breaking thought barriers

Angular Training

Session -14

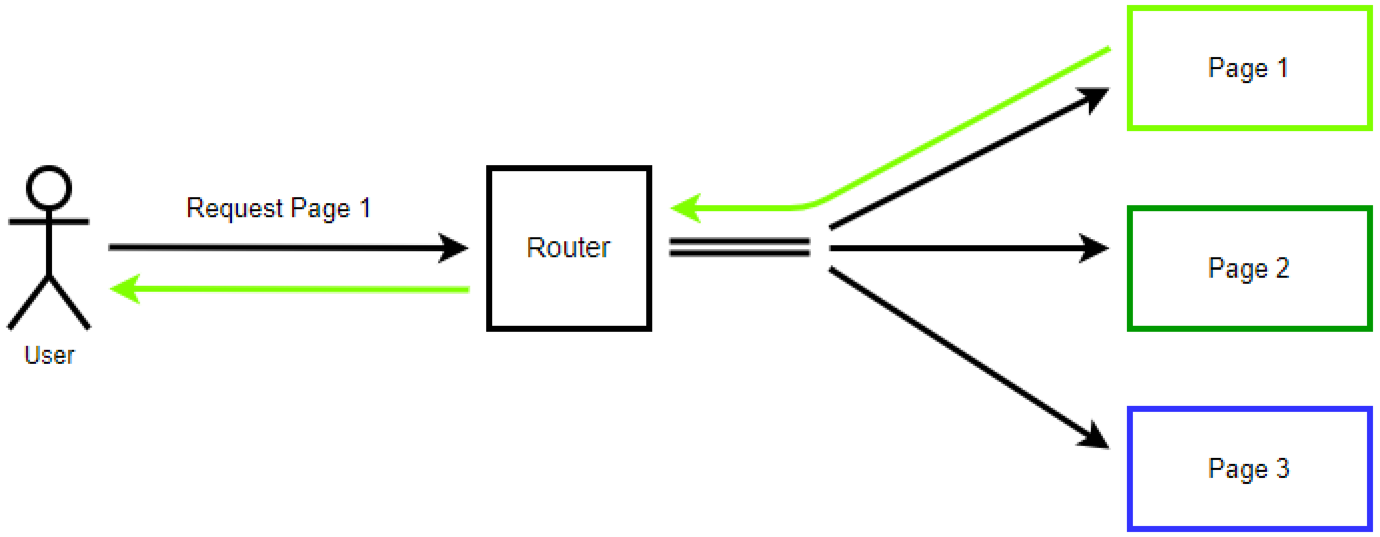
Outlines

Routing in Angular

What is routing in Angular?

Routing means navigating between different pages.

Routing in Angular allows us to move from one part of the application to another part, or from one view to another view.



In a single-page application, we change what the user sees by showing and hiding parts of the display according to that specific component, rather than going out to the server to load a whole new page.

It makes the Angular applications faster than usual applications.

Routing is an important part of this behaviours that SPA exhibits.

How to do routing ?

1. Create Separate Module for Routing

2. Import Router Module ,Routes In your AppRouting Module

```
import { Routes,RouterModules} from '@angular.router'
```

3. Create a Routes Config

```
const appRoutes :Routes =[
    {path:'home',component :HomeComponent},
    {path:'user-list',component:UserListComponent},
    {path:' ',redirectTo:'/home',pathMatch:'full'}
];
```

4. Call RouterModule.forRoot() and give it the Routes config

5. Export this module into you **RootModule**

```
@NgModule({
    imports :[RouterModule.forRoot(appRoutes)] ,
    exports :[RouterModule]
})
```

6. Place a **<route-outlet></router-outlet>** tag in your template where you to perform it.

7. Place links that will take your user to those Routes and use **routerLink** attribute to give them links.

THE ROUTER-OUTLET

The Router-Outlet is a directive that’s available from the router library where the Router inserts the component that gets matched based on the current browser’s URL.

You can add multiple outlets in your Angular application which enables you to implement advanced routing scenarios.

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

Any component that gets matched by the Router will render it as a sibling of the Router outlet.

ROUTES AND PATHS

- Routes are definitions (objects) comprised from at least a path and a component (or a redirectTo path) attributes.

- The path refers to the part of the URL that determines a unique view that should be displayed, and component refers to the Angular component that needs to be associated with a path.

- Based on a route definition that we provide (via a static RouterModule.forRoot(routes) method), the Router is able to navigate the user to a specific view.

- Each Route maps a URL path to a component.

- The path can be empty which denotes the default path of an application and it’s usually the start of the application.

ROUTES AND PATHS

- The path can take a wildcard string (**). The router will select this route if the requested URL doesn’t match any paths for the defined routes. This can be used for displaying a “Not Found” view or redirecting to a specific view if no match is found.

- **{ path: 'contacts', component: ContactListComponent}**

- If this route definition is provided to the Router configuration, the router will render ContactListComponent when the browser URL for the web application becomes /contacts.

ROUTE MATCHING STRATEGIES

- The Angular Router provides different route matching strategies.
- The default strategy is simply checking if the current browser's URL is prefixed with the path.
- For example our previous route:

```
{ path:  'contacts', component:  ContactListComponent}
```

- Could be also written as:

```
{ path:  'contacts', pathMatch: 'prefix', component:  ContactListComponent}
```

- The patchMath attribute specifies the matching strategy. In this case, it's prefix which is the default.

ROUTE MATCHING STRATEGIES

- The second matching strategy is **full**. When it’s specified for a route, the router will check if the the path is exactly **equal** to the path of the current browser’s URL:

```
{ path:  'contacts',pathMatch: 'full', component:  ContactListComponent}
```

ROUTE PARAMS

- Creating routes with parameters is a common feature in web apps.
- Angular Router allows you to access parameters in different ways:

  A. Using the ActivatedRoute service,

  B. Using the ParamMap observable available starting with v4.

- You can create a route parameter using the colon syntax.

{ path: 'contacts/:id', component: ContactDetailComponent}

NAVIGATION DIRECTIVE

The Angular Router provides the routerLink directive to create navigation links.

This directive takes the path associated with the component to navigate to.

For example:

<a [routerLink]="'/contacts'">Contacts</a>