**Lesson09 Angular reactive forms validation**

**Notes:-**

**1-we apply the module validator to apply the validation**

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

|  |  |
| --- | --- |
| **Function** | **Description** |
| required | Validate that a field has a value. Used for required fields. For example, Name is required. |
| requiredTrue | Validate that the field value is true. This validator is commonly used on a required checkbox. For example, "I Agree to the terms" checkbox must be checked to submit the form. |
| email | Validate that the field value has a valid email pattern. For example, abc is not a valid email. |
| pattern | Validate that the field value matches the specified regex pattern. |
| min | Validate that the field value is greater than or equal to the provided number. For example, minimum age to vote is 18. |
| max | Validate that the field value is less than or equal to the provided number. For example, people over the age of 90 are not eligible for this insurance policy. |
| minLength | The number of characters in the field must be greater than or equal to the provided minimum length. For example, Full Name must be at least 3 characters. |
| maxLength | The number of characters in the field must be less than or equal to the provided maximum length. For example, Description cannot exceed 500 characters. |

**Example:-**

**In the following example, we see that we apply validation on the form**

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 see that we apply the validation on the Form Control

//(we see that we apply dependency injection on form builder)

this.empForm = this.fb.group({

FullName : ['',[Validators.required,Validators.minLength(2),Validators.maxLength(10)]],

Email : [''],

skills : this.fb.group({

skillName: [''],

experienceInYears:[''],

proficiency:['beginner']

})})}

onSubmit(): void {

debugger;

console.log(this.empForm.value);}

onLoadDataClick(): void {

debugger;

this.empForm.setValue({

FullName: 'Pragim Technologies',

Email: 'pragim@pragimtech.com',

skills: {

skillName: 'C#',

experienceInYears: 5,

proficiency: 'beginner'

}});}}

**2-In the html page we see the following validation with following notes**

**Touched: - means that the element is touched**

**Dirty: - means that the element is write first characters**

<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">

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

//If the element clicked and not write and click tab it will show errors

<span style="color:red;" class="help-block" \*ngIf="((empForm.get('FullName').touched || empForm.get('FullName').dirty) && empForm.get('FullName').errors)">

//If the element not write any thing it will show this erros

<span \*ngIf="empForm.get('FullName').errors.required">

Full Name is required</span>

//If the element write less than 2 and more than 10 it will show this error

<span \*ngIf="empForm.get('FullName').errors.minlength || empForm.get('FullName').errors.maxlength">

Full Name must be greater than 2 characters and less than 10 characters

</span>

</span>

</div></div>

<div class="form-group">

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

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

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

</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">

<input type="text" class="form-control" id="skillName" placeholder="Name" formControlName="skillName"></div>

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

<input type="text" placeholder="Experience in Years" class="form-control" formControlName="experienceInYears">

</div>

</div>

<div class="form-group">

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

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

<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">Advanced

</label>

</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>