**REACTIVE FORM**

**Question 1**

**Reactive Form - New Component and Name field inclusion**  
  
Similar to the hands on done for Template Driven Form, we will develop the same edit employee form using Reactive Forms. The form will be gradually built with increasing complexity.  
  
In this hands, we will create a new component and add the name field for employee.  
  
Follow instructions below:

* Create a new component named 'edit-emp-reactive'
* Include ReactiveFormsModule in app.module.ts
* Import FormControl in edit-emp-reactive component.ts.
* Include a new instance variable in component.ts. Refer code below

name = new FormControl('');

* Refer below code to include the form element and map it to the form control

Name: <input type="text" [formControl]="name">

* Set a default value for the field by changing the name property as specified below.

name = new FormControl('John');

**Sample Layout**

|  |
| --- |
| https://cognizant.e-box.co.in/uploads/Image/01fseangular/tdf2.png |

**Answer 1**

*All other files same as previous.*

**app.module.ts**

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

import { FormsModule, ReactiveFormsModule} from '@angular/forms'

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './app.component';

import { ViewEmpComponent } from './view-emp/view-emp.component';

import { EditEmpComponent } from './edit-emp/edit-emp.component';

import { QuantityIncrementComponent } from './quantity-increment/quantity-increment.component';

import { QuantitySelectorComponent } from './quantity-selector/quantity-selector.component';

import { EditEmpTemplateDrivenComponent } from './edit-emp-template-driven/edit-emp-template-driven.component';

import { EditEmpReactiveComponent } from './edit-emp-reactive/edit-emp-reactive.component';

@**NgModule**({

declarations: [

AppComponent,

ViewEmpComponent,

EditEmpComponent,

QuantityIncrementComponent,

QuantitySelectorComponent,

EditEmpTemplateDrivenComponent,

EditEmpReactiveComponent

],

imports: [

BrowserModule,

AppRoutingModule,

FormsModule,

ReactiveFormsModule

],

providers: [],

bootstrap: [AppComponent]

})

export class **AppModule** { }

**app.component.html**

<b>Angular Learning</b>&nbsp;&nbsp;&nbsp;<a *routerLink*="view">View Employee</a>

&nbsp;&nbsp;<a *routerLink*="edit">Edit Employee</a>

&nbsp;&nbsp;<a *routerLink*="edit-emp-temp">Edit Employee (Template Driven)</a>

&nbsp;&nbsp;<a *routerLink*="edit-emp-react">Edit Employee (Reactive)</a>

&nbsp;&nbsp;<a *routerLink*="quant-incr">Quantity Selector</a>

&nbsp;&nbsp;<a *routerLink*="quant-sel">Quantity Selector</a>

<hr>

<router-outlet></router-outlet>

**app-routing.module.ts**

import { NgModule } from '@angular/core';

import { Routes, RouterModule } from '@angular/router';

import { QuantityIncrementComponent } from './quantity-increment/quantity-increment.component';

import { QuantitySelectorComponent } from './quantity-selector/quantity-selector.component';

import { EditEmpComponent } from './edit-emp/edit-emp.component';

import { ViewEmpComponent } from './view-emp/view-emp.component';

import { EditEmpTemplateDrivenComponent } from './edit-emp-template-driven/edit-emp-template-driven.component';

import { EditEmpReactiveComponent } from './edit-emp-reactive/edit-emp-reactive.component';

const routes: **Routes** = [

{

path:'view',

component:ViewEmpComponent

},

{

path:'edit',

component:EditEmpComponent

},

{

path:'quant-incr',

component:QuantityIncrementComponent

},

{

path:'quant-sel',

component:QuantitySelectorComponent

},

{

path:'edit-emp-temp',

component:EditEmpTemplateDrivenComponent

},

{

path:'edit-emp-react',

component:EditEmpReactiveComponent

}

];

@**NgModule**({

imports: [RouterModule.**forRoot**(routes)],

exports: [RouterModule]

})

export class **AppRoutingModule** { }

**edit-emp-reactive.component.ts**

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

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

@**Component**({

selector: 'app-edit-emp-reactive',

templateUrl: './edit-emp-reactive.component.html',

styleUrls: ['./edit-emp-reactive.component.css']

})

export class **EditEmpReactiveComponent** implements **OnInit** {

reactiveForm: **FormGroup**;

constructor() {

}

**ngOnInit**() {

this.**initForm**();

}

private **initForm**() {

this.reactiveForm = new **FormGroup**({

'empName': new **FormControl**("John")

});

}

}

**edit-emp-reactive.component.html**

<h1> Edit Employee </h1>

<form *[formGroup]*="reactiveForm" *(ngSubmit)*="onSubmit()">

<label *for*="Name">Name : </label>

<br/>

<input *type*="text" *id*="empName" *class*="form-control" *formControlName*="empName">

</form>

**Question 2**

**Reactive Form - Include form group and validation for name field**  
  
Instructions for getting employee data from instance variable and include validations:

* Import Employee, Skill and Department into component.ts
* Copy and paste the Employee test data from template driven form instance variable
* Set the data for form control from employee property. Refer code below.

name = new FormControl(this.employee.name);

* Include FormGroup and Validators in import

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

* Define form group property in component.ts as specified below:

this.empForm = new FormGroup({

'name': new FormControl(this.employee.name, [

Validators.required,

Validators.minLength(4),

Validators.maxLength(20)

])

});

* Include getter method for name. This is for the validator to retrieve the value in the form field.

get name() { return this.empForm.get('name'); }

* Modify template with inclusion of FormGroup and use the template driven form validation code.

**Answer 2**

**edit-emp-reactive.component.ts**

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

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

import { Department } from '../Department';

import { Employee } from '../Employee';

@**Component**({

selector: 'app-edit-emp-reactive',

templateUrl: './edit-emp-reactive.component.html',

styleUrls: ['./edit-emp-reactive.component.css']

})

export class **EditEmpReactiveComponent** implements **OnInit** {

reactiveForm: **FormGroup**;

constructor() {

}

 departments: **Department**[]=[

    {id:1,name:"HelpDesk"},

    {id:2,name:"HR"},

    {id:3,name:"IT"},

    {id:4,name:"Payroll"},

    {id:5,name:"admin"}

  ];

  emp:**Employee**={

    id:null,

    name:null,

    salary:null,

    permanent:null,

    department:{

      id:1,

      name:'Payroll'

    },

    skills:[

      {

        id:1,

        name:'HTML'

      },

      {

        id:2,

        name:'CSS'

      },

      {

        id:3,

        name:'Javascript'

      }

    ],

    dateofBirth: new **Date**('04/20/2019')

}

**ngOnInit**() {

this.**initForm**();

}

get **empName**() {

return this.reactiveForm.**get**('empName');

}

private **initForm**() {

this.reactiveForm = new **FormGroup**({

'empName': new **FormControl**(this.emp.name,[

Validators.required,

Validators.**minLength**(4),

Validators.**maxLength**(20)

])

});

}

}

**edit-emp-reactive.component.html**

<h1> Edit Employee </h1>

<form *[formGroup]*="reactiveForm" *(ngSubmit)*="onSubmit()">

<label *for*="Name">Name : </label>

<br/>

<input *type*="text" *id*="empname" *class*="form-control" *formControlName*="empName">

<span *class*="help-block" *\*ngIf*="empName.errors?.required && empName.touched">

Name is Required

</span>

<span *class*="help-block" *\*ngIf*="empName.errors?.minlength">

Name Must be at least 2 characters long

</span>

<span *class*="help-block" *\*ngIf*="empName.errors?.maxlength">

Name can have maximum of 10 characters

</span>

</form>

**Question 3**

**Reactive Form - Include salary, permanent and department fields**  
  
Implement rest of the form fields for edit employee form. Refer information below

* Utilize the HTML template code implemented in template driven form including display of employee JSON and form JSON
* Define department as form group within form group
* Include additional getter methods for each field
* Implement onSubmit() method to print the form details in console by using the value property of form group

**Answer 3**

**edit-emp-reactive.component.ts**

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

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

import { Department } from '../Department';

import { Employee } from '../Employee';

@**Component**({

selector: 'app-edit-emp-reactive',

templateUrl: './edit-emp-reactive.component.html',

styleUrls: ['./edit-emp-reactive.component.css']

})

export class **EditEmpReactiveComponent** implements **OnInit** {

reactiveForm: **FormGroup**;

constructor() {

}

 departments: **Department**[]=[

    {id:1,name:"HelpDesk"},

    {id:2,name:"HR"},

    {id:3,name:"IT"},

    {id:4,name:"Payroll"},

    {id:5,name:"admin"}

  ];

  emp:**Employee**={

    id:null,

    name:null,

    salary:null,

    permanent:null,

    department:{

      id:1,

      name:'Payroll'

    },

    skills:[

      {

        id:1,

        name:'HTML'

      },

      {

        id:2,

        name:'CSS'

      },

      {

        id:3,

        name:'Javascript'

      }

    ],

    dateofBirth: new **Date**('04/20/2019')

}

**ngOnInit**() {

this.**initForm**();

}

get **empName**() {

return this.reactiveForm.**get**('empName');

}

**onSubmit**() {

**console**.**log**("Name is : " + this.reactiveForm.value.empName); **console**.**log**("Salary is : " + this.reactiveForm.value.empSalary); **console**.**log**("Permananent or Not : " + this.reactiveForm.value.empPermananent); **console**.**log**("Department is : " + this.reactiveForm.value.empDept);

}

private **initForm**() {

this.reactiveForm = new **FormGroup**({

'empName': new **FormControl**(this.emp.name,[

Validators.required,

Validators.**minLength**(4),

Validators.**maxLength**(20)

]),

'empSalary':new **FormControl**(this.emp.salary),

'empPermananent':new **FormControl**(this.emp.permanent),

'empDept':new **FormControl**(this.emp.department)

}

);

}

}

**edit-emp-reactive.component.html**

<div *style*="float: left;width: 33%;">

<h1> Edit Employee </h1>

<form *[formGroup]*="reactiveForm" *(ngSubmit)*="onSubmit()">

<div *class*="form-group">

<label *for*="Name">Name : </label>

<br/>

<input *type*="text" *id*="empname" *formControlName*="empName">

<span *class*="help-block" *\*ngIf*="empName.errors?.required && empName.touched">

Name is Required

</span>

<span *class*="help-block" *\*ngIf*="empName.errors?.minlength">

Name Must be at least 2 characters long

</span>

<span *class*="help-block" *\*ngIf*="empName.errors?.maxlength">

Name can have maximum of 10 characters

</span>

</div>

<div *class*="form-group">

<label *for*="salary">Salary</label>

<br/>

<input *type*="number" *name*="salary" *id*="salary" *formControlName*="empSalary"/>

</div>

<div *class*="form-group">

<div>

<label *for*="control-label">Is Permanent or not ?</label>

<br/>

<label *class*="radio-inline">

<input *type*="radio" *name*="empPermananent" *id*="permanent" *value*="true" *formControlName*="empPermananent" /> Yes

</label>

<label *class*="radio-inline">

<input *type*="radio" *name*="empPermananent" *id*="permanent" *val*="false" *formControlName*="empPermananent" /> No

</label>

</div>

</div>

<div *class*="form-group">

<label *for*="department">Department : </label>

<br/>

<select *id*="department" *name*="empDept" *class*="form-group" *formControlName*="empDept" >

<option *\*ngFor*="let dept of departments" *[value]*="dept.id">

{{dept.name}}

</option>

</select>

</div>

<br/>

<div *style*="margin-left:150px;margin-bottom: 20px;">

<button *type*="submit" *class*="btn" *[disabled]*="!reactiveForm.valid">Save</button>

</div>

</form>

</div>

<div *style*="float:left;width: 33%" >

<strong>Employee Model</strong>

<br/>

<pre>{{emp|json}}</pre>

</div>

<div *style*="float: left;width:33%">

<strong>Angular Genrated Form Model</strong>

<br/>

<pre>{{reactiveForm.value|json}}</pre>

</div>

----------------------------

**Question 4**

**Reactive Form - Using FormBuilder**  
  
Implement employee edit using FormBuilder. Refer instructions below:

* Create a new component named 'edit-emp-form-builder
* Copy existing html and ts code from reactive form
* Modify FormControl defintion to use FormBuilder
* Inject form builder in constructor

**Answer 4**

**Question 5**

**Reactive Form - Dynamic Forms**  
  
Implement adding multiple skills in edit employee form. This should dynamically add new textboxes for each skill.  
  
Refer live example [here](https://stackblitz.com/angular/olrgkrandqm) and implement this functionality.

**Answer 5**