LAPORAN MOBILE & WEB SERVICE PRAKTIK

Laporan ini untuk memenuhi tugas matakuliah Mobile & Web Service Praktik Pertemuan 2

Nama : I Nyoman Darmayoga
NPM : 5220411265

PROGRAM STUDI INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS TEKNOLOGI YOGYAKARTA

2024

Github: https://github.com/darmayo/Mobile---Web-Service-Praktik.git

Pengertian Flutter Widget/Component

Flutter Widget adalah elemen grafis atau komponen dasar yang digunakan untuk membuat pengguna (User Interface) di aplikasi Flutter. Segala sesuatu yang terlihat pada layar aplikasi Flutter adalah widget. Misalnya, tombol, teks, gambar, dan bahkan struktur tata letak seperti grid dan list semuanya adalah widget.

Widget pada Flutter bisa statis (tidak berubah) atau dinamis (berubah seiring waktu atau input pengguna). Berdasarkan hal ini, widget dibagi menjadi dua kategori utama:

1. Stateless Widget: Widget yang tidak berubah selama aplikasi berjalan.
2. Stateful Widget: Widget yang dapat berubah saat aplikasi berjalan, bergantung pada interaksi pengguna atau perubahan data.

Jenis-Jenis Widget

Berikut adalah jenis-jenis Widget, yaitu:

a. **MaterialApp**

MaterialApp adalah widget utama yang digunakan untuk memulai aplikasi Flutter. Widget ini mengatur tampilan aplikasi berdasarkan desain material dan menyediakan fitur penting seperti navigasi antar halaman (rute), pengaturan tema, serta aksesibilitas. MaterialApp juga digunakan untuk seluruh struktur widget dalam aplikasi, sehingga sangat penting untuk mengatur tampilan dan fungsionalitas aplikasi secara menyeluruh.

b. **Scaffold**

Scafffold ini adalah widget tata letak yang menyediakan kerangka kerja untuk membangun desain visual berbasis material. Widget ini mencakup fitur seperti bilah aplikasi, laci navigasi, bilah navigasi bawah, dan tombol aksi mengambang, yang memudahkan pembuatan antarmuka pengguna (UI) yang konsisten dan menarik secara visual.

c. **Container**

Widget Container adalah widget yang fleksibel dan kuat untuk mengatur tata letak dan gaya. Widget ini dapat menampung beberapa widget di dalamnya serta memungkinkan pengaturan properti seperti padding, margin, dekorasi, dan perataan. Container sering digunakan untuk membuat tata letak yang responsif dan elemen UI yang disesuaikan.

d. **Row and Column**

Row dan Column adalah widget tata letak yang digunakan untuk menyusun widget secara horizontal (Row) dan vertikal (Column). Widget ini fleksibel dan responsif, sehingga memudahkan pembuatan tata letak UI yang lebih kompleks. Anda juga bisa menggabungkan Row dan Column untuk menciptakan desain yang lebih rumit.

e. **ListView**

ListView adalah daftar widget yang bisa digulir secara vertikal atau horizontal. Biasanya digunakan untuk menampilkan banyak data atau konten yang dibuat secara dinamis. ListView memiliki berbagai opsi penyesuaian, seperti arah gulir, jarak antar item, dan pengaturan perilaku gulir.

f. **GestureDetector**

GestureDetector adalah widget yang sangat fleksibel yang berfungsi untuk mendeteksi berbagai gerakan, seperti ketukan, seret, serta gerakan memperkecil dan memperbesar. Widget ini membungkus widget lainnya dan menyediakan fungsi callback untuk menangani berbagai jenis gerakan, sehingga memudahkan Anda menambahkan interaktivitas ke dalam aplikasi.

g. **TextField**

TextField adalah widget yang digunakan untuk menerima input teks dari pengguna. Widget ini menawarkan berbagai opsi untuk menyesuaikan penampilan, memvalidasi input, dan memformat teks yang dimasukkan. TextField sangat penting untuk membuat formulir dan antarmuka input pengguna dalam aplikasi Flutter.

h. **Image**

Image widget ini digunakan untuk menampilkan gambar dalam aplikasi Flutter. Widget ini mendukung berbagai format gambar dan menawarkan opsi untuk mengubah ukuran, memotong, dan menyimpan gambar dalam cache. Gambar dapat dimuat dari sumber lokal, URL jaringan, atau dari memori.

i. **RaiseButton and FlatButton**

RaisedButton dan FlatButton adalah widget yang digunakan untuk membuat tombol dengan tampilan yang berbeda. Widget ini memungkinkan Anda menyesuaikan elemen seperti teks, warna, dan fungsi yang dipanggil saat tombol ditekan. Tombol ini sangat penting untuk interaksi dan navigasi pengguna dalam aplikasi Flutter.

j. **FutureBuilder**

FutureBuilder adalah widget yang digunakan untuk memuat dan menampilkan data secara asinkron dalam aplikasi Flutter. Widget ini memungkinkan Anda menjalankan tugas di latar belakang dan memperbarui antarmuka pengguna (UI) setelah data siap. FutureBuilder membuat penanganan operasi asinkron lebih mudah dan menyediakan cara yang sederhana untuk mengelola status pemuatan.

Cara Instal Widget

Berikut langkah demi langkah untuk menginstal Flutter dan mulai menggunakan widget.

1. Install Flutter SDK

- Download Flutter SDK dari situs resminya: Flutter SDK.
- Ekstrak file zip yang sudah diunduh dan simpan di lokasi yang diinginkan di komputermu.
- Tambahkan flutter/bin ke dalam PATH environment variables di sistemmu.

2. Cek Instalasi Flutter

- Buka terminal (Command Prompt di Windows) dan jalankan perintah: **flutter doctor**
- Flutter akan memeriksa apakah ada dependensi yang belum terpasang, seperti Android Studio, Xcode (untuk Mac), dan SDK lainnya yang diperlukan.

3. Install Editor

- Kita bisa menggunakan **VS Code**, **Android Studio**, atau **IntelliJ IDEA** untuk pengembangan Flutter. Pastikan sudah memasang **Flutter plugin** di editor pilihanmu.
- Di **VS Code**, buka menu Extensions dan cari **Flutter** lalu klik install.

4. Install Android SDK (Opsional jika belum terpasang)

- Jika Kita menggunakan Android Studio, pastikan Android SDK sudah terpasang dengan membuka **SDK Manager** dari Android Studio.
- Install SDK versi terbaru dan pastikan semua **tools** seperti **Android Virtual Device (AVD)** sudah terpasang.

5. Tambahkan Flutter Widget

- Buka file lib/main.dart di proyekmu.
- Kita bisa mulai menambahkan widget dengan mengganti kode bawaan dengan yang Kita butuhkan. Berikut adalah code yang telah saya buat sebagai contoh:

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

void main() {
  runApp(MaterialApp(
    title: 'My Flutter App',
    theme: ThemeData(
      primarySwatch: Colors.blue,
      visualDensity: VisualDensity.adaptivePlatformDensity,
    ),
    home: MyApp(),
  ));
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('My App'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          children: <Widget>[
            Container(
              padding: EdgeInsets.all(16.0),
              margin: EdgeInsets.symmetric(vertical: 16.0),
              decoration: BoxDecoration(
                color: Colors.blue,
                borderRadius: BorderRadius.circular(8.0),
              ),
              child: Text(
                'Container Widget',
                style: TextStyle(color: Colors.white, fontSize: 18),
              ),
            ),
            Row(
              mainAxisAlignment: MainAxisAlignment.spaceAround,
              children: <Widget>[
                Icon(Icons.star, color: Colors.blue),
                Text('Row Widget', style: TextStyle(fontSize: 16)),
              ],
            ),
            SizedBox(height: 20),
            Column(
              children: <Widget>[
                Text('Column Widget', style: TextStyle(fontSize: 16)),
                Icon(Icons.star, color: Colors.blue),
              ],
            ),
            SizedBox(height: 20),
            Expanded(
              child: ListView.builder(
                itemCount: 10, // Adjust based on your data
                itemBuilder: (BuildContext context, int index) {
                  return ListTile(
                    title: Text('Item ${index + 1}'),
                  );
                },
              ),
            ),
            GestureDetector(
```

```dart
            GestureDetector(
              onTap: () {
                // Handle tap gesture
                ScaffoldMessenger.of(context).showSnackBar(
                  SnackBar(content: Text('Tapped!')),
                );
              },
              child: Container(
                color: Colors.blue,
                width: 100.0,
                height: 100.0,
                child: Center(
                  child: Text(
                    'Tap me',
                    style: TextStyle(color: Colors.white),
                  ),
                ),
              ),
            ),
            SizedBox(height: 20),
            TextField(
              decoration: InputDecoration(
                labelText: 'Enter your name',
                border: OutlineInputBorder(),
              ),
            ),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: () {
                // Handle ElevatedButton press
              },
              child: Text('Elevated Button'),
            ),
            SizedBox(height: 10),
            TextButton(
              onPressed: () {
                // Handle TextButton press
              },
              child: Text('Text Button'),
            ),
            SizedBox(height: 20),
            FutureBuilder<String>(
              future: fetchData(),
              builder: (BuildContext context, AsyncSnapshot<String> snapshot)
{
                if (snapshot.connectionState == ConnectionState.waiting) {
                  return CircularProgressIndicator();
                } else if (snapshot.hasError) {
                  return Text('Error: ${snapshot.error}');
                } else {
                  return Text('Data: ${snapshot.data}');
                }
              },
            ),
          ],
        ),
      ),
    );
  }

  Future<String> fetchData() async {
    await Future.delayed(Duration(seconds: 2)); // Simulate network delay
    return 'Hello from Future!';
  }
}
```

9. Build dan Jalankan

- Jalankan aplikasi Flutter di perangkat dengan mengetikkan: **flutter run**
- Kita juga bisa menjalankannya langsung dari editor dengan klik tombol "Play" atau "Run".
- Hasil run code yang telah saya buat adalah sebagai berikut: