**Section 11 – Forms**

Chapter 138 – V-Model And Inputs

1. Two-way binding (v-model), you can not just listen to what the user entered, but you can also overwrite what’s in the form, and you can change the value which is stored in the input.

Chapter 139 – Working With V-Model Modifiers And Numbers

1. Type: number.

Note: When you print in console.log, the printed value is blue it means type number, black means type string.

Note: v-model.number can convert string value into number value.

Chapter 140 – V-Model And Dropdowns

1. Dropdown.

this.referrer = 'wom';

wom is one of the dropdown value, if we fill the value with this, so the dropdown default will change into index ‘wom’.

Chapter 141 – Using V-Model With Checkboxes And Radio Buttons

1. Checkbox

You need to give value (used to uncheck one by one) and v-model (used to check one by one)

1. Radio button

Same with checkbox

Chapter 142 – Adding Basic Form Validation

1. Using @blur=”” on the textbox to validate the inputted value, validate before move and fill another textbox. (Live Validation)

<div class="form-control" :class="{invalid: userNameValidity === 'invalid'}">

      <label for="user-name">Your Name</label>

      <input id="user-name" name="user-name" type="text" v-model.trim="userName" @blur="validateInput"/>

      <p v-if="userNameValidity === 'invalid'">Please entera valid name !</p>

</div>

Chapter 143 – Building A Custom Control Component

1. How if we have 4 buttons in 1 form and only 1 submit button we will use ? Another button have another function. Add type=”button” to another button.

<li><button type="button">Poor</button></li>

Chapter 144 – Using V-Model On Custom Component

1. How to reset button class after submit ? Using props like below.

export default {

    props: ['modelValue'],

    emits: ['update:modelValue'],

    // data() {

    //     return {

    //         activeOption: this.modelValue

    //     };

    // },

    // computed: {

    //     activeOption() {

    //         return this.modelValue;

    //     }

    // },

    methods: {

        activate(option) {

            // this.activeOption = option;

            this.$emit('update:modelValue', option)

        }

    }

}

Tag HTML will be like below.

<li :class="{active: modelValue === 'poor'}">

<button type="button" @click="activate('poor')">Poor</button>

</li>

Chapter 145 – Module Summary

1. Making input type text, number, checkbox, and radio button.
2. Using props and emits.