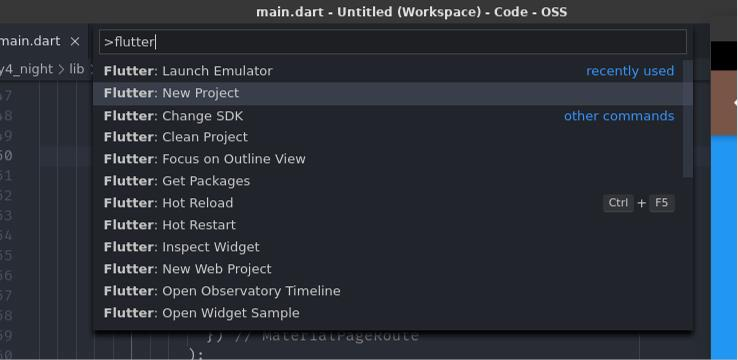
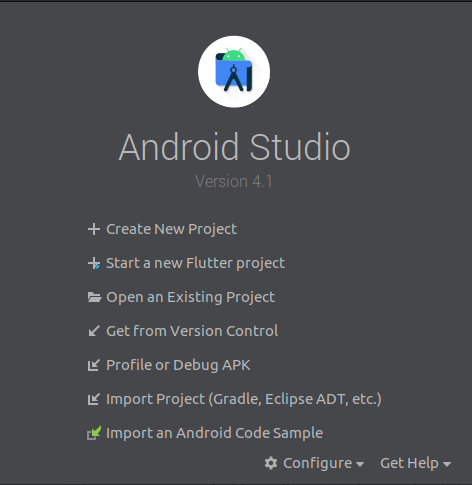
**6. Begin Create a Project!**

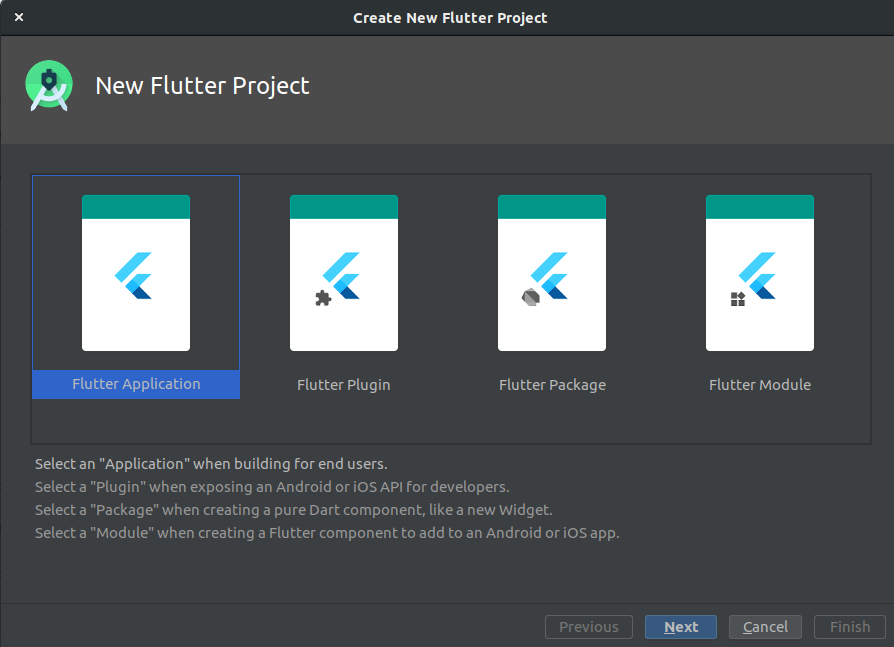
1. **Visual Studio Code** Untuk membuat flutter dapat menggunakan command melalui shortcut **Ctrl+Shift+P**, lalu pilih Flutter : New Project dan tunggu hingga proses selesai.



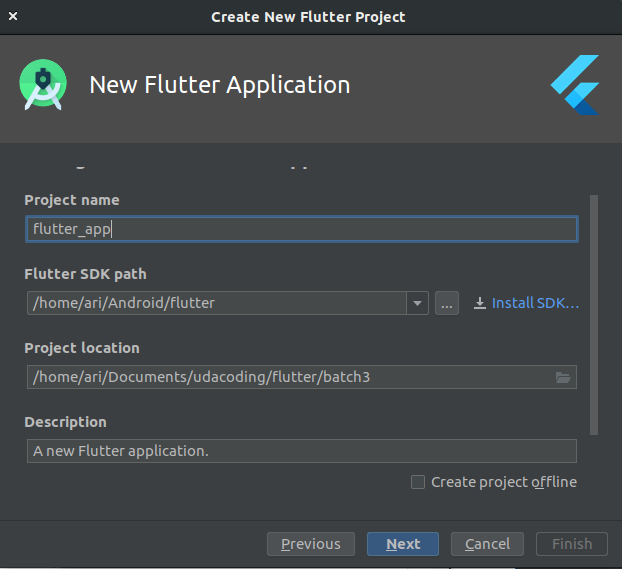
2. **Android Studio** ada beberapa langkah untuk membuat project flutter .



Pilih Start new Flutter project



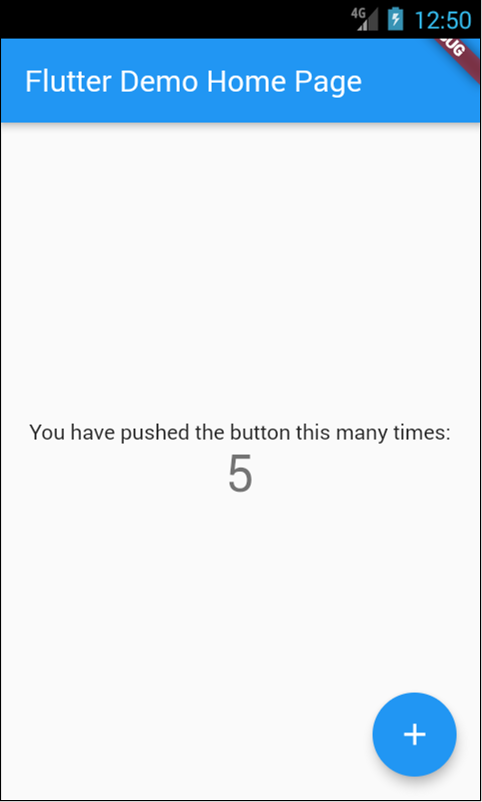
Klik Next pada Flutter Application



Buat nama project anda di Project name

Tentukan SDK anda pada Flutter SDK path

Pilih lokasi penyimpanan project anda pada Project Location



**A. Penjelasan Singkat**

import 'package:flutter/material.dart';

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

class **MyApp** extends **StatelessWidget** {

// This widget is the root of your application.

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Flutter Demo',

theme: ThemeData(

primarySwatch: Colors.blue,

),

home: MyHomePage(title: 'Flutter Demo Home Page'),

);

}

}

Pada kode tersebut kita melakukan import material.dart fungsinya agar bisa menggunakan widget-widget yang telah disediakan.

Lalu pada fungsi main() kita memanggil class MyApp dan melakukan extend StatelessWidget

class **MyHomePage** extends **StatefulWidget** {

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

final String title;

@override

\_MyHomePageState 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>[

Text(

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

),

Text(

'$\_counter',

style: Theme.of(context).textTheme.display1,

),

], ), ),

floatingActionButton: FloatingActionButton(

onPressed: \_incrementCounter,

tooltip: 'Increment',

child: Icon(Icons.add),

), // This trailing comma makes auto-formatting nicer for build

methods.

);

}

}

Pada class MyApp kita mengembalikan/return MaterialApp(), MaterialApp merupakan pondasi dari aplikasi kita berjalan nantinya. Lalu didalamnya terdapat property seperti title, theme, dan home, pada title kita isikan dengan nama aplikasi kita, pada theme kita bisa mengatur tema dari aplikasi kita, lalu pada home inilah kita mengisikan halaman awal pada saat aplikasi dijalankan, disini kita memanggil MyHomePage sebagai halaman awal kita. Pada MyHomePage kita memanggil \_MyHomePageState dan mererturn sebuah Scaffold, scaffold adalah Widget yang bisa kita isi dengan tampilantampilan umum pada aplikasi android seperti dukungan AppBar, bagian body, ataupun floating action button.

**B. Perbedaan Stateless dan Stateful**

Pada flutter ada dua widget yang akan sangat sering digunakan yaitu Stateless Widget dan StatefulWidget, lalu apa beda keduanya ?

* **StatefullWidget** adalah widget yang digunakan untuk menampilkan datadata yang dinamis atau datanya mengalami perubahan, seperti data dari database atau semisalnya.
* **StalessWidget** adalah widget yang berfungsi untuk menampilkan hal-hal yang sifatnya statis.

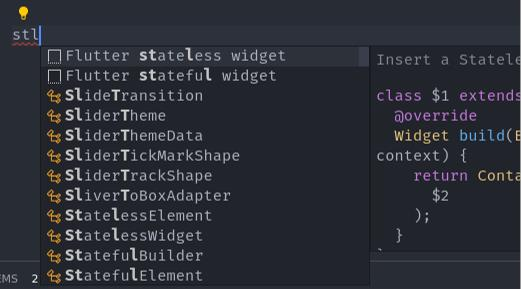
Sebagai contoh pada saat membuat project baru, maka sudah ada kode default yang apabila di run maka akan tampil seperti pada gambar sebelumnya

Pada aplikasi flutter tersebut terdapat 1 buah tombol dan terdapat text yang akan berubah dan melakukan penambahan angkanya pada saat mengklik tombolnya, hal ini bisa dilakukan apabila kita menggunakan Stateful widget, pada stateful widget terdapat fungsi setState yang artinya kita mengupdate sebuah nilai dari suatu variabel dan widget yang menggunakan atau menampilkan nilai tersebut akan di update tampilannya.

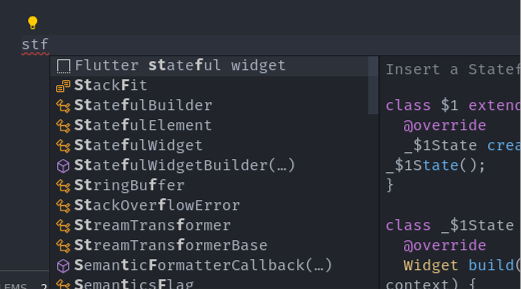
Sedangkan pada stateless tidak terdapat setState artinya widget-widget yang tampil akan statis.

Ada cara yang mudah untuk membuat stateless dan statefull, caranya cukup mudah yaitu dengan menggunakan autocomplete yang telah disediakan yaitu dengan mengetikkan:

**stl** = untuk stateless lalu enter



**stf** = untuk statefull lalu enter



**C. Property Child dan Children**

**1. Child**

Sesuai dengan namanya child, dalam bahasa indonesia artinya adalah anak berarti widget tersebut hanya bisa memiliki 1 widget di bawahnya.

Sebagai contoh :

class **MyHomePage** extends **StatelessWidget** {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Property Child"),

),

body: Container(

child: Text("Halo !!!"),

color: Colors.yellow,

padding: EdgeInsets.all(16.0),

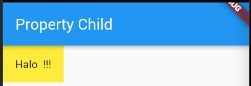
),

);

}

}

Hasilnya seperti berikut



Container memiliki properti child, sehingga hanya bisa menampung satu buah widget.

**2. Children**

Children atau anak-anak (jamak) berarti widget tersebut bisa berisi dengan banyak widget.

class **MyHomePage** extends **StatelessWidget** {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Property Children"),

),

body: Column(

children: <Widget>[

Container(

child: Text("Halo 1 !!!"),

color: Colors.lime,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 2 !!!"),

color: Colors.purple,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 3 !!!"),

color: Colors.lightBlue,

padding: EdgeInsets.all(16.0),

),

],

));

}

Hasilnya seperti berikut



**D. Widget Layout**

Untuk mengatur tata letak Terdapat dua widget yaitu Row dan Column

**1. Row**

Pada Row widget-widget akan tampil dengan arah horizontal atau sebaris, widget Row menggunakan property children artinya widget ini bisa diisi oleh banyak widget.

class **MyHomePage** extends **StatelessWidget** {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Row"),

),

body: Row(

mainAxisAlignment: MainAxisAlignment.spaceBetween,

children: <Widget>[

Container(

child: Text("Halo 1 !!!"),

color: Colors.lime,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 2 !!!"),

color: Colors.purple,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 3 !!!"),

color: Colors.lightBlue,

padding: EdgeInsets.all(16.0),

),

],

));

}

}

Hasilnya seperti berikut



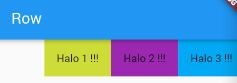
Kita juga bisa menambahkan pengaturan posisi menggunakan mainAxisAlignment, ada beberapa opsi diantaranya :

1. MainAxisAlignment.spaceBetween = akan tampil seperti diatas.

2. MainAxisAlignment.spaceEvenly = akan seperti berikut :



3. mainAxisAlignment: MainAxisAlignment.end = akanseperti berikut :



4. dan masih ada yang lain seperti center, start, end

**2. Column**

Sedangkan pada Column berlaku sebaliknya widget akan mengarah secara vertikal atau lurus kebawah.

Sebagai contoh :

class **MyHomePage** extends **StatelessWidget** {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Column"),

),

body: Column(

mainAxisAlignment: MainAxisAlignment.spaceBetween,

children: <Widget>[

Container(

child: Text("Halo 1 !!!"),

color: Colors.lime,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 2 !!!"),

color: Colors.purple,

padding: EdgeInsets.all(16.0),

),

Container(

child: Text("Halo 3 !!!"),

color: Colors.lightBlue,

padding: EdgeInsets.all(16.0),

),

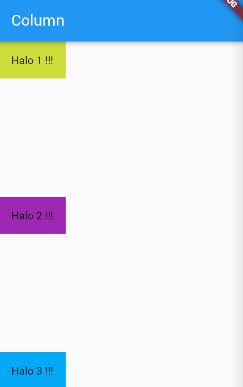
],

));

}

}

Hasilnya seperti berikut



Kita juga bisa mengatur mainAxisAlignment, bedanya hanya pada arahnya saja.