Routing

Agenda

- Конфігурація роутів
- Роут лінки
- Події роутів
- Передача параметрів через роут
- Поточний роут
- Навігація з компоненти
- Guards
- Lazy Loading

Що таке Router?

- дозволяє здійснювати навігацію з одного шаблону на інший, коли користувач переходить на посилання чи здійснює певні дії

Конфігурування Router

- app.module

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

const routes: Routes = [
  { path: 'posts', component: PostsComponent },            // конфігурування шляхів
  { path: 'post/:id', component: SinglePostsComponent },
];

@NgModule({
  imports: [ RouterModule.forRoot(routes) ],                    // підключаємо модуль
})
```

404, редірект

```
{ path: '**', component: PageNotFoundComponent }  // якщо не знайдено жодного роута

{ path: '',   redirectTo: '/post', pathMatch: 'full' }, // задаємо редірект на роут post
```

Router посилань

```
<router-outlet></router-outlet>          // вказує місце де мають відображатись компоненти залежно від роута

<a routerLink="/posts" routerLinkActive="active">Posts</a> // вказується шлях куди має перейти, і додається клас active для активного посилання

<div class="card" *ngFor="let post of allPost">              // роут на один пост
     <a routerLink="/posts/{{post.id}}">Posts</a>
</div>
```

Передача параметрів

```
<a [routerLink]="['/post', post.id]">            // параметри для роута передаються як властивості коспоненти, і згенерується такий url localhost:4200/post/1
```

в компоненті потрібно отримати цей параметр і зробити запит

```
import { Router, ActivatedRoute, ParamMap } from '@angular/router'; // імпортуємо
constructor(
   private router: Router,
  private route: ActivatedRoute,                  // інджектимо залежності
  private data: PostService
) {}
ngOnInit() {
          let id = this.route.snapshot.paramMap.get('id');  // отримуємо id з параметра
     this.data.getSinglePost(id).subscribe( (post: IPost) => {
          this.post = post;                                  // викликаємо метод сервіса із запитом
          })
}
```

Навігація з компоненти

```
gotoPosts() {                                                    // перехід по
вказаному роуту при виклику методу
  this.router.navigate(['/posts']);
}

 import {Location} from '@angular/common';

 constructor(private location: Location) {}

 goBack() {

     this.location.back(); }                          // крок назад, аналогічний як у браузері

 }
```

Допоміжний router-outlet

можна одночасно показувати кілька роутів, але тоді їх потрібно називати

```
<router-outlet name="list"></router-outlet>

<router-outlet name="bio"></router-outlet>
```

https://onehungrymind.com/named-router-outlets-in-angular-2/

https://www.techiediaries.com/angular-router-multiple-outlets/

Guards

Не завжди потрібно, щоб користувач мав доступ до будь-якого місця програми. Наприклад:

- спочатку користувач повинен увійти (аутентифікувати).
- слід отримати деякі дані, перш ніж відображати цільовий компонент.

Щоб це реалізувати потрібно додати **guards** до конфігурації маршруту, щоб обробляти ці сценарії.

- Якщо **guards** повертає true, процес навігації продовжується.
- Якщо **guards** повертає false, процес навігації зупиняється

https://angular.io/guide/router#milestone-5-route-guards

- створення guard(cli) `ng g g <name>`

```
import {CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot} from "@angular/router";

export class AuthGuard implements CanActivate{
    canActivate(route: ActivatedRouteSnapshot, state: RouterStateSnapshot) : boolean{
        return true;
    }
}
```

Якщо AuthGuard дозволяє перехід, то метод canActivate () повинен повертати true. Якщо доступ заборонений, то метод повертає false

```
{ path: 'about', component: ProfileComponent, canActivate: [AuthGuard]} - // перед тим як перейти по роуту перевіряється чи guard повертає true
```

CanDeactivate перевіряє можливість перехід з певного компонента

Firebase Guard

https://github.com/angular/angularfire/blob/master/docs/auth/router-guards.md

Lazy Loading

Із збільшенням функціоналу аплікації і збільшується розмір файлів. Щоб з цим боротися можна застосувати техніку **lazy loading**, коли певні модулі завантажуються за певним запитом. Для цього аплікацію потрібно розбити на модулі.

```
{                                                                    // модуль завантажується коли користувач перейде на роут
  path: 'admin',
  loadChildren: import('./admin/admin.module').then(m => m.AdminModule),
  data: { preload: true }     // модуль завантажеться через кілька секунд після завантаження аплікації
},
```

Посилання

https://metanit.com/web/angular2/7.1.php

https://angular.io/guide/router

https://angular-2-training-book.rangle.io/handout/routing/routeparams.html

https://medium.com/@ryanchenkie_40935/angular-authentication-using-route-guards-bf7a4ca13ae3

https://angularfirebase.com/lessons/how-to-lazy-load-components-in-angular-4-in-three-steps/

https://codeburst.io/using-angular-route-guard-for-securing-routes-eabf5b86b4d1

https://stackblitz.com/docs варіанти імпорту/експорту stackblitz

Відео

https://www.youtube.com/watch?v=KJ5WYDN3Zng

https://www.youtube.com/watch?v=cKod7WX0qUc&list=PLqHlAwsJRxANhhHlAlazVrbX69UMJ9Bcu