Atelier 3 :
Routage

Objectifs

Cet atelier vise à utiliser la technique de routage (routing) du framework Angular. A la fin de cet atelier vous serez capable de :

1. définir et configurer une route,
2. utiliser la route et ses informations.

Enoncé

À la fin de l'atelier précédent, nous avons abouti à une application de boutique en ligne qui dispose d'un catalogue de produits de base. L'application n'a pas d'états variables ou de navigation. Il existe une seule URL, qui affiche toujours la page "DSI Store" avec une liste de produits et leur description.

Dans cette section, nous allons étendre l'application pour afficher les détails complets du produit dans des pages séparées, avec leurs propres URL.

Pour ce faire, nous utiliserons le routeur d'Angular, qui permet d'afficher différents components et données à l'utilisateur en fonction de l'emplacement de l'utilisateur dans l'application.

Travail à faire

L'application est déjà configurée pour utiliser le routeur d'Angular et utiliser le routage pour accéder au component de la liste de produits que vous avez modifié précédemment (voir le fichier (`app.module.ts`). Définissons un itinéraire pour afficher les détails de chaque produit.

1. Nous commençons par générer un nouveau component qu'on nomme `details-produit`.
2. Une route associe un ou plusieurs chemins d'URL à un composant. Dans le fichier `app.module.ts`, et à l'intérieur du tableau des routes, nous ajoutons un nouveau chemin (qui prend pour paramètres :
    a. URL : `produits/:produitId`
    b. Réponse : le component créé dans l'étape précédente

3. Nous allons maintenant définir un lien en utilisant la directive `RouterLink`. Cette directive définit le mode de navigation de l'utilisateur vers la route (ou l'URL) de manière déclarative dans le template du component. Nous voulons que lorsque l'utilisateur clique sur un nom de produit nous lui affichons les détails de ce produit. Pour le faire, nous allons mettre à jour le fichier `liste-produit.component.html` par :
    a. ajouter à la directive `*ngFor` la variable `produitId` qui recevra les index du tableau des `produits` dans chaque itération,
    b. ajouter une liaison de propriété `routerLink` pour lui assigner le chemin défini précédemment ainsi que la variable `produitId`.

Le component `details-produit` gère l'affichage de chaque produit. Le routeur d'Angular affiche les components en fonction de l'URL du navigateur et des routes que nous avons définis. Nous utiliserons le routeur d'Angular pour combiner les données du produit et les informations de routage pour afficher les détails spécifiques à chaque produit.

4. Dans le fichier `details-produit.component.ts`, nous commençons par importer le `ActivatedRoute` depuis le package `@angular/router` ainsi que le tableau de `produits` depuis le fichier `produits.ts`.
5. Dans le même fichier nous définissons (à l'intérieur de la classe) la propriété `produit`, puis nous injectons, dans le constructeur, le `ActivatedRoute`. Notons bien que le `ActivatedRoute` est spécifique à chaque component routé, il sera chargé par le routeur d'Angular. Il contient des informations sur la route, ses paramètres et des données supplémentaires qui lui sont associées.
6. Nous ajoutons ensuite dans le méthode `ngOnInit()` le code suivant :

```
ngOnInit() {
  this.route.paramMap.subscribe(params => {
    this.produit = produits[+params.get('produitId')];
  });
}}
```

7. Nous mettons à jour maintenant le template `details-produit.component.html` pour afficher les détails du produit.