



Modul Praktikum **Pemrograman Mobile**



STATEFUL WIDGET

Tujuan Pembelajaran

- A. Mahasiswa memahami stateful dalam flutter
- B. Mahasiswa memahami penggunaan stateful
- C. Mahasiswa memahami penggunaan textfield
- D. Mahasiswa memahami switch pada flutter

Dasar Teori

1. State

Sebelum kita belajar tentang **Stateful Widget**, kita harus tahu dulu apa itu **state**. Di dalam reaktif framework seperti Flutter, kalian bisa pikirkan jika UI (User Interface) adalah hasil *return* dari sebuah functions. Dan fungsi itu dipanggil menggunakan satu argumen, yaitu **state**. Jadi, **state** adalah data yang berubah selama lifecycle app kalian.

$$\text{UI} = f(\text{state})$$

The layout on the screen Your build methods The application state

Intinya ketimbang cuman mengubah satu nilai, Flutter menghapus UI-nya lalu membuat ulang semua UI-nya dan Flutter cukup cepat dalam melakukan hal itu.

2. Stateful Widget

Kita akan coba latihan membuat Stateful widget dan membandingkannya dengan Stateless Widget.

Pertama, buat dulu project Flutter baru (Tekan CTRL + SHIFT + P di VSCode), lalu tulis *Flutter: New Project*. Jika sudah, hapus semua boiler code di file *main.dart*. Selanjutnya, buat sebuah **Stateless Widget** bernama **MyApp**.

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

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

class MyApp extends StatelessWidget {
  const MyApp({super.key});
```



```
@override
Widget build(BuildContext context) {
  return MaterialApp(
    title: 'Praktikum Mobile',
    theme: ThemeData(
      primarySwatch: Colors.blue,
    ),
    home: const MyHomePage(),
  );
}
```

Lalu, buat sebuah **Statefull Widget** dengan nama **MyHomePage**. Cukup ketik *st/* lalu tekan tab, otomatis terbuat.

```
class MyHomePage extends StatefulWidget {
  const MyHomePage({super.key});

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  @override
  Widget build(BuildContext context) {
    return Container();
  }
}
```

Hapus line 28 - 32 dan ganti dengan kodingan di bawah ini.

```
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Praktikum Mobile'),
      backgroundColor: Colors.blue,
    ),
    body: Center(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          const Text(
            'You have pushed the button this many times:',
          ),
          Text(
            '$_counter',
          ),
        ],
      ),
    ),
  );
}
```

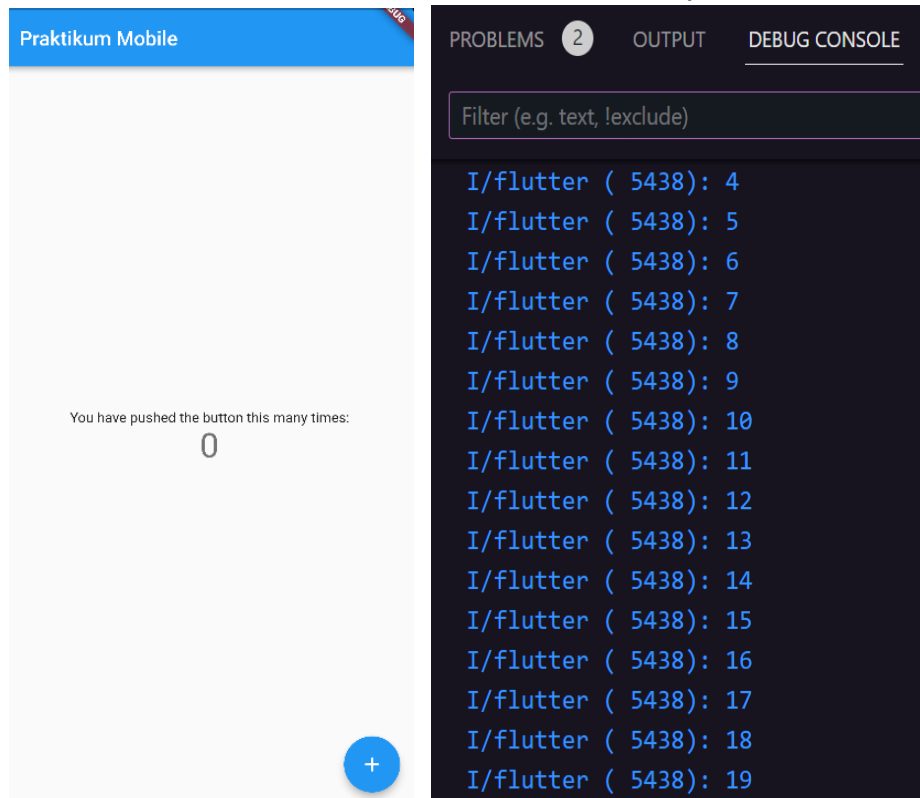


```
        style: Theme.of(context).textTheme.headline4,  
      ),  
    ],  
  ),  
),  
floatingActionButton: FloatingActionButton(  
  onPressed: _incrementCounter,  
  tooltip: 'Increment',  
  child: const Icon(Icons.add),  
),  
);  
}
```

Jika sudah, buat sebuah variable dan fungsi untuk mengatur state.

```
int _counter = 0;  
  
void _incrementCounter() {  
  _counter++;  
  print(_counter);  
}  
  
}
```

Jalankan, lalu tekan tombol coba + di bawah kanan, maka output akan terlihat seperti ini.

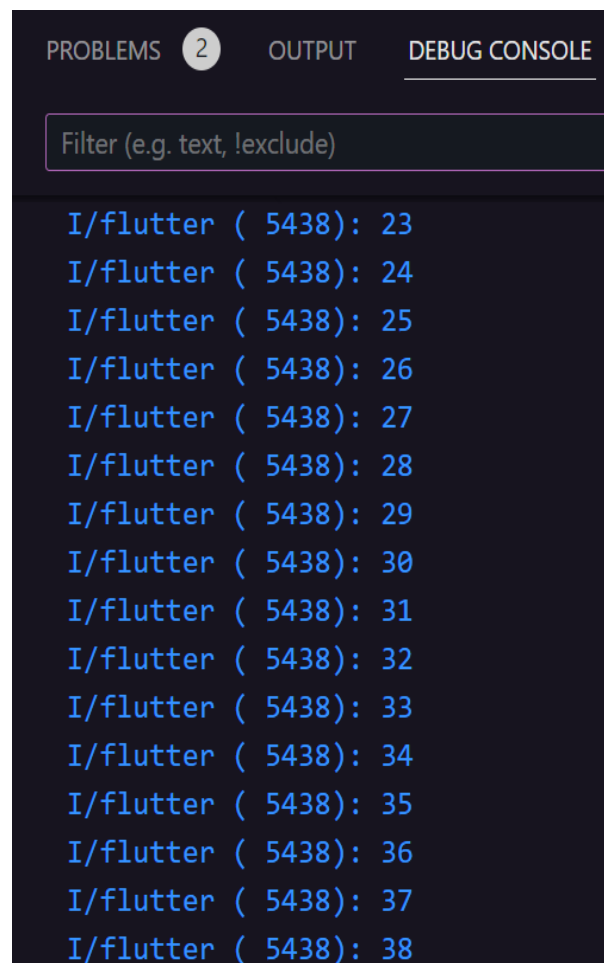
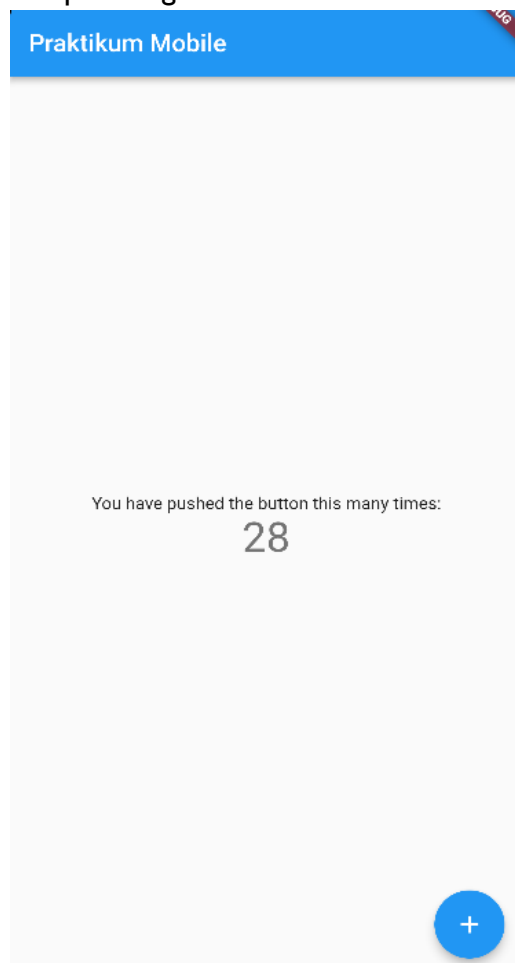




Bisa kita lihat, nilai pada variable berubah tapi tidak dengan nilai pada UI. Maka dari itu, kita butuh memanggil sebuah fungsi yang berguna untuk mengubah state pada UI. (Ingat, UI hanyalah sebuah kembalian dari fungsi dengan argumen berupa state). Cukup tambahkan **setState** pada fungsi **_incrementCounter()**.

```
void _incrementCounter() {  
  setState(() {  
    _counter++;  
  });  
  print(_counter);  
}
```

Output Program:





3. Button

Pada contoh di atas, kalian sudah melihat ada penggunaan button, yaitu *FloatingActionButton*. Di dalam Flutter ada beberapa button yang bisa kalian pakai.

Old Widget	Old Theme	New Widget	New Theme
<code>FlatButton</code>	<code>ButtonTheme</code>	<code>TextButton</code>	<code>TextButtonTheme</code>
<code>RaisedButton</code>	<code>ButtonTheme</code>	<code>ElevatedButton</code>	<code>ElevatedButtonTheme</code>
<code>OutlineButton</code>	<code>ButtonTheme</code>	<code>OutlinedButton</code>	<code>OutlinedButtonTheme</code>

Jika kalian lihat tabel (atau gambar) di atas, ada beberapa button yang di sarankan untuk dipakai lagi, seperti *FlatButton* dan lebih disarankan memakai penggantinya yaitu *Text Button*.

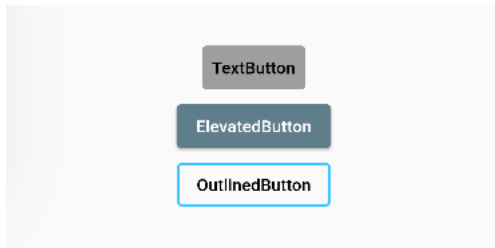
- **TextButton** merupakan widget button yang memiliki tampilan yang polos selayaknya *Text*. *TextButton* umumnya digunakan pada toolbars, dialog, atau bersama komponen button lain.
- **ElevatedButton** merupakan bagian dari material design widget dari flutter. *ElevatedButton* memiliki 2 parameter yaitu **onPressed** dan **child**. Parameter **onPressed** merupakan sebuah *function event* ketika tombol ditekan dan parameter **child** diisi oleh widget pada umumnya
- **OutlinedButton** juga merupakan bagian dari material design yang menyediakan tampilan *TextButton* dengan tambahan outline.

```
TextButton(  
  onPressed: () {},  
  style: TextButton.styleFrom(  
    foregroundColor: Colors.black,  
    backgroundColor: Colors.grey),  
  child: Text('TextButton'),  
),  
ElevatedButton(  
  onPressed: () {},  
  style: ElevatedButton.styleFrom(  
    foregroundColor: Colors.white,  
    backgroundColor: Colors.blueGrey, // foreground  
  ),  
  child: Text('ElevatedButton'),  
),  
OutlinedButton(  
  onPressed: () {},
```



```
style: OutlinedButton.styleFrom(  
  foregroundColor: Colors.black,  
  side: BorderSide(  
    width: 2,  
    color: Colors.lightBlueAccent  
  ),  
),  
child: Text('OutlinedButton'),  
),
```

Output program:



4. TextField

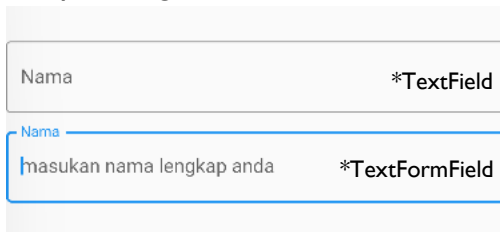
Widget **TextField** merupakan salah satu input pengguna yang umumnya pada aplikasi mobile dan Widget **Textfield** menerima input berupa text yang berasal dari Keyboard. Dalam **TextField** ada juga yang namanya **TextFormField**, **TextFormField** berfungsi sebagai input field yang biasanya digunakan untuk input nama lengkap, searching, email, password dan input lainnya.

Pertama-pertama, kita akan membahas cara membuat **TextField** yang paling gampang.

```
const TextField(  
  decoration: InputDecoration(  
    border: OutlineInputBorder(),  
    labelText: 'Nama',  
  ),  
)  
const SizedBox(height: 15),  
TextFormField(  
  decoration: InputDecoration(  
    hintText: "masukan nama lengkap anda",  
    labelText: "Nama",  
    border: OutlineInputBorder(  
      borderRadius: BorderRadius.circular(5.0)),  
    ),  
)
```



Output Program:



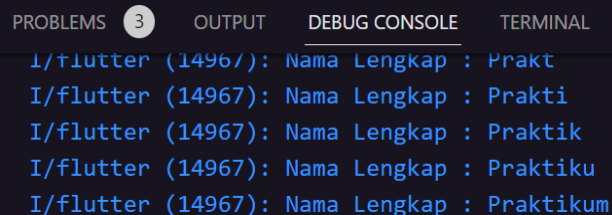
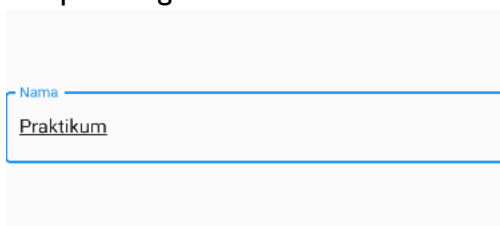
Dalam TextField kita bisa mengambil value dari teks yang kita inputkan, ada beberapa cara yaitu menggunakan parameter **onChanged()** dan **Controller**.

onChanged() berisi sebuah fungsi yang akan dipanggil setiap terjadi perubahan inputan pada TextField. Sedangkan **Controller** atau biasanya dalam TextField disebut **TextEditingController**, dimana kita menggunakan controller dari TextField untuk memasukkan inputan.

Contoh TextField menggunakan onChanged() dan outputnya

```
TextFormField(  
  decoration: InputDecoration(  
    hintText: "masukan nama lengkap anda",  
    labelText: "Nama",  
    border: OutlineInputBorder(  
      borderRadius: BorderRadius.circular(5.0)),  
  ),  
  controller: _controller,  
  onChanged: (String value) {  
    print("Nama Lengkap : $value");  
  },  
),
```

Output Program:



Contoh membuat **TextEditingController()**

```
final TextEditingController _controller = TextEditingController();  
String? name = '';
```

Pertama buat dulu variabel untuk **TextEditingController()** dan variabel untuk inputan. Ketika menggunakan controller, pastikan untuk menghapus controller ketika



halaman atau widget sudah tidak digunakan. Ini bertujuan supaya tidak menimbulkan kebocoran memori.

```
@override
void dispose() {
  _controller.dispose();
  super.dispose();
}
```

Karena sebelumnya kita menghapus controller jika dia tidak dibutuhkan lagi, berarti kita juga harus membuat controller dipanggil setiap UI di rebuild. Selanjutnya kita akan membuat sebuah fungsi untuk menampilkan teks yang diinputkan.

```
@override
void initState() {
  super.initState();
  _controller.addListener(_printLatestValue);
}

void _printLatestValue() {
  print('Nama : ${_controller.text}');
}
```

Setelah membuat fungsi-fungsi yang dibutuhkan, selanjutnya kita akan membuat TextField dengan menambahkan controller ke dalamnya.

```
Text(
  '$name',
  style: const TextStyle(
    fontSize: 20,
    fontWeight: FontWeight.bold
  ),
),
const SizedBox(
  height: 15,
),
SizedBox(
  height: 80,
  width: 350,
  child: TextField(
    decoration: InputDecoration(
      hintText: "masukan nama lengkap anda",
      labelText: "Nama",
      border: OutlineInputBorder(
        borderRadius: BorderRadius.circular(5.0)
      ),
    ),
  ),
),
```

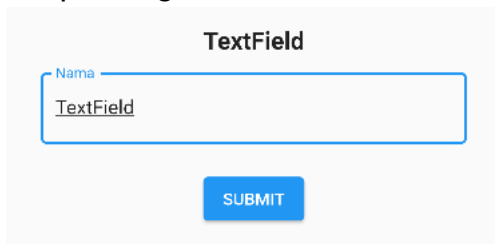


```
),  
  controller: _controller,  
),  
),
```

Langkah terakhir kita akan membuat button untuk mengambil inputan agar dapat menampilkan hasil inputan.

```
ElevatedButton(  
  onPressed: (() {  
    setState(() {  
      name = _controller.text;  
    });  
  }),  
  style: ElevatedButton.styleFrom(  
    foregroundColor: Colors.blue  
  ),  
  child: const Text(  
    'SUBMIT',  
    style: TextStyle(color: Colors.white),  
  ),  
),
```

Output Program:



```
I/flutter (17308): Nama : TextF  
I/flutter (17308): Nama : TextFi  
I/flutter (17308): Nama : TextFie  
I/flutter (17308): Nama : TextFiel  
I/flutter (17308): Nama : TextField
```

5. Radio dan Check Button

Radio merupakan inputan yang digunakan untuk memilih salah satu dari beberapa pilihan dalam suatu kelompok, sedangkan **Checkbox** merupakan inputan benar atau salah. Checkbox akan berisi centang jika nilainya adalah benar dan kosong jika salah.

Berikut contoh membuat **Radio Button**

Pertama kita akan membuat variable terlebih dahulu

```
var radioValue = "";
```

Setelah itu kita akan membuat kode program sederhana radio button

```
ListTile(  
  leading: Radio(  

```



```
        value: 'Button',
        groupValue: radioValue,
        onChanged: (String? Value) {
            setState(() {
                radioValue = value!;
            });
        },
    ),
    title: const Text('Button'),
),
ListTile(
    leading: Radio(
        value: 'TextField',
        groupValue: radioValue,
        onChanged: (String? Value) {
            setState(() {
                radioValue = value!;
            });
        },
    ),
    title: const Text('TextField'),
),
ListTile(
    leading: Radio(
        value: 'Radio Button',
        groupValue: radioValue,
        onChanged: (String? Value) {
            setState(() {
                radioValue = value!;
            });
        },
    ),
    title: const Text('Radio Button'),
),
Text(
    radioValue,
    style: const TextStyle(
        fontSize: 20,
        fontWeight: FontWeight.bold,
    ),
),
),
```



Output Program:

- ☐ Button
- ☐ TextField
- ☒ Radio Button

Radio Button

Contoh membuat **Checkbox**

Hampir sama dengan radio button tadi, pertama kita buat variabel dulu tetapi variabel yang digunakan pada **checkbox** bertipe *Boolean* yang gunanya untuk menampung nilai checkbox tersebut.

```
bool? check1 = false;  
bool? check2 = false;  
bool? check3 = false;
```

setelah itu buatlah widget checkbox dengan variable Boolean yang dibuat tadi.

```
ListTile(  
  leading: Checkbox(  
    value: check1,  
    onChanged: (bool? value) {  
      setState(() {  
        check1 = value;  
      });  
    },  
  ),  
  title: const Text('Pertemuan 1'),  
)  
ListTile(  
  leading: Checkbox(  
    value: check2,  
    onChanged: (bool? value) {  
      setState(() {  
        check2 = value;  
      });  
    },  
  ),  
  title: const Text('Pertemuan 2'),  
)  
ListTile(  
  leading: Checkbox(  
    value: check3,  
    onChanged: (bool? value) {
```



```
        setState(() {  
            check3 = value;  
        });  
    },  
    ),  
    title: const Text('Pertemuan 3'),  
),
```

Output Program:

- ☒ Pertemuan 1
- ☒ Pertemuan 2
- ☐ Pertemuan 3

6. Switch

Switch pada flutter yaitu inputan yang mengembalikan nilai boolean true atau false. Switch umumnya digunakan sebagai konfigurasi on/off pada halaman setting.

Berikut contoh cara membuat **Switch**

Buat variable terlebih dahulu untuk memanggil nilai dari switch. Sama seperti radio button di switch juga menggunakan variable tipe Boolean.

```
bool switchOn = false;
```

Setelah membuat variabel kita akan membuat switch menggunakan widget switch yang sudah disediakan dalam flutter.

```
Switch(  
    value: switchOn,  
    activeColor: Colors.black,  
    onChanged: (bool value) {  
        setState(() {  
            switchOn = value;  
        });  
    }  
)
```

Output Program:

