

# Mentoring Menjadi Master Flutter From Zero to Hero

## Week 1



#Zero to Hero

# Mentor

Dibimbing oleh Para Mentor yang berpengalaman di bidangnya



**Rizki Syaputra**

CEO & Founder  
at Udacoding



**Syahrizal Akbar**

Mobile Developer  
at Udacoding



**Yusmi Putra W**

Mobile Developer  
at Udacoding



**M Alamsyah**

Mobile Developer  
at Udacoding

[udacoding.com](https://udacoding.com)

## **BAB 1 : Basic Dart**

1.1. Pengenalan Dart

1.2. Mengapa Dart

## **BAB 2 : Perkenalan**

2.1. Sejarah Flutter

2.2. Kelebihan Flutter

2.3. Instalasi

2.4. Hal yang Perlu diketahui

2.5 Hot Reload

## **BAB 3: Memulai Project**

3.1. Penjelasan Singkat

3.2. Perbedaan Stateless dan Stateful

3.3. Property Child dan Children

3.3.1. Child

3.3.2. Children

3.4. Widget Layout

3.4.1. Row

3.4.2. Column

## **BAB 4 : Widget Umum**

4.1. Center

4.2. Text

4.3. Icon

4.4 Container

4.5. SizedBox

4.6. Button

4.7. Text Form Field

4.8. InkWell

## **Membuat Apk Android**

## Bab1. Basic Dart

### 1.1 Pengenalan Dart



# Dart

Dart 1.0 telah dirilis pada tanggal 14, November 2013 oleh Google dan didirikan oleh Lars Bak dan Kasper Lund. Ini bertujuan untuk membantu pengembang membangun aplikasi web dan mobile modern. Ini mencakup klien, server, dan sekarang seluler dengan Flutter. Hadir dengan berbagai alat termasuk mesin virtual, dan repositori manajemen paket, alat ini memberikan cukup amunisi bagi Anda untuk memulai proyek berikutnya.

Sebagai bahasa yang berorientasi object (object Oriented) dengan sintaksis (Syntax) C-style yang dapat diubah secara opsional menjadi JavaScript. Keunggulan yang sangat terlihat pada Dart yaitu mendukung berbagai macam alat bantu pemrograman seperti antarmuka (interface), class, collection, generics, dan opsional typing.

Alasan mengapa bahasa dart begitu cepat populer yaitu karena kita bisa menggunakan Dart untuk membuat aplikasi Web Android iOS dan juga menjalankan Server.

Mudahnya saat kita sedang menggunakan dart yaitu kita dapat membuat UI(User Interface) yang indah serta berkualitas pada setiap device dengan menggunakan:

- Bahasa yang mengoptimalkan client

Dart pertama kali dioptimalkan untuk web apps dan berevolusi untuk membantu pengembangan Mobile App. Dart juga dapat kita gunakan untuk menjalankan Command Line dan Server-Side.

- Kaya akan Framework

Baik untuk pembuatan aplikasi berbasis android maupun iOS secara maksimal fungsi yang tersedia pada framework flutter dapat digunakan dengan baik di 2 device tersebut.

- Tool yang Fleksibel dan menyenangkan

Flutter adalah tool yang sangat direkomendasikan oleh dart karena pada flutter bisa digunakan untuk berbagai tujuan.

## **1.2 Mengapa Dart**

Beberapa kelebihan bahasa pemrograman Dart menjadikan bahasa ini menjadi salah satu bahasa baru yang dapat dipelajari oleh para developer maupun calon developer. Developers yang bekerja di Google dan perusahaan besar lainnya menggunakan dart untuk membangun aplikasi Android iOS dan Web yang berkualitas. Dart memberikan fitur yang Client Side Development (Pengembangan dari sisi client) yang oleh karena inilah banyak developer yang memilih menggunakan Dart.

### **1. Mudah dipelajari**

- Dart memiliki banyak kemiripan dengan bahasa pemrograman yang banyak digunakan oleh developers (Java, C++, PHP, JavaScript...dll).
- Kita bisa jadi tanpa disadari sudah dapat menggunakan dart karena kemiripannya.
- Dart akan lebih mudah dipelajari jika kita sebelumnya sudah memiliki pengalaman dalam menggunakan bahasa pemrograman yang bersifat Object Oriented seperti Java ataupun C++.

