deti universidade de aveiro
departamento de electrónica,
telecomunicações e informática

Tecnologias e Programação Web
2019/2020

Angular Framework

Angular Framework

Routing in Angular

Angular – Routing (i)

- The browser is a familiar model of application navigation:
  - Enter a URL in the address bar and the browser navigates to a corresponding page;
  - Click links on the page and the browser navigates to a new page;
  - Click the browser's back and forward buttons and the browser navigates backward and forward through the history of pages you've seen.

Angular – Routing (ii)

- The Angular Router is based on this model:
  - It can interpret a browser URL as an instruction to navigate to a client-generated view.
  - It can pass optional parameters along to the supporting view component that help it decide what specific content to present.
  - You can bind the router to links on a page and it will navigate to the appropriate application view when the user clicks a link.

Angular – Routing (iii)

- The Angular Router is based on this model:
  - You can navigate imperatively when the user clicks a button, selects from a drop box, or in response to some other stimulus from any source.
  - And the router logs activity in the browser's history journal so the back and forward buttons work as well.

Routing – Developing (i)

- An Angular best practice is to load and configure the router in a separate, top-level module that is dedicated to routing and imported by the root AppModule.

- By convention, the module class name is AppRoutingModule and it resides in the app-routing.module.ts file, in the src/app folder.

Routing – Developing (ii)

- Let’s create the AppRoutingModule, using the following command line, in project folder:

```
ng generate module app-routing --flat --module=app
```

  - --flat: puts the file in src/app instead of its own folder.
  - --module=app: tells the CLI to register it in the imports array of the AppModule.

Routing – Developing (iii)

- Router Imports
  - The Angular Router is an optional service that presents a particular component view for a given URL. So, it must be imported correctly.
  - Modify “app-routing.module.ts” as following:

app-routing.module.ts

```
1  import { NgModule } from '@angular/core';
2  import { RouterModule, Routes} from '@angular/router';
3
```

Routing – Developing (iv)

- Router Configuration
  - Export RouterModule. It makes router directives available for use in the AppModule components that will need them.

app-routing.module.ts

```
1  import { NgModule } from '@angular/core';
2  import { RouterModule, Routes} from '@angular/router';
3
4  @NgModule({
5    exports: [
6      RouterModule
7    ]
8  })
9  export class AppRoutingModule { }
```

Routing – Developing (v)

- Adding Routes
  - Routes tell the router which view to display when a user clicks a link or pastes a URL into the browser address bar.
  - A typical Angular Route has two properties:
    - path: a string that matches the URL in the browser address bar.
    - component: the component that the router should create when navigating to this route.

Routing – Developing (vi)

- Adding Routes (cont.)
  - We want to navigate to the AuthorsComponent when the URL is something like localhost:4200/authors.
  - Import the AuthorsComponent so you can reference it in a Route. Then define an array of routes with a single route to that component.

app-routing.module.ts

```
1  import { NgModule } from '@angular/core';
2  import { RouterModule, Routes} from '@angular/router';
3
4  import { AuthorsComponent} from './authors/authors.component';
5
6  const routes: Routes = [
7    {path: 'authors', component: AuthorsComponent}
8  ];
```

Routing – Developing (vii)

- Adding Routes (cont.)
  - Then, the router must be initialized in order to start listening for browser location changes.
  - Add RouterModule to @NgModule.imports array, like this:

app-routing.module.ts

```
10  @NgModule({
11    exports: [
12      RouterModule
13    ],
14    imports: [
15      RouterModule.forRoot(routes)
16    ]
17  })
18  export class AppRoutingModule { }
```

Routing – Developing (viii)

- Adding Routes (cont.)
  - If need, add AppRoutingModule to “app.module.ts”, like this:

app.module.ts

```
1   import { BrowserModule } from '@angular/platform-browser';
2   import { NgModule } from '@angular/core';
3   import { FormsModule} from '@angular/forms';
4
5   import { AppComponent } from './app.component';
6   import { AppRoutingModule } from './/app-routing.module';
7   import { AuthorsComponent } from './authors/authors.component';
8
9   @NgModule({
10    declarations: [
11      AppComponent,
12      AuthorsComponent,
13    ],
14    imports: [
15      BrowserModule,
16      FormsModule,
17      AppRoutingModule
18    ],
```

Routing – Developing (ix)

- Adding Routes (cont.)
  - The RouterOutlet is one of the router directives that became available to the AppComponent when RouterModule is exported.
  - Modify “src/app/app.component.html” like this, deleting <app-authors>:

app.component.html

```
1  <h1>{{ title }}</h1>
2  <router-outlet></router-outlet>
3
```

  - Now, you can already try the URL “http://localhost:4200/authors”

Routing – Developing (x)

- Adding Routes (cont.)
  - You can also add a navigation link to navigate from home to authors:

app.component.html

```
1  <h1>{{ title }}</h1>
2
3  <nav>
4    <a routerLink="/authors">Authors</a>
5  </nav>
6
7  <router-outlet></router-outlet>
```

  - Using routerLink attribute is a better way than href attribute. Try to use both to see the differences.

