Angular 8 : Online Class

Sahosoft

23-August-2019

By: Sahosoft Solutions

Presented by : Chandan Kumar

Angular 8 : Online Class

Sahosoft

Router Outlet

By: Sahosoft Solutions

Presented by : Chandan Kumar

Router Outlet

Router-outlet in Angular works as a placeholder which is used to load the different components dynamically based on the activated component or current route state. Navigation can be done using router-outlet directive and the activated component will take place inside the router-outlet to load its content.

To enable routing, we need to use router-outlet into our HTML template like this.
<router-outlet></router-outlet>

Sahosoft

- ✓ The Router outlet is a placeholder that gets filled dynamically by Angular, depending on the current router state.
- ✓ **Router outlet** is a dynamic component that the router uses to displays views based on router navigations.

- ✓ Router outlet is a *Routing component*. An Angular component with a RouterOutlet that displays views based on router navigations.

- ✓ The role of **<router-outlet>** is to mark where the router displays a view. (This is the location where Angular will insert the component we want to route to on the view)

- ✓ The **<router-outlet>** tells the router where to display routed views.

Router Outlet

- ✓ The RouterOutlet is one of the router directives that became available to the AppComponent because AppModule imports AppRoutingModule which exported RouterModule.

```
<nav>
<a routerLink="/dashboard">Dashboard</a> |
<a routerLink="/about">About us</a> |
<a routerLink="/contact">Contact us</a>
</nav>

<router-outlet>

</router-outlet>
```

Sahosoft

Angular 8 : Online Class

Sahosoft

Router Link

By: Sahosoft Solutions

Presented by : Chandan Kumar

Router Link

With the help of the routerLink directive, we can link to routes of our application right from the html template. Just add the directive to an HTML-Element. When the user clicks on that element, angular navigates to the specified location.

The routerLink is the selector for the RouterLink directive that turns user clicks into router navigations. you can assign a string to the routerLink

This directive generates our link based on the route path.

Sahosoft

Router Link - Client Side

Syntax

```
template:
<h1>Angular Router</h1>
<div>
<a [routerLink]="['/student’]">Student</a>
</div>

 <router-outlet></router-outlet>
```

Sahosoft

Router Link - Server-Side

For your application to work with server-side rendering, the element hosting the directive has the be a link (a) element.

It is also possible, to navigate to a route from code. To do so, we need to the angular router.

```
import { Router } from '@angular/router';
constructor(private router: Router) {}
```

Once we have that router, navigation is quite simple. Just call the navigate function of Router. This function takes an array. The first entry of the array defines the string of the route, we want to navigate to. The second is optional and allows us to pass in a route parameter.

```
this.router.navigate(['/student']);
```

Sahosoft

Angular 8 : Online Class

Sahosoft

Creating a Routing Module

By: Sahosoft Solutions

Presented by : Chandan Kumar

Creating a Routing Module

By using the command line parameter "--routing", we’re able to create a routing module in the Angular application. So, by using any of the following commands, we can create a routing module in Angular with the default routing set up.

When you generate a module, you can use the **--routing** option like **ng g module my-module --routing** to create a separate file my-module-routing.module.ts to store the module routes.
The file includes an empty Routes object that you can fill with routes to different components and/or modules.
Example : create module file

```
ng g m modulename
```

Sahosoft

```
ng g m modulename --routing
```

g-> generate
m-> module

```
ng g m modulename --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

```
ng g m modulename --routing –-flat
ng generate module modulename --flat --module=anyroute
```

Create Component/ Service without spec.ts

Create Component without spec.ts

```
ng g c student
ng g c student --spec=false
Option "spec" is deprecated: Use "skipTests" instead.
ng g c student --spec false
Option "spec" is deprecated: Use "skipTests" instead.
ng g c student --nospec
Option "spec" is deprecated: Use "skipTests" instead.
ng g s student --skipTests
```

Create Service without spec.ts

```
ng g s my-service1
ng g s student --spec=false
Option "spec" is deprecated: Use "skipTests" instead.
ng g s student --spec false
Option "spec" is deprecated: Use "skipTests" instead.
ng g s student --nospec
Option "spec" is deprecated: Use "skipTests" instead.
ng g s my-service --skipTests
```

Sahosoft

Create new component/directive/class/guard

Create new component/directive/class/guard

ng generate component <name> [options]

OPTION	DESCRIPTION
--skipTests=true|false	When true, does not create "spec.ts" test files for the new **component**/directive/**class**/guard. Default: false
--spec=true|false	***Deprecated***: Use "skipTests" instead. When true (the default), generates a "spec.ts" test file for the new component. Default: true

Difference-between-the-href-and-routerLink

Href

Href - stands for hyper reference , generally used for making link to switch between multiple pages for simple HTML.

href is html anchor tag attribute to navigate to another page. Here a new page will be loaded.

RouterLink

RouterLink is used to achieve the same functionality but angular 2 or above are single page applications, where the page should not reload. RouterLink navigates to New Url and the component is rendered in place of routeroutlet without reloading the page.

routerLink - functionality similar to href but this is in the form of angular2, means to say that routerLink is used in angular 2 for routing purpose and behind the scene routerLink is being converted into href for switch between pages (routes in angular's term)

Sahosoft

Create new component/directive/class/guard

Add multiple module then add new component cli cmd

Sahosoft