### **2. CodeBase yang sudah di compile Natively (bawaan)**

- Framework lain memberikan kita sedikit akses untuk menggunakan codingan kita pada Platform yang berbeda. Berbeda dengan Dart.
- Dart memberikan kita izin penuh untuk membuat satu aplikasi yang codingannya dapat digunakan di berbagai platform. Aplikasi yang kita buat akan dapat digunakan pada Android juga iOS.
- Dart tidak hanya dapat kita gunakan untuk mobile develop kita juga dapat menggunakan Dart untuk Web Development.

### **3. Produktif**

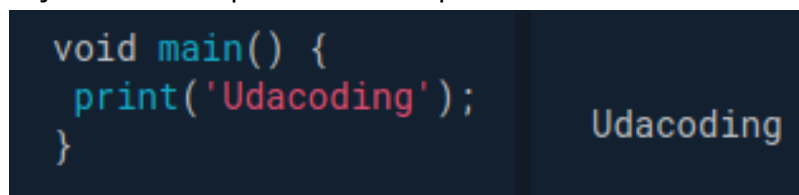
- Cepat dan mudah dalam Layouting dan menambahkan Feature pada Project.
- Layout juga dapat kita buat dengan menggunakan Codingan.

#### 4. Compile AoT dan JiT.

- Perubahan pada project dapat kita lihat secara Instan, di aplikasi.
- Tidak perlu melakukan Recompile yang memakan banyak waktu.
- Melihat perubahan tidak perlu untuk menunggu project di load.
- Anda hanya perlu untuk save dan perubahan akan terlihat.
- Ini dikarenakan Framework yang dapat meng compile Ahead of Time (AoT) / Lebih Cepat dan Just in Time(JiT) / Tepat waktu.

#### First Code on Dart

Dart menyediakan editor online di <https://dartpad.dartlang.org/>. Editor Dart dapat menjalankan skrip dan menampilkan HTML serta Console Output.



```
void main() {  
  print('Udacoding');  
}
```

Udacoding

**Fungsi Main()** Setiap aplikasi terdapat point dalam programnya yang berfungsi untuk masuk ke aplikasi. Ketika sebuah aplikasi dijalankan, dimulai dari point masuk yang ditentukan itu. Di dart, point untuk masuknya adalah fungsi main (). **print( )** adalah fungsi yang telah ditetapkan untuk mencetak string atau nilai tertentu ke output.

Berikut ini beberapa info yang perlu diketahui sebelum memulai Coding menggunakan Dart.

### Identifier pada Dart

Pengidentifikasi (identifier) adalah nama yang diberikan kepada elemen dalam program seperti variabel, fungsi, dll. Rules (Aturan) untuk identifier adalah

- Identifier dapat menyertakan karakter, dan digit(angka). Namun, identifier tidak dapat dimulai dengan digit.
- Identifier tidak dapat berisikan simbol-simbol spesial selain underscore ( \_ ) dan tanda dolar ( \$ ).
- Identifier tidak dapat berisikan Keyword \*
- Identifier harus unik - Identifier tidak dapat berisikan spasi(space)
- Berikut beberapa contoh identifier yang valid(dapat digunakan) dan invalid identifier (yang tidak dapat digunakan).

Valid	Invalid
firstName	Var
first_name	first name
num1	first-name
\$result	1number

## Keyword in Dart

Keywords memiliki makna spesial dalam context bahasa pemrograman. Dibawah ini adalah list dari Keyword yang ada pada Dart.

<a href="#">abstract</a> <sup>2</sup>	<a href="#">else</a>	<a href="#">import</a> <sup>2</sup>	<a href="#">super</a>
<a href="#">as</a> <sup>2</sup>	<a href="#">enum</a>	<a href="#">in</a>	<a href="#">switch</a>
<a href="#">assert</a>	<a href="#">export</a> <sup>2</sup>	<a href="#">interface</a> <sup>2</sup>	<a href="#">sync</a> <sup>1</sup>
<a href="#">async</a> <sup>1</sup>	<a href="#">extends</a>	<a href="#">is</a>	<a href="#">this</a>
<a href="#">await</a> <sup>3</sup>	<a href="#">extension</a> <sup>2</sup>	<a href="#">library</a> <sup>2</sup>	<a href="#">throw</a>
<a href="#">break</a>	<a href="#">external</a> <sup>2</sup>	<a href="#">mixin</a> <sup>2</sup>	<a href="#">true</a>
<a href="#">case</a>	<a href="#">factory</a> <sup>2</sup>	<a href="#">new</a>	<a href="#">try</a>
<a href="#">catch</a>	<a href="#">false</a>	<a href="#">null</a>	<a href="#">typedef</a> <sup>2</sup>
<a href="#">class</a>	<a href="#">final</a>	<a href="#">on</a> <sup>1</sup>	<a href="#">var</a>
<a href="#">const</a>	<a href="#">finally</a>	<a href="#">operator</a> <sup>2</sup>	<a href="#">void</a>
<a href="#">continue</a>	<a href="#">for</a>	<a href="#">part</a> <sup>2</sup>	<a href="#">while</a>
<a href="#">covariant</a> <sup>2</sup>	<a href="#">Function</a> <sup>2</sup>	<a href="#">rethrow</a>	<a href="#">with</a>
<a href="#">default</a>	<a href="#">get</a> <sup>2</sup>	<a href="#">return</a>	<a href="#">yield</a> <sup>3</sup>
<a href="#">deferred</a> <sup>2</sup>	<a href="#">hide</a> <sup>1</sup>	<a href="#">set</a> <sup>2</sup>	
<a href="#">do</a>	<a href="#">if</a>	<a href="#">show</a> <sup>1</sup>	
<a href="#">dynamic</a> <sup>2</sup>	<a href="#">implements</a> <sup>2</sup>	<a href="#">static</a> <sup>2</sup>	

## Whitespace dan Line Breaks

Dart mengabaikan spasi, tab, dan baris baru yang muncul dalam program. Kita dapat menggunakan spasi, tab, dan baris baru secara bebas dalam program kita dan kita bebas untuk memformat dan memasukkan program kita dengan cara yang rapi dan konsisten yang membuat kode mudah dibaca dan dipahami.

## Dart adalah case-sensitive

Ini berarti bahwa Dart membedakan antara huruf besar dan huruf kecil. Kesalahan penggunaan huruf besar dan huruf kecil sering terjadi pada pembuatan identifier dan class. Untuk itu harap lebih diperhatikan lagi dalam penamaan pada Dart!.

## Pernyataan (statement) diakhiri dengan Semicolon

Setiap baris instruksi disebut pernyataan. Setiap pernyataan Dart harus diakhiri dengan titik koma (;). Satu baris dapat berisi beberapa pernyataan. Namun, pernyataan ini harus dipisahkan dengan titik koma.

## Comment di Dart

Komentar (Comment) adalah cara untuk meningkatkan keterbacaan suatu program. Comment dapat digunakan untuk memasukkan informasi tambahan tentang program seperti penulis kode, petunjuk tentang fungsi / konstruksi dll. Comment akan diabaikan oleh compiler. Dart dapat berisikan jenis komentar berikut

- **Single Line Comment ( // )** ⇒ Teks apapun yang berada setelah " // " dalam satu line diperlakukan sebagai komentar
- **Multi Line Comments ( /\* \*/ )** ⇒ Teks apapun yang berada diantara "/\* \*/" diperlakukan sebagai komentar, Komentar ini dapat mencakup banyak line.

Single Line Comments	Multi Line Comments
<pre>void main(){ //Single Line Comment print( "Udacoding"); }</pre>	<pre>void main(){ /*Multi Line Comment */ print( "Udacoding"); }</pre>



## Syntax pada Dart

Syntax mendefinisikan seperangkat aturan untuk menulis program. Setiap spesifikasi bahasa mendefinisikan sintaksnya sendiri. Sintaks Dart terdiri dari:

- Variable
- Operators
- Class
- Function
- Loop
- Comments
- Library
- Package
- Typedefs
- Decision making
- Expression dan Programming Construct (konstruksi pemrograman)
- Data Structure yang direpresentasikan sebagai Generics/Collections

## Tipe Data Pada Dart

Salah satu karakteristik paling mendasar dari bahasa pemrograman adalah himpunan tipe data yang dimiliki oleh bahasa pemrograman. Ini adalah jenis nilai yang dapat direpresentasikan dan dimanipulasi dalam bahasa pemrograman.

Dart memiliki tipe-tipe data berikut :

- Numbers
- Strings
- Booleans
- Lists
- Maps

### ● Numbers

Numbers (Angka) dalam Dart digunakan untuk mewakili literal numerik.

Number pada Dart memiliki dua tipe :

- **Integer** sebuah tipe data dari Number yang nilainya berbentuk bilangan bulat. Nilainya harus bersifat angka dan tidak boleh ada koma.
- **Double** digunakan untuk merepresentasikan angka float / decimal.

Sintaks untuk menyatakan number adalah seperti yang diberikan di bawah ini :

```
int nama_var;    // mendeklarasikan variable integer
double nama_var; // mendeklarasikan variable variable
```

```
void main(){
  // deklarasi integer
  int num1 = 10;|
  print(num1); // print variable num1

  // deklarasi double value
  double num2 = 10.50;
  print(num2); // print variable num2
}
```

10  
10.5

---

*NOTE: Dart akan throw exception (memberikan error) jika nilai pecahan diletakkan pada variabel integer.*

---

## Parsing Number

Fungsi statis `parse()` memungkinkan penguraian string yang berisi numerik literal menjadi angka. contoh berikut menunjukkan hal yang sama :

```
void main() {
  print(num.parse('12'));
  print(num.parse('10.91'));
}
```

12  
10.91

## Property dari Number

Tabel berikut mencantumkan properti yang dimiliki oleh Number pada Dart

Property	Deskripsi	Contoh
<b>hashCode</b>	Me-return kode hash untuk nilai numerik.	<pre>void main() {   int n = 5000;   print(n.hashCode); }</pre>
<b>isFinite</b>	<b>true</b> , jika jumlahnya terbatas. <b>false</b> , jika jumlahnya tidak terbatas.	<pre>void main() {   int n = 5000;   print(n.isFinite); }</pre>
<b>isInfinite</b>	<b>true</b> , jika jumlahnya adalah infinity (tidak terbatas) positif atau tidak terbatasnya negatif. <b>false</b> , jika tidak infinity.	<pre>void main() {   int n = 5000;   print(n.isInfinite); }</pre>
<b>isNegative</b>	<b>true</b> , jika jumlahnya negatif. <b>false</b> , jika sebaliknya.	<pre>void main() {   int posNum = 10;   int posNeg = -10;   print(posNum.isNegative);   print(posNeg.isNegative); }</pre>
<b>sign</b>	Return minus satu, nol atau plus satu tergantung pada tanda dan nilai numerik dari angka tersebut.	<pre>void main() {   int posNum = 10;   int posNeg = -12;   int valZero = 0;   print(posNum.sign);   print(posNeg.sign);   print(valZero.sign); }</pre>
<b>isEven</b>	Me-return nilai true jika nomornya adalah bilangan genap.	<pre>void main() {   int posNum = 10;   print(posNum.isEven); }</pre>

<b>isOdd</b>	Me-return nilai true jika nomor tersebut adalah angka ganjil.	<pre>void main() {     int posNum = 10;     print(posNum.isOdd); }</pre>
--------------	---	--

## Method dari Number

Dibawah ini adalah List Method yang dapat kita gunakan pada Numbers

Method dan Deskripsi	Contoh
<b>abs</b> Return nilai absolute dari sebuah number.	<pre>void main() { var a = -2; print(a.abs()); }</pre>
<b>ceil</b> Return bilangan bulat terkecil yang tidak lebih kecil dari angka.	<pre>void main() { var a = 2.4; print("ceiling value dari 2.4 = + \${a.ceil()}"); }</pre>
<b>compare</b> Membandingkan number ini dengan number lain.	<pre>void main() { var a = 2.4;     print(a.compareTo(12)); print(a.compareTo(2.4));     print(a.compareTo(0)); }</pre>
<b>Floor</b> Return bilangan bulat terbesar tidak lebih besar dari angka saat ini.	<pre>void main() { var a = 2.9; print("floor value 2.9 = \${a.floor()}"); }</pre>
<b>remainder</b> Return sisa terpotong setelah membagi dua angka.	<pre>void main() { var a = 10; var b = 17;     print(a.remainder(2)); print(b.remainder(2)); }</pre>
<b>Round</b> Return integer terdekat dari number saat ini.	<pre>void main() { double n1 = 12.023; double n2 = 12.89;     var value = n1.round(); print(value);     value = n2.round(); print(value); }</pre>
<b>toDouble</b> Return double yang setara dengan number saat ini.	<pre>void main() { int n1 = 2; var value =     n1.toDouble(); print("Output = \${value}"); }</pre>

<b>toInt</b> Return int yang setara dengan number saat ini.	<i>void main() { int n1 = 2.0; var value = n1.toInt(); print("Output = \${value}"); }</i>
<b>toString</b> Return string yang setara dengan number saat ini.	<i>void main() { int n1 = 2; var value = n1.toString(); print(value is String); }</i>
<b>Truncate</b> Return integer setelah membuang digit pecahan apapun.	<i>void main() { double n1 = 2.123; var value = n1.truncate(); print("truncate value dari 2.123 = \${value}"); }</i>

### • Strings

Tipe data String mewakili urutan karakter. String Dart adalah urutan unit kode UTF 16 . Nilai string di Dart dapat direpresentasikan menggunakan kutip tunggal ( ' ' ) atau ganda ( " " ) atau tiga ( " " " " ). String baris tunggal direpresentasikan menggunakan tanda kutip tunggal atau ganda. Kutipan rangkap(kutip tiga) digunakan untuk mewakili string multi-line.

```
//kutip satu dan dua untuk string satu line (tunggal)
String variable_name = 'value';
String variable_name = "value";

//kutip rangkap (tiga) untuk string multi-line
String variable_name = '''line1
line2''';
String variable_name= '''line1
Line2'''''';
```

### Contoh

```
void main() {
  String str1 = 'string satu baris';
  String str2 = "string satu baris";
  String str3 = '''multiline string''';
  String str4 = """multiline string""";
  print(str1);
  print(str2);
  print(str3);
  print(str4);
}
```

```
string satu baris
string satu baris
multiline string
multiline string
```

## String Interpolation (interpolasi)

Proses membuat string baru dengan menambahkan nilai ke string statis disebut sebagai penggabungan atau interpolasi. Dengan kata lain, itu adalah proses menambahkan string ke string lain.

Operator plus ( + ) adalah mekanisme yang biasa digunakan untuk menyatukan / menyisipkan string.

```
void main() {
  String str1 = "hello";
  String str2 = "world";
  String tambah = str1+str2;
  print("string yang digabungkan :
  ${tambah}");
}
```

string yang digabungkan :  
helloworld

### Contoh 2

Kita dapat menggunakan " \$ { } ", dapat digunakan untuk menginterpolasi nilai ekspresi Dart dalam string. Contoh berikut menggambarkan hal yang sama.

```
void
main() {
  int n=1+1;
  String str1 = "1 tambah 1 sama dengan${n}";
  print(str1);
  String str2 = "2 tambah 2 sama dengan${2+2}";
  print(str2);
}
```

1 tambah 1 sama dengan2  
2 tambah 2 sama dengan4

## String Properties

Properti string pada tabel berikut hanya dapat dibaca(read-only properties).

Properti dan Deskripsinya	Contoh
<b>codeUnits</b> Me-return daftar unit kode UTF-16 yang tidak dapat dimodifikasi dari string ini.	<pre>void main() {   String str = "Hello";   print(str.codeUnits); }</pre>
<b>isEmpty</b> Return true jika string tidak memiliki value atau empty (kosong).	<pre>void main() {   String str = "Hello";   print(str.isEmpty); }</pre>

<p><b>Length</b></p> <p>Return panjang (jumlah karakter) pada string termasuk spasi, tab dan karakter baris baru (enter).</p>	<pre>void main() { String str = "Hello All"; print("Length dari string adalah \${str.length}"); }</pre>
---	---

## BAB 2 : Perkenalan



### 2.1. Sejarah Flutter

Flutter adalah framework untuk pengembangan aplikasi mobile open source yang dikembangkan oleh Google, versi pertama dari flutter dinamai dengan codename Sky dan berjalan pada sistem operasi Android, lalu pada Desember 2018 flutter stabil versi 1 dirilis.

### 2.2. Kelebihan Flutter

memiliki beberapa kelebihan yaitu :

1. **Hot Reload** merupakan sebuah fitur ketika developer mengembangkan sebuah aplikasi, lalu melakukan perubahan pada kode program, program akan otomatis mengupdate perubahan, maka dengan adanya hot reload kita tidak perlu melakukan build ulang, cukup save lalu akan otomatis mengupdate sesuai dengan apa yang telah kita ubah, sehingga sangat mempercepat proses pengembangan program.
2. **Multiplatform** artinya kode yang telah kita buat maka akan bisa dijalankan baik di ios, android, web, maupun desktop tanpa perlu melakukan porting secara manual.
3. **UI yang Cantik** UI flutter telah didesain dengan sangat baik, juga untuk menggunakannya sangat mudah sekali.





## 2.3. Instalasi

Untuk menggunakan flutter, maka ada 2 IDE yang direkomendasikan , yaitu Visual Studio Code dan Android Studio, untuk mendapatkannya download melalui link berikut :

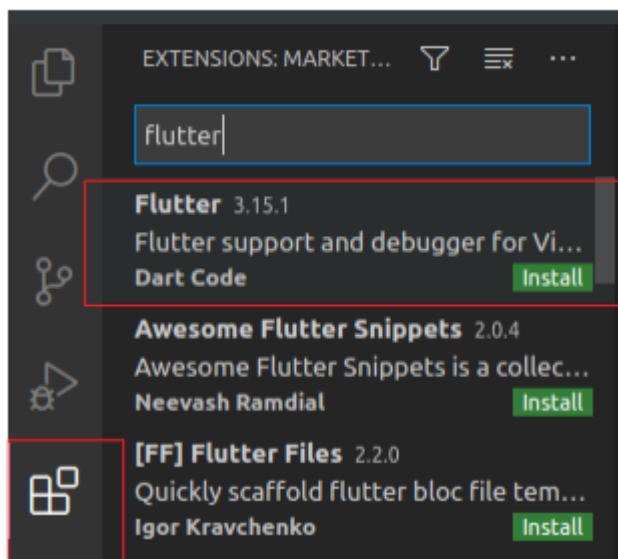
1. Android Studio : <https://developer.android.com/studio>
2. Visual Studio Code : <https://code.visualstudio.com/>

Setelah menginstall IDE maka diperlukan Flutter SDK, untuk mendapatkannya bisa didownload melalui link <https://flutter.dev/docs/get-started/install>

Setelah itu install plugin Flutter :

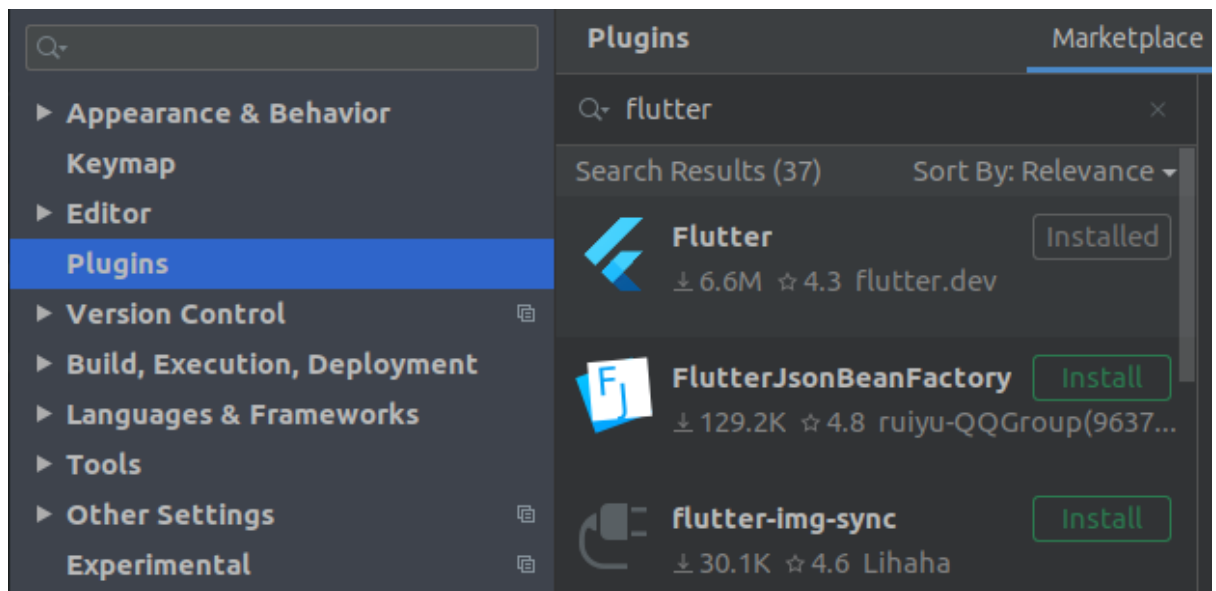
### 1. Visual Studio Code :

Install plugin melalui menu extension, lalu search Flutter dan install



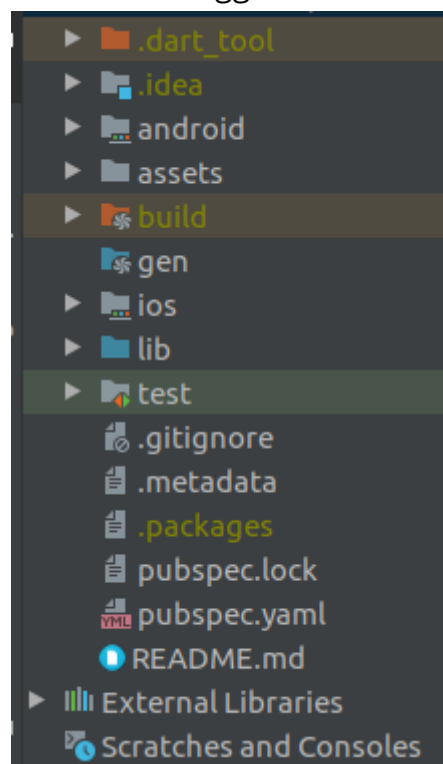
### 2. Android Studio

Buka Settings > Plugins > Marketplace, lalu search Flutter dan install



## 2.4. Hal yang Perlu diketahui

Hal yang perlu diketahui ketika menggunakan flutter yaitu :



1. **folder lib** pada folder lib inilah kode kita akan dibuat, dan biasanya di folder ini bisa berisi beberapa folder agar file kodingan kita bisa lebih rapi.
2. **pubspec.yaml** pada flutter semua gambar atau assets yang ditambahkan ke project maka dia wajib didaftarkan di file pubspec.yaml, file ini juga berfungsi apabila kita ingin menambah dependensi pada project kita, maka kita perlu menemukannya disini lalu setelah itu kita tinggal menjalankan perintah flutter pub get pada terminal.

**3. build.gradle** pada umumnya fungsinya untuk menambah dependensi seperti pada project android.

**4. Info.plist** Pada file ini berisi konfigurasi permission untuk ios.

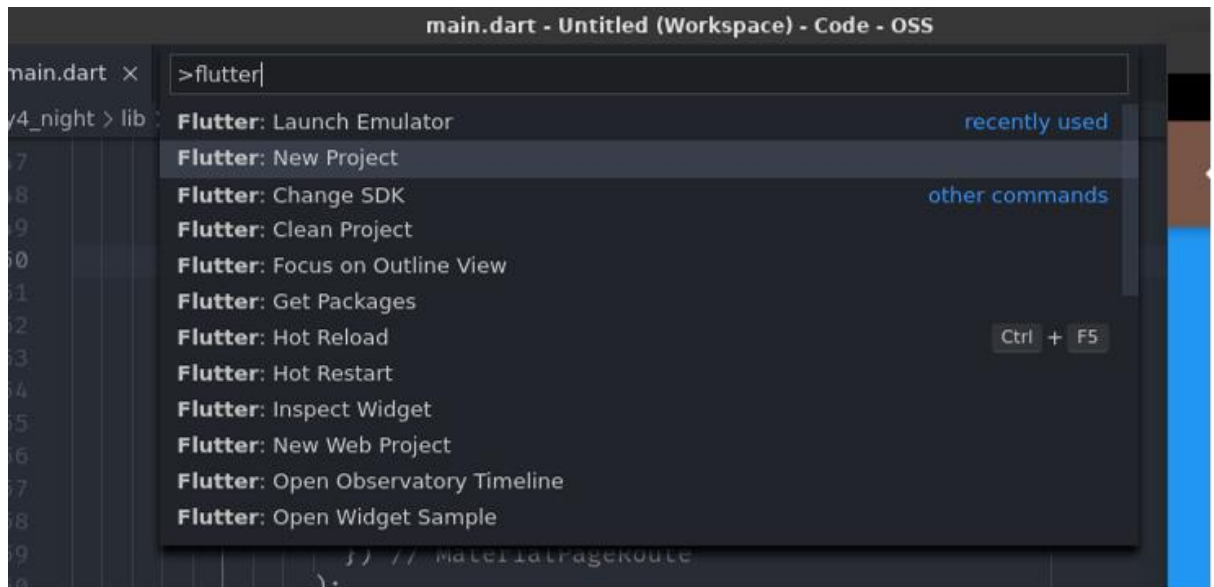
## **2.5 Hot Reload**

Hot reload flutter mempunyai fitur pengembangan cepat / hot reload, dimana saat kita memperbarui kode kita tidak perlu membuild aplikasinya. Sama halnya saat kita koding di web, hanya save dan lihat hasilnya,

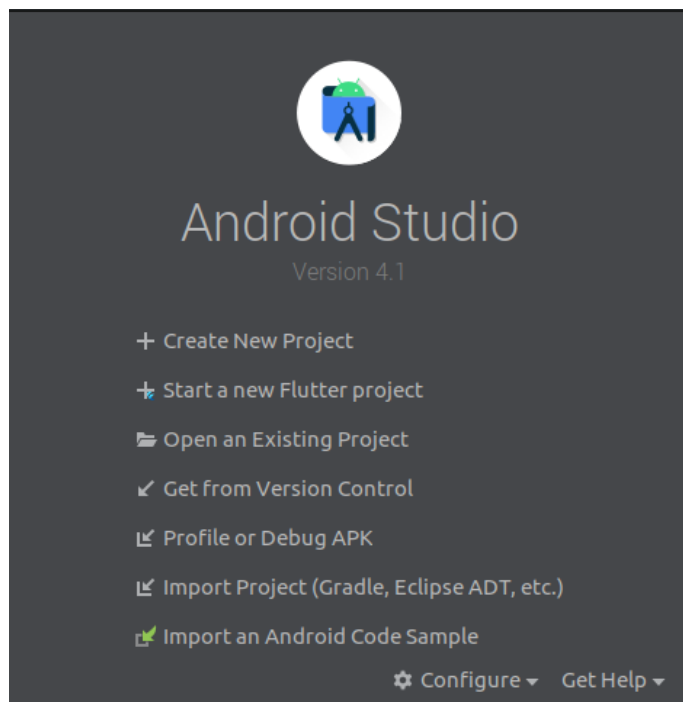
Untuk melakukan build project flutter, kita dapat menggunakan sistem operasi apapun seperti Windows, Linux, dan juga MacOS. Namun apabila kita ingin build project untuk iOS, maka diperlukan perangkat berupa MacBook, dan menggunakan XCode sebagai emulatoanya.

## BAB 3 : Memulai Project

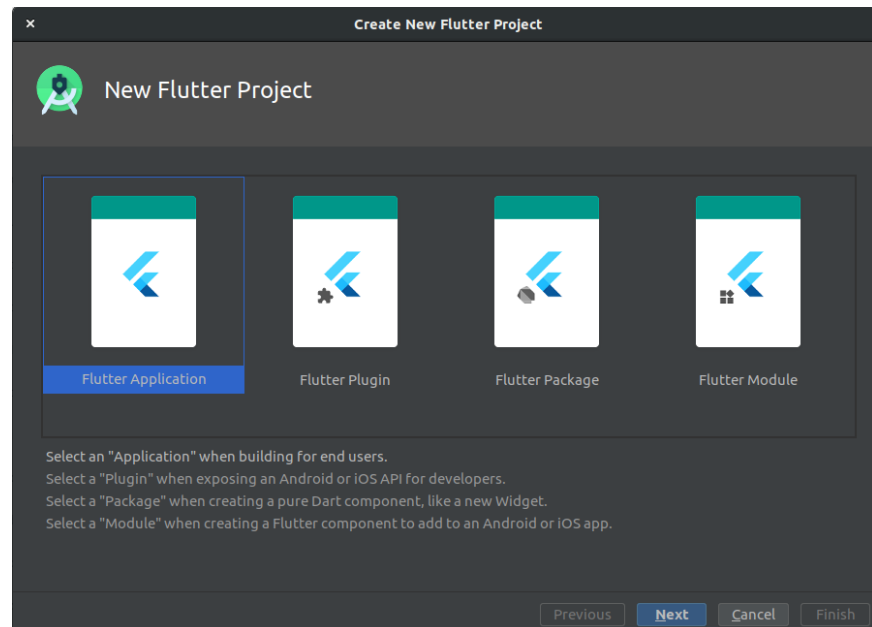
1. **Visual Studio Code** Untuk membuat flutter dapat menggunakan command melalui shortcut **Ctrl+Shift+P**, lalu pilih Flutter : New Project dan tunggu hingga proses selesai.



