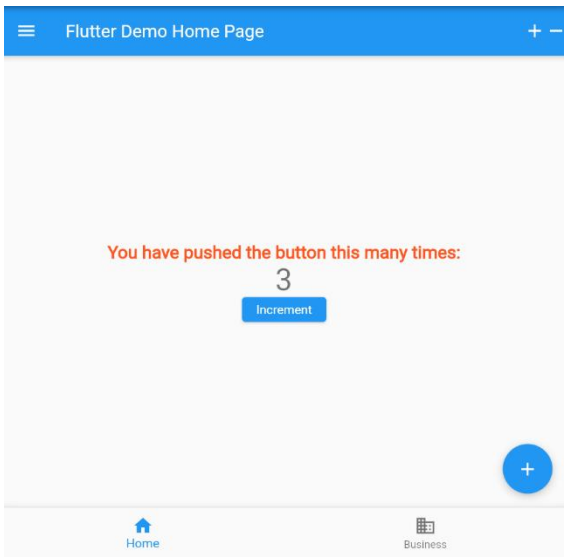
Week3

Flutter Basics, Introduction to Widgets and Styling

Other Topics

- pubspec.yaml has some libraries (open it and click pub get)
- We pass a widget to runApp. If it is a statefull widget, the state is also created, when the time comes build method of the widget is called
- view>tool window>structure you can see the tree structure of the widgets
- or you can use flutter outline (please refresh dart analysis)
- when you use flutter inspector while running the app, it shows the realized tree
- when a screen needs to be refreshed widgets build function is called
- hot reload: ctrl+s or lightning icon: build method of root widget is called
- if you add new objects to the state you have to use hot restart instead of hot reload (which starts the app from the main function)
- if you add new libraries you need to stop and run the app
- flutter inspector helps us: find a widgets code, which widget handles a particular area of the screen, what is the exact boundaries of a widget.
- click toggle select widget mode in the inspector and click an item in the emulator.

Modifying the Default App



- Modify the default flutter app: to add a navigation button to the bottom add these codes return scaffold part:

```
bottomNavigationBar: BottomNavigationBar(
  items:[
    BottomNavigationBarItem(
      icon: Icon(Icons.home),
      label: 'Home',
    ),
```

```
        BottomNavigationBarItem(
          icon: Icon(Icons.business),
          label: 'Business',
        ),
      ]
```

- AppBar: manages the top navigation
    - leading (left menu)
    - title (center)
    - actions (right)

- Modify the appBar part:

```
    appBar: AppBar(
      leading: Icon(Icons.menu),
      title: Text(widget.title),
      actions: [
        Icon(Icons.add),
        Icon(Icons.remove),
      ],
    ),
```

- Text: use whenever you need to type a text
- If you need to change style of a word in a text (such as bold) use RichText

```
const Text(
'You have pushed the button this many times:',
style: TextStyle(
  color: Colors.deepOrange,
  fontSize: 20,
  fontWeight: FontWeight.bold,
),
```

- Button widgets: ElevatedButton, TextButton, IconButton, FloatingActionButton
- Buttons has child and onPressed properties.

```
Text(
  '$_counter',
  style: Theme.of(context).textTheme.headline4,
),
ElevatedButton(
  child: Text("Increment"
  ),
  onPressed: (){
    print("increment pressed");
    _incrementCounter();
  },
```

Building an App from Scratch

- Create an app from scratch:

```
import 'package:flutter/material.dart';
void main(){
 runApp(MyApp());
}
class MyApp extends StatelessWidget{
 @override
  Widget build(BuildContext context){
   return MaterialApp(
     home: Text('Hello'),
   );
  }
}
```

- Each class should be inherited from a stateful or stateless widget class and should have a build method.
- Please note that home is a named parameter, and its type is also widget.
- To add styling and background:
    - add a Scaffold to home parameter (creates a base page design)
    - type ctlr+space inside the scope of Scaffold to see what named parameters it has
    - add appBar (for the title of the page) and body (for the remaining blank space)

```
import 'package:flutter/material.dart';
void main() {
 runApp(MyApp());
}
class MyApp extends StatelessWidget {
 @override
 Widget build(BuildContext context){
   return MaterialApp(
     home: Scaffold(
       appBar: AppBar(
         title: Text("My first app"),
       ), //appBar
       body: Text("This is the default text"),
     ),//scaffold
   );//materialapp
  }
}
```

- To make styling we use invisible widgets in the body such as Row, Column, ListView and Container
- Click Text widget in the body parameter and hit alt+enter then select Wrap with Column.
- Add Elevated button under the text widget of body>column>children
- ElevatedButton(child: Text("Answer 1"), onPressed: null),
- Add an void anwerQuestion() function just over the build method and write the name of this function to onPressed parameters of ElevatedButtons.
- To use anonymous functions: onPressed: () { print("answer 2 is chosen"); }),

- Add a list variable in the build method: var questions=['What is your fav color?','What is your fav animal?'];
- Change the body>column>children>text like Text(questions.elementAt(0)),
- Add this variable in the class a scope: var questionIndex=0; and add this into answerQuestion function: questionIndex++;
- Run the app and click answers. See console window is changing but the question is not.
- We are trying to change the internal state of the widget but our widget is stateless.

```
import 'package:flutter/material.dart';
void main() {
 runApp(MyApp());
}
class MyApp extends StatelessWidget {
 var questionIndex=0;
 void answerQuestion() {
  questionIndex++;
  print("button click $questionIndex");
 }
 @override
 Widget build(BuildContext context) {
  var questions=['What is your fav color?','What is your fav animal?'];
  return MaterialApp(
   home: Scaffold(
    appBar: AppBar(
     title: Text("My first app"),
    ), //appBar
    body: Column(
     children: [
      Text(questions.elementAt(questionIndex)),
      ElevatedButton(child: Text("Answer 1"), onPressed: answerQuestion),
      ElevatedButton(
        child: Text("Answer 2"),
        onPressed: () {
         print("answer 2 is chosen");
        }),
      ElevatedButton(child: Text("Answer 3"), onPressed: answerQuestion),
     ],
    ),
   ), //scaffold
  ); //materialapp
 }
}
```

- What is the difference between stateless and stateful widgets?
- StatelessWidget can’t re-run build() when its internal properties change.
- To convert a stateless widget to statefulwidget you can use quick fix (alt+enter) super-fast. But the long way is like this:
- Change the class inheritance statelesswidget to stateful widget.

- Divide the myapp class into two classes:

```
import 'package:flutter/material.dart';
void main() {
  runApp(MyApp());
}
class MyApp extends StatefulWidget {
  @override
  State<MyApp> createState() => _MyAppState();
}
class _MyAppState extends State<MyApp> {
  var questionIndex=0;
  void answerQuestion() {
    setState(() {
      questionIndex++;
    });
    print("button click $questionIndex");
  }
  @override
  Widget build(BuildContext context) {
    var questions=['What is your fav color?','What is your fav animal?'];
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text("My first app"),
        ), //appBar
        body: Column(
          children: [
            Text(questions.elementAt(questionIndex)),
            ElevatedButton(child: Text("Answer 1"), onPressed: answerQuestion),
            ElevatedButton(
              child: Text("Answer 2"),
              onPressed: () {
                print("answer 2 is chosen");
              }),
            ElevatedButton(child: Text("Answer 3"), onPressed: answerQuestion),
          ],
        ),
      ), //scaffold
    ); //materialapp
  }
}
```

References

- https://www.udemy.com/course/learn-flutter-dart-to-build-ios-android-apps/
- https://developers.google.com/community/experts/directory/profile/profile-gazihan-alankus