TUGAS PENDAHULUAN 13
PEMROGRAMAN PERANGKAT BERGERAK

MODUL XIII
NETWORKING

DISUSUN OLEH:
KHOLIFAHDINA
2211104004

SE 06 01

S1 REKAYASA PERANGKAT LUNAK
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
2024

1. Apa yang dimaksud dengan state management pada Flutter?

   State management dalam Flutter adalah pendekatan untuk mengelola dan memanipulasi state (data atau informasi aplikasi yang berubah-ubah) agar aplikasi dapat merespons interaksi pengguna atau perubahan data secara real-time.

2. Sebut dan jelaskan komponen-komponen yang ada di dalam GetX.

   a. **State Management**

   Mengelola data secara reaktif menggunakan **Obx** dan **Rx variables**:

```
var count = 0.obs; // Reactive state
count.value++;     // Update state
```

   b. **Dependency Injection (DI)**

   Mengatur dependency dengan mudah menggunakan **Get.put()** atau **Get.find()**.

```
Get.put<MyController>(MyController());
var controller = Get.find<MyController>();
```

   c. **Route Management**

   Navigasi antar halaman tanpa boilerplate:

```
Get.to(NextPage());
Get.back();
```

   d. **Controller**

   Class untuk logika bisnis, menggunakan **GetxController**.

```
class MyController extends GetxController {
  var count = 0.obs;
  void increment() => count++;
}
```

   e. **Bindings**

   Mengatur dependency untuk halaman tertentu.

   f. **Middleware**

   Mengontrol navigasi, seperti cek otentikasi.

   g. **Snackbar, Dialog, BottomSheet**

   Menampilkan notifikasi kepada pengguna:

```
Get.snackbar('Title', 'Message');
Get.defaultDialog(title: 'Title');
```

   h. **Worker**

   Memantau perubahan state dan menjalankan aksi.

```
ever(controller.count, (value) => print(value));
```

3. Lengkapilah code di bawah ini, dan tampilkan hasil outputnya serta jelaskan.

- file main.dart

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

/// Controller untuk mengelola state counter
class CounterController extends GetxController {
  // Variabel untuk menyimpan nilai counter
  var counter = 0.obs;

  // Fungsi untuk menambah nilai counter
  void increment() {
    counter++;
  }

  // Fungsi untuk mereset nilai counter
  void reset() {
    counter.value = 0;
  }
}

class HomePage extends StatelessWidget {
  final CounterController controller = Get.put(CounterController());

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Counter App")),
      body: Center(
        child: Obx(() {
          // Menampilkan nilai counter
          return Text(
            "${controller.counter}",
            style: TextStyle(fontSize: 48),
          );
        }),
      ),
      floatingActionButton: Column(
        mainAxisAlignment: MainAxisAlignment.end,
        children: [
          FloatingActionButton(
            onPressed: () {
```

```
              // Menambah nilai counter
              controller.increment();
            },
            child: Icon(Icons.add),
          ),
          SizedBox(height: 10),
          FloatingActionButton(
            onPressed: () {
              // Mereset nilai counter
              controller.reset();
            },
            child: Icon(Icons.refresh),
          ),
        ],
      ),
    );
  }
}

void main() {
  runApp(MaterialApp(
    debugShowCheckedModeBanner: false,
    home: HomePage(),
  ));
}
```

- file main_test.dart

```
import 'package:cobaa/main.dart';
import 'package:flutter/material.dart';
import 'package:flutter_test/flutter_test.dart';

void main() {
  testWidgets('Counter increments smoke test', (WidgetTester tester)
async {
    // Build aplikasi dan render frame pertama
    await tester.pumpWidget(MaterialApp(home: HomePage()));

    // Verifikasi bahwa counter dimulai dari 0
    expect(find.text('0'), findsOneWidget);
    expect(find.text('1'), findsNothing);
```

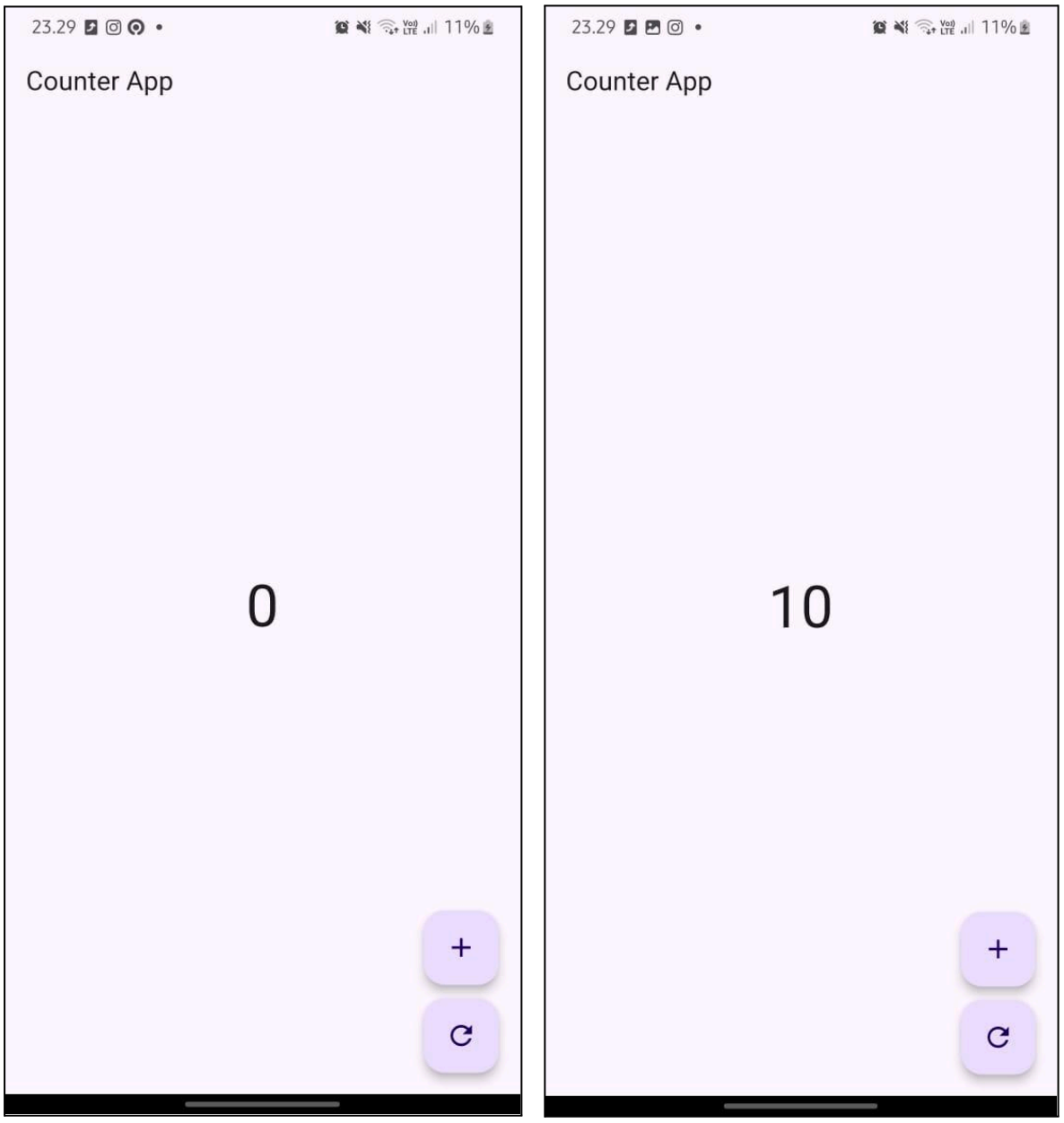
```
    // Tekan ikon "+" dan render frame berikutnya
    await tester.tap(find.byIcon(Icons.add));
    await tester.pump();

    // Verifikasi bahwa counter telah bertambah
    expect(find.text('0'), findsNothing);
    expect(find.text('1'), findsOneWidget);

    // Tekan ikon "refresh" dan render frame berikutnya
    await tester.tap(find.byIcon(Icons.refresh));
    await tester.pump();

    // Verifikasi bahwa counter telah direset ke 0
    expect(find.text('0'), findsOneWidget);
    expect(find.text('1'), findsNothing);
  });
}
```

Output:



Program ini adalah aplikasi sederhana berbasis **Flutter** yang menggunakan **GetX** untuk mengelola state (nilai) counter. Program ini memiliki fitur utama untuk menambah nilai counter secara real-time saat tombol "+" ditekan, serta mereset nilai counter kembali ke nol menggunakan tombol "refresh". Algoritma utama bekerja dengan memanfaatkan **Reaktif State Management** dari GetX, di mana nilai counter disimpan dalam variabel `RxInt` (reaktif integer) dan otomatis memperbarui tampilan (UI) setiap kali nilainya berubah. Aplikasi ini memanfaatkan widget `Obx` untuk memantau perubahan nilai counter secara reaktif dan memperbarui teks di layar. Output yang dihasilkan adalah tampilan nilai counter di tengah layar yang terus diperbarui secara real-time sesuai interaksi pengguna.