

## Angular 2 Routing Lab

In this exercise, we will implement the routing feature in Angular 2. We will be implementing this lab using Visual Studio Code (VSCode).

**Step 1:** Create a Workspace (ws) folder and open it in the VSCode, using Open Folder menu item.

**Step 2:** In the workspace folder add a new folder of name **app**.

**Step 3:** In the Workspace folder add a new file of name package.json and define the following package in it.

```
{
  "name": "ng2-routing",
  "version": "1.0.0",
  "scripts": {
    "start": "concurrently \"tsc -w\" \"node appserver.js\"",
    "tsc": "tsc",
    "tsc:w": "tsc -w",
    "lite": "lite-server",
    "typings": "typings",
    "postinstall": "typings install"
  },
  "license": "ISC",
  "dependencies": {
    "@angular/common": "2.0.0",
    "@angular/compiler": "2.0.0",
    "@angular/core": "2.0.0",
    "@angular/forms": "2.0.2",
    "@angular/http": "2.0.0",
    "@angular/platform-browser": "2.0.0",
    "@angular/platform-browser-dynamic": "2.0.0",
    "@angular/router": "3.0.2",
    "bootstrap": "3.3.7",
    "es6-shim": "0.35.1",
    "koa": "1.2.4",
    "koa-static": "2.0.0",
    "livereload": "0.5.0",
    "reflect-metadata": "0.1.8",
    "rxjs": "5.0.0-beta.12",
    "systemjs": "0.19.39",
    "zone.js": "0.6.25"
  },
  "devDependencies": {
    "concurrently": "3.1.0",
    "lite-server": "2.2.2",
    "node-gyp": "3.4.0",
    "typescript": "2.0.3",
    "typings": "1.4.0"
  }
}
```

**Step 4:** In the folder add a new html file of name index.html. Right-Click on this html and select open in command prompt. This will open command prompt. Run the following commands from the command prompt

```
Npm install -g typescript
```

```
Npm install -g typings
```

```
Npm install -concurrently
```

```
Npm install
```

```
typings install
```

**Step 6:** In the ws folder add a new file of name appserver.js, this will contains code for creating server hosting using koa

```
var koa = require("koa");
var serve = require("koa-static");
var livereload = require("livereload");

var app = koa();
var server = livereload.createServer();

server.watch(__dirname + "/app/*.js");

app.use(serve("."));

app.listen(5000);
```

**Step 7:** To define system configuration, add a new file of name systemjs.config.js in ws folder and register following packages in it

```
var map = {
  "rxjs": "node_modules/rxjs",
  "@angular/common": "node_modules/@angular/common",
  "@angular/forms": "node_modules/@angular/forms",
  "@angular/compiler": "node_modules/@angular/compiler",
  "@angular/core": "node_modules/@angular/core",
  "@angular/platform-browser": "node_modules/@angular/platform-browser",
  "@angular/platform-browser-dynamic": "node_modules/@angular/platform-browser-dynamic",
  "@angular/router": "node_modules/@angular/router"
};
var packages = {
  "rxjs": { "defaultExtension": "js" },
  "@angular/common": { "main": "bundles/common.umd.js", "defaultExtension": "js" },
  "@angular/forms": { "main": "bundles/forms.umd.js", "defaultExtension": "js" },
  "@angular/compiler": { "main": "bundles/compiler.umd.js", "defaultExtension": "js" },
  "@angular/core": { "main": "bundles/core.umd.js", "defaultExtension": "js" },
```

```

        "@angular/platform-browser": { "main": "bundles/platform-browser.umd.js",
"defaultExtension": "js" },
        "@angular/platform-browser-dynamic": { "main": "bundles/platform-browser-
dynamic.umd.js", "defaultExtension": "js" },
        "@angular/router": { "main": "bundles/router.umd.js", "defaultExtension": "js" },
        "app": {
            format: 'register',
            defaultExtension: 'js'
        }
    };

    var config = {
        map: map,
        packages: packages
    };

    System.config(config);

```

**Step 8:** Add a tsconfig.json file in the ws folder (the tsc --init command can also be used to add this file from the command prompt) the file should contain following definitions

```

{
  "compilerOptions": {
    "target": "es5",
    "module": "system",
    "moduleResolution": "node",
    "sourceMap": true,
    "emitDecoratorMetadata": true,
    "experimentalDecorators": true,
    "removeComments": false,
    "noImplicitAny": false
  },
  "exclude": [
    "node_modules",
    "typings/main",
    "typings/main.d.ts"
  ]
}

```

**Step 9:** In the app folder add a new file of name emp.mode.ts. This will contain model class as shown below

```

export class Employee{
  constructor(
    public empNo:number,
    public empName:string,
    public deptName:string

```

```

    })
  }

  export const Employees=[
    {empNo:101,empName:'MS',deptName:'IT'},
    {empNo:102,empName:'LS',deptName:'HR'},
    {empNo:103,empName:'TS',deptName:'SL'},
    {empNo:104,empName:'VB',deptName:'IT'},
    {empNo:105,empName:'PB',deptName:'HR'},
    {empNo:106,empName:'AB',deptName:'SL'}
  ];

```

**Step 10:** In the app folder add following component files for the business logic

app.employees.component.ts

```

import { Component, OnInit } from '@angular/core';
import { Employee, Employees } from './emp.model';
import { Router } from "@angular/router";
@Component({
  selector: 'emp-list',
  templateUrl: './app/listemployees.html'
})
export class ListEmployeeComponent implements OnInit {
  employees = Employees;
  constructor() { }

  ngOnInit() { }
}

```

This class expose the Employees array to the template URL

App.newemployee.component.ts

```

import { Component, OnInit } from '@angular/core';
import { Employee, Employees } from './emp.model';
import { Router } from "@angular/router";
@Component({
  selector: 'new-emp',
  templateUrl: './app/newemployee.html'
})
export class AddEmployeeComponent implements OnInit {
  employee: Employee;
  employees=Employees;
  constructor(private router: Router) {
    this.employee = new Employee(0, "", "");
  }

  ngOnInit() { }
}

```

```

clear(){
this.employee = new Employee(0,"");

}
save(){
  this.employees.push(this.employee);
  this.router.navigate(['']);
}
}

```

This class expose Employee object so that it can be pushed into an employees array. The save() function push the employee data and navigate to the route expression using route.

App.editemployee.component.ts

```

import { Component, OnInit } from '@angular/core';
import { Employee, Employees } from './emp.model';
import { Router, ActivatedRoute } from "@angular/router";
@Component({
  selector: 'edit-emp',
  templateUrl: './app/editemployee.html'
})
export class EditEmployeeComponent implements OnInit {
  employee:Employee;
  employees=Employees;
  empNo:number;
  sub:any;
  constructor(private router: ActivatedRoute,private route:Router) {
    this.employee = new Employee(0,"");
  }

  ngOnInit() {
    this.sub = this.router.params.subscribe(params => {
      this.empNo = +params['id'];
      for(let e of this.employees){
        if(e.empNo==this.empNo){
          this.employee.empNo = this.empNo;
          this.employee.empName =e.empName;
          this.employee.deptName = e.deptName;
        }
      }
    });
  }

  clear(){
    this.employee = new Employee(0,"");
  }
}

```

```

    }
    save(){
      this.employees.push(this.employee);
      this.route.navigate(['']);
    }
  }
}

```

The above class is used to edit the employee.

App.main.component.ts

```

import { Component, OnInit } from '@angular/core';
import { ListEmployeeComponent } from './app.employees.component';
@Component({
  selector: 'main-page',
  template: `<nav class="navbar navbar-default">
    <div class="container-fluid">
      <div class="navbar-header">
        <span class="navbar-brand">DEMO Routing</span>
      </div>
      <div class="collapse navbar-collapse">
        <ul class="nav navbar-nav">
          <li><a class="btn btn-link"
[routerLink]="['']">Employee List</a></li>
          <li><a class="btn btn-link"
[routerLink]="['addEmployee']">Add Employee</a></li>
        </ul>
      </div>
    </div>
  </nav>
<router-outlet></router-outlet>`
})
export class MainComponent {
}

```

This class defines routeLink based on the route table. The <router-outlet></router-outlet> is a platform where views will be injected.

**Step 11:** In the app folder add a new file for defining routes, name this file as app.router.ts

```

import { ModuleWithProviders } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { ListEmployeeComponent } from './app.employees.component';
import { AddEmployeeComponent } from './app.newemployee.component';
import { EditEmployeeComponent } from './app.editemployee.component';

```

```

let appRoutes : Routes=[
  {path:"",component:ListEmployeeComponent},
  {path:"addEmployee",component:AddEmployeeComponent},
  {path:"editEmployee/:id",component:EditEmployeeComponent}
];

export const routing: ModuleWithProviders
  = RouterModule.forRoot(appRoutes);

```

This defines the route array with path and component.

**Step 12:** Add boot.ts in the app folder for bootstrapping

```

import { NgModule } from '@angular/core';
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
import { BrowserModule } from '@angular/platform-browser';
import { ListEmployeeComponent } from './app.employees.component';
import { AddEmployeeComponent } from './app.newemployee.component';
import { EditEmployeeComponent } from './app.editemployee.component';
import { MainComponent } from './app.main.component';
import { routing } from './app.router';
import { FormsModule } from '@angular/forms';
@NgModule({
  imports: [BrowserModule,FormsModule,routing],
  declarations: [MainComponent,ListEmployeeComponent,
    AddEmployeeComponent,EditEmployeeComponent],
  bootstrap:[MainComponent]
})
export class AppModule { }

platformBrowserDynamic().bootstrapModule(AppModule);

```

This loads all components for the application

**Step 13:** In app folder add following HTML

Listemployee.html

```

<table class="table table-striped table-bordered">
  <thead>
    <tr>
      <td>EmpNo</td>
      <td>EmpName</td>
      <td>DeptName</td>
      <td></td>
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let emp of employees"

```

```

    >
    <td>{{emp.empNo}}</td>
    <td>{{emp.empName}}</td>
    <td>{{emp.deptName}}</td>
    <td>
        <a [routerLink]="['/editEmployee', emp.empNo]">{{emp.empNo}}</a>
    </td>
</tr>
</tbody>
</table>

```

This contains route ling to edit employee.

Newemployee.html

```

<table class="table table-stripped table-bordered">
  <tr>
    <td>EmpNo</td>
    <td>
      <input type="text" [(ngModel)]= "employee.empNo"
      class="form-control">
    </td>
  </tr>
  <tr>
    <td>EmpName</td>
    <td>
      <input type="text"
      class="form-control" [(ngModel)]= "employee.empName">
    </td>
  </tr>
  <tr>
    <td>DeptName</td>
    <td>
      <input type="text"
      class="form-control" [(ngModel)]= "employee.deptName">
    </td>
  </tr>
  <tr>
    <td>
      <input type="button" value="clear" (click)="clear()"
      class="btn btn-default">
    </td>
    <td>
      <input type="button" (click)="save()" value="save"
      class="btn btn-success">
    </td>
  </tr>
</table>

```



Editemployee.html

```
<table class="table table-stripped table-bordered">
  <tr>
    <td>EmpNo</td>
    <td>
      <input type="text" [(ngModel)]="employee.empNo"
        class="form-control">
    </td>
  </tr>
  <tr>
    <td>EmpName</td>
    <td>
      <input type="text"
        class="form-control" [(ngModel)]="employee.empName">
    </td>
  </tr>
  <tr>
    <td>DeptName</td>
    <td>
      <input type="text"
        class="form-control" [(ngModel)]="employee.deptName">
    </td>
  </tr>
  <tr>
    <td>
      <input type="button" value="clear" (click)="clear()"
        class="btn btn-default">
    </td>
    <td>
      <input type="button" (click)="save()" value="save"
        class="btn btn-success">
    </td>
  </tr>
</table>
```

**Step 14:** Modify the index.html by adding following markup

```
<!DOCTYPE html>
<html>

<head>
  <base href="/" />
  <title>Angular 2 Two Binding</title>
  <script>
    document.write('<script src="http://' + (location.host || 'localhost').split(':')[0] +
      ':35729/livereload.js?snipver=1"></' + 'script>')
  </script>
```

```

<link rel="stylesheet" href="node_modules/bootstrap/dist/css/bootstrap.min.css"></lin
</head>

<body>

  <h1>Angular 2 Routing</h1>
<main-page></main-page>

  <script src="node_modules/es6-shim/es6-shim.min.js"></script>
  <script src="node_modules/reflect-metadata/Reflect.js"></script>
  <script src="node_modules/systemjs/dist/system.src.js"></script>
  <script src="node_modules/zone.js/dist/zone.js"></script>
  <script src="systemjs.config.js"></script>
  <script>
    System.import('./app/boot')
      .then(null, console.error.bind(console));
  </script>
</body>

</html>

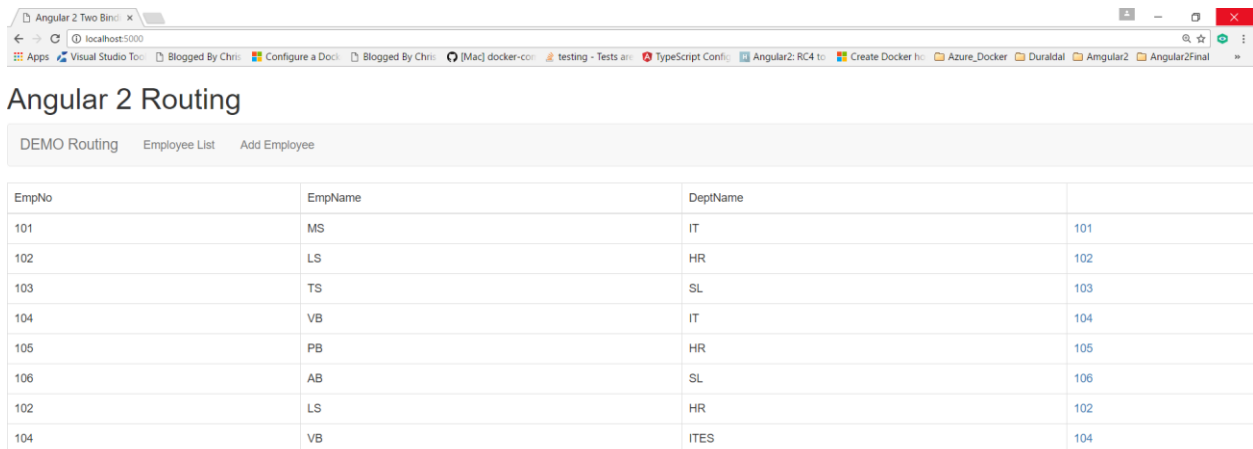
```

Note that the file must contain base url in the header section. This will load the default view.

From the command prompt run the following command

Npm run start

The following result will be displayed



EmpNo	EmpName	DeptName	
101	MS	IT	101
102	LS	HR	102
103	TS	SL	103
104	VB	IT	104
105	PB	HR	105
106	AB	SL	106
102	LS	HR	102
104	VB	ITES	104

Click on Add Employee, the add employee view will be displayed. Similarly when the link for the EmpNo is clicked the edit employee view will be displayed.

