

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 (3 étapes)

1 app-routing.module.ts

```
const route:Routes = [  
  {  
    path: "",  
    component: ListComponent  
  },  
  {  
    path: 'detail',  
    component: DetailComponent  
  }  
]
```

app.component.html

```
<nav>  
  2 <a routerLink="/">Home</a>  
  <div routerLink="/detail">....</div>  
</nav>  
  
<router-outlet></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

```
<a routerLink="/login">
```

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

```
<a [routerLink]="[ '/users'  
{name:userName, id:userId} ]">
```

routing dynamique avec 2 paramètres.
Passés dans un tableau d'objets. (/users/fred/2)

```
<a routerLink="/user/2"  
  fragment="email">
```

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

```
const route:Routes = [  
  
  {  
    path: 'user/:name/:id',  
    component: UserComponent  
  },  
  
]
```

app.component.ts

```
<ul>  
  <li routerLink='/user/1'>Fred</li>  
  2 <li routerLink='/user/2'>Marie</li>  
    <li routerLink='/user/3'>Noah</li>  
</ul>  
  
<router-outlet></router-outlet> 3
```

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: classDuComposantACharger }`

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

user-detail.component.ts

```
constructor( private route:ActivatedRoute ) {}  
ngOnInit() {  
  
  this.userId = this.route.snapshot.params.id  
  
  // ou  
  
  this.route.params.subscribe( routeParams => {  
    this.userId = routeParams.id  
  });  
}
```

Le component cible peut récupérer les paramètres de l'url

Soit de manière statique

Soit de manière dynamique

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

5 La class router pour naviguer via ses méthodes

user-detail.component.ts

```
constructor( private router:Router ) {}

gotoUsersPage() {
  this.router.navigate(['users']);
}

gotoUserDetailPage(userId) {
  this.router.navigate(['users', {id:userId} ]);
}
```

La méthode navigate() permet de naviguer vers une page

en spécifiant l'url

en spécifiant l'url, et en lui passant des paramètres 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>