

Modul 2

State Management

setState vs Provider

Module Overview

State management pada sebuah aplikasi memungkinkan pemrogram mengatur data/state bekerja pada sebuah aplikasi. Flutter sebagai framework memiliki state bawaan yaitu setState yang umumnya digunakan untuk mengatur local state UI, namun memiliki keterbatasan sehingga terdapat beberapa pilihan pendekatan state management pada flutter, salah satunya adalah provider.

Module Objectives

Setelah mempelajari dan mempraktikkan modul ini, mahasiswa diharapkan dapat:

- Memahami konsep state management pada flutter
- Menerapkan setState dan Provider pada flutter
- Mengetahui perbedaan kapan harus menggunakan setState dan Provider.

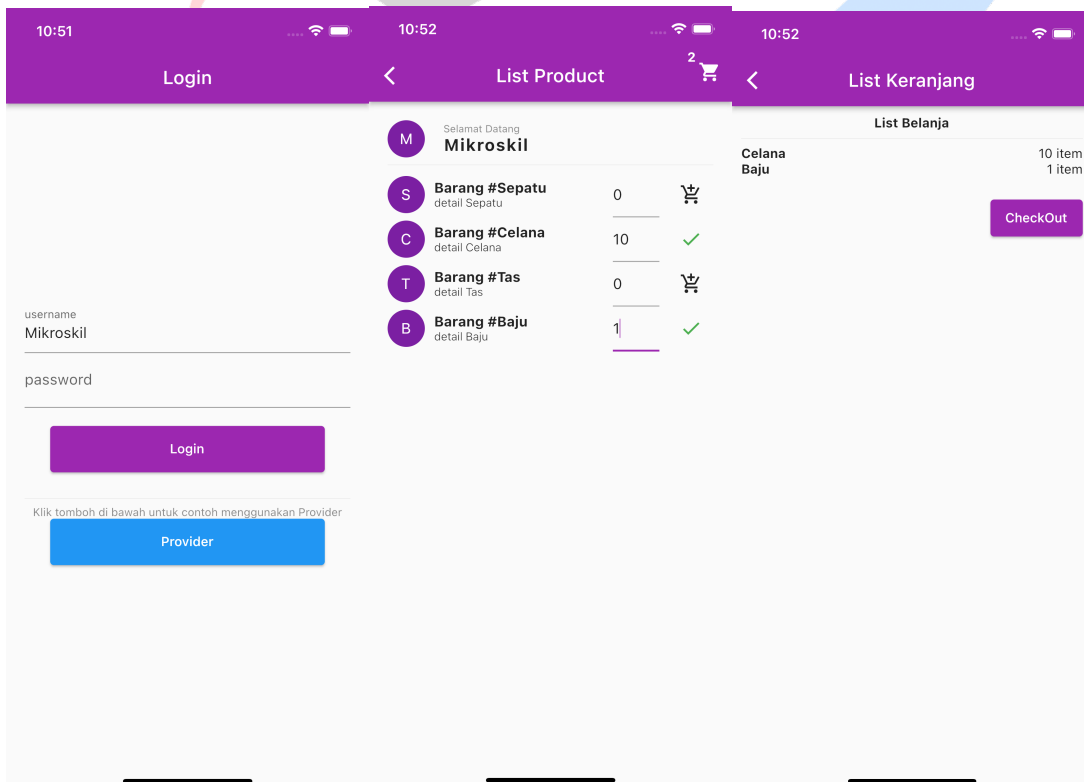
setState

setState adalah basic state management pada flutter dengan fungsi `setState()`. Dikenal sebagai Ephemeral state atau UI state atau local state untuk mengatur kondisi saat ini, dan data yang dinamis.

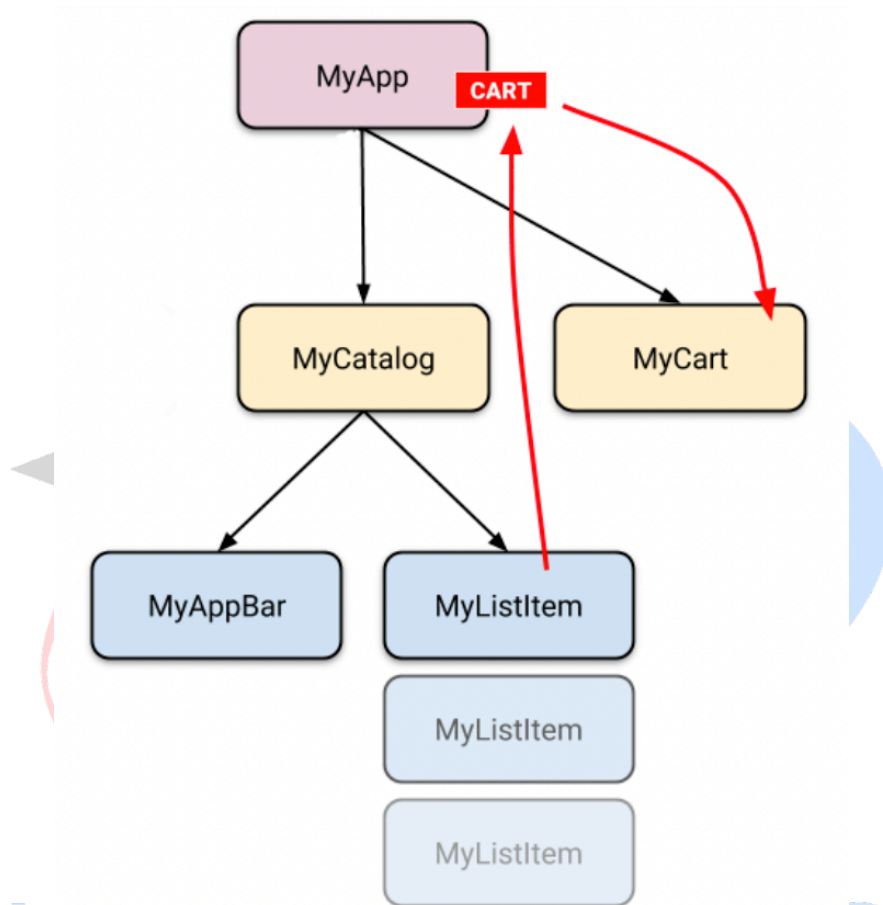
Menggunakan setState, maka pemrogram dapat menyelesaikan 3 masalah berikut:

- Mengirim state ke widget turunan (widgets tree)
- Rebuild widgets ketika terjadi perubahan/update pada state
- Navigasi antara screen dan mempertahankan sinkronisasi state

Membuat aplikasi keranjang belanja menggunakan setState

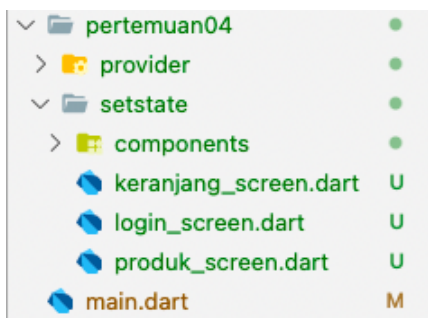


Gambaran aplikasi ini berupa tree berikut:



Langkah-langkah pembuatan:

1. Buatlah sebuah folder dan file dengan struktur file seperti gambar berikut:



pada folder **pertemuan04**, kita akan membuat 2 folder untuk bahasan **setstate** dan provider nantinya. **Fokus pada folder **setstate****, buatlah sebuah file **login_screen.dart**.

Atur home screen pada main.dart ke LoginScreen();

2. Buatlah sebuah halaman menggunakan **statefulWidget** dengan nama class **LoginScreen**.

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

class LoginScreen extends StatefulWidget {
  LoginScreen({Key? key}) : super(key: key);

  @override
  State<LoginScreen> createState() => _LoginScreenState();
}

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

3. Gunakan **scaffold** pada widget build agar aplikasi dapat menggunakan material component. Ganti return container() dengan scaffold.

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar( // AppBar ...
    body: SafeArea( // SafeArea ...
  ); // Scaffold
}
```

4. Pada property appBar gunakan Widget **AppBar()** untuk menampilkan title.

```
appBar: AppBar(
  title: Text(loginText),
```

),

5. Pada property body, tambahkan gunakan Widget SafeArea() > Container() > Center() > Column ()

```
body: SafeArea(
  child: Container(
    padding: EdgeInsets.all(defaultPadding),
    height: MediaQuery.of(context).size.height,
    child: Center(
      child: Column( // Column // Center --
    ), // Container
  ), // SafeArea
```

6. Pada widget column, tambahkan beberapa widget untuk membuat halaman login.

```
-child: Column(
  mainAxisAlignment: MainAxisAlignment.center,
  children: [
    //Spacer widget seperti flex untuk mengatur jarak antara widget di flex container, column, atau row.
    const Spacer(),

    //LoginForm
    SizedBox( // Column // SizedBox --
    SizedBox(height: defaultPadding),

    //LoginButton
    ElevatedButton( // ElevatedButton --
  ]), // Column // Center
```

7. Buatlah login form menggunakan textfield
pada masing-masing login form, gunakan properti controller untuk handle terima input text field.

UNIVERSITAS
MIKROSKIL

```
//LoginForm
- SizedBox(
  child: Column(
    children: [
      TextField(
        controller: usernameController,
        decoration: InputDecoration(
          label: const Text('username'),
          hintText: 'Masukkan Username',
          errorText: isUsernameEmpty == true
            ? 'username harus diisi'
            : null, // InputDecoration
        ), // TextField
      TextField(
        controller: passwordController,
        decoration: InputDecoration(
          label: const Text('password'),
          hintText: 'Masukkan Password',
          errorText: isPasswordEmpty == true
            ? 'Kata sandi harus diisi'
            : null, // InputDecoration
        ), // TextField
    ],
  ), // Column // SizedBox
```

8. Tambahkan TextEditingController sebagai **state** pada class _LoginScreenState

```
class _LoginScreenState extends State<LoginScreen> {
  //Contoh LocalVariabel
  TextEditingController usernameController = TextEditingController();
  TextEditingController passwordController = TextEditingController();
```

State diinisialisasi pada bagian ini untuk digunakan pada local class **LoginScreen()**

```
//Contoh LocalVariabel
TextEditingController usernameController = TextEditingController();
TextEditingController passwordController = TextEditingController();

String loginText = "Login";
double defaultPadding = 20;

//status username dan password kosong!
bool? isUsernameEmpty;
bool? isPasswordEmpty;

@override
void initState() {
  //Contoh inisialisasi default state. Nilai ini akan tampil saat pertama kali aplikasi run.
  usernameController.text = 'Mikroskil';
  isUsernameEmpty = false;
  isPasswordEmpty = false;

  super.initState();
}
```

Inisialisasi state **loginText** untuk digunakan sebagai title pada **AppBar()**.

Pada contoh kasus ini, digunakan **widget lifecycle iniState()** untuk inisialisasi value saat widget dibuat. Contohnya untuk kebutuhan property **errorText** pada **TextField()**, maka diinisialisasi **isUsernameEmpty = false;**



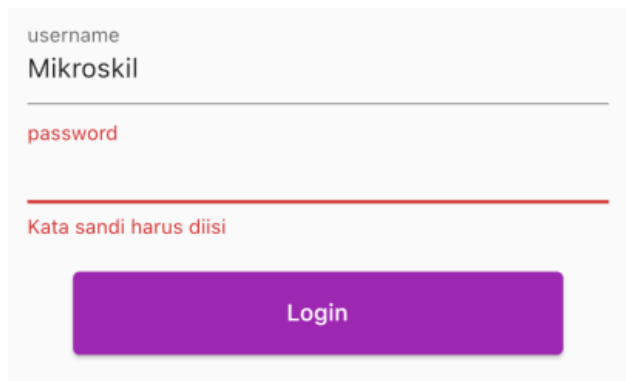
9. Pada bagian LoginButton tambahkan gunakan widget ElevatedButton, pada fungsi onPressed lakukan logika sederhana untuk mengecek jika username atau password kosong, maka tampilkan pesan error pada property **errorText** TextField.

```
//LoginButton
ElevatedButton(
  onPressed: () {
    //contoh validasi jika username dan password empty maka tampilkan pesan error
    if (usernameController.text.isEmpty) {
      setState(() {
        isUsernameEmpty = true;
      });
    }
    if (passwordController.text.isEmpty) {
      setState(() {
        isPasswordEmpty = true;
      });
    }
    //Jika tidak ada kesalahan navigasi ke halaman produk
    else {
      setState(() {
        isUsernameEmpty = false;
        isPasswordEmpty = false;
      });
      Navigator.push(context, MaterialPageRoute(
        builder: (context) {
          //passing data to another screen
          return ProdukScreen(
            username: usernameController.text); // ProdukScreen
        },
      )); // MaterialPageRoute
    }
  },
  child: Text(loginText),
  style: ElevatedButton.styleFrom(
    minimumSize: Size(
      MediaQuery.of(context).size.width / 2 + 100, 50)), // Size
), // ElevatedButton
```


Pengecekan menggunakan percabangan if, jika usernameController atau passwordController.text kosong maka gunakan **setState()** untuk memperbaharui state UI.

```
setState(() {
  isUsernameEmpty = true;
});
```

jika kondisinya logika ini true, maka akan mempengaruhi state **isUsernameEmpty** menjadi **true** dan menampilkan pesan error seperti gambar di bawah.



The screenshot shows a login form with two text input fields. The first field is labeled 'username' and contains the text 'Mikroskil'. The second field is labeled 'password' and is empty. Below the password field, there is a red error message that reads 'Kata sandi harus diisi'. At the bottom of the form is a purple button labeled 'Login'.

Full code **login_screen.dart**

```
import 'package:flutter/material.dart';
import 'package:flutter_application/pertemuan04/provider/produk2_screen.dart';

class LoginScreen extends StatefulWidget {
  LoginScreen({Key? key}) : super(key: key);

  @override
  State<LoginScreen> createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  //Contoh LocalVariabel
  TextEditingController usernameController = TextEditingController();
  TextEditingController passwordController = TextEditingController();

  String loginText = "Login";
  double defaultPadding = 20;
```

```
//status username dan password kosong!
bool? isUsernameEmpty;
bool? isPasswordEmpty;

@override
void initState() {
  //Contoh inisialisasi default state. Nilai ini akan tampil saat pertama kali
  aplikasi run.
  usernameController.text = 'Mikroskil';
  isUsernameEmpty = false;
  isPasswordEmpty = false;

  super.initState();
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text(loginText),
    ),
    body: SafeArea(
      child: Container(
        padding: EdgeInsets.all(defaultPadding),
        height: MediaQuery.of(context).size.height,
        child: Center(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              //Spacer widget seperti flex untuk mengatur jarak antara widget
              di flex container, column, atau row.
              const Spacer(),

              //LoginForm
              SizedBox(
                child: Column(
                  children: [
                    TextField(
                      controller: usernameController,
                      decoration: InputDecoration(
                        label: const Text('username'),
                        errorText: isUsernameEmpty == true
                          ? 'username harus diisi'
                          : null),
                    ),
                    TextField(
                      controller: passwordController,
```

```

        decoration: InputDecoration(
          label: const Text('password'),
          errorText: isPasswordEmpty == true
            ? 'Kata sandi harus diisi'
            : null),
        ),
      ],
    )),
    SizedBox(height: defaultPadding),

    //LoginButton
    ElevatedButton(
      onPressed: () {
        //contoh validasi jika username dan password empty maka
        //tampilkan pesan error
        if (usernameController.text.isEmpty) {
          setState(() {
            isUsernameEmpty = true;
          });
        }
        if (passwordController.text.isEmpty) {
          setState(() {
            isPasswordEmpty = true;
          });
        }
        //Jika tidak ada kesalahan navigasi ke halaman produk
        else {
          setState(() {
            isUsernameEmpty = false;
            isPasswordEmpty = false;
          });
          Navigator.push(context, MaterialPageRoute(
            builder: (context) {
              //passing data to another screen
              return ProdukScreen(
                username: usernameController.text);
            },
          ));
        }
      },
      child: Text(loginText),
      style: ElevatedButton.styleFrom(
        minimumSize: Size(
          MediaQuery.of(context).size.width / 2 + 100, 50)),
    ),

    SizedBox(height: defaultPadding),

```

```
const Divider(),
const Text(
  'Klik tombol di bawah untuk contoh menggunakan Provider',
  textAlign: TextAlign.center,
  style: TextStyle(fontSize: 12, color: Colors.black45),
),
ElevatedButton(
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: ((context) => ProdukScreen2())));
  },
  child: Text('Provider'),
  style: ElevatedButton.styleFrom(
    primary: Colors.blue,
    minimumSize: Size(
      MediaQuery.of(context).size.width / 2 + 100, 50)),
),

const Spacer(),
]),
),
),
);
}
}
```

UNIVERSITAS
MIKROSKIL

10. Buatlah file **produk_screen.dart** yang menggunakan widget **StatefulWidget**

```

1  import 'package:flutter/material.dart';
2  import 'package:flutter_application/pertemuan04/setstate/components/produk_widget.dart';
3  import 'package:flutter_application/pertemuan04/setstate/keranjang_screen.dart';
4
5  class ProdukScreen extends StatefulWidget {
6    final String username;
7    ProdukScreen({Key? key, required this.username}) : super(key: key);
8
9    @override
10   State<ProdukScreen> createState() => _ProdukScreenState();
11 }
12
13 class _ProdukScreenState extends State<ProdukScreen> {
14   double defaultPadding = 20;
15   String titleScreen = 'List Product';
16
17   TextEditingController sepatuCtrl = TextEditingController();
18   TextEditingController tasCtrl = TextEditingController();
19   TextEditingController celanaCtrl = TextEditingController();
20   TextEditingController bajuCtrl = TextEditingController();
21
22   List<Map<String, String>> keranjang = [];
23
24   bool? isSepatuAdd;
25   bool? isCelanaAdd;
26   bool? isTasAdd;
27   bool? isBajuAdd;
28
29   @override
30   void initState() {
31     //Default ctrl sepatu
32     sepatuCtrl.text = 0.toString();
33     tasCtrl.text = 0.toString();
34     celanaCtrl.text = 0.toString();
35     bajuCtrl.text = 0.toString();
36
37     //Status Baju
38     isSepatuAdd = false;
39     isCelanaAdd = false;
40     isTasAdd = false;
41     isBajuAdd = false;
42
43     super.initState();
44   }
45
46   @override
47   Widget build(BuildContext context) {
48     return Scaffold(
49       appBar: AppBar(title: Text(titleScreen), actions: [ // AppBar --
50       body: SafeArea( // SingleChildScrollView // SafeArea --
51     ); // Scaffold
52   }
53 }
54
55 
```

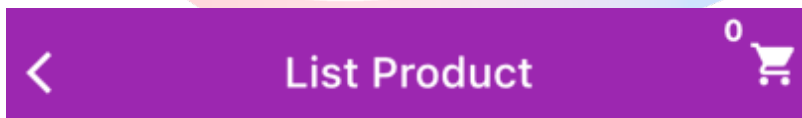
Pada halaman ini kita menggunakan beberapa state untuk mengatur UI state seperti status add produk bool? **isSepatuAdd**, **isCelanaAdd** dst.

List keranjang juga diinisialisasi sebagai dengan list Map kosong [].

11. Buatlah icon untuk menunjukkan total keranjang belanja pada AppBar()

```
-appBar: AppBar(title: Text(titleScreen), actions: [
  Stack(children: [
    IconButton(
      icon: const Icon(Icons.shopping_cart),
      onPressed: () {
        // handle the press
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) =>
              KeranjangScreen(keranjang: keranjang))); // MaterialPageRoute
      },
    ), // IconButton
    //keranjang belanja
    Positioned(
      child: Text(
        keranjang.length.toString(),
        style: const TextStyle(fontWeight: FontWeight.bold),
      ) // Text // Positioned
    ), // Stack
  ]), // AppBar
);
```

Pada bagian digunakan widget **Stack** pada property **actions:[]**



12. body menggunakan widget SafeArea() > SingleChildScrollView() > Padding() > Column()

```
-body: SafeArea(
  child: SingleChildScrollView(
    child: Padding(
      padding: EdgeInsets.all(defaultPadding),
      child: Column( // Column // Padding ...
    ), // SingleChildScrollView // SafeArea
  );
```

13. Pada widget Column() tambahkan beberapa widget sebagai children seperti:

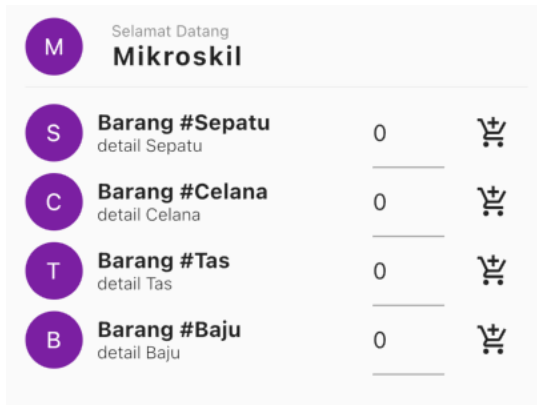
```

child: Column(
  children: [
    //header
    Row(
      children: [
        CircleAvatar(
          child: Text(widget.username.substring(0, 1)),
        ), // CircleAvatar
        SizedBox(width: defaultPadding),
        Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            const Text(
              'Selamat Datang',
              style: TextStyle(fontSize: 11, color: Colors.black45),
            ), // Text
            Text(
              widget.username,
              style: const TextStyle(
                fontWeight: FontWeight.bold,
                fontSize: 18,
                letterSpacing: 1.0), // TextStyle
            ), // Text
          ],
        ) // Column
      ],
    ), // Row
    const Divider(),
    Column(children: [
      // Untuk kebutuhan data dinamis, level expert: https://stackoverflow.com/questions/50766139/flutter-create-dynamic-number-of-texteditingcontrollers

      // contoh produk statik dengan textediting statik.
      ProdukWidget(
        namaProduk: 'Sepatu',
        ctrl: sepatuCtrl,
        status: isSepatuAdd,
        press: () {
          print('sepatu');
          setState(() {
            isSepatuAdd = true;
            keranjang
              .add({"title": 'Sepatu', "total": sepatuCtrl.text});
          });
        },
      ), // ProdukWidget
      ProdukWidget( // ProdukWidget --
      ProdukWidget( // ProdukWidget --
      ProdukWidget( // ProdukWidget --
    ]) // Column
  )

```

Pada bagian //header tampilkan **username** yang telah dikirim dari halaman login_screen. Berikut tampilan body.



14. Buat sebuah file **produk_widget.dart** untuk widget child produk bernama **ProdukWidget()** menggunakan StatelessWidget

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

class ProdukWidget extends StatelessWidget {
  final String namaProduk;
  final TextEditingController ctrl;
  final VoidCallback press;
  final bool? status;
  const ProdukWidget(
    {Key? key,
    required this.namaProduk,
    required this.ctrl,
    required this.status,
    required this.press})
    : super(key: key);

  @override
  Widget build(BuildContext context) {
    double defaultPadding = 20;

    return Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        //kiri
        Row(
          children: [
            CircleAvatar(
              child: Text(namaProduk.substring(0, 1)),
            ),
```



```

        SizedBox(width: defaultPadding / 2),
        Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Text(
              'Barang #${namaProduk}',
              style: const TextStyle(
                fontSize: 15, fontWeight: FontWeight.bold),
            ),
            Text(
              'detail ${namaProduk}',
              style: const TextStyle(
                fontSize: 12, fontWeight: FontWeight.w300),
            )
          ],
        ),
      ],
    ),
  ),
  //kanan
  Row(
    children: [
      Container(
        width: 50,
        child: TextField(
          controller: ctrl,
        ),
      ),
      SizedBox(width: defaultPadding / 2),
      Center(
        child: IconButton(
          onPressed: status! ? null : press,
          icon: status!
            ? const Icon(
                Icons.check,
                color: Colors.green,
              )
            : const Icon(
                Icons.add_shopping_cart_outlined,
              )
        ),
      ),
    ],
  ),
);
;
}
}

```

Widget ini memiliki property `final` String `namaProduk`; `final` `TextEditingController` `ctrl`; `final` `VoidCallback` `press`; `final` `bool?` `status`; yang akan dikirim dari parent yaitu `ProdukScreen()`.

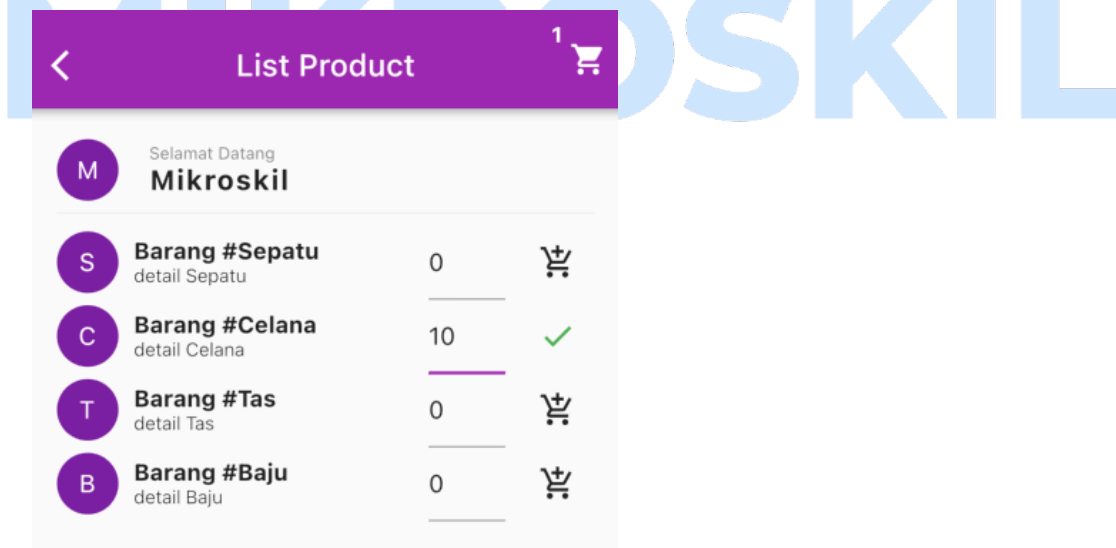
widget ini dapat digunakan (reusable) pada screen lain lain, pada case ini digunakan pada halaman `produk_screen()`.

```
-Column(children: [
  // Untuk kebutuhan data dinamis, level expert: https://stackoverflow.com/questions/50766139/flutter-create-dynamic-number-of-texteditingcontrollers

  // contoh produk statik dengan textediting statik.
  ProdukWidget(
    namaProduk: 'Sepatu',
    ctrl: sepatuCtrl,
    status: isSepatuAdd,
    press: () {
      print('sepatu');
      setState(() {
        isSepatuAdd = true;
        keranjang
          .add({"title": 'Sepatu', "total": sepatuCtrl.text});
      });
    },
  ),
  // ProdukWidget
  ProdukWidget( // ProdukWidget --
  ProdukWidget( // ProdukWidget --
  ProdukWidget( // ProdukWidget --
  // Column
```

Pada widget **ProdukWidget()** terdapat property `press()` yang berfungsi jika `IconButton` keranjang diklik maka akan **mengubah UI state** status `isSepatuAdd` menjadi `true` dan nemambah **list keranjang** dengan `title` dan `total`.

Berikut adalah tampilannya:



Full source code produk_screen.dart

```
import 'package:flutter/material.dart';
import
'package:flutter_application/pertemuan04/setstate/components/produk_widget.dart';
import 'package:flutter_application/pertemuan04/setstate/keranjang_screen.dart';

class ProdukScreen extends StatefulWidget {
  final String username;
  ProdukScreen({Key? key, required this.username}) : super(key: key);

  @override
  State<ProdukScreen> createState() => _ProdukScreenState();
}

class _ProdukScreenState extends State<ProdukScreen> {
  double defaultPadding = 20;
  String titleScreen = 'List Product';

  TextEditingController sepatuCtrl = TextEditingController();
  TextEditingController tasCtrl = TextEditingController();
  TextEditingController celanaCtrl = TextEditingController();
  TextEditingController bajuCtrl = TextEditingController();

  List<Map<String, String>> keranjang = [];

  bool? isSepatuAdd;
  bool? isCelanaAdd;
  bool? isTasAdd;
  bool? isBajuAdd;

  @override
  void initState() {
    //Default ctrl sepatu
    sepatuCtrl.text = 0.toString();
    tasCtrl.text = 0.toString();
    celanaCtrl.text = 0.toString();
    bajuCtrl.text = 0.toString();

    //Status Baju
    isSepatuAdd = false;
    isCelanaAdd = false;
    isTasAdd = false;
    isBajuAdd = false;

    super.initState();
  }
}
```

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(title: Text(titleScreen), actions: [
      Stack(children: [
        IconButton(
          icon: const Icon(Icons.shopping_cart),
          onPressed: () {
            // handle the press
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: ((context) =>
                  KeranjangScreen(keranjang: keranjang))));
          },
        ),
        //keranjang belanja
        Positioned(
          child: Text(
            keranjang.length.toString(),
            style: const TextStyle(fontWeight: FontWeight.bold),
          ))
      ]),
    body: SafeArea(
      child: SingleChildScrollView(
        child: Padding(
          padding: EdgeInsets.all(defaultPadding),
          child: Column(
            children: [
              //header
              Row(
                children: [
                  CircleAvatar(
                    child: Text(widget.username.substring(0, 1)),
                  ),
                  SizedBox(width: defaultPadding),
                  Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                      const Text(
                        'Selamat Datang',
                        style: TextStyle(fontSize: 11, color: Colors.black45),
                      ),
                      Text(

```

```

        widget.username,
        style: const TextStyle(
          fontWeight: FontWeight.bold,
          fontSize: 18,
          letterSpacing: 1.0),
      ),
    ],
  ),
],
),
const Divider(),
Column(children: [
  // Untuk kebutuhan data dinamis, level expert:
https://stackoverflow.com/questions/50766139/flutter-create-dynamic-number-of-texteditingcontrollers

  // contoh produk statik dengan textediting statik.
  ProdukWidget(
    namaProduk: 'Sepatu',
    ctrl: sepatuCtrl,
    status: isSepatuAdd,
    press: () {
      print('sepatu');
      setState(() {
        isSepatuAdd = true;
        keranjang
          .add({"title": 'Sepatu', "total": sepatuCtrl.text});
      });
    },
  ),
  ProdukWidget(
    namaProduk: 'Celana',
    ctrl: celanaCtrl,
    status: isCelanaAdd,
    press: () {
      print('Celana');
      setState(() {
        isCelanaAdd = true;
        keranjang
          .add({"title": 'Celana', "total": celanaCtrl.text});
      });
    },
  ),
  ProdukWidget(
    namaProduk: 'Tas',
    ctrl: tasCtrl,
    status: isTasAdd,

```

```

        press: () {
          print('Tas');
          setState(() {
            isTasAdd = true;
            keranjang.add({"title": 'Tas', "total": tasCtrl.text});
          });
        },
      ),
    ProdukWidget(
      namaProduk: 'Baju',
      ctrl: bajuCtrl,
      status: isBajuAdd,
      press: () {
        print('Baju');
        setState(() {
          isBajuAdd = true;
          keranjang
            .add({"title": 'Baju', "total": bajuCtrl.text});
        });
      },
    ),
  ],
),
),
);
}
}

```

15. Buatlah sebuah halaman untuk menampilkan isi keranjang dengan file **keranjang_screen.dart**

```

import 'package:flutter/material.dart';

class KeranjangScreen extends StatefulWidget {
  final List<Map<String, String>> keranjang;
  KeranjangScreen({Key? key, required this.keranjang}) : super(key: key);

  @override
  State<KeranjangScreen> createState() => _KeranjangScreenState();
}

class _KeranjangScreenState extends State<KeranjangScreen> {
  @override
  Widget build(BuildContext context) {

```

```

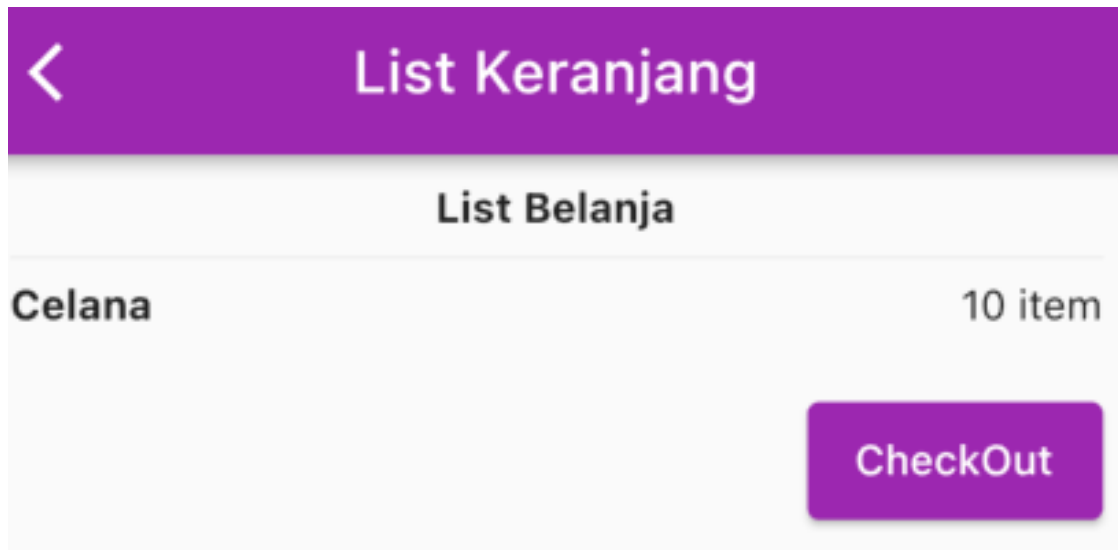
return Scaffold(
  appBar: AppBar(
    title: const Text('List Keranjang'),
  ),
  body: Padding(
    padding: EdgeInsets.all(10),
    child: Column(
      children: [
        const Text(
          'List Belanja',
          style: TextStyle(fontWeight: FontWeight.bold, fontSize: 14),
        ),
        Divider(),
        //List Keranjang
        Column(
          children: widget.keranjang.map((val) {
            return Row(
              mainAxisAlignment: MainAxisAlignment.spaceBetween,
              children: [
                //kiri
                Text(
                  '${val['title']}',
                  style: TextStyle(fontWeight: FontWeight.bold),
                ),
                //kanan
                Text('${val['total']} item')
              ],
            );
          }).toList(),
        ),
        const SizedBox(height: 20),
        //Button CheckOut
        Align(
          alignment: Alignment.bottomRight,
          child: ElevatedButton(
            onPressed: () {
              //Bagaimana cara hapus list keranjang belanja?
            },
            style: ElevatedButton.styleFrom(minimumSize: Size(100, 40)),
            child: const Text('CheckOut')),
        ),
      ],
    ),
  );

```

```
}  
}
```

Halaman ini memiliki property keranjang yang akan diterima dari parent melalui navigasi yang dikirim dari halaman produk_screen.dart (**klik icon keranjang belanja**).

Tampilan:



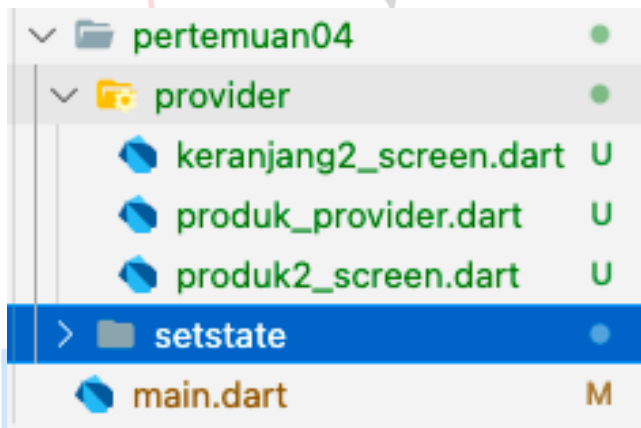
UNIVERSITAS
MIKROSKIL

Provider

Provider merupakan salah satu state manajemen pada flutter yang dikembangkan oleh komunitas dan direkomendasikan oleh flutter untuk mengatur state aplikasi. State pada provider dapat digunakan untuk memisahkan logic dan UI dan state dapat digunakan oleh banyak child. Tidak seperti setState yang valuenya harus diinisialisasi secara local state dan jika ingin digunakan oleh screen lain maka harus dikirim ke childnya.

Menggunakan provider:

1. Pastikan telah install package provider: <https://pub.dev/packages/provider>
2. Buatlah sebuah stuktur folder sebagai berikut:



3. buat sebuah file produk_provider.dart

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

class ProdukProvider extends ChangeNotifier {
  String titleScreen = 'List Product';

  final List<Map<String, String>> _keranjang = [];
  List<Map<String, String>> get keranjang => _keranjang;
  set isiKeranjang(val) {
    _keranjang.add(val);
  }
}
```

```

        notifyListeners();
    }

    TextEditingController tasCtrl = TextEditingController();

    bool _isTasAdd = false;
    bool get isTasAdd => _isTasAdd;
    set setTasStatus(val) {
        _isTasAdd = val;
        notifyListeners();
    }
}

```

Keterangan:

- Gunakan ChangeNotifier pada class ProdukProvider, sehingga aplikasi dapat menggunakan fungsi seperti **notifyListeners()**, ini akan mempengaruhi UI state (mirip seperti setState())
 - notifyListeners() di case ini digunakan jika ada yang isiKeranjang dengan value dengan kata kunci set **isiKeranjang(val)** dan **setTasStatus(val)**.
 - **set dan get** adalah method getters dan setters yang menyediakan akses baca dan tulis pada objek properti.
4. Agar bisa menggunakan provider ini, maka perlu diinisialisasi pada parent widget, contohnya: pada myApp di main.dart

```

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return ChangeNotifierProvider(
      create: (context) => ProdukProvider(),
      child: MaterialApp(
        title: 'Flutter Demo',
        debugShowCheckedModeBanner: false,
        theme: ThemeData(
          primarySwatch: Colors.purple,
        ),
        home: LoginScreen(),
      ),
    );
  }
}

```

```
);  
}  
}
```

Disini menggunakan widget `ChangeNotifierProvider` yang disediakan oleh provider (pastikan import provider di halaman ini).

Catatan: jika terdapat lebih dari 1 (satu) provider, maka gunakan property `MultiProvider`. contoh:

```
MultiProvider(  
  providers: [  
    Provider<Something>(create: (_) => Something()),  
    Provider<SomethingElse>(create: (_) => SomethingElse()),  
    Provider<AnotherThing>(create: (_) => AnotherThing()),  
  ],  
  child: someWidget,  
)
```

5. Buatlah sebuah halaman `produk2_screen.dart` untuk menampilkan list produk

```
import 'package:flutter/material.dart';  
import  
'package:flutter_application/pertemuan04/provider/keranjang2_screen.dart';  
import  
'package:flutter_application/pertemuan04/provider/produk_provider.dart';  
import  
'package:flutter_application/pertemuan04/setstate/components/produk_widget.dart';  
import 'package:provider/provider.dart';  
  
class ProdukScreen2 extends StatelessWidget {  
  const ProdukScreen2({Key? key}) : super(key: key);  
  
  @override  
  Widget build(BuildContext context) {  
    return Scaffold(  
      appBar: AppBar(  

```

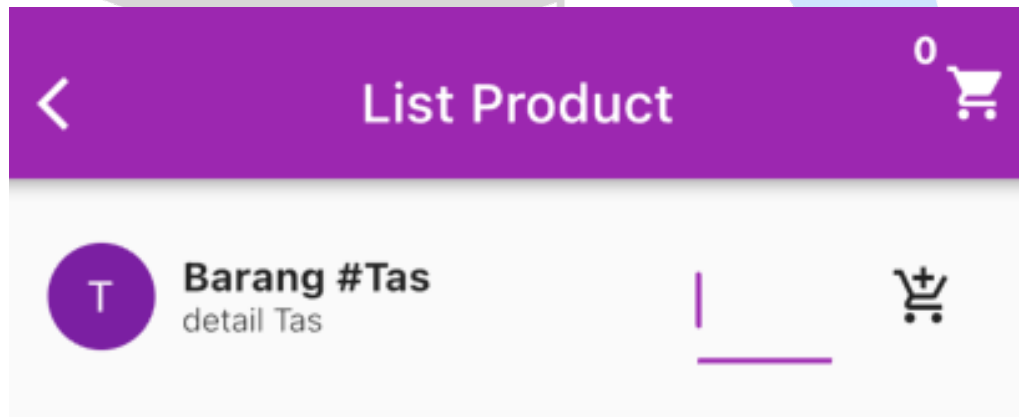
```

        title: Text(context.read<ProdukProvider>().titleScreen),
        actions: [
            Stack(children: [
                IconButton(
                    icon: const Icon(Icons.shopping_cart),
                    onPressed: () {
                        Navigator.push(context,
                            MaterialPageRoute(builder: ((context) =>
                                Keranjang2())));
                    },
                ),
            ],
        ),
        //keranjang belanja
        Positioned(
            child: Text(
                context.watch<ProdukProvider>().keranjang.length.toString(),
                style: const TextStyle(fontWeight: FontWeight.bold),
            ),
        ),
    ],
),
body: Padding(
    padding: const EdgeInsets.all(20),
    child: Column(children: [
        ProdukWidget(
            namaProduk: 'Tas',
            ctrl: context.watch<ProdukProvider>().tasCtrl,
            status: context.watch<ProdukProvider>().isTasAdd,
            press: () {
                print('Tas');
                context.read<ProdukProvider>().setTasStatus = true;
                context.read<ProdukProvider>().isiKeranjang = {
                    "title": 'Tas',
                    "total": context.read<ProdukProvider>().tasCtrl.text
                };
            },
        ),
    ],
),
);
}
}

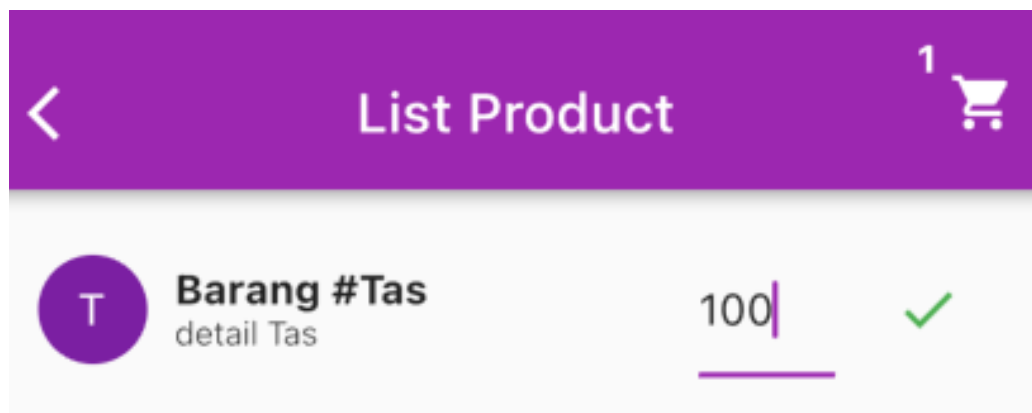
```

Keterangan:

- `context.watch<ProdukProvider>().keranjang.length.toString()` digunakan untuk menampilkan data, jika ada perubahan maka nilai keranjang disini akan berubah.
- `context.read<ProdukProvider>().tasCtrl.text` digunakan untuk membaca atau menulis data tanpa melihat perubahan yang terjadi. Contoh lain `context.read<ProdukProvider>().setTasStatus = true;` untuk menulis atau mengubah `tasStatus`.



Tampilan jika ada perubahan UI State



6. Buatlah sebuah halaman Keranjang2 pada file keranjang2_screen.dart

```
import 'package:flutter/material.dart';
import
'package:flutter_application/pertemuan04/provider/produk_provider.dart';
import 'package:provider/provider.dart';

class Keranjang2 extends StatelessWidget {
  const Keranjang2({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    final provider = context.read<ProdukProvider>();
    return Scaffold(
      appBar: AppBar(
        title: const Text('List Keranjang'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(10),
        child: Column(
          children: [
            const Text(
              'List Belanja',
              style: TextStyle(fontWeight: FontWeight.bold, fontSize: 14),
            ),
            Divider(),
            //List Keranjang
            Column(
              children: provider.keranjang.map((val) {
                return Row(
                  mainAxisAlignment: MainAxisAlignment.spaceBetween,
                  children: [
                    //kiri
                    Text(
                      '${val['title']}',
                      style: const TextStyle(fontWeight: FontWeight.bold),
                    ),
                    //kanan
                    Text('${val['total']} item')
                  ],
                );
              }).toList(),
            ),
            const SizedBox(height: 20),
          ],
        ),
      ),
    );
  }
}
```

```

//Button CheckOut
Align(
  alignment: Alignment.bottomRight,
  child: ElevatedButton(
    onPressed: () {
      //Bagaimana cara hapus list keranjang belanja?
    },
    style: ElevatedButton.styleFrom(minimumSize: Size(100,
      40)),
    child: const Text('CheckOut')),
  ),
),
);
}
}

```

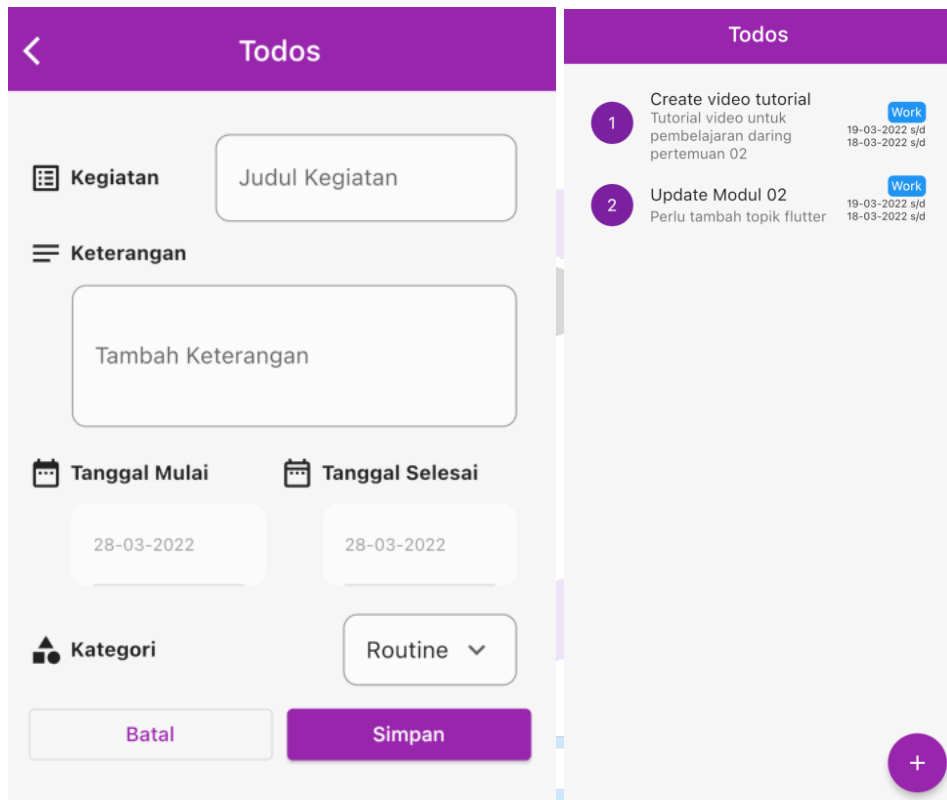
Keterangan:

- Karena disini digunakan provider, maka halaman Keranjang2 dapat menggunakan data dari ProdukProvider dengan membaca data keranjang.

UNIVERSITAS MIKROSKIL

Latihan:

1. Menggunakan **setState** lengkapi halaman pada **aplikasi todos** pada latihan Minggu03 sehingga bisa menampilkan list todos. contoh:



Keterangan:

- Menerapkan `TextEditingController` pada `textfield`. Default value seperti pada tampilan, contoh: "judul kegiatan", "Tambah Keterangan".
- Kolom tanggal menggunakan `textfield`, initial value pada adalah tanggal hari ini gunakan `DateTime.Now` **
- Bagian kategori menggunakan widget **DropDownButton** **tidak wajib ada.

- Menggunakan **provider** buatlah aplikasi perhitungan luas segitiga dengan ui state sebagai berikut.

Hitung Keliling Segitiga sama sisi

Masukkan ke-3 sisinya

Hitung

UI state jika sisi tidak sama

Hitung Keliling Segitiga sama sisi

Masukkan ke-3 sisinya

harus sama

Coba Lagi!

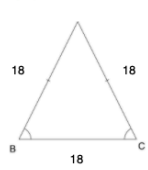
UI state jika berhasil

Hitung Keliling Segitiga sama sisi

Masukkan ke-3 sisinya

Hitung

Hasil: 54 cm



UNIVERSITAS
MIKROSKIL