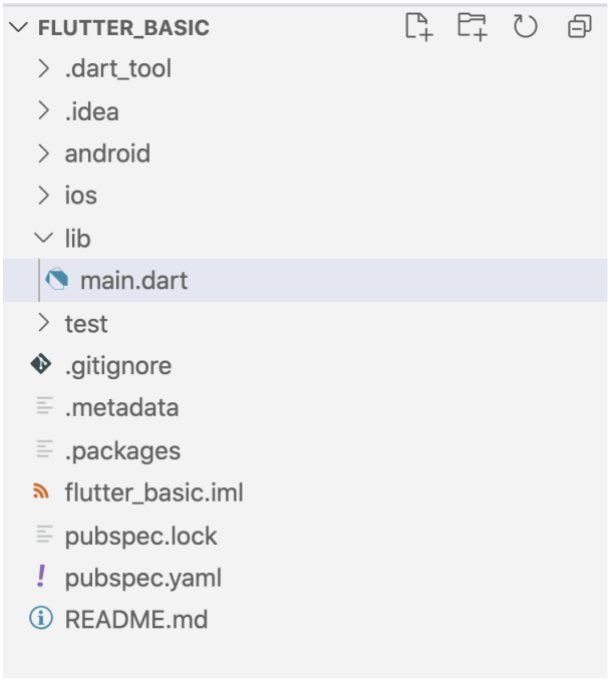
BAB 1. BASIC APLIKASI FLUTTER

1.1 Struktur Project Flutter

1.1.1 Struktur folder

Ketika membuat project flutter secara default berikut adalah struktur folder dan filenya. Dapat kita lihat strukturnya terdiri dari .dart_tool, .idea, android, ios, lib, test, .gitignore, .metadata, .packages, .flutter_basic.iml, pubspec.lock, pubspec.yaml, README.md.



Berikut adalah penjelasan yang lebih detail tentang struktur files dari flutter.

- .dart_tools : Konfigurasi untuk dart language
- .idea : Konfigurasi untuk android studio
- gitignore : File git yang digunakan untuk mengelola source code. Hal ini akan berguna jika developer sudah bekerja dengan git.
- metadata : File yang berisi metadata dari project
- packages : File yang berisi alamat path
- flutter_basic.iml: File yang berisi detail dari project.
- pubspec.lock : File yang berisi versi library atau package yang digunakan pada project yang degenerate sesuai dengan file pubspec.yaml.
- pubspec.yaml : File yang berisi library atau package yang dibutuhkan untuk pengembangan aplikasi.
- Readme.md : File markdown yang dapat digunakan untuk menjelaskan cara setup

aplikasi atau informasi penting yang perlu untuk diketahui oleh developer lain.

Untuk struktur folder akan dijelaskan pada subbab 2.1.2 sampai 2.1.7.

1.1.2 Multi os ios dan android

Perhatikan pada folder project flutter terdapat dua folder yaitu folder ios dan folder android, dengan menggunakan kedua folder tersebut flutter dapat membuat aplikasi berbasis ios dan berbasis android dalam satu project.

1.1.3 Folder android

Folder android berisi file file pendukung untuk mengenerate project android dan akan dikompilasi menjadi sebuah apk pada folder build. Namun anda jarang atau bahkan tidak perlu mengedit yang ada di folder android. Anda akan banyak bekerja di folder lib.

1.1.4 Folder ios

Berisi project ios, folder ini sama dengan folder android, sangat jarang dan bahkan tidak perlu untuk mengubah apapun pada folder ios. Folder ios dan android dikelola oleh flutter sdk yang akan dimerge (disatukan) dengan code yang ada di folder lib untuk membuat aplikasi ios dan android.

1.1.5 Folder lib

Folder lib berisi kode program dengan bahasa dart yang berupa widget yang dapat dibuat sesuai dengan kebutuhan aplikasi anda.

1.1.6 Folder test

Berisi source code dart yang digunakan untuk melakukan test secara otomatis pada aplikasi flutter.

1.1.7 Pubspec.yaml

Pada file ini berisi konfigurasi konfigurasi project flutter yang dibuat dimana anda dapat mendata asset berupa font, gambar dan lain lain. Pada file ini anda juga dapat mengkonfigurasi flutter sdk dan konfigurasi yang terkait flutter.

1.2 Flutter Hot Reload

Pada flutter terdapat fungsi hot reload dan hot restart yang digunakan untuk pengembangan aplikasi dengan flutter. Hote reload mencompile source code yang baru ditambahkan dan dikirimkan ke dart virtual machine diupdate. Setelah selesai update, dart virtual machine akan memperbarui UI sesuai dengan perubahan. Keunggulan hot reload adalah waktu prosenya cepat disbanding hot restart. Akan tetapi hot reload mempertahan state yang ada sehingga jika menggunakan state maka nilai dari widget tidak akan berubah.

1.3 Flutter Hot Restart

Hot restart akan mencompile ulang aplikasi dan mereset (destroy) state yang ada. Jadi hot restart akan membuild ulang widget tree sesuai dengan code yang telah diperbarui.

1.4 Bedah Hello World Project

1.4.1 Import Statement

Seperti halnya kode program pada umumnya dart dapat menggunakan statement import untuk mengimport package, library, atau file lain yang digunakan pada file yang dieksekusi.

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

1.4.2 Main function

Main function pada flutter dibuat dengan menggunakan kode program berikut ini dimana semua proses aplikasi dimulai dari mengeksekusi fungsi main.

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

Perhatikan pada fungsi main ini yang di eksekusi adalah class MyApp dimana class MyApp harus mengextend salah satu StatlessWidget atau StatefullWidget, ingat bahwa flutter membangun UI menggunakan widget.

1.4.3 Stateless Widget

Flutter menggunakan Widget sebagai elemen elemen pembangun UI, widget ini adalah kode program yang diterjemahkan menjadi tampilan yang dapat dilihat dan diinteraksikan oleh pengguna. Stateless widget bersifat statis / final dimana nilai atau konfigurasi telah diinisiasi sejak awal, nilai variabel pada widget ini tidak dapat diubah oleh widget ini sendiri tetapi dapat diubah oleh parent widget nya jika parent nya adalah StatefullWidget. Struktur dasar stateless widget adalah sebagai berikut:

```
class exampleStateless extends StatelessWidget{
  @override
  Widget build(BuildContext context) {

  }
}
```

1.4.4 Statefull Widget

Statefull widget bersifat dinamis, widget ini dapat diperbarui ketika dibutuhkan sesuai dengan action pengguna atau jika ada ada perubahan data. Perubahan data pada statefull widget di trigger oleh perubahan state oleh karena itu sebuah StatefullWidget selalu memiliki State. Struktur dasar statefull widget adalah sebagai berikut:

```
class exampleStateless extends StatefulWidget{
  @override
  State<StatefulWidget> createState() {
  }
}
```

1.5 Build in Widget

Pada framework flutter terdapat banyak widget. Widget yang telah disediakan dapat digunakan mengembangkan aplikasi yang mobile, desktop dan web yang memiliki tampilan menarik. Secara lebih detail widget yang tersedia di flutter dapat dicek pada https://flutter.dev/docs/development/ui/widgets. Berikut adalah beberapa contoh widget yang sering digunakan pada pembuatan aplikasi dengan flutter.

1.5.1 Text Widget

Text widget digunakan untuk menampilkan string yang dapat terdiri satu baris maupun beberapa baris. Contoh penggunaan text widget pada source code dan ouputnya adalah sebagai berikut:

```
class MyApp extends StatelessWidget {
      @override
      Widget build(BuildContext context) {
         return MaterialApp(
         home : Text('This is Text Widget',),
         );
```

```
        }
}
```

1.5.2 Image Widget

Image widget digunakan untuk menampilkan image. Contoh penggunaan image widget pada source code dan ouputnya adalah sebagai berikut:

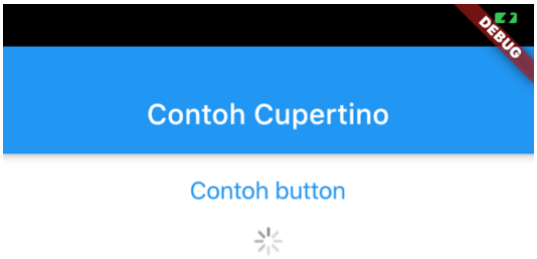
```
class MyApp extends StatelessWidget {
      @override
      Widget build(BuildContext context) {
         return MaterialApp(
         home:Image(image:  NetworkImage('https://flutter.github.io/assets-for-api-docs/assets/widgets/owl.jpg'),),
         );
      }
}
```

1.5.3 Material Design dan iOS Cupertino

Cupertino widget digunakan untuk mendesain sesuai dengan standar desain pada iOS. Contoh penggunaan cupertino widget pada source code dan ouputnya adalah sebagai berikut:

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

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Container(
        margin: EdgeInsets.only(top: 30),
        color: Colors.white,
        child: Column(
          children: <Widget>[
            AppBar(title: Text('Contoh Cupertino')),
            CupertinoButton(
              child: Text("Contoh button"),
              onPressed: () {},
            ),
            CupertinoActivityIndicator(),
```

```
            ],
          ),
        ),
      );
    }
}
```

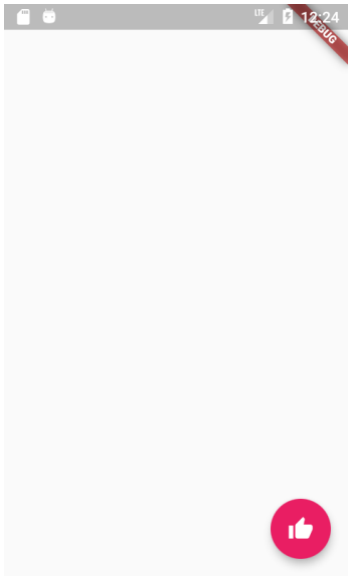


1.5.4 Button

Button widget terdapat beberapa macam pada flutter yaitu antara lain ButtonBar, DropdownButton, FlatButton, FloatingActionButton, IconButton, OutlineButton, PopupMenuButton, dan RaisedButton. Contoh penggunaan cupertino widget pada source code dan ouputnya adalah sebagai berikut:

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        floatingActionButton:FloatingActionButton(
```

```
          onPressed: () {
            // Add your onPressed code here!
          },
          child: Icon(Icons.thumb_up),
          backgroundColor: Colors.pink,
        ),
      ),
    );
  }
}
```

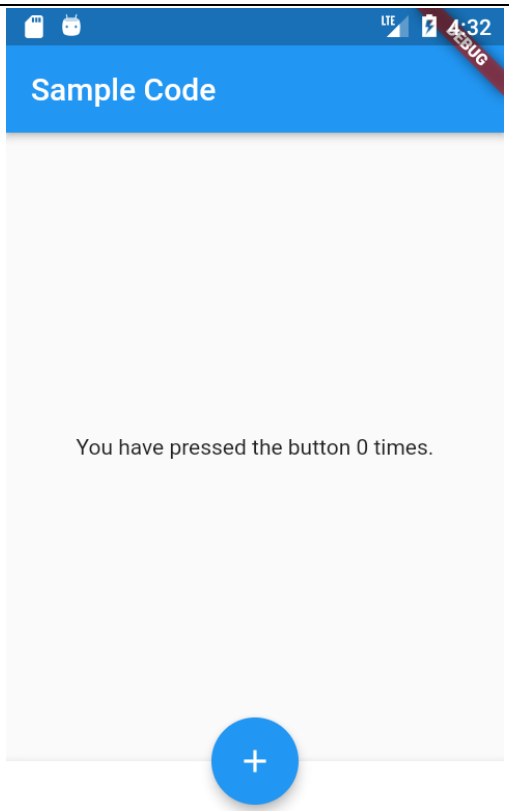


1.5.5 Scaffold

Scaffold widget digunakan untuk mengatur tata letak sesuai dengan material design. Contoh penggunaan scaffold widget pada source code dan ouputnya adalah sebagai berikut:

```
class MyApp extends StatelessWidget {

  int _count = 0;
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Sample Code'),
        ),
        body: Center(
          child: Text('You have pressed the button $_count times.'),
```

```
        ),
        bottomNavigationBar: BottomAppBar(
          child: Container(
            height: 50.0,
          ),
        ),
        floatingActionButton: FloatingActionButton(
          onPressed: () => 0,
          tooltip: 'Increment Counter',
          child: Icon(Icons.add),
        ),
        floatingActionButtonLocation:
FloatingActionButtonLocation.centerDocked,
      ),
    );
  }
}
```

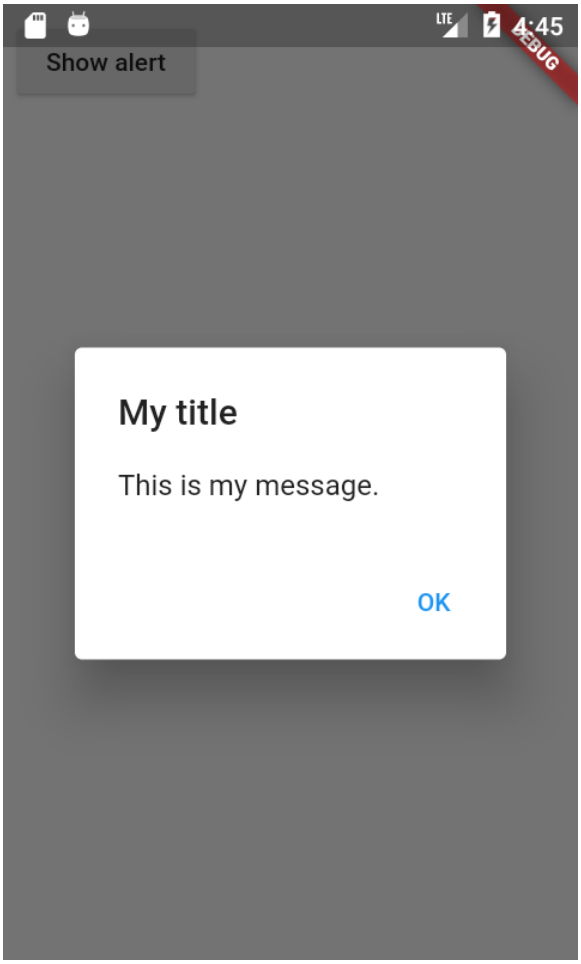


1.5.6 Dialog

Dialog widget pada flutter memiliki dua jenis dialog yaitu AlertDialog dan SimpleDialog. Contoh penggunaan AlertDialog widget pada source code dan ouputnya adalah sebagai berikut:

```
class MyApp extends StatelessWidget {
  int _count = 0;
  @override
  Widget build(BuildContext context) {
    return MaterialApp(

      home: Scaffold(
       body:MyLayout(),
      ),
    );
  }
}

class MyLayout extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.all(8.0),
      child: RaisedButton(
        child: Text('Show alert'),
        onPressed: () {
          showAlertDialog(context);
        },
      ),
    );
  }
}

showAlertDialog(BuildContext context) {

  // set up the button
  Widget okButton = FlatButton(
    child: Text("OK"),
    onPressed: () { },
```

```
  );

  // set up the AlertDialog
  AlertDialog alert = AlertDialog(
    title: Text("My title"),
    content: Text("This is my message."),
    actions: [
      okButton,
    ],
  );

  // show the dialog
  showDialog(
    context: context,
    builder: (BuildContext context) {
      return alert;
    },
  );
}
```

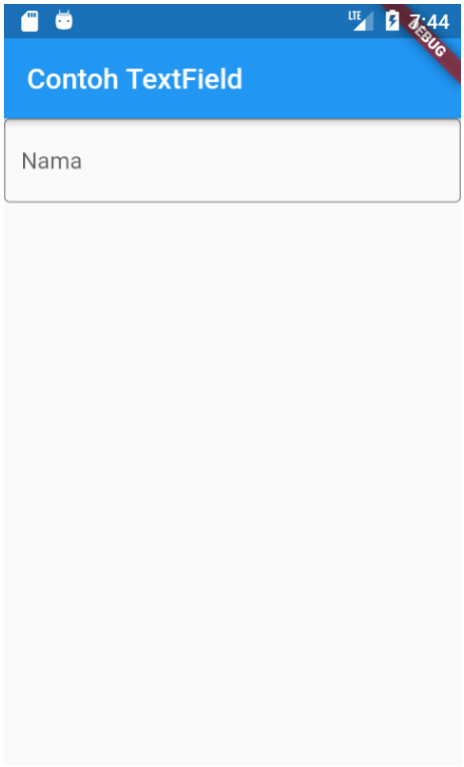


1.5.7 Input dan Selection Widget

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 pada source code dan ouputnya adalah sebagai berikut:

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(

      home: Scaffold(
       appBar: AppBar(
          title: Text("Contoh TextField")
        ),
       body: TextField(
              obscureText: false,
              decoration: InputDecoration(
                border: OutlineInputBorder(),
```

```
                labelText: 'Nama',
              ),
            ),
        ),
    );
  }
}
```



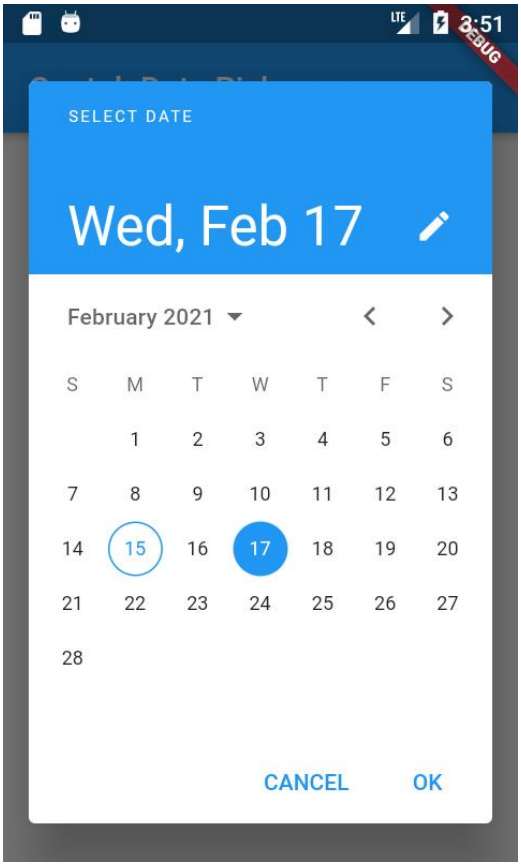
1.5.8 Date and Time Pickers

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(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Contoh Date Picker',
      home: MyHomePage(title: 'Contoh Date Picker'),
    );
```

```
  }
}

class MyHomePage extends StatefulWidget {
  MyHomePage({Key key, 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<Null> _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]),
            SizedBox(height: 20.0,),
            RaisedButton(
              onPressed: () => {
                _selectDate(context),
                print(selectedDate.day     +     selectedDate.month     +
selectedDate.year )
              },

              child: Text('Pilih Tanggal'),
            ),
```

```
            ],
          ),
        ),
      );
    }
}
```



1.6 Build in Layout Widget

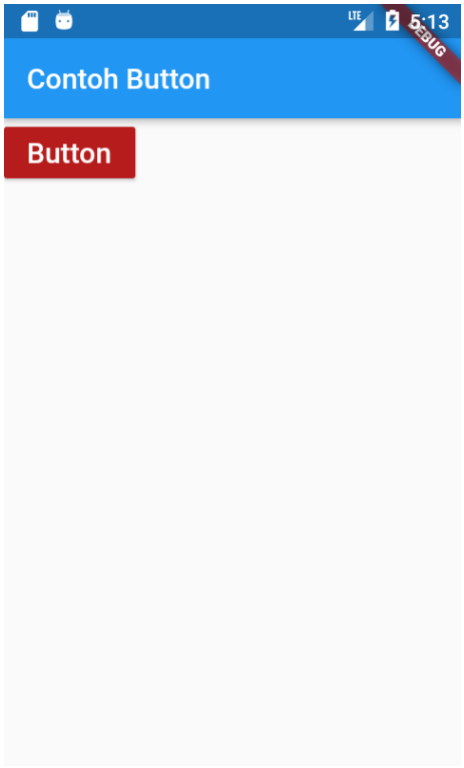
1.6.1 Container

Container widget berguna untuk menyimpan berbagai macam attribute dan menampung berbagai macam fungsi objek. Container merupakan **single child objek** yang artinya hanya dapat memiliki satu buah child widget, akan tetapi dalam sebuat container kita dapat menempatkan row, column, text dan container lain. Container memiliki beberapa property yaitu:

A. property child : digunakan untuk membuat menampung widget didalam container. Widget yang ditampung antara lain Text, Column, ListView, Buton dan lain sebagainya.

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
```

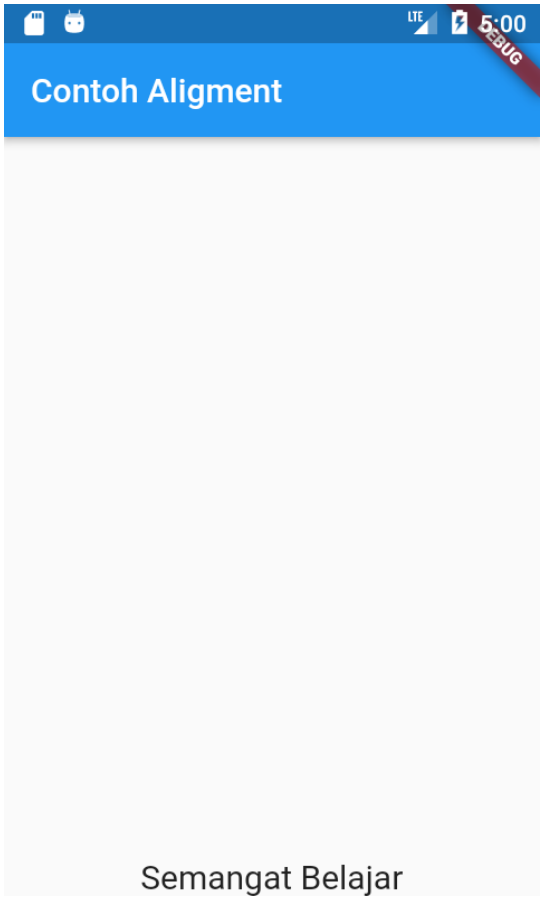
```
      return MaterialApp(
        home: Scaffold(
            appBar: AppBar(title: Text("Contoh Button")),
            body: Container(
                child: RaisedButton(
                  textColor: Colors.white,
                  onPressed: (){},
                  color: Colors.red[900],
                  child: Text(
                          "Button",
                          style: TextStyle(fontSize: 20)
                        ),
                ))),
      );
    }
  }
```



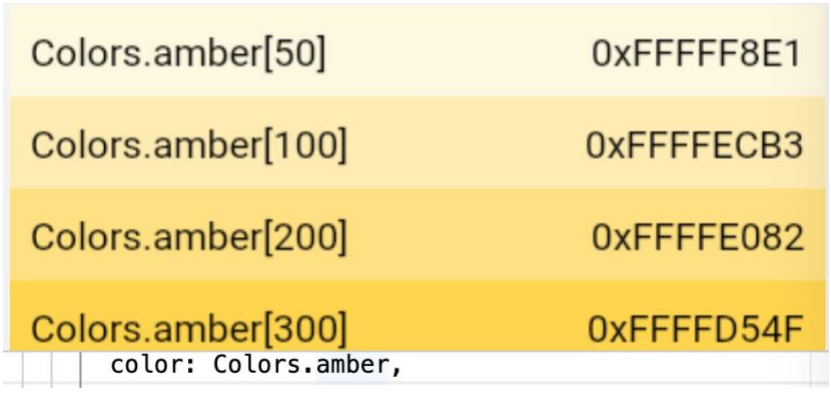
B. property alignment : mengatur posisi child widget menggunakan property Alignment.

```
  class MyApp extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
      return MaterialApp(
        home: Scaffold(
            appBar: AppBar(title: Text("Contoh Aligment")),
            body: Container(
                alignment: Alignment.bottomCenter,
                child: Text(
                  'Semangat Belajar',
```

```
                    style: TextStyle(
                      fontSize: 20,
                    ),
                  ))),
        );
      }
    }
```

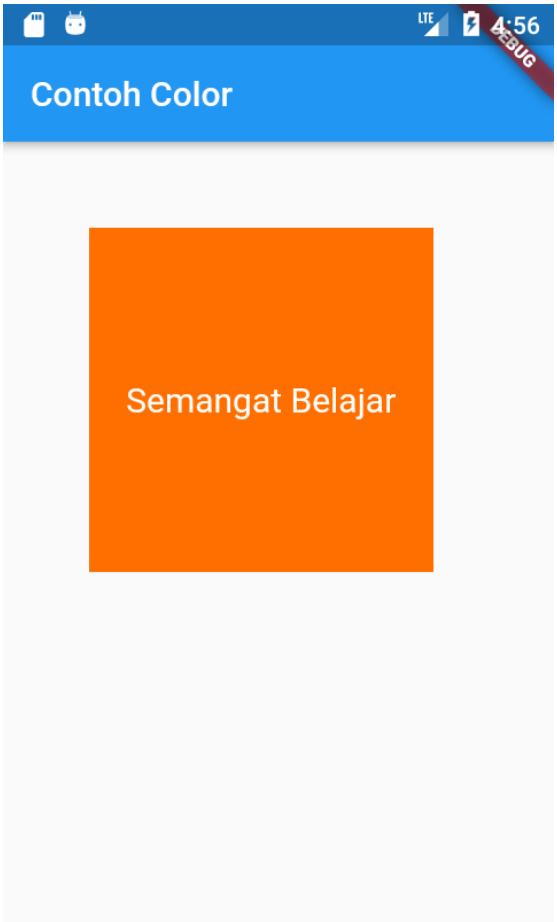


C. property color : digunakan untuk mengubah wara latar belakang container. Untuk memilih warna dapat menghover warna maka akan muncul pilihan warna yang dapat kita gunakan seperti berikut:



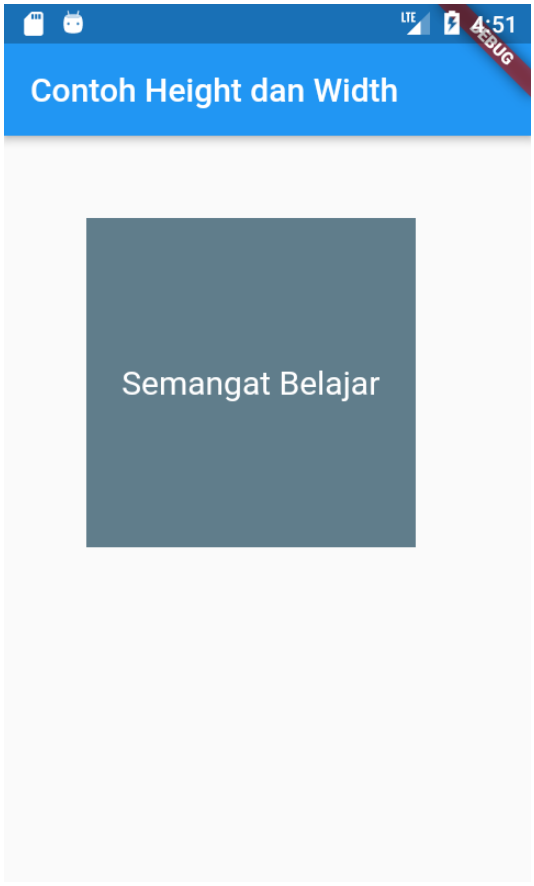
Contoh penggunaan color pada container adalah sebagai berikut:

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
          appBar: AppBar(title: Text("Contoh Color")),
          body: Container(
              margin: EdgeInsets.all(50),
              height: 200,
              width: 200,
              alignment: Alignment.center,
              color: Colors.amber[900],
              child: Text(
                'Semangat Belajar',
                style: TextStyle(fontSize: 20, color: Colors.white),
              ))),
    );
  }
}
```



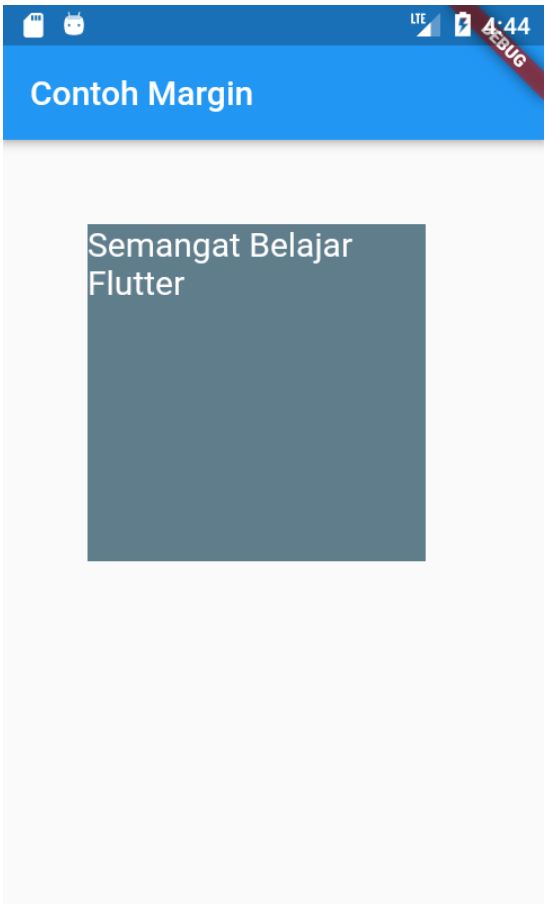
D. Property height dan width : Secara default ukuran container menyesuaikan dengan ukuran body layar, maka untuk mengatur layoutnya dapat menggunakan property height dan width.

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
          appBar: AppBar(title: Text("Contoh Height dan Width")),
          body: Container(
              margin: EdgeInsets.all(50),
              height: 200,
              width: 200,
              alignment: Alignment.center,
              color: Colors.blueGrey,
              child: Text(
                'Semangat Belajar',
                style: TextStyle(fontSize: 20, color: Colors.white),
              ))),
    );
  }
}
```



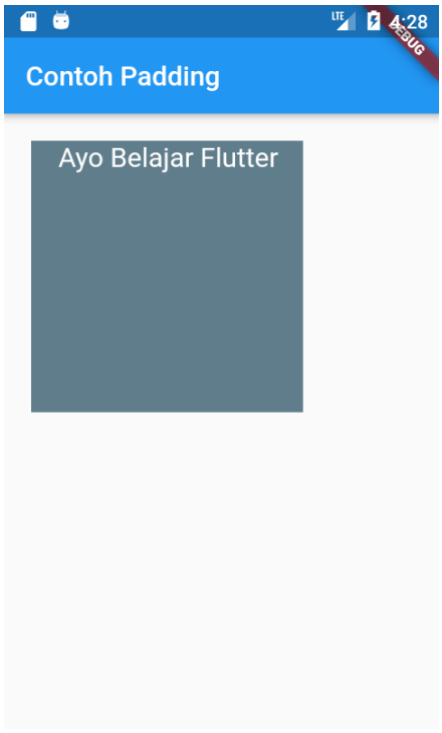
E. property margin : membuat jarak container dengan dengan widget lainnya.

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
          appBar: AppBar(title: Text("Contoh Margin")),
          body: Container(
              margin: EdgeInsets.all(50),
              height: 200,
              width: 200,
              alignment: Alignment.topLeft,
              color: Colors.blueGrey,
              child: Text(
                'Semangat Belajar Flutter',
                style: TextStyle(fontSize: 20, color: Colors.white),
              ))),
    );
  }
}
```



F. property padding : digunakan untuk menambahkan jarak antara container dengan widget yang ada didalam container tersebut.

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
          appBar: AppBar(title: Text("Contoh Padding")),
          body: Container(
              padding: EdgeInsets.only(left: 20),
              margin: EdgeInsets.all(20),
              height: 200,
              width: 200,
              alignment: Alignment.topLeft,
              color: Colors.blueGrey,
              child: Text(
                'Ayo Belajar Flutter',
                style: TextStyle(fontSize: 20, color: Colors.white),
              ))),
    );
  }
}
```



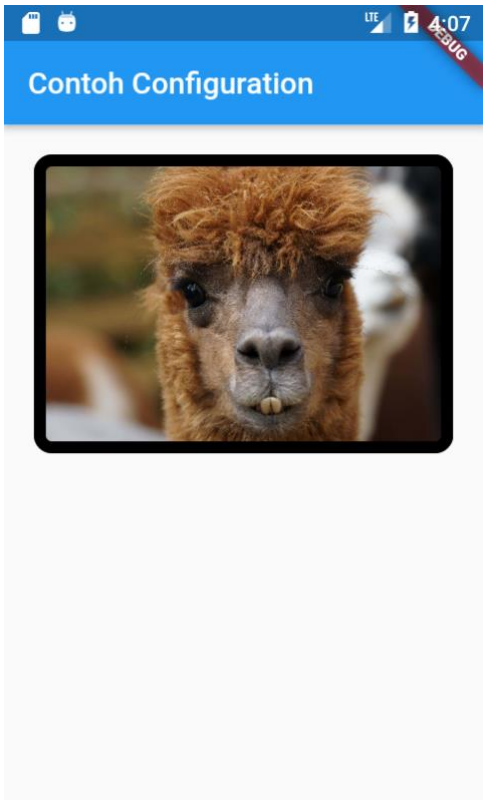
G. property transform : berfungsi untuk melakukan rotasi pada container dengan melakukan dari berbagai sumbu putar misalnya X,Y, dan Z.

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
          appBar: AppBar(title: Text("Contoh Transform")),
```

```
            body: Container(
                decoration: BoxDecoration(
                  image: const DecorationImage(
                    image: NetworkImage(
                        'https://pixnio.com/free-
images/2017/03/07/2017-03-07-10-59-39-900x600.jpg'),
                    fit: BoxFit.fitWidth,
                  ),
                  border: Border.all(
                    color: Colors.black,
                    width: 8,
                  ),
                ),
                height: 200,
                width: 300,
                margin: const EdgeInsets.only(left: 30.0, right: 30.0,
top: 30),
                transform: Matrix4.rotationZ(-0.1),
              )
          ),
      );
    }
}
```

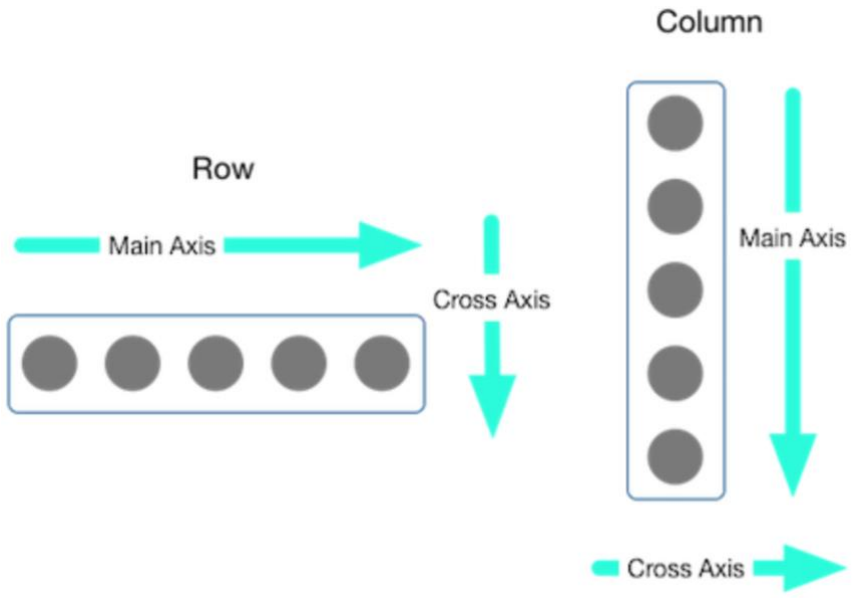
H. property decoration : untuk mencustom container dengan berbagai macam efek misalnya dengan mengubah warna border, memberikan gambar, dan membuat efek bayangan.

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
          appBar: AppBar(title: Text("Contoh Configuration")),
          body: Container(
              decoration: BoxDecoration(
                color: const Color(0xff7c94b6),
                image: const DecorationImage(
                  image: NetworkImage(
                      'https://pixnio.com/free-
images/2018/12/02/2018-12-02-19-17-12.jpg'),
                  fit: BoxFit.fitWidth,
                ),
                border: Border.all(
                  color: Colors.black,
                  width: 8,
                ),
                borderRadius: BorderRadius.circular(12),
              ),
              height: 200,
              width: 300,
              margin: EdgeInsets.all(20))),
    );
  }
}
```



1.6.2 Row dan Column

Column widget digunakan untuk mangatur tata letak widget secara vertikal. Sedangkan row digunakan untuk mengatur tata letak widget secara horizontal. Berikut adalah gambaran perbedaan row dan widget adalah sebagai berikut:

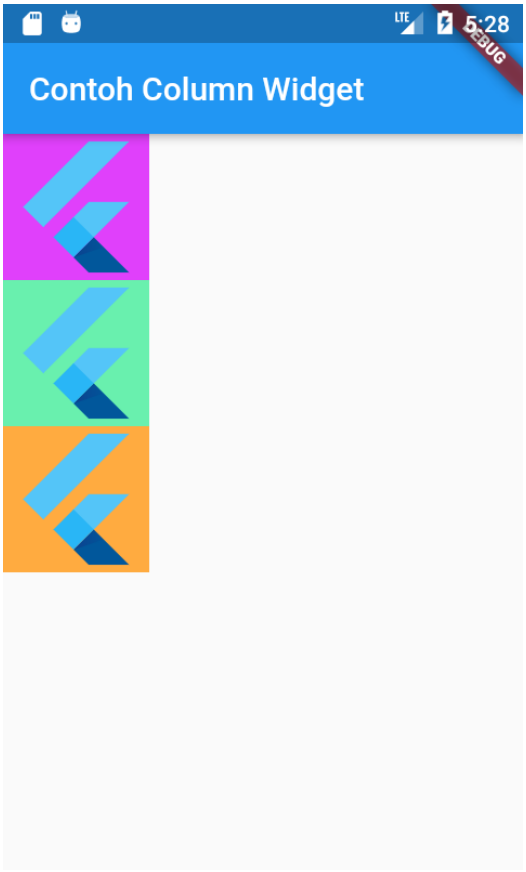


(sumber : https://flutter.dev/docs)

- Berikut adalah contoh penggunaan Column Widget

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Contoh Column Widget")),
        body: Column(
          children: [
            Container(
              color: Colors.purpleAccent,
              child: FlutterLogo(
                size: 90.0,
              ),
            ),
            Container(
              color: Colors.greenAccent,
              child: FlutterLogo(
                size: 90.0,
              ),
```

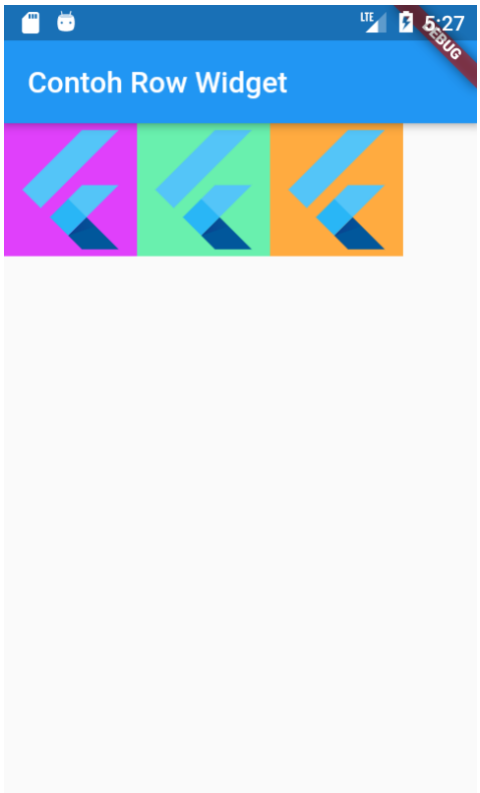
```
              ),
              Container(
                color: Colors.orangeAccent,
                child: FlutterLogo(
                  size: 90.0,
                ),
              ),
            ],
          ),
        ),
      );
    }
}
```



- Berikut adalah contoh penggunaan Row Widget

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Contoh Row Widget")),
        body: Row(
          children: [
            Container(
              color: Colors.purpleAccent,
```

```
                child: FlutterLogo(
                  size: 90.0,
                ),
              ),
              Container(
                color: Colors.greenAccent,
                child: FlutterLogo(
                  size: 90.0,
                ),
              ),
              Container(
                color: Colors.orangeAccent,
                child: FlutterLogo(
                  size: 90.0,
                ),
              ),
            ],
          ),
        ),
      );
    }
  }
```

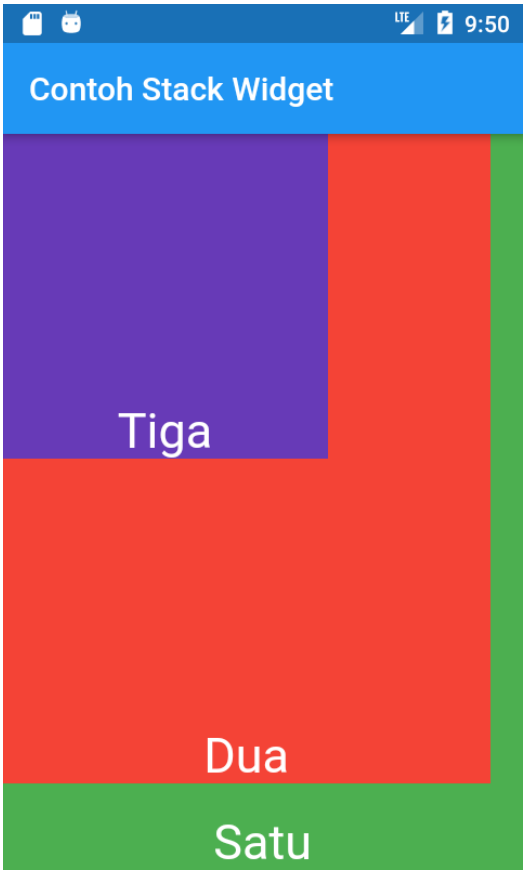


1.6.3 Stack

Stack Widget digunakan untuk menumpuk beberapa widget pada beberapa lapisan.

```
class MyApp extends StatelessWidget {
  @override
```

```
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(
          title: Text("Contoh Stack Widget"),
        ),
        body: Stack(
          children: <Widget>[
            Container(
              color: Colors.green,
              alignment:Alignment.bottomCenter,
              child:  Text("Satu",  style:  TextStyle(fontSize:30,color:
Colors.white)),
            ),
            Container(
              color: Colors.red,
              alignment:Alignment.bottomCenter,
              child:  Text("Dua",  style:  TextStyle(fontSize:30,color:
Colors.white)),
              height: 400.0,
              width: 300.0,
            ),
            Container(
              color: Colors.deepPurple,
              alignment:Alignment.bottomCenter,
              child:  Text("Tiga",  style:  TextStyle(fontSize:30,color:
Colors.white)),
              height: 200.0,
              width: 200.0,
            ),
          ],
        ),
      ),
    );
  }
}
```

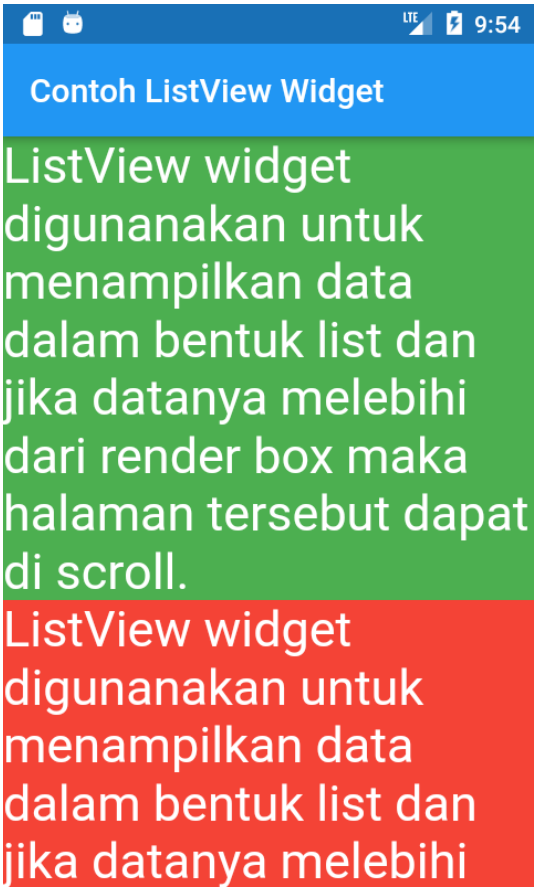


1.6.4 ListView

ListView widget digunanakan untuk menampilkan data dalam bentuk list dan jika datanya melebihi dari render box maka halaman tersebut dapat di scroll.

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(
          title: Text("Contoh ListView Widget"),
        ),
        body: ListView(
          children: <Widget>[
            Container(
              color: Colors.green,
              alignment:Alignment.topLeft,
              child: Text("ListView widget digunanakan untuk menampilkan data dalam bentuk list dan jika datanya melebihi dari render box maka halaman tersebut dapat di scroll.", style: TextStyle(fontSize:30,color: Colors.white)),
            ),
```

```
            Container(
              color: Colors.red,
              alignment:Alignment.topLeft,
              child: Text("ListView widget digunanakan untuk menampilkan
data dalam bentuk list dan jika datanya melebihi dari render box maka
halaman tersebut dapat di scroll.", style: TextStyle(fontSize:30,color:
Colors.white)),
              height: 400.0,
              width: 300.0,
            ),
            Container(
              color: Colors.deepPurple,
              alignment:Alignment.topLeft,
              child: Text("ListView widget digunanakan untuk menampilkan
data dalam bentuk list dan jika datanya melebihi dari render box maka
halaman tersebut dapat di scroll.", style: TextStyle(fontSize:30,color:
Colors.white)),
              height: 200.0,
              width: 200.0,
            ),
          ],
        ),
      ),
    );
  }
}
```

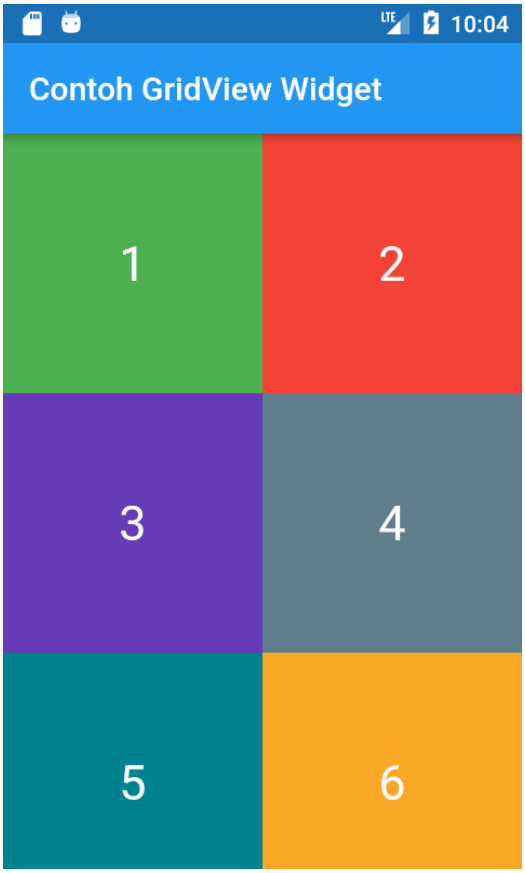


1.6.5 GridView

Gridview digunakan untuk menata tata letak widget pada list 2 dimensi. GridView juga secara otomatis menyediakan scrolling ketika konten melebihi render box.

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(
          title: Text("Contoh GridView Widget"),
        ),
        body: GridView.count(
          crossAxisCount: 2,
          children: <Widget>[
            Container(
              color: Colors.green,
              alignment:Alignment.center,
              child:    Text("1",    style:    TextStyle(fontSize:30,color:
Colors.white)),
            ),
```

```
            Container(
              color: Colors.red,
              alignment:Alignment.center,
              child:    Text("2",    style:    TextStyle(fontSize:30,color:
Colors.white)),
              height: 400.0,
              width: 300.0,
            ),
            Container(
              color: Colors.deepPurple,
              alignment:Alignment.center,
              child:    Text("3",    style:    TextStyle(fontSize:30,color:
Colors.white)),
              height: 200.0,
              width: 200.0,
            ),
            Container(
              color: Colors.blueGrey,
              alignment:Alignment.center,
              child:    Text("4",    style:    TextStyle(fontSize:30,color:
Colors.white)),
              height: 200.0,
              width: 200.0,
            ),
            Container(
              color: Colors.cyan[800],
              alignment:Alignment.center,
              child:    Text("5",    style:    TextStyle(fontSize:30,color:
Colors.white)),
              height: 200.0,
              width: 200.0,
            ),
            Container(
              color: Colors.yellow[800],
              alignment:Alignment.center,
              child:    Text("6",    style:    TextStyle(fontSize:30,color:
Colors.white)),
              height: 200.0,
              width: 200.0,
            ),
          ],
        ),
      ),
    );
  }
}
```



1.7 Praktikum 1 Membuat UI Sederhana

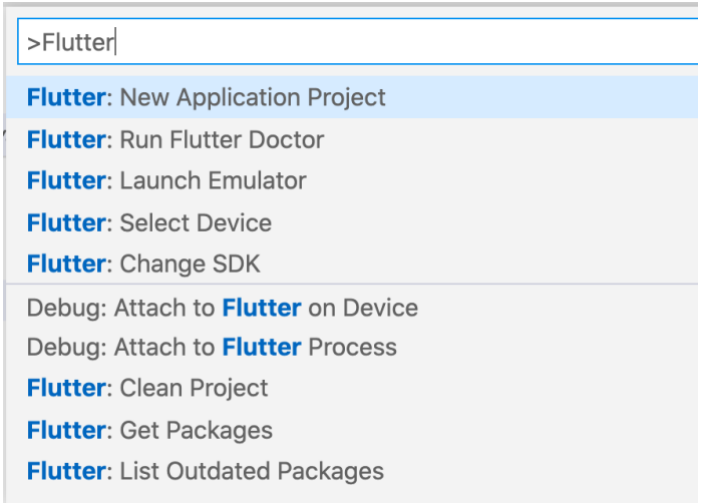
1.7.1 Alat dan Bahan

Pada praktikum pertama anda akan membuat sebuah project memanfaatkan widget flutter.

1. Visual Studio Code
2. Flutter SDK
3. Emulator atau device android

1.7.2 Langkah Praktikum

1. Buat project baru dengan menggunakan vscode misalnya dengan nama flutter_basic. Pada vscode dapat dilakukan dengan cara memilih menu '**View => Command Pallete** selanjutnya klik Flutter akan muncul tampilan seperti pada gambar di bawah ini. Selanjutnya pilih **Flutter: New Application Project** , anda akan diminta untuk menentukan path dimana project akan dibuat dan menentukan nama project di folder tersebut.



Untuk membuat project flutter dapat juga dilakukan pada terminal atau cmd dengan mengunakan perintah seperti berikut:

```
flutter create nama_project
```

2. Dengan informasi yang anda dapatkan dari teori diatas buatlah UI sederhana seperti tampilan dibawah ini menggunakan flutter.

8:11

MyApp

BERITA TERBARU PERTANDINGAN HARI INI

Costa Mendekat Ke Palmeiras

Transfer

Pique Bilang Wasit Untungkan Madrid, Koeman Tepok Jidat

Barcelona Feb 13, 2021

Pique Bilang Wasit Untungkan Madrid, Koeman Tepok Jidat