Aplikasi Pertama dan Widget Dasar Flutter

Pertemuan 5

Nama: Muhammad Taufiq Abdus Salam

Kelas: SIB 2D

**Langkah-Langkah Praktikum:**

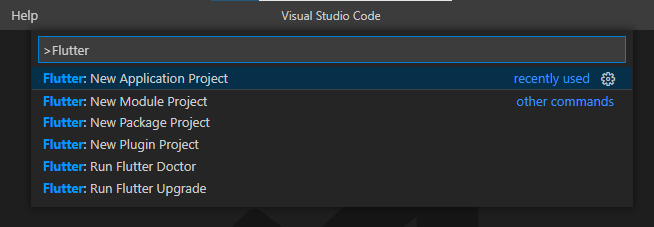
## Praktikum 1: Membuat Project Flutter Baru

Selesaikan langkah-langkah praktikum berikut ini menggunakan editor Visual Studio Code (VS Code) atau Android Studio atau code editor lain kesukaan Anda.

**Perhatian:** Diasumsikan Anda telah berhasil melakukan setup environment Flutter SDK, VS Code, Flutter Plugin, dan Android/iOS SDK pada [Codelabs#1](https://jti-polinema.github.io/flutter-codelab/01-pengenalan/).

## ****Langkah 1:****

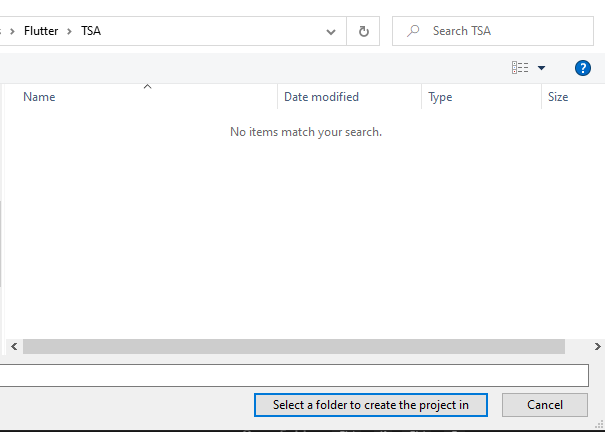
Buka VS Code, lalu tekan tombol **Ctrl + Shift + P** maka akan tampil Command Palette, lalu ketik **Flutter**. Pilih **New Application Project**.



**Catatan:** Anda dapat mengakses **Command Palette** dengan cara lain, yaitu pilih menu **View > Command Palette**. Tombol shortcut **Ctrl + Shift + P** mungkin hanya berlaku di Windows.

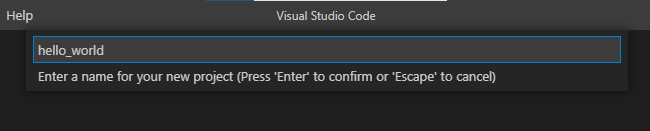
## ****Langkah 2:****

Kemudian buat folder **sesuai style** laporan praktikum yang Anda pilih. Disarankan pada folder dokumen atau desktop atau alamat folder lain yang tidak terlalu dalam atau panjang. Lalu pilih **Select a folder to create the project in**.



## ****Langkah 3:****

Buat nama project flutter **hello\_world** seperti berikut, lalu tekan **Enter**. Tunggu hingga proses pembuatan project baru selesai.



**Perhatian:** Nama project ini harus lowercase (huruf kecil semua) tanpa menggunakan spasi. Untuk memisahkan kata, bisa menggunakan underline (garis bawah). Nama project tidak dapat diawali dengan angka atau karakter khusus lain. Nama project ini bukan nama aplikasi yang akan tampil di Play Store atau App Store. Untuk nama aplikasi, nanti dapat diatur ketika melakukan deployment.

## ****Langkah 4:****

Jika sudah selesai proses pembuatan project baru, pastikan tampilan seperti berikut. Pesan akan tampil berupa "**Your Flutter Project is ready!**" artinya Anda telah berhasil membuat project Flutter baru.

