Angular Créer de la navigation

Dans une application web, le routing est une brique essentielle pour des interfaces navigables.

Pour cela Angular offre une série d'outils simples à mettre en place

Un interface navigable est faite de composants web, qui seront appelés au clic de l'utilisateur sur des liens de menu proposant la navigation, la plus souvent la navbar sert de menu principal.

1 CRÉER UNE ROUTE

2 LA DIRECTIVE routerLink

3 PASSER DES PARAMÈTRES DANS L'URL

4 LA CLASS ActivatedRoute

5 LA CLASS *Router* et la méthode navigate()

1 CRÉER UNE ROUTE (<u>3</u> étapes)

1 app-routing.module.ts

app.component.html

```
<nav>
2      <a routerLink='/'>Home</a>
      <div routerLink='/detail'>....</div>
      <nav>

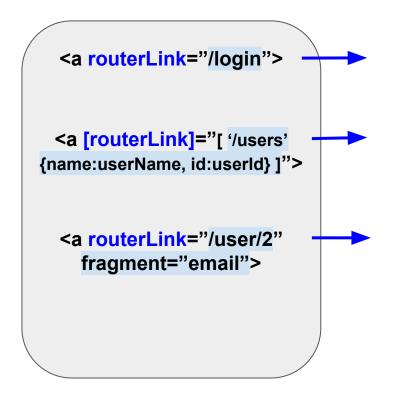
<router-outlet>
3
```

Le composant **<router-outlet>** se charge d'afficher dans la vue HTML, le bon composant qui appelé par l'url / ou /login

Angular propose de créer un tableau de routes. Une route est un objet qui prend 2 propriétés.

{ path: urlCible, component: classDuComponentACharger }

2 LA DIRECTIVE routerLink



routing statique. Affichera le component qui match avec l'url (/login)

routing dynamique avec 2 paramètres.

Passés dans un tableau d'objets. (/users/fred/2)

routing avec un fragment. (/users/2#email)

routerLink est un attribut spécifique que l'on peut appliquer sur n'importe quel élément HTML. Il permet le changement d'url lorsque l'utilisateur cliquera sur cet élément, déclenchant ainsi le router-outlet.

https://angular.io/api/router/RouterLink

3 PASSER DES PARAMÈTRES DANS L'URL

1 app-routing.module.ts

app.component.ts

```
            routerLink='/user/1'>Fred
            routerLink='/user/2'>Marie
            routerLink='/user/3'>Noah

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

Le composant router-outlet se chargera d'afficher le bon composant. Et celui-ci pourra récupérer les paramètres de l'url via la class ActivatedRoute (page suivante)

On définit des paramètres nommés lorsque l'on crée une route { path: urlCible/:param1/:param2, component: classDuComponentACharger }

4 La class ActivatedRoute pour récupérer les paramètres de l'url

user-detail.component.ts Le component cible peut récupérer constructor(private route:ActivatedRoute) {} les paramètres de l'url ngOnInit() { Soit de manière statique this.userId = this.route.snapshot.params.id // ou Soit de manière dynamique this route.params.subscribe(routeParams => { this.userId = routeParams.id **})**;

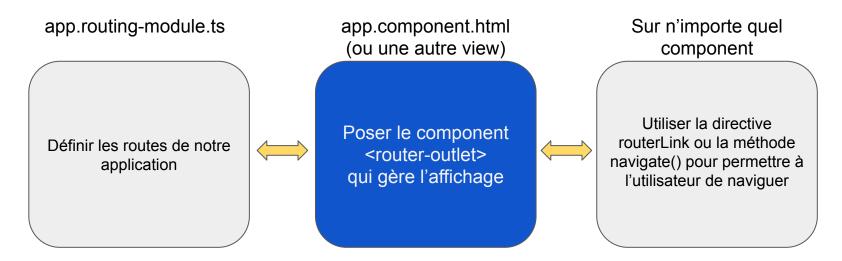
https://angular.io/api/router/ActivatedRoute

5 La class router pour naviguer via ses méthodes

user-detail.component.ts La méthode navigate() permet de constructor(private router:Router) {} naviguer vers une page gotoUsersPage() { en spécifiant l'url this.router.navigate(['users']); gotoUserDetailPage(userId) { en spécifiant l'url, et en lui passant des paramètres this router.navigate(['users', {id:userId}]); nommés dans un objet

https://angular.io/api/router/Router

Pour résumer : le routing c'est ...



Le principe du routing est de permettre à l'interface d'être navigable.

Angular permet de mettre cela en place facilement.

Le routing permet également de charger de modules à la demande lors de la navigation. C'est le *lazy-loading*. voir le lien https://angular.io/guide/lazy-loading-ngmodules