F01P. Widgets

Requirements

- We make a Flutter Stateful Widget app:

1. It is a Google-style app (MaterialApp).
2. It displays the "Flutter Demo Home Page" header.
3. It shows the count in the body.
4. It has a button that is clicked to

How to make a Stateful Widget?

1. We should make a state.
2. The state has all the GUI variables.
3. The state is created with createState() in the StatefulWidget.

```
import 'package:flutter/material.dart';

void main() => runApp(MaterialApp(home:MyApp()));
class MyApp extends StatefulWidget {
  const MyApp();
  @override
  State<MyApp> createState() => _MyAppState();
}
class _MyAppState extends State<MyApp> {
   ...
}
```

Scaffold

- We will discuss the Scaffold in the next section.

```
class MyApp extends StatelessWidget {
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(...)
      body: Text(...)
      floatingActionButton: FloatingActionButton(
      ),
    );
  }
}
```

How do you connect the button and the counter widgets?

- Google engineers copied this idea to solve this problem.

1. They separate the variable and state.
2. The GUI is updated only when the state is changed.

- For the counter app, we need a variable to store the count information.

1. This is a GUI variable that should be in the State.
2. Changing the variable changes anything about the GUI.

- We should call `setState()` API to redraw GUI.

1. This is the most important trait of GUI programming in Flutter.
2. This is a great design idea for implementing cross-platform GUI programming.

- We provide `_incrementConter()` service function to call setState().

```
class _MyHomePageState extends State<MyHomePage> {
  int _counter = 0; // variable

  void _incrementCounter() {
    setState(() {_counter++;}); // redraw GUI
  }
}
```

- When the button is clicked, the `_incrementCounter` service function is called to increase the counter and redraw the GUI.

```
Scaffold(
  appBar: AppBar(title: Text(widget.title),),
  body: Text('$_counter'),
  floatingActionButton: FloatingActionButton(
    onPressed: _incrementCounter,
    child: Icon(Icons.add),
  ),
```

Practice 1

- Implement the Counter Stateful Widget using the MaterialApp Scaffold.
  - You can discuss how to implement the application with your team members.
  - You can use LLM to get hints.
  - You can read Counter1.dart if you

Practice 2 (HW)

- This time, you should solve the problem independently without any help.
  - Add one more button (-) to decrease the value when clicked.

Self-grading for HW

- You solved this problem using only Practice 1 (100%).
- You solved this problem with outside help (80%).
- You solved this problem by reading the Practice 2 answer (40%).