## Praktikum 2: Menghubungkan Perangkat Android atau Emulator

Melanjutkan dari praktikum 1, Anda diminta untuk menjalankan aplikasi ke perangkat fisik (device Android atau iOS). Silakan ikuti langkah-langkah pada codelab tautan berikut ini.

<https://developer.android.com/codelabs/basic-android-kotlin-compose-connect-device?hl=id#0>

Praktikum 3: Membuat Repository GitHub dan Laporan Praktikum

Melanjutkan dari praktikum 2, silakan selesaikan langkah-langkah berikut ini.

**Perhatian:** Diasumsikan Anda telah mempunyai akun GitHub dan Anda telah memahami konsep dasar dalam bekerja menggunakan Git pada pertemuan pertama.

Praktikum ini dapat Anda **lewati langsung ke langkah 11**jika sudah paham cara membuat laporan praktikum sesuai *style*yang Anda pilih.

**Langkah 1:**

Login ke akun [GitHub](https://github.com/new" \t "_blank) Anda, lalu buat repository baru dengan nama "**flutter-fundamental-part1**"

**Langkah 2:**

Lalu klik tombol "**Create repository**" lalu akan tampil seperti gambar berikut.

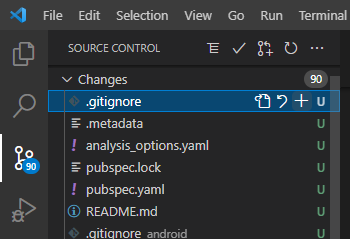
**Langkah 3:**

Kembali ke VS code, project flutter hello\_world, buka terminal pada menu **Terminal > New Terminal**. Lalu ketik perintah berikut untuk inisialisasi git pada project Anda.

git init

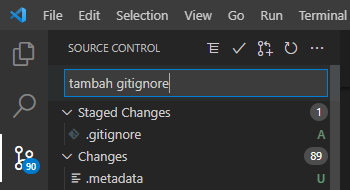
**Langkah 4:**

Pilih menu **Source Control** di bagian kiri, lalu lakukan **stages** (+) pada file **.gitignore** untuk mengunggah file pertama ke repository GitHub.



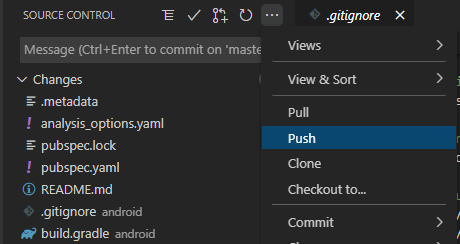
**Langkah 5:**

Beri pesan commit "**tambah gitignore**" lalu klik **Commit** (✔)



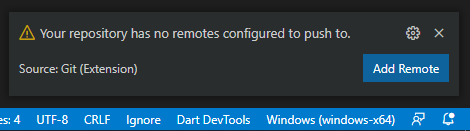
**Langkah 6:**

Lakukan push dengan klik bagian menu titik tiga > **Push**



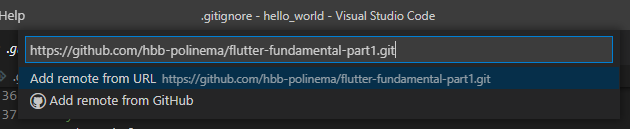
**Langkah 7:**

Di pojok kanan bawah akan tampil seperti gambar berikut. Klik "**Add Remote**"

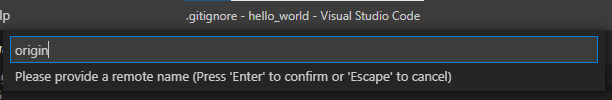


**Langkah 8:**

Salin tautan repository Anda dari browser ke bagian ini, lalu klik **Add remote**



Setelah berhasil, tulis remote name dengan "**origin**"



**Langkah 9:**

Lakukan hal yang sama pada file **README.md** mulai dari Langkah 4. Setelah berhasil melakukan push, masukkan username GitHub Anda dan password berupa token yang telah dibuat (pengganti password konvensional ketika Anda login di browser GitHub). Reload halaman repository GitHub Anda, maka akan tampil hasil push kedua file tersebut seperti gambar berikut.

**Perhatian:** Personal access token GitHub dapat Anda buat melalui <https://github.com/settings/tokens/new> atau dapat membaca dokumentasi [GitHub Token](https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token" \t "_blank).

**Langkah 10:**

Lakukan push juga untuk semua file lainnya dengan pilih **Stage All Changes**. Beri pesan commit "**project hello\_world**". Maka akan tampil di repository GitHub Anda seperti berikut.

**Langkah 11:**

Kembali ke VS Code, ubah platform di pojok kanan bawah ke emulator atau device atau bisa juga menggunakan browser Chrome. Lalu coba running project **hello\_world** dengan tekan **F5** atau **Run > Start Debugging**. Tunggu proses kompilasi hingga selesai, maka aplikasi flutter pertama Anda akan tampil seperti berikut.

**Perhatian:** Proses Run atau kompilasi untuk pertama kali akan memakan waktu cukup lama (3-5 menit). Namun, proses kompilasi berikutnya menjadi lebih cepat. Berbeda ketika melakukan kompilasi ke device, hal ini akan memakan waktu lebih lama lagi.

**Langkah 12:**

Silakan screenshot seperti pada Langkah 11, namun teks yang ditampilkan dalam aplikasi berupa nama lengkap Anda. Simpan file screenshot dengan nama **01.png** pada folder **images**(buat folder baru jika belum ada) di project hello\_world Anda. Lalu ubah isi README.md seperti berikut, sehingga tampil hasil screenshot pada file README.md. Kemudian push ke repository Anda.

**Perhatian:** Lakukan proses screenshot seperti pada Langkah 12 untuk setiap Laporan Praktikum yang Anda akan buat pada praktikum selanjutnya hingga pertemuan project final.

## Praktikum 4: Menerapkan Widget Dasar

Selesaikan langkah-langkah praktikum berikut ini dengan melanjutkan dari praktikum sebelumnya.

## ****Langkah 1: Text Widget****

Buat folder baru **basic\_widgets** di dalam folder **lib**. Kemudian buat file baru di dalam basic\_widgets dengan nama text\_widget.dart. Ketik atau salin kode program berikut ke project hello\_world Anda pada file text\_widget.dart.

import 'package:flutter/material.dart';

class MyTextWidget extends StatelessWidget {

  const MyTextWidget({Key? key}) : super(key: key);

  @override

  Widget build(BuildContext context) {

    return const Text(

      "Nama saya Muhammad Taufiq Abdus Salam, sedang belajar Pemrograman Mobile",

      style: TextStyle(color: Colors.red, fontSize: 14),

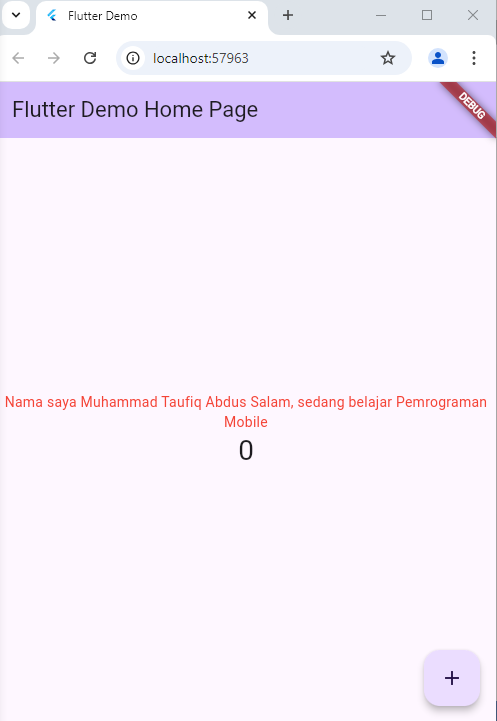
      textAlign: TextAlign.center);

  }

}

**Perhatian:** Gantilah teks **Fulan** dengan nama lengkap Anda.

Lakukan import file text\_widget.dart ke main.dart, lalu ganti bagian text widget dengan kode di atas. Maka hasilnya seperti gambar berikut. Screenshot hasil milik Anda, lalu dibuat laporan pada file README.md.



## ****Langkah 2: Image Widget****

Buat sebuah file image\_widget.dart di dalam folder basic\_widgets dengan isi kode berikut.

import 'package:flutter/material.dart';

class MyImageWidget extends StatelessWidget {

  const MyImageWidget({Key? key}) : super(key: key);

  @override

  Widget build(BuildContext context) {

    return const Image(

      image: AssetImage("logo\_polinema..jpg")

    );

  }

}

Lakukan penyesuaian asset pada file pubspec.yaml dan tambahkan file logo Anda di folder assets project hello\_world.

Jangan lupa sesuaikan kode dan import di file main.dart kemudian akan tampil gambar seperti berikut.



Praktikum 5: Menerapkan Widget Material Design dan iOS Cupertino

Selesaikan langkah-langkah praktikum berikut ini dengan melanjutkan project hello\_world Anda. Lakukan langkah yang sama seperti pada Praktikum 3, yaitu setiap widget dibuat file sendiri lalu import ke main.dart dan screenshot hasilnya.

**Langkah 1: Cupertino Button dan Loading Bar**

Buat file di basic\_widgets > loading\_cupertino.dart. Import stateless widget dari material dan cupertino. Lalu isi kode di dalam method Widget build adalah sebagai berikut.

import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

class MyLoadingCupertino extends StatelessWidget {

  const MyLoadingCupertino ({Key? key}) : super(key: key);

  @override

  Widget build(BuildContext context) {

    return MaterialApp(

      home: Container(

        margin: const EdgeInsets.only(top: 30),

        color: Colors.white,

        child: Column(

          children: <Widget>[

            CupertinoButton(

              child: const Text("Contoh button"),

              onPressed: () {},

            ),

            const CupertinoActivityIndicator(),

          ],

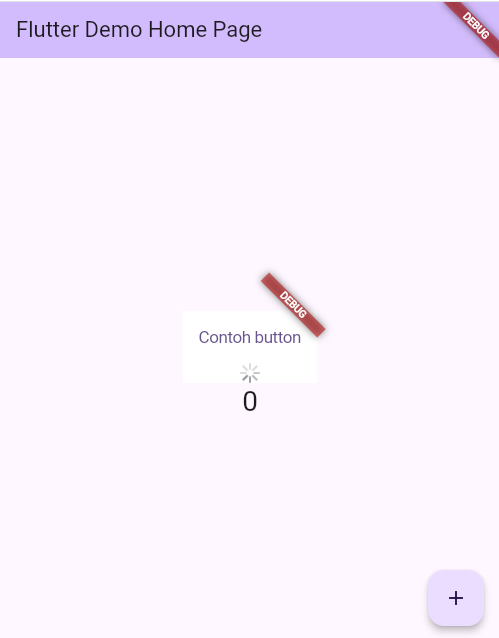
        ),

      ),

    );

  }

}



**Langkah 2: Floating Action Button (FAB)**

Button widget terdapat beberapa macam pada flutter yaitu ButtonBar, DropdownButton, TextButton, FloatingActionButton, IconButton, OutlineButton, PopupMenuButton, dan ElevatedButton.

Buat file di basic\_widgets > fab\_widget.dart. Import stateless widget dari material. Lalu isi kode di dalam method Widget build adalah sebagai berikut.

return MaterialApp(

home: Scaffold(

floatingActionButton: FloatingActionButton(

onPressed: () {

*// Add your onPressed code here!*

},

child: const Icon(Icons.thumb\_up),

backgroundColor: Colors.pink,

),

),

);

**Langkah 3: Scaffold Widget**

Scaffold widget digunakan untuk mengatur tata letak sesuai dengan material design.

Ubah isi kode main.dart seperti berikut.

import 'package:flutter/material.dart';

void main() {

runApp(const MyApp());

}

class MyApp extends StatelessWidget {

const MyApp({Key? key}) : super(key: key);

*// This widget is the root of your application.*

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Flutter Demo',

theme: ThemeData(

primarySwatch: Colors.red,

),

home: const MyHomePage(title: 'My Increment App'),

);

}

}

class MyHomePage extends StatefulWidget {

const MyHomePage({Key? key, required this.title}) : super(key: key);

final String title;

@override

State<MyHomePage> createState() => \_MyHomePageState();

}

class \_MyHomePageState extends State<MyHomePage> {

int \_counter = 0;

void \_incrementCounter() {

setState(() {

\_counter++;

});

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text(widget.title),

),

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

const Text(

'You have pushed the button this many times:',

),

Text(

'$\_counter',

style: Theme.of(context).textTheme.headline4,

),

],

),

),

bottomNavigationBar: BottomAppBar(

child: Container(

height: 50.0,

),

),

floatingActionButton: FloatingActionButton(

onPressed: \_incrementCounter,

tooltip: 'Increment Counter',

child: const Icon(Icons.add),

),

floatingActionButtonLocation: FloatingActionButtonLocation.centerDocked,

);

}

}

**Langkah 4: Dialog Widget**

Dialog widget pada flutter memiliki dua jenis dialog yaitu AlertDialog dan SimpleDialog.

Ubah isi kode main.dart seperti berikut.

class MyApp extends StatelessWidget {

const MyApp({Key? key}) : super(key: key);

@override

Widget build(BuildContext context) {

return const MaterialApp(

home: Scaffold(

body: MyLayout(),

),

);

}

}

class MyLayout extends StatelessWidget {

const MyLayout({Key? key}) : super(key: key);

@override

Widget build(BuildContext context) {

return Padding(

padding: const EdgeInsets.all(8.0),

child: ElevatedButton(

child: const Text('Show alert'),

onPressed: () {

showAlertDialog(context);

},

),

);

}

}

showAlertDialog(BuildContext context) {

*// set up the button*

Widget okButton = TextButton(

child: const Text("OK"),

onPressed: () {

Navigator.pop(context);

},

);

*// set up the AlertDialog*

AlertDialog alert = AlertDialog(

title: const Text("My title"),

content: const Text("This is my message."),

actions: [

okButton,

],

);

*// show the dialog*

showDialog(

context: context,

builder: (BuildContext context) {

return alert;

},

);

}

**Langkah 5: Input dan Selection Widget**

Flutter menyediakan widget yang dapat menerima input dari pengguna aplikasi yaitu antara lain Checkbox, Date and Time Pickers, Radio Button, Slider, Switch, TextField.

Contoh penggunaan TextField widget adalah sebagai berikut:

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("Contoh TextField")),

body: const TextField(

obscureText: false,

decoration: InputDecoration(

border: OutlineInputBorder(),

labelText: 'Nama',

),

),

),

);

}

}

**Langkah 6: Date and Time Pickers**

Date and Time Pickers termasuk pada kategori input dan selection widget, berikut adalah contoh penggunaan Date and Time Pickers.

import 'dart:async';

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 const MaterialApp(

title: 'Contoh Date Picker',

home: MyHomePage(title: 'Contoh Date Picker'),

);

}

}

class MyHomePage extends StatefulWidget {

const MyHomePage({Key? key, required this.title}) : super(key: key);

final String title;

@override

\_MyHomePageState createState() => \_MyHomePageState();

}

class \_MyHomePageState extends State<MyHomePage> {

*// Variable/State untuk mengambil tanggal*

DateTime selectedDate = DateTime.now();

*// Initial SelectDate FLutter*

Future<void> \_selectDate(BuildContext context) async {

*// Initial DateTime FIinal Picked*

final DateTime? picked = await showDatePicker(

context: context,

initialDate: selectedDate,

firstDate: DateTime(2015, 8),

lastDate: DateTime(2101));

if (picked != null && picked != selectedDate) {

setState(() {

selectedDate = picked;

});

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text(widget.title),

),

body: Center(

child: Column(

mainAxisSize: MainAxisSize.min,

children: <Widget>[

Text("${selectedDate.toLocal()}".split(' ')[0]),

const SizedBox(

height: 20.0,

),

ElevatedButton(

onPressed: () => {

\_selectDate(context),

*// ignore: avoid\_print*

print(selectedDate.day + selectedDate.month + selectedDate.year)

},

child: const Text('Pilih Tanggal'),

),

],

),

),

);

}

}