BUT2 RA Développement mobile - cours 5 :

Idéalement, utiliser VSCode et Linux pour ce cours. Sinon, utilisez DartPad : `https://dartpad.dev/`

Ce cours/TD est la suite directe du cours précédant.

La suppression d'un élément

Passer une fonction en paramètre

La suppression d'une ligne implique de passer une fonction en paramètre de notre widget `TheAmazingRow`.

```
const TheAmazingRow({
  ...
  required this.onDelete,
});
...
final Function(String) onDelete;
```

La fonction sera appelée au niveau du bouton de suppression:

```
IconButton(
  icon: const Icon(Icons.delete),
  onPressed: () => onDelete(label),
),
```

La fonction de suppression

On écrit notre fonction de suppression dans notre widget `HomePage` (ou plutot dans son state `_HomePageState`), car c'est ici que se trouve notre liste `pokedex`. Vous pouvez la placer en dessous de la méthode `build`.

```
void onDeletePokemon(String label) {
  pokedex.removeWhere((e) => e.name.toLowerCase() == label.toLowerCase());
  setState(() {});
}
```

On passe la fonction en paramètre :

```
TheAmazingRow(
  label: pokemon.name.toUpperCase(),
  icon: pokemon.illustration,
  onDelete: onDeletePokemon, // <==== pas besoin des parenthèses
)
```

La navigation entre pages

Documentation officielle

La documentation officielle est très claire : `https://docs.flutter.dev/cookbook/navigation/named-routes`

Préparer une deuxième page

Le but est de cliquer sur une ligne et d'ouvrir une page qui concerne un pokémon. Vous pouvez créez un fichier `detail_page.dart` et mettre dedans le widget page `SecondRoute` de la documentation officielle. Renommez le widget en `DetailPage`

Définir les noms de pages

```dart
class PageName {
  static const String home = '/';
  static const String detail = '/detail_page';
}
```

Définir le routeur

Notre `MaterialApp` devient :

```
  return MaterialApp(
        theme: ... ,
        darkTheme: ... ,
        themeMode: ... ,
        initialRoute: PageName.home,
        routes: {
          PageName.home: (context) => const HomePage(),
          PageName.detail: (context) => const DetailPage(),
        },
      );
```

Notez bien que la documentation offficielle utilise les String `'/'` et `'/second'` en nom de page, tandis que nous utilisons des variables statiques `PageName.home` et `PageName.detail`

Passer à une autre page

Dans notre widget `TheAmazingRow`, on wrap le contenu avec un bouton `InkWell`. On peut ainsi appeler la fonction de navigation. On ajoute également dans `arguments` le nom du pokémon, afin de pouvoir l'utiliser dans la nouvelle page.

```
InkWell(
  onTap: () => Navigator.pushNamed(
    context,
    PageName.detail,
    arguments: label,
  ),
  child: ...
```

Récupérer l'argument dans la nouvelle page

Pour récupérer le nom du pokémon passé en argument du `Navigator`, on met dans la méthode `build` de `DetailPage`

```
class DetailPage extends StatelessWidget {
  const DetailPage({super.key});

  @override
  Widget build(BuildContext context) {

    final pokemonName = ModalRoute.of(context)!.settings.arguments as String;

    ...
```

Vous devez obtenir le résultat suivant :

En cliquant sur la zone rouge	la page ci-essous s'ouvre