = event.clientY;

        },

        dummy: function(event) {

            this.stopPropagation();

        },

        alertMe: function(event) {

            alert('Alert !');

        }

    }

})

Chapter 348 – Chapter 352

1. Computed (like or similar to function) only concern for the variable that computed, but method always aware with all the variable, even that variable is not processed in the function in method. Computed and method work synchronously, but watch works asynchronously.
2. Two Way Binding

Index.html

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

<div id="app">

    <input type="text" v-model="name">

    <p>{{ name }}</p>

</div>

<script src="app.js"></script>

App.js

new Vue({

    el: '#app',

    data: {

        name: 'Daiva'

    }

})

1. Computed Properties

Index.html

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

<div id="app">

    <!-- <button v-on:click="increase">Increase</button>

    <button v-on:click="decrease">Decrease</button> -->

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

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

    <button v-on:click="secondCounter++">Incrase Second</button>

    <p>Counter: {{ counter }} | {{ secondCounter }}</p>

    <!-- <p>Result: {{ result }}</p> -->

    <p>Result: {{ result() }} | {{ output }}</p>

</div>

<script src="app.js"></script>

App.js

new Vue({

    el: '#app',

    data: {

        counter: 0,

        // result: ''

        // It does not work

        // result: this.counter > 5 ? 'Greater 5' : 'Smaller 5'

        secondCounter: 0

    },

    computed: {

        output: function() {

            console.log('Computed');

            return this.counter > 5 ? 'Greater 5' : 'Smaller than 5';

        }

    },

    watch: {

        counter: function(value) {

            var vm = this;

            setTimeout(function() {

                vm.counter = 0;

            }, 2000);

        }

    },

    methods: {

        // increase: function() {

        //  this.counter++;

        //  this.result = this.counter > 5 ? 'Greater 5' : 'Smaller 5';

        // },

        // decrease: function() {

        //  this.counter--;

        //  this.result = this.counter > 5 ? 'Greater 5' : 'Smaller 5';

        // }

        result: function() {

            console.log('Method');

            return this.counter > 5 ? 'Greater 5' : 'Smaller 5';

        }

    }

})

Chapter 353 – Chapter 357

1. With CSS

Index.html

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

<div id="app">

    <div

        class="demo"

        @click="attachRed = !attachRed"

        :class="divClasses"></div>

    <div class="demo" :class="{red: attachRed}"></div>

    <div class="demo" :class="color"></div>

    <hr>

    <input type="text" v-model="color">

</div>

<script src="app.js"></script>

<style>

.demo {

    width: 100px;

    height: 100px;

    background-color: gray;

    display: inline-block;

    margin: 10px;

}

.red {

    background-color: red;

}

.green {

    background-color: green;

}

.blue {

    background-color: blue;

}

</style>

App.js

new Vue({

    el: '#app',

    data: {

        attachRed: false,

        color: 'green'

    },

    computed: {

        divClasses: function() {

            return {

                red: this.attachRed,

                blue: !this.attachRed

            };

        }

    }

})

1. Without CSS

Index.html

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

<div id="app">

    <div class="demo" :style="{backgroundColor: color}"></div>

    <div class="demo" :style="myStyle"></div>

    <div class="demo" :style="[myStyle, {height: width + 'px'}]"></div>

    <hr>

    <input type="text" v-model="color">

    <input type="text" v-model="width">

</div>

<script src="app.js"></script>

<style>

.demo {

    width: 100px;

    height: 100px;

    background-color: gray;

    display: inline-block;

    margin: 10px;

}

</style>

App.js

new Vue({

    el: '#app',

    data: {

        color: 'gray',

        width: 100

    },

    computed: {

        myStyle: function() {

            return {

                backgroundColor: this.color,

                width: this.width + 'px'

            }

        }

    }

})