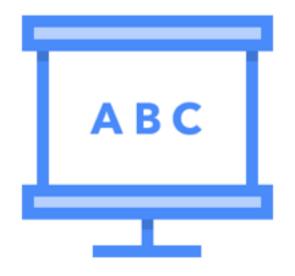


4. Démarrer un nouveau projet lonic v4



4.1. Navigation et routing

Avec le framework Ionic, le routing de l'application se fait comme avec une application Angular. On utilise simplement un « Wrapper » Ionic/Angular pour modifier le comportement de animation lors de la navigation, offrant ainsi des transition équivalente aux application « Native »

Doc Routing:

https://angular.io/guide/router

https://angular.io/guide/lazy-loading-ngmodules

4.1.1. Navigation avec le composant « ion-tabs »

Ionic propose un composant nommé « Tabs » qui permet de créer facilement un système de navigation avec des « Tabs ».

Doc: https://github.com/ionic-team/ionic/tree/master/core/src/components/tabs

Celui-ci est un grand classique des applications mobile. Voici comment l'implanter.

- Créer un module pour chaque « Tab » avec un routing et un composant basic.
- Modifier le fichier

 ./src/app/app.routing.ts
 comme ceci:

 Modifier le fichier `./src/features/ index-page/index-page.routing.ts:

```
. . .
const routes: Routes = [
    path: 'tabs',
    component: IndexPageComponent,
    children: [
          path: 'one',
          outlet: 'one',
          loadChildren: '../tab-one/tab-one.module#TabOneModule'
          path: 'two',
          outlet 'two',
          loadChildren: '../tab-two/tab-two.module#TabTwoModule'
          path: 'three',
          outlet: 'three',
          loadChildren: '../tab-three/tab-three.module#TabThreeModule'
    redirectTo: '/tabs/(one:one)'
```

modifier le fichier `./src/app/features/index-page/containers/index-page/index-page.component.html`

fichier `./src/app/features/tab-one/containers/tab-one/tab-one.component.html`

```
<ion-header>
 <ion-toolbar color="primary">
   <ion-title>
     Ionic Blank
   </ion-title>
 </ion-toolbar>
</ion-header>
<ion-content padding>
 Tab One work!
</ion-content>
```

- fichier `./src/app/features/tab-one/tab-one.routing.ts`

```
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { TabOneComponent } from './containers/tab-one/tab-one.component';
const routes: Routes = [
    path: '',
    component: TabOneComponent,
    outlet: 'one'
];
@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class TabOneRoutingModule { }
```

4.1.2. Navigation avec le composant « ion-split-pane »

Un autre grand classique c'est le menu de navigation « swipable » lonic propose aussi une intégration de cette fonctionnalité.

Doc:

https://github.com/ionic-team/ionic/tree/master/core/src/components/split-pane https://github.com/ionic-team/ionic/tree/master/core/src/components/nav https://github.com/ionic-team/ionic/tree/master/core/src/components/menu https://github.com/ionic-team/ionic/tree/master/core/src/components/toolbar https://github.com/ionic-team/ionic/tree/master/core/src/components/menu-toggle Voici un exemple de son implémentation:

- fichier `app.component.html`

```
<ion-app>
    <ion-split-pane>
        <ion-menu [content]="content">
          <ion-header>
            <ion-toolbar color="primary">
              <ion-title>Menu</ion-title>
            </ion-toolbar>
          </ion-header>
          <ion-content>
            menu items
          </ion-content>
        </ion-menu>
    <ion-nav #nav #content main>
      <ion-router-outlet></ion-router-outlet>
    </ion-nav>
 </ion-split-pane>
</ion-app>
```

sur chaque fichiers `tab-XXX.component.html:

```
• • •
<ion-header>
  <ion-toolbar color="primary">
    <ion-buttons slot="start">
      <ion-menu-toggle>
        <ion-button>
          <ion-icon slot="icon-only" name="menu"></ion-icon>
        </ion-button>
      </ion-menu-toggle>
    </ion-buttons>
    <ion-title>
      Ionic Blank
    </ion-title>
  </ion-toolbar>
</ion-header>
```