Développement d'applications mobiles

Chapitre 3 - Flutter - concepts de base

Daniel Schreurs

20 septembre 2022

Haute École de la Province de Liège

Table des matières i

Introduction

Introduction : In Flutter, everything is a widget.

- Flutter est composé de widgets...beaucoup de widgets ;
- Un widget permet de décrire :
  - Une interface ;
  - Une fonctionnalité ;

Material Component

Material Component : Scaffol

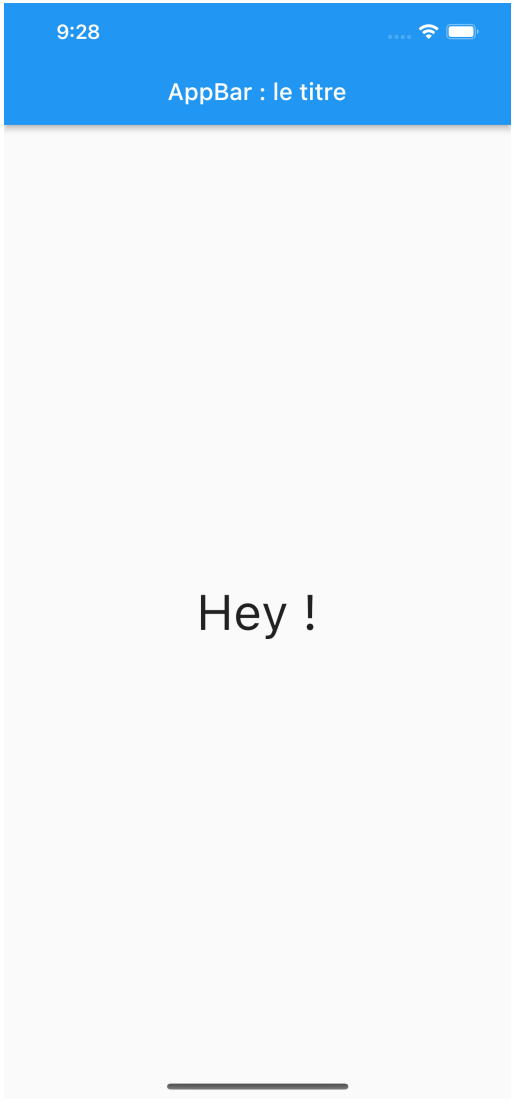
- Le Scaffol est la structure de base des applications Android ;
- Un ensemble pouvant regrouper plusieurs éléments comme :
  - Une *appBar* ;
  - Un *body* ;
  - une *BottomNavigationBar* ;
  - Et beaucoup d'autres encore...

```
return Scaffold(
    body: Text("Hey"),
);
```

Material Component : AppBar

- L'AppBar est un widget ;
- Se situer en haut de l'écran ;
- Enfant de *Scaffold*.

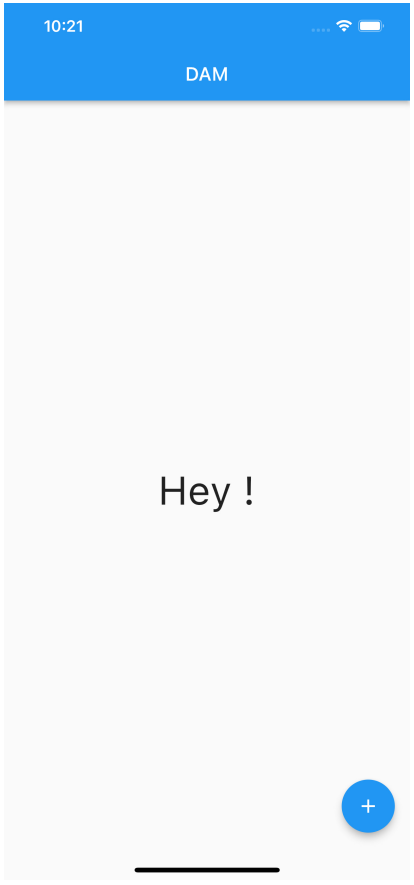
```
return Scaffold(
    appBar: AppBar(
        title: Text(
            widget.title
            ),
    ),
    body: Text("Hey"),
);
```



Material Component : floatingActionButton

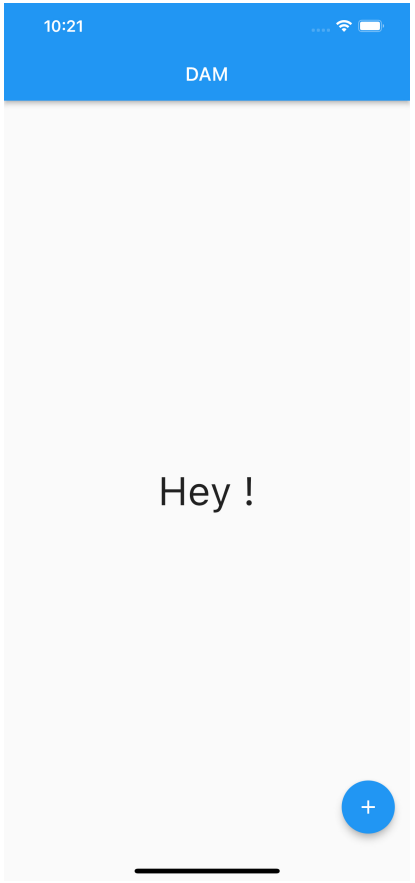
- Le floatingActionButton est un widget ;
- Consiste en un bouton flottant ;
- Qui peut être placé : centerDocked, bottomNavigationBar, centerFloat, endDocked,bottomNavigationBar, bottomNavigationBar, endTop,miniStartTop, startTop.

```
floatingActionButton:
    FloatingActionButton(
    onPressed: () => {print("
        Bonjour")},
    child: const Icon(Icons.
        add),
    ),
```



Material Component : bottomNavigationBar

- La bottomNavigationBar est une barre de navigation ;
- Située en bas de l'écran ;
- Par défaut, sa taille est nulle ;
- Il convient d'y inclure un Container par exemple ;



Material Component : bottomNavigationBar

```
floatingActionButtonLocation:
    FloatingActionButtonLocation.centerDocked,
bottomNavigationBar: BottomAppBar(
    shape: CircularNotchedRectangle(),
    child: Container(
    height: 50,
    child: Row(
        mainAxisAlignment: MainAxisAlignment.spaceAround
            ,
        children: [
        IconButton(
            onPressed: pressed,
            icon: Icon(Icons.add_location),
        ),
        IconButton(
            onPressed: pressed,
            icon: Icon(Icons.forward),
        ),],),),),
```

Cupertino (Style iOS)

Cupertino (Style iOS)

- Style présent sur iOS ;
- Nom donné en référence à la ville où se situe le siège d'Apple ;

Nous ne couvrons pas ces widgets.

Détection de la plateforme

Détection de la plateforme

- Pour se fondre dans le style de son système d'exploitation hôte, il convient alors de lui faire posséder les deux styles : Material Design et Cupertino ;
- Créer deux Widgets, un pour Android et l'autre pour iOS ;
- Il faut importer `Platform`.

Détection de la plateforme

Ces propriétés retournent `true` si la plateforme est détectée au runtime.

- `Platform.isAndroid` pour détecter si le programme s'exécute sur ;
- `Platform.isFuchsia` pour détecter si le programme s'exécute sur Fuchsia ;
- `Platform.isIOS` pour détecter si le programme s'exécute sur iOS ;
- `Platform.isLinux` pour détecter si le programme s'exécute sur Linux ;
- `Platform.isMacOS` pour détecter si le programme s'exécute sur macOS.

Détection de la plateforme

```
1 @override
2 Widget build(BuildContext context) {
3 return Platform.isIOS ? myCupertinoApp() :
      myCupertinoApp();
4 }
```

`myCupertinoApp` et `myCupertinoApp` étant des classes `Widget` personelles.