**Lesson12 Angular Move validation messages to the component class in reactive form**

**Notes:-**

**1-The benefits of move validation to the component class**

**A-Easily unit test validation logic**

**B-Instead of hard-coding validation messages in the application, we can load them from an external source like a database or a file.**

**C-Change validation dynamically at run-time based on the decisions made in code or user selections**

import { Component, OnInit } from '@angular/core';

// Import FormGroup and FormControl classes

import { FormGroup, FormControl,FormBuilder, Validators } from '@angular/forms';

import { ReactiveFormsModule } from '@angular/forms';

@Component({

selector: 'app-employee-react-comp',

templateUrl: './employee-react-comp.component.html',

styleUrls: ['./employee-react-comp.component.css']})

export class EmployeeReactCompComponent implements OnInit {

empForm:FormGroup;

constructor(private fb:FormBuilder) { }

ngOnInit() {

//we apply the validation on the form group

this.empForm = this.fb.group({

//if we want to apply multiple validation, we use this below validation

FullName : ['',Validators.compose([Validators.required, Validators.minLength(2),Validators.maxLength(15)])],

Email : [''],

skills : this.fb.group({

skillName: ['C#'],

experienceInYears:[''],

proficiency:['beginner']

})})}

// This object will hold the messages to be displayed to the user

// Notice, each key in this object has the same name as the

// corresponding form control

formErrors = {

'FullName': '',

'Email': '',

'skillName': '',

'experienceInYears': '',

'proficiency': ''};

// This object contains all the validation messages for this form

validationMessages = {

'FullName': {

'required': 'Full Name is required.',

'minlength': 'Full Name must be greater than 2 characters.',

'maxlength': 'Full Name must be less than 10 characters.'},

'Email': {

'required': 'Email is required.'},

'skillName': {

'required': 'Skill Name is required.'},

'experienceInYears': {

'required': 'Experience is required.'},

'proficiency': {

'required': 'Proficiency is required.'}};

//we apply the validation on each formControl and set value for it

loadControls(frm:FormGroup){

Object.keys(frm.controls).forEach(key =>{

var abstractobj = frm.get(key);

if(abstractobj instanceof FormGroup){

this.loadControls(abstractobj);}

else{

//we reset the value of key validation for each control

this.formErrors[key] = '';

//if the element is not valid and is clicked and drop or write then delete

if(abstractobj && !abstractobj.valid && (abstractobj.touched || abstractobj.dirty)){

//we get the values based on key for the array variable

//example for FullName

//'required': 'Full Name is required.',

//'minlength': 'Full Name must be greater than 2 characters.',

//'maxlength': 'Full Name must be less than 10 characters.'

const messages = this.validationMessages[key];

//we make for looping for each control the validation has occur

//example if you click one character it will show the 1 validator

//'minlength': 'Full Name must be greater than 2 characters.',

for(const errorKey in abstractobj.errors){

if(errorKey){

//we store the validation inside the formErrors variable

this.formErrors[key] += messages[errorKey]+'';

}}}}})}

onLoadDataClick(): void {

debugger;

this.loadControls(this.empForm);

console.log(this.formErrors);}

onSubmit(): void {

console.log(this.empForm.value);

}}

**2-in the html page we don’t change anything**

<form class="form-horizontal" [formGroup]="empForm" (ngSubmit)="onSubmit()" >

<div class="panel panel-primary">

<div class="panel-heading"><h3 class="panel-title">Create Employee</h3>

</div>

<div class="panel-body">

<div class="form-group">

<label class="col-sm-2 control-label" for="fullName">Full Name</label>

<div class="col-sm-8">

//to fix the click and move to another control then we use the (blur)

<input id="fullName" type="text" class="form-control" formControlName="FullName" (blur)="loadControls()" />

//we see that we apply the validation on the formErrors.FullName has value it will show the error

<span style="color:red;" \*ngIf="formErrors.FullName" >

{{formErrors.FullName}}

</span>

</div></div>

<div class="form-group">

<label class="col-sm-2 control-label" for="email">Email</label>

<div class="col-sm-8">

//to fix the click and move to another control then we use the (blur)

<input id="email" type="text" class="form-control" formControlName="Email" (blur)="loadControls()"/>

//we see that we apply the validation on the formErrors.FullName has value it will show the error

<span style="color:red;" \*ngIf="formErrors.Email" >

{{formErrors.Email}}

</span>

</div>

</div>

<div formGroupName="skills">

<div class="form-group">

<label class="col-sm-2 control-label" for="skillName">Skill</label>

<div class="col-sm-4">

//to fix the click and move to another control then we use the (blur)

<input type="text" class="form-control" id="skillName" formControlName="skillName" (blur)="loadControls()">

//we see that we apply the validation on the formErrors.FullName has value it will show the error

<span style="color:red;" \*ngIf="formErrors.skillName" >

{{formErrors.skillName}}

</span>

</div>

<label class="col-sm-2 control-label" for="skillName">Experience In Years</label>

<div class="col-sm-4">

//to fix the click and move to another control then we use the (blur)

<input type="text" class="form-control" formControlName="experienceInYears" (blur)="loadControls()">

//we see that we apply the validation on the formErrors.FullName has value it will show the error

<span style="color:red;" \*ngIf="formErrors.experienceInYears" >

{{formErrors.experienceInYears}}

</span></div></div>

<div class="form-group">

<label class="col-md-2 control-label">Proficiency</label>

<div class="col-md-10">

<label class="radio-inline">

<input id="proficiency" type="radio" value="beginner" formControlName="proficiency">Beginner

</label>

<label class="radio-inline">

<input id="proficiency" type="radio" value="intermediate" formControlName="proficiency">Intermediate

</label>

<label class="radio-inline">

<input id="proficiency" type="radio" value="advanced" formControlName="proficiency" (blur)="loadControls()">Advanced

</label>

<span style="color:red;" \*ngIf="formErrors.proficiency" >

{{formErrors.proficiency}}

</span>

</div>

</div>

</div>

</div>

<div class="panel-footer">

<button class="btn btn-primary" type="submit">Save</button>

<button class="btn btn-primary" type="button" (click)="onLoadDataClick()">Load Data</button>

</div>

</div>

</form>

<table>

<tr>

<th style="padding: 10px">FormGroup</th>

<th style="padding: 10px">FormControl (fullName)</th>

</tr>

<tr>

<td style="padding: 10px">

Form Touched : {{empForm.touched}}<br />

Dirty : {{empForm.dirty}}<br />

valid : {{empForm.valid}}<br />

FormValues : {{empForm.value | json}}

</td>

<td style="padding: 10px">

Full Name Touched : {{empForm.controls.FullName.touched}}<br/>

Full Name Dirty : {{empForm.get('FullName').dirty}}<br/>

Full Name valid : {{empForm.get('FullName').valid}}<br/>

Full Name Value : {{empForm.get('FullName').value}}

</td>

</tr>

</table>