ICT4153

Mobile Application Development

Practical 06

Flutter – Widgets

Objective

The objective of this practical sheet is to familiarize learners with essential Flutter widgets, enabling them to build interactive and visually appealing user interfaces

What is a widget in Flutter?

A widget in Flutter is a building block of the user interface. Everything in Flutter is a widget, including structural elements (like Scaffold), layout elements (like Row, Column), and interactive elements (like TextField, Button). Widgets define the visual structure and behavior of an app, and they can be stateless (do not change) or stateful (can change dynamically).

Category of Widgets

There are mainly 14 categories into which the flutter widgets are divided. They are mainly segregated on the basis of the functionality they provide in a flutter application.

Widgets	Description
Accessibility	These are the set of widgets that make a Flutter app more easily accessible.
Animation and Motion	These widgets add animation to other widgets.
Assets, Images, and Icons	These widgets take charge of assets such as display images and show icons.

Widgets	Description
Async	These provide async functionality in the Flutter application.
Basics	These are the bundle of widgets that are absolutely necessary for the development of any Flutter application.
Cupertino	These are the iOS-designed widgets.
Input	This set of widgets provides input functionality in a Flutter application.
Interaction Models	These widgets are here to manage touch events and route users to different views in the application.
Layout	This bundle of widgets helps in placing the other widgets on the screen as needed.
Material Components	This is a set of widgets that mainly follow material design by Google.
Painting and effects	This is the set of widgets that apply visual changes to their child widgets without changing their layout or shape.
Scrolling	This provides scrollability of to a set of other widgets that are not scrollable by default.
Styling	This deals with the theme, responsiveness, and sizing of the app.
Text	This displays text.

Types of Widgets

There are broadly two types of widgets in the flutter:

- Stateless Widget
- Stateful Widget

✓ **Stateless Widget**

A Stateless Widget is a type of widget which once built, then it's properties and state can't be changed. These widgets are immutable and, once created can't be modified.

Note: *These are used for static content or UI content that don't need a change after time.*

Key Characteristics of Stateless Widgets are: ***Immutable , No State and Lightweight.***

Examples: Display Text , Icons, Images, etc.

✓ **Stateful Widget**

Stateful Widgets is a type of widget that can change state. It can maintain and update the appearance in the response to change in state.

Note: *These are used for dynamic change in the properties and appearance over the time.*

Key Characteristics of Stateful Widgets are: ***Mutable State , State Lifecycle and Dynamic Updates.***

Examples: Buttons, Sliders, Text Fields, etc.

1. Stateful vs Stateless Widgets

Concept Explanation:

- **Stateless Widget:** A widget that does not change its state after being built.
- **Stateful Widget:** A widget that can change dynamically based on user interaction.

Exercise:

1. Create a Flutter app with a Stateless widget that displays a text message.
2. Modify the app to use a Stateful widget and add a button that updates a counter when pressed.

Stateless Widget Example:

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        body: Center(child: Text("Hello, Flutter!")),
      ),
    );
  }
}
```

Stateful Widget Example:

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatefulWidget {
  @override
  _MyAppState createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  int counter = 0;
```

```
  void incrementCounter() {
    setState(() {
      counter++;
    });
  }

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Stateful Widget Example")),
        body: Center(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              Text("Counter: \$counter"),
              ElevatedButton(
                onPressed: incrementCounter,
                child: Text("Increment"),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```

2. Container Widget

Concept Explanation:

- The Container widget is a fundamental building block in Flutter, used for styling and layout purposes.
- It allows you to apply **padding, margins, borders, colors, and transformations** to child widgets.
- It helps structure UI components efficiently by wrapping them inside flexible boxes.

Key Properties:

- height & width: Defines the size of the container.
- color: Sets the background color.
- padding & margin: Adds space inside and outside the container.
- alignment: Positions the child widget within the container.
- decoration: Allows styling with borders, gradients, and rounded corners.

Exercise:

1. Create a Container with a width of 200, height of 200, and a red background color.
2. Add a Text widget inside the container and center-align it.
3. Apply padding of 20 pixels and rounded corners with a border radius of 15.

Example Code:

```
Container(
  height: 200,
  width: 200,
  padding: EdgeInsets.all(20),
  decoration: BoxDecoration(
    color: Colors.red,
    borderRadius: BorderRadius.circular(15),
    boxShadow: [
      BoxShadow(
        color: Colors.black26,
        blurRadius: 10,
        spreadRadius: 2,
      ),
    ],
  ),
  alignment: Alignment.center,
  child: Text("Hello", style: TextStyle(color: Colors.white, fontSize: 20)),
)
```

Example code 2:

```
import 'package:flutter/material.dart';
void main() => runApp(const MyApp());
class MyApp extends StatelessWidget {
const MyApp({Key? key}) : super(key: key);
@override
Widget build(BuildContext context) {
        return MaterialApp(
        home: Scaffold(
                appBar: AppBar(
                title: const Text("Container example"),
                ),
                body: Container(
                height: 200,
                width: double.infinity,
                color: Colors.purple,
                alignment: Alignment.center,
                margin: const EdgeInsets.all(20),
                padding: const EdgeInsets.all(30),
                transform: Matrix4.rotationZ(0.1),
                child: const Text("Hello! i am inside a container!",
                        style: TextStyle(fontSize: 20)),
                ),
        ),
        );
}
}
```

3. Scaffold Widget

Concept Explanation:

- The Scaffold widget provides a basic layout structure for a Flutter app.
- It supports an **AppBar, Drawer, BottomNavigationBar, FloatingActionButton, and a body section**.
- It helps in building consistent UI structures efficiently.

Key Properties:

- appBar: Displays a header at the top.
- body: Main content of the page.
- floatingActionButton: A button that floats above content.
- drawer: A navigation drawer.
- bottomNavigationBar: Navigation buttons at the bottom.

Exercise:

1. Create a Scaffold with an AppBar containing a title.
2. Add a FloatingActionButton that displays a snackbar when clicked.
3. Implement a BottomNavigationBar to switch between two pages.
4. Add a Drawer with navigation options.

Example code

```
import 'package:flutter/material.dart';
void main() {
  runApp(MyApp());
}
class MyApp extends StatefulWidget {
  @override
  _MyAppState createState() => _MyAppState();
}
class _MyAppState extends State<MyApp> {
  int _selectedIndex = 0;
  void _onItemTapped(int index) {
    setState(() {
      _selectedIndex = index;
    });
  }
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Scaffold Example")),
        body: Center(child: Text("Page \${_selectedIndex + 1}")),
        floatingActionButton: FloatingActionButton(
          onPressed: () {
            ScaffoldMessenger.of(context).showSnackBar(
              SnackBar(content: Text("FAB Clicked!")),
            );
          },
          child: Icon(Icons.add),
        ),
        bottomNavigationBar: BottomNavigationBar(
          items: [
            BottomNavigationBarItem(icon: Icon(Icons.home), label: "Home"),
```

```
          BottomNavigationBarItem(icon: Icon(Icons.settings), label: "Settings"),
        ],
        currentIndex: _selectedIndex,
        onTap: _onItemTapped,
      ),
      drawer: Drawer(
        child: ListView(
          children: [
            DrawerHeader(
              decoration: BoxDecoration(color: Colors.blue),
              child: Text("Menu", style: TextStyle(color: Colors.white)),
            ),
            ListTile(
              leading: Icon(Icons.home),
              title: Text("Home"),
              onTap: () {
                Navigator.pop(context);
              },
            ),
            ListTile(
              leading: Icon(Icons.settings),
              title: Text("Settings"),
              onTap: () {
                Navigator.pop(context);
              },
            ),
          ],
        ),
      ),
    ),
  );
 }
}
```

4. MaterialApp Widget

MaterialApp Class: MaterialApp is a predefined class or widget in a flutter. It is likely the main or core component of a flutter app. The MaterialApp widget provides a wrapper around other Material Widgets. We can access all the other components and widgets provided by Flutter SDK. Text
widget, DropdownButton widget, AppBar widget, Scaffold widget, ListView widget, StatelessWidget, StatefulWidget, IconButton widget, TextField widget, Padding widget, ThemeData widget, etc. are the widgets that can be accessed using MaterialApp class. There are many more widgets that are accessed using MaterialApp class. Using this widget, we can make an attractive app that follows the Material Design guidelines.

Example code:

```
import 'package:flutter/material.dart';
void main() {
 runApp(MyApp());
}
class MyApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   title: 'Flutter Demo',
   theme: ThemeData(
    primarySwatch: Colors.blue,
   ),
   home: Scaffold(
    appBar: AppBar(title: Text("MaterialApp Example")),
    body: Center(child: Text("Welcome to MaterialApp")),
   ),
  );
 }
}
```

5. BottomNavigationBar Widget

Example code:

```
import 'package:flutter/material.dart';
void main() {
  runApp(MyApp());
}
class MyApp extends StatefulWidget {
  @override
  _MyAppState createState() => _MyAppState();
}
class _MyAppState extends State<MyApp> {
  int _selectedIndex = 0;
  static List<Widget> _widgetOptions = <Widget>[
    Text('Home Screen'),
    Text('Settings Screen'),
  ];
  void _onItemTapped(int index) {
    setState(() {
      _selectedIndex = index;
    });
  }
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Bottom Navigation Example")),
        body: Center(child: _widgetOptions.elementAt(_selectedIndex)),
        bottomNavigationBar: BottomNavigationBar(
          items: const <BottomNavigationBarItem>[
```

```
          BottomNavigationBarItem(icon: Icon(Icons.home), label: 'Home'),
          BottomNavigationBarItem(icon: Icon(Icons.settings), label: 'Settings'),
        ],
        currentIndex: _selectedIndex,
        onTap: _onItemTapped,
      ),
    ),
  );
 }
}
```

6. ClipRRect Widget

```dart
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("ClipRRect Example")),
        body: Center(
          child: ClipRRect(
            borderRadius: BorderRadius.circular(20),
            child: Image.network('https://via.placeholder.com/150'),
          ),
        ),
      ),
    );
  }
}
```

7. Drawer Widget

```dart
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("Drawer Example")),
        drawer: Drawer(
          child: ListView(
            children: [
              DrawerHeader(
                decoration: BoxDecoration(color: Colors.blue),
                child: Text("Menu", style: TextStyle(color: Colors.white)),
              ),
              ListTile(
                title: Text("Home"),
                onTap: () {},
              ),
              ListTile(
                title: Text("Settings"),
                onTap: () {},
              ),
            ],
          ),
        ),
        body: Center(child: Text("Drawer Example")),
      ),
    );
  }
}
```

8. Opacity Widget

```dart
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("Opacity Example")),
        body: Center(
          child: Opacity(
            opacity: 0.5,
            child: Text("Transparent Text"),
          ),
        ),
      ),
    );
  }
}
```

9. RotatedBox Widget

```dart
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("RotatedBox Example")),
        body: Center(
          child: RotatedBox(
            quarterTurns: 1,
            child: Text("Rotated Text"),
          ),
        ),
      ),
    );
  }
}
```

10. RichText Widget

```
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("RichText Example")),
        body: Center(
          child: RichText(
            text: TextSpan(
              style: TextStyle(fontSize: 24, color: Colors.black),
              children: <TextSpan>[
                TextSpan(text: "Bold Text", style: TextStyle(fontWeight: FontWeight.bold)),
                TextSpan(text: " Normal Text"),
              ],
            ),
          ),
        ),
      ),
    );
  }
}
```

11. OctoImage Widget

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("OctoImage Example")),
        body: Center(
          child: OctoImage(
            image: NetworkImage('https://via.placeholder.com/150'),
            placeholderBuilder:
OctoPlaceholder.blurHash('LEHV6nWB2yk8pyo0adR*.7kCMdnj'),
            errorBuilder: OctoError.icon(color: Colors.red),
          ),
        ),
      ),
    );
  }
}
```

12. AppBar Widget

```dart
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("AppBar Example"),
          actions: [
            IconButton(icon: Icon(Icons.search), onPressed: () {}),
          ],
        ),
        body: Center(child: Text("AppBar Example")),
      ),
    );
  }
}
```

- **Find more about the above-given widgets.**
- **Practice all the above codes and upload your output as screenshots attached in a word document to the given link in LMS.**