Ex 1 : Basics / Widgets / Styling

Starting date: @2020년 9월 14일

Deadline: @2020년 9월 21일

Instructions

For this exercise you will have to create 3 screens : The homepage, the detail page and the discovery page with some statics data or get the dummy data in the data folder. You don't have to link it or create the navigation yet, it will be on the next exercise. If there are thing you already know how to do it like dynamic data or navigation you can do it but don't forget about the deadline.

I will also help me to know what you know in Flutter.

Objectives

- Use Flutter Widgets
- Know how to style and gave a global style
- Create layouts
- Use image (local and web)
- Know how to position element on the screen
- Know the issue you can have depended on the device and how to solve it
- Create you own components
- Use custom font (you can use any font you want)
- Use custom icons (here icomoon)
- Know the difference between stateless and stateful widget

Documentation

You don't have to read all the documentation (because it gonna takes a long time) but you can refers to it when you don't know how to do something or when you have issues.

For all the widgets that already exist and that are provided by Flutter :

- https://flutter.dev/docs/development/ui/widgets
- https://flutter.dev/docs/development/ui/widgets/material
- https://gallery.flutter.dev/#/ (widget examples)
- https://flutter.dev/docs/cookbook

Stateful and Stateless widget :

- https://api.flutter.dev/flutter/widgets/StatefulWidget-class.html
- https://api.flutter.dev/flutter/widgets/StatelessWidget-class.html

For the Theme (global style, color, etc) :

- https://api.flutter.dev/flutter/material/ThemeData-class.html

Assets :

- https://flutter.dev/docs/development/ui/assets-and-images

Image :

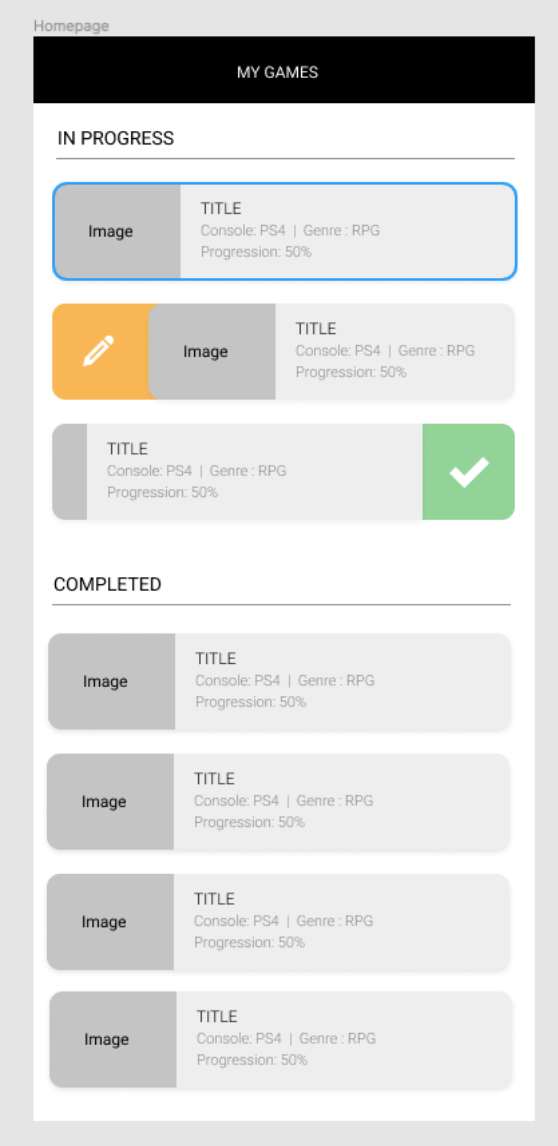
- https://flutter.dev/docs/cookbook/images/network-image#:~:text=Displaying images is fundamental for,the Image.network() constructor

Custom font :

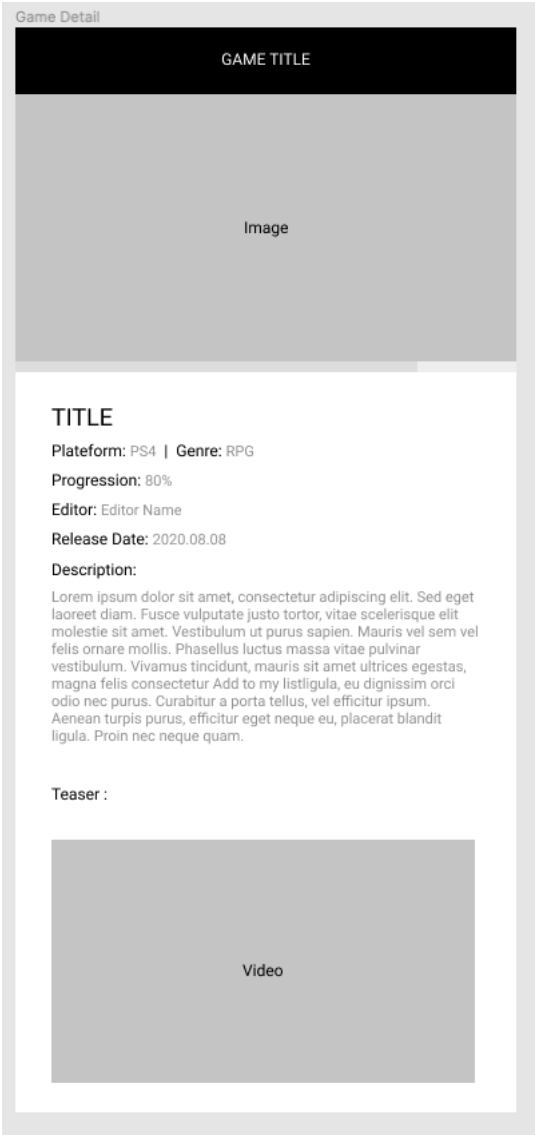
- https://flutter.dev/docs/cookbook/design/fonts

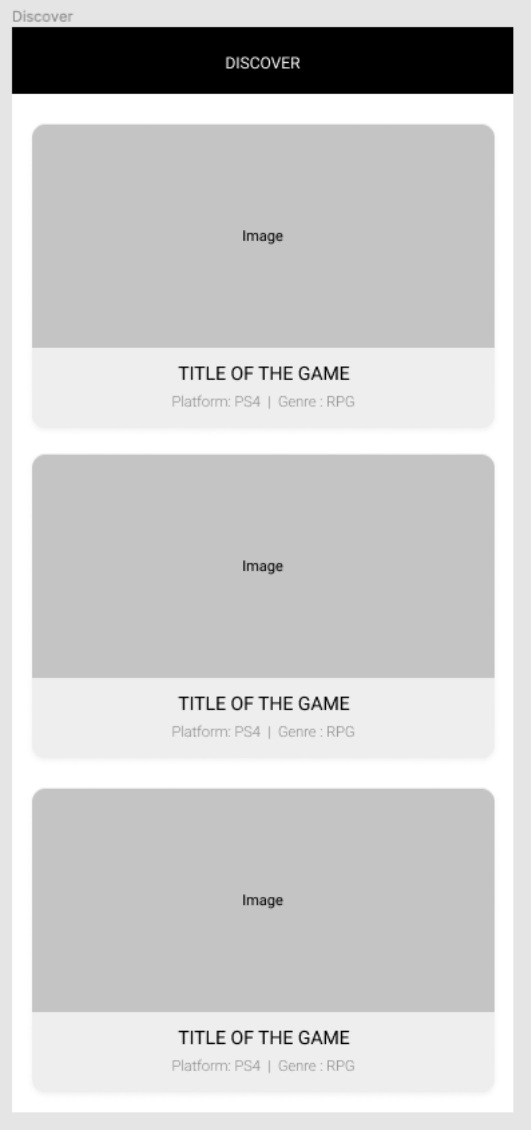
Icons that are provided by Flutter :

- https://api.flutter.dev/flutter/material/Icons-class.html


Homepage
MY GAMES
IN PROGRESS
Image
TITLE
Console: PS4 | Genre : RPG
Progression: 50%
Image
TITLE
Console: PS4 | Genre : RPG
Progression: 50%
TITLE
Console: PS4 | Genre : RPG
Progression: 50%
COMPLETED
Image
TITLE
Console: PS4 | Genre : RPG
Progression: 50%
Image
TITLE
Console: PS4 | Genre : RPG
Progression: 50%
Image
TITLE
Console: PS4 | Genre : RPG
Progression: 50%
Image
TITLE
Console: PS4 | Genre : RPG
Progression: 50%

Screens


Game Detail
GAME TITLE
Image
TITLE
Plateform: PS4 | Genre: RPG
Progression: 80%
Editor: Editor Name
Release Date: 2020.08.08
Description:
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed eget laoreet diam. Fusce vulputate justo tortor, vitae scelerisque elit molestie sit amet. Vestibulum ut purus sapien. Mauris vel sem vel felis ornare mollis. Phasellus luctus massa vitae pulvinar vestibulum. Vivamus tincidunt, mauris sit amet ultrices egestas, magna felis consectetur Add to my listligula, eu dignissim orci odio nec purus. Curabitur a porta tellus, vel efficitur ipsum. Aenean turpis purus, efficitur eget neque eu, placerat blandit ligula. Proin nec neque quam.
Teaser :
Video



- **Homepage** : contains the list of the user's games. At first there are the games he haven't finished / actually playing then there are the games he finished and then the games that he have but haven't start playing yet. When you swipe the tile to the left the game is considered as completed. And when you swipe it to the right you go to the page where you can modify the progression.
- **Detail page** : contains information on a game. Here there is the title, the platform, the genre, the progression (if he owns the game), the editor, the release date, a description. (optional a teaser - image or video). Feels free to add other information. At the bottom of the image (if the user owns the game), there is a progress bar which indicate his progression.
- **Discover page**: contains a list of all the games / or the games that the user don't have yet (your choice). Each games are displayed as a card with an image, title, platform and genre.

Widgets

If you already know how to display, position and use the flutter widget or if you want to figure it out yourself, you don't have to read this part . There are just small hint on which widgets to use.

▼ Hints (click here to see)

💡 Hint
For the child to follow his parent round border use ClipRRect
https://api.flutter.dev/flutter/widgets/ClipRRect-class.html

💡 Hint
If you have issue with ListView widget it might be because you need to transform it to a list with .toList()

💡 Hint
For the tiles on the Homepage you can use the Dismissible widget :
https://flutter.dev/docs/cookbook/gestures/dismissible

💡 Hint
For the cards on the Discover page, there is already a Card widget that already exists:
https://api.flutter.dev/flutter/material/Card-class.html

💡 Hint
For the progress bar on the Detail page, you can use the LinearProgressIndicator or create a progress bar component :
https://api.flutter.dev/flutter/material/LinearProgressIndicator-class.html
https://www.youtube.com/watch?v=O-rhXZLtpv0&ab_channel=Flutter

💡 Hint
For the line under the In progress or completed on the Homepage you can use Divider :
https://api.flutter.dev/flutter/material/Divider-class.html

💡 Hint
For the header you can use the AppBar widget. It will be easier later to add icons.
https://api.flutter.dev/flutter/material/AppBar-class.html

Layout

For the Layout, if you find similarities in the layout (for example grid layout) you can create a layout then use it on different screens. For example :

```
/**************************/
/// CENTERED CONTAINER WITH BUTTON
/// content center (vertically and horizontally) with a button at the bottom right part of the page and loader if loading state is tru
/// Parameters :
/// children (List<Widget> - required) - button (FlatButton) - buttonPadding (EdgeInsets)
/**************************/

import 'package:flutter/material.dart';
import 'package:provider/provider.dart';
import 'package:smile_bb/models/global.dart';
import 'package:smile_bb/shared/widgets/Loader.dart';

class CenteredContainerWithButton extends StatelessWidget {

  final List<Widget> children;
  final FlatButton button;
  final EdgeInsets buttonPadding;

  CenteredContainerWithButton({
    @required this.children,
    this.button,
    this.buttonPadding
  });


  @override
  Widget build(BuildContext context) {
    double width = MediaQuery.of(context).size.width;
    double height = MediaQuery.of(context).size.height;

    final global = Provider.of<GlobalModel>(context);

    return Container(
      height: height,
      child: Stack(children: <Widget>[
        Container(
          width: width,
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            crossAxisAlignment: CrossAxisAlignment.center,
            children: children,
          ),
        ),
        Align(
          alignment: Alignment.bottomRight,
          child: buttonPadding != null
          ? Padding(
              padding: buttonPadding,
              child: button,
            )
          : button
        ),
        Consumer<GlobalModel>(
          builder: (context, global, _) {
            return global.isLoading
              ? Positioned.fill(
                  child: Opacity(
                    opacity: 0.5,
                    child: Container(
                      color: Colors.black,
                        child: Center(child: Loader())
                    )
                  )
              )
              : Container();
          },
        )
      ],)
    );
  }
}
```

Components

If you have elements that you might reused multiple time, you can create widgets. Whenever you create your own component be careful if it's a Stateful or a Stateless widget.

Here is an example of a custom widget :

```
/**************************/
/// IMAGE BUTTON
/// A button that contains only an image
/// Parameters :
/// imageWidth (double) - imageHeight (double - required) - margin (double) - imagePath (String - required)
/// buttonText (String) - textColor (Color) - textSize (double) - onTapFunction (Function - required)
/**************************/

import 'package:flutter/material.dart';

class ImageButton extends StatelessWidget {

  final double imageWidth;
  final double imageHeight;
  final double margin;
  final String imagePath;
  final String buttonText;
  final Color textColor;
  final double textSize;
  final Function onTapFunction;

  ImageButton({
    this.imageWidth,
    @required this.imageHeight,
    this.margin,
    @required this.imagePath,
    this.buttonText,
    this.textColor,
    this.textSize,
    @required this.onTapFunction
  });

  @override
  Widget build(BuildContext context) {
    double width = margin != null ? MediaQuery.of(context).size.width - margin : MediaQuery.of(context).size.width;

    return GestureDetector(
      child: Container(
        width: imageWidth ?? width,
        height: imageHeight,
        decoration: BoxDecoration(
          color: Colors.transparent,
          image: DecorationImage(
            image: AssetImage(imagePath),
            fit: BoxFit.contain
          )
        ),
        child: buttonText != null
          ? Text(
              buttonText,
              style: TextStyle(
                color: textColor,
                fontSize: textSize
              )
            )
          : Container()
      ), onTap: onTapFunction
    );
  }
}
```

Global style

For the global style use a **ThemeData** widget that you gonna link to the main **MaterialApp**. For example :

```
ThemeData _buildAppTheme() {
    return ThemeData(
      accentColor: Color(0xff6432f7),
      primaryColor: Color(0xff151c2a),
      buttonColor: Color(0xff6432f7),
      dividerColor: Color(0xffcccccc),
      backgroundColor: Color(0xfff2f2f2),
      appBarTheme: AppBarTheme(color: Colors.blue),
      fontFamily: 'NoteSans',
      iconTheme: IconThemeData(
        color: Colors.white,
      ),
      textTheme: TextTheme(
```

```
        headline: TextStyle(fontWeight: FontWeight.w500),
        title: TextStyle(fontWeight: FontWeight.w500),
        body1: TextStyle(fontSize: 14.0, color: Color(0xff151c2a)),
        body2: TextStyle(fontSize: 16.0, color: Color(0xff151c2a)),
        button: TextStyle(fontSize: 15.0, color: Colors.white),
      ),
    );
  }
}
```

```
MaterialApp(
  title: 'My app',
  theme: _buildAppTheme(),
)
```

The to use it :

```
style: TextStyle(
  color: Theme.of(context).accentColor
),
```

If you have style that you reuse a lot you can create dart file that contains it and import the elements. For example :

```
/**************************/
/// TEXT INPUT DECORATION
/// general decoration for input
/**************************/

import 'package:flutter/material.dart';

const textInputDecoration = InputDecoration(
  fillColor: Colors.white,
  filled: true,
  contentPadding: EdgeInsets.symmetric(horizontal: 10.0),
  border: OutlineInputBorder(
    borderRadius: const BorderRadius.all(
      const Radius.circular(5.0),
    ),
    borderSide: BorderSide(width: 0.0)
  ),
);
```

```
import 'package:smile_bb/shared/style/textInputDecoration.dart';

TextFormField(
  decoration: textInputDecoration.copyWith(hintText: '메일주소를 입력해주세요'),
  ...
)
```

You can also create separate file to get the style. For example :

```
const double defaultPadding = 20;

const Color textColor = Color(0xff333333);
```

Screen issue

Depend on the device you have, you can have some issue. For example on iPhone 11 you have an issue because of the camera on the upper part. So your contains will cut by the camera. For that use SafeArea widget. For more information : https://api.flutter.dev/flutter/widgets/SafeArea-class.html

Use Dummy Data

Example how to use the dummy data :

```
import "./data/dummy_data.dart"

final platformGames = DUMMY_GAMES.where((game) {
  return game.platforms.contains(platformId);
}).toList();


return ListView.builder(itemBuilder: (ctx, index) {
  return Text(platformGames[index].title);
}, itemCount: platformGames.length),
```

Custom icons

For the custom icons, you can either use image (png, jpg, etc) or font. The issue with image is that you have to get good quality images (especially for iPhone, iPad, etc) and if you change the size of it or the color you have to get another one. It can be really heavy. So for this project we are using font. For that you have to use SVG with all the strokes combine in one. Then you upload it on the website of you choice which usually will generate a font for you.

Icomoon

Website: https://icomoon.io/

Some icons that I already prepared and added to the project :

> https://s3-us-west-2.amazonaws.com/secure.notion-static.com/6b489a3a-874d-4212-924e-687b723d8bbf/icomoon_(3).zip

You can see the list of the icons if you open the **demo.html** file. If you are building a website, you have the css that is already prepared in the **style.css** file. For this project what we get is the fonts in the fonts folder. Those fonts are the one that are in the assets/fonts/icomoon folder.

If you want to add other icons, you have to go to the website then go to the Icomoon App. There you create a project and load the project. Once you are on the page, you can click on import icons and select the **selection.json** file. It will load all the icons. From that you can import other icons (SVG), select them and generate the font.

The you will have to replace the font in the project and add the icons in the **helpers/icomoon.dart** file by getting the code.

If you need to display the icon, use the icoMoon class (don't forger to import it). For example :

```
icon: Icon(IconMoon.iback, color: Colors.black87, size: 15.0),
```

Flutter icon

Website: https://www.fluttericon.com/

You can also use this website the generate a font. The dart file with the class will be in the zip that you will get so you will just have to copy/paste.

Tutorial : https://medium.com/flutterpub/how-to-use-custom-icons-in-flutter-834a079d977

Naming convention

Local variable (so for example a variable that is only use in the widget) always start with an underscore : _

They will be only available in the dart file they are defined