Angular Framework

Multiple Views in Angular

Multi-Views

- Adding multiple views:
  - Adding an Overview panel.
  - Adding the ability to navigate between Authors and Overview views.
  - When users click an author in either view, it navigates to a details view of the selected author.
  - When users click a deep link in an email, it opens the details view for a particular author.

Multi-Views – Overview view (i)

- Create Overview component:
  - Command line: ng generate component overview
- In “src/app/overview.component.html” put the following html code:

overview.component.html

```
1  <h3>Some Authors</h3>
2  <div class="grid grid-pad">
3    <a *ngFor="let author of authors" class="col-1-4">
4      <div class="module author">
5        <h4>{{author.name}}</h4>
6      </div>
7    </a>
8  </div>
```

Multi-Views – Overview view (ii)

- Code “overview.component.ts” in the same way as “authors.component.ts”, with light difference:

overview.component.ts

```
1  import { Component, OnInit } from '@angular/core';
2  import { Author } from '../author';
3  import {AUTHORS} from '../authorslist';
4
5
6  @Component({
7    selector: 'app-overview',
8    templateUrl: './overview.component.html',
9    styleUrls: ['./overview.component.css']
10 })
11 export class OverviewComponent implements OnInit {
12   authors: Author[];
13
14   constructor() {
15     this.authors = AUTHORS.slice(0, 4);
16   }
17
18   ngOnInit() {
19   }
```

Multi-Views – Overview view (iii)

- Substitute the Overview Component CSS file with the one given in moodle “overview.component.css”.

overview.component.css

```
1
2   [class*='col-'] {
3     float: left;
4     padding-right: 20px;
5     padding-bottom: 20px;
6   }
7   [class*='col-']:last-of-type {
8     padding-right: 0;
9   }
10  a {
11    text-decoration: none;
12  }
13  *, *:after, *:before {
14    -webkit-box-sizing: border-box;
15    -moz-box-sizing: border-box;
16    box-sizing: border-box;
17  }
```

Multi-Views – Overview view (iv)

- Add the Overview route:

app-routing.module.ts

```
1   import { NgModule } from '@angular/core';
2   import { RouterModule, Routes} from '@angular/router';
3
4   import { AuthorsComponent} from './authors/authors.component';
5   import { OverviewComponent } from './overview/overview.component';
6
7   const routes: Routes = [
8     { path: 'authors', component: AuthorsComponent },
9     { path: 'overview', component: OverviewComponent }
10  ];
```

Multi-Views – Overview view (v)

- Add links to the main component:

app.component.html

```
1  <h1>{{ title }}</h1>
2
3  <nav>
4    <a routerLink="">Home</a>
5    <a routerLink="/overview">Overview</a>
6    <a routerLink="/authors">Authors</a>
7  </nav>
8
9  <router-outlet></router-outlet>
```

- Give it a try …

Multi-Views – Details view (i)

- Create AuthorDetails component:
  - Command line:
    ng generate component author-details
- Cut and paste the html code relative to details from “authors.component.html” to “author-details.component.html” and make the following modifications:

author-details.component.html

```
<div *ngIf="author">
  <h2>Information on {{ author.name | uppercase }} </h2>
  <div>
    <label>Num:
      <input [ngModel]="author.num" readonly>
    </label>
  </div>
  <div>
    <label>Name:
      <input [(ngModel)]="author.name" placeholder="name">
    </label>
  </div>
  <div>
    <label>Email:
      <input [(ngModel)]="author.email" placeholder="email">
    </label>
  </div>
</div>
```

Multi-Views – Details view (ii)

- In “author-details.component.ts” file put following lines:

author-details.component.ts

```
1   import { Component, OnInit } from '@angular/core';
2   import { Author} from '../author';
3
4   @Component({
5     selector: 'app-author-details',
6     templateUrl: './author-details.component.html',
7     styleUrls: ['./author-details.component.css']
8   })
9   export class AuthorDetailsComponent implements OnInit {
10    author: Author;
11
12    constructor() {}
```

Multi-Views – Details view (iii)

- Substitute the AuthorDetails Component CSS file with the one given in moodle “author-details.component.css”.

author-details.component.css

```
1
2   label {
3     display: inline-block;
4     width: 3em;
5     margin: .5em 0;
6     color: #607D8B;
7     font-weight: bold;
8   }
9   input {
10    height: 2em;
11    font-size: 1em;
12    padding-left: .4em;
13  }
14  button {
15    margin-top: 20px;
16    font-family: Arial;
17    background-color: #eee;
18    border: none;
19    padding: 5px 10px;
20    border-radius: 4px;
21    cursor: pointer; cursor: hand;
22  }
```

Multi-Views – Details view (iv)

- Add the author details route:

app-routing.module.ts

```
1   import { NgModule } from '@angular/core';
2   import { RouterModule, Routes} from '@angular/router';
3
4   import { AuthorsComponent} from './authors/authors.component';
5   import { OverviewComponent } from './overview/overview.component';
6   import { AuthorDetailsComponent } from './author-details/author-details.component';
7
8   const routes: Routes = [
9     { path: 'authors', component: AuthorsComponent },
10    { path: 'overview', component: OverviewComponent },
11    { path: 'authordetails/:num', component: AuthorDetailsComponent }
12  ];
```

Multi-Views – Details view (v)

- Update “authors.component.html” like this:

authors.component.html

```
1
2   <h2>Authors</h2>
3   <ul class="authors">
4     <li *ngFor="let author of authors">
5       <a routerLink="/authordetails/{{author.num}}">
6         <span class="badge">{{ author.num }}</span> {{ author.name }}
7       </a>
8     </li>
9   </ul>
```

- AuthorsComponent doesn’t use “selectedAuthor” anymore, so its code can be deleted from “authors.component.ts”.

Multi-Views – Details view (vi)

- Update “overview.component.html” like this:

overview.component.html

```
1  <h3>Some Authors</h3>
2  <div class="grid grid-pad">
3    <a *ngFor="let author of authors" class="col-1-4" routerLink="/authordetails/{{author.num}}">
4      <div class="module author">
5        <h4>{{author.name}}</h4>
6      </div>
7    </a>
8  </div>
```

Multi-Views – Details view (vii)

- Get the route that led to the view. Add the following:

author-details.component.ts

```
1  import { Component, OnInit, Input } from '@angular/core';
2  import { Author} from '../author';
3
4  import { ActivatedRoute } from '@angular/router';
5  import { Location } from '@angular/common';
6
7  @Component({
8    selector: 'app-author-details',
9    templateUrl: './author-details.component.html',
10   styleUrls: ['./author-details.component.css']
11 })
12 export class AuthorDetailsComponent implements OnInit {
13   @Input() author: Author;
14
15   constructor(
16     private route: ActivatedRoute,
17     private location: Location
18   ) {}
```

Multi-Views – Details view (viii)

- Extract the num parameter from route.
- Add following lines:

author-details.component.ts

```
1  import { Component, OnInit, Input } from '@angular/core';
2  import { Author} from '../author';
3  import { AUTHORS } from '../authorslist';
```

```
21   ngOnInit() {
22     this.getAuthor();
23   }
24
25   getAuthor(): void {
26     const num = +this.route.snapshot.paramMap.get('num');
27     this.author = AUTHORS.find( predicate: author => author.num === num);
28   }
```

Multi-Views – Details view (ix)

- Add a goBack button to the view:
  - In “author-details.component.ts” file add the following:

```
30    goBack(): void {
31      this.location.back();
32    }
```

  - In “author-details.component.html” file add the following:

```
12    <div>
13      <label>Email:
14        <input [(ngModel)]="author.email" placeholder="email">
15      </label>
16    </div>
17    <br>
18    <button (click)="goBack()">go back</button>
19  </div>
```

Angular Framework

Bootstrap and Angular

Bootstrap & Angular

- One easy way to use Bootstrap in Angular is to import all needed files in “index.html” file:

index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Books</title>
  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css"
  <link rel="stylesheet" href="sticky-footer-navbar.css" >
</head>
<body>
  <app-root></app-root>
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41J
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" integrity="sha3
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js" integrity="sha384-
</body>
</html>
```

Bootstrap & Angular

- Example: