INF 354 Notes

Angular

Contents

[1. Angular CLI 2](#_Toc133946226)

[2. App Component structure 2](#_Toc133946227)

[Component 2](#_Toc133946228)

[Template 3](#_Toc133946229)

[3. Services 4](#_Toc133946230)

[4. Routing and Navigation with pages 7](#_Toc133946231)

# Angular CLI

npm install -g @angular/cli – install angular

ng new my-app – create my app

ng build – when ready to build

cd my-app – cd into directory

code . – open VS code

ng serve – open on localhost – will open angular landing page

<https://material.angular.io/> - design components for angular

# App Component structure

Interpolation binding = {{ }} - binds the data one way, which means that data moves in one direction from the components to HTML elements

Property binding = [ ] - one-way data binding mechanism that allows you to set the properties for HTML elements

Event binding = ( ) - when information flows from the view to the component when an event is triggered. The event could be a mouse click or keypress

Two-way data binding = [( )] - data flows from the component to the view and back. This binding ensures that the component and view are always in sync

ng generate component components/header – create files for header

ngOnInit() – life cycle method what we use to initialize some code eg. Http request

## Component

hero-details.component.ts:

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

import { ActivatedRoute } from '@angular/router';

import { HeroService } from 'src/app/services/hero.service';

import { Hero } from 'src/app/shared/hero';

@Component({

selector: 'app-hero-details',

templateUrl: './hero-details.component.html',

styleUrls: ['./hero-details.component.scss']

})

export class HeroDetailsComponent implements OnInit {

hero:any

constructor(private heroService: HeroService, private route: ActivatedRoute) { }

ngOnInit(): void {

this.heroService.getHero(+this.route.snapshot.params['id']).subscribe((hero: any) => {this.hero = hero}) // This line retrieves the hero object from the server using the HeroService and the route parameter id obtained from the ActivatedRoute service. The + sign is used to convert the id parameter from a string to a number. The subscribe method is used to handle the response from the server. When the hero object is received, it is assigned to the hero property of the component.

}

}

## Template

hero-details.html:

<div class="flex-container" fxLayout.xs="column">

<div class="flex-item">Age: {{hero?.age}}</div>

<div class="flex-item">Birthday: {{hero?.birthday}}</div>

<div class="flex-item"></div>

<div class="flex-item">Height: {{hero?.height}}</div>

<div class="flex-item">Alive: {{hero?.alive}}</div>

<div class="flex-item"></div>

</div>

<button mat-button [routerLink]="['/heroes']">

<mat-icon>home</mat-icon>

return

</button>

# Services

hero.service.ts:

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

import { Observable, of} from 'rxjs';

import { Hero } from '../shared/hero';

@Injectable({

providedIn: 'root'

})

export class HeroService {

constructor() {

if(!localStorage.getItem('heroes')) {

let heroes = [{

"id": 1,

"name": "Tony Stark (Iron Man)",

"age": 53,

"birthday": "May 29",

"height": "185cm",

"image": "assets/images/tony-stark-iron-man.webp",

"alive": false,

},{cont…}

]

localStorage.setItem('heroes', JSON.stringify(heroes))

}

}

getHeroes(): Observable<any[]> {

let heroes:any[]=[]

if (localStorage.getItem('heroes'))

{

heroes = JSON.parse(localStorage.getItem('heroes')!)

}

return of(heroes)

}

getHero(id:number): Observable<any>

{

let heroes:Hero[] = [];

if (localStorage.getItem('heroes'))

{

heroes = JSON.parse(localStorage.getItem('heroes')!)

}

let hero:any = heroes.find(hero => hero.id === id)

return of(hero)

}

async deleteHero(id: any){

let heroes:Hero[] = []

if (localStorage.getItem('heroes'))

{

heroes = JSON.parse(localStorage.getItem('heroes')!)

}

let hero = heroes.find(hero => hero.id === id)

if (hero)

{

let index = heroes.indexOf(hero)

heroes.splice(index, 1)

await localStorage.setItem('heroes', JSON.stringify(heroes))

}

}

}

# Routing and Navigation with pages

app-routing.module.ts: // where you set up and configure your application routing

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

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

import { DashboardComponent } from './dashboard/dashboard.component';

import { HeroDetailsComponent } from './heroes/hero-details/hero-details.component';

import { HeroesComponent } from './heroes/heroes.component';

const routes: Routes = [

{path: 'dashboard', component:DashboardComponent},

{path: 'heroes', component:HeroesComponent},

{path: 'hero/:id', component:HeroDetailsComponent },

{path: '', redirectTo: '/dashboard', pathMatch:'full'}

];

@NgModule({

imports: [RouterModule.forRoot(routes)],

exports: [RouterModule]

})

export class AppRoutingModule { }

app.component.html:

<mat-sidenav #sidenav mode="side" opened="true" class="side-container">

<mat-nav-list>

<a mat-list-item [routerLink]="'/dashboard'"> Dashboard </a>

<a mat-list-item [routerLink]="'/heroes'"> Heroes </a>

</mat-nav-list>

</mat-sidenav>

Another example:

hero-details.component.html:

<td mat-cell \*matCellDef="let element"> <button mat-button [routerLink]="['/hero', element.id]">

hero-details.component.ts:

export class HeroDetailsComponent implements OnInit {

hero:any

constructor(private heroService: HeroService, private route: ActivatedRoute) { }

ngOnInit(): void {

this.heroService.getHero(+this.route.snapshot.params['id']).subscribe((hero: any) => {this.hero = hero})

}

}