WSA | Forward looking IT finishing school

Routing And Navigation

Angular

www.webstackacademy.com

Routing And Navigation

(Navigate from one component to another)

Routing And Navigation

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.

The Angular Router

- 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.
- 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 And Navigation

Agenda:

1. Configuring the Routes
2. Implementing SPA
3. Working with Route and Query parameters
4. Programmatic Navigation

Routing And Navigation

1. Configure the routes: import router module to app.module.ts

```
import { RouterModule } from '@angular/router';

imports: [
    BrowserModule,
    HttpModule,
    RouterModule.forRoot([
      { path: '', component: HomeComponent },
      { path: 'followers/:id', component: GithubprofileComponent },
      { path: 'followers', component: GithubfollowersComponent },
      { path: 'posts', component: PostsComponent },
      { path: '**', component: NotfoundComponent },
    ])
  ],
```

2. Configure the routes: add router outlet to app.component.html

```
<app-navbar></app-navbar>
<router-outlet></router-outlet>
```

Routing And Navigation

Link Navigation: Navigating from Navbar menus

```
<nav class="navbar navbar-expand-sm bg-light">
  <ul class="navbar-nav">
    <li class="nav-item">
       <a class="nav-link" href="/followers">Followers</a>
    </li>
    <li class="nav-item">
       <a class="nav-link" href="/posts">Posts</a>
    </li>
  </ul>
</nav>
```

The problem:

- **Using href in anchor tag will reinitialize the component each time whenever user clicks on the link.**
- **If the project is very big, waiting time will be increased.**
- **To fix this instead of using href use Router Link. Details are given in the next slide**

Routing And Navigation

Router Link Directive: Dynamic Navigation

- Navigating the link with dynamic link is achieved by using the router link binding as shown below,

```
<div class="media-body">
   <h4 class="media-heading">
     <a [routerLink]="['/followers',follower.id]">{{ follower.login }}</a>
   </h4>
   <a href="follower.html_url">{{ follower.html_url }}</a>
 </div>
```

Dynamic Active class

- Changing the active class dynamically depending on the current page user is accessing is achieved by using routerLinkActive directive.
- Apply this directive on all the list of navbar.

Routing And Navigation

Accessing route parameter from URL

- Angular provides us with the ActivatedRoute object.
- We can access the URL through this object, but first, you have to inject it into your component. Inject it like any other service:

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

@Component({
  selector: 'app-githubprofile',
  templateUrl: './githubprofile.component.html',
  styleUrls: ['./githubprofile.component.css']
})
export class GithubprofileComponent implements OnInit
{
  constructor(private route: ActivatedRoute) { }

}
```

Routing And Navigation

Accessing route parameter from URL

- Fetch the parameter from URL by subscribing to the activated route object.
- Using the subscription is the same as any other subscription. If there is a change then the observable’s value will get pushed to the callback function.

```
export class GithubprofileComponent implements OnInit
{

  constructor(private route: ActivatedRoute) { }
  ngOnInit() {
    this.route.paramMap
      .subscribe(params => {
        console.log(params);
        let id = +params.get('id');
        console.log("The ID is" + id);
      });
  }
 }
```

Routing And Navigation

Accessing query parameters from URL

- Accessing query string parameters is similar to accessing URL parameters.
- It’s just a different property on the ActivatedRoute object; queryParamMap. So all the same principles apply, but make sure to use the right property.

```
export class GithubprofileComponent implements OnInit {
  constructor(private route: ActivatedRoute) { }
  ngOnInit() {
    this.route.queryParamMap.
      subscribe(params => {
        console.log("The optional parameters are" + params);

        let pageNumber = params.get('page');
        let sortOrder = params.get('order');

        console.log("The actual values" + pageNumber + sortOrder);
      });
  }
 }
```

Exercise

- Create a GitHub profile page by accessing REST API provided by the GIT
  - Example URL: https://api.github.com/users/

- Display the current user details, which should have the following details:
  - ✓ Name of the user
  - ✓ Profile pic
  - ✓ Number of followers
  - ✓ Place

Thank you

WebStack Academy

#83, Farah Towers,
1st Floor, MG Road,
Bangalore – 560001

M: +91-809 555 7332
E: training@webstackacademy.com

WSA in Social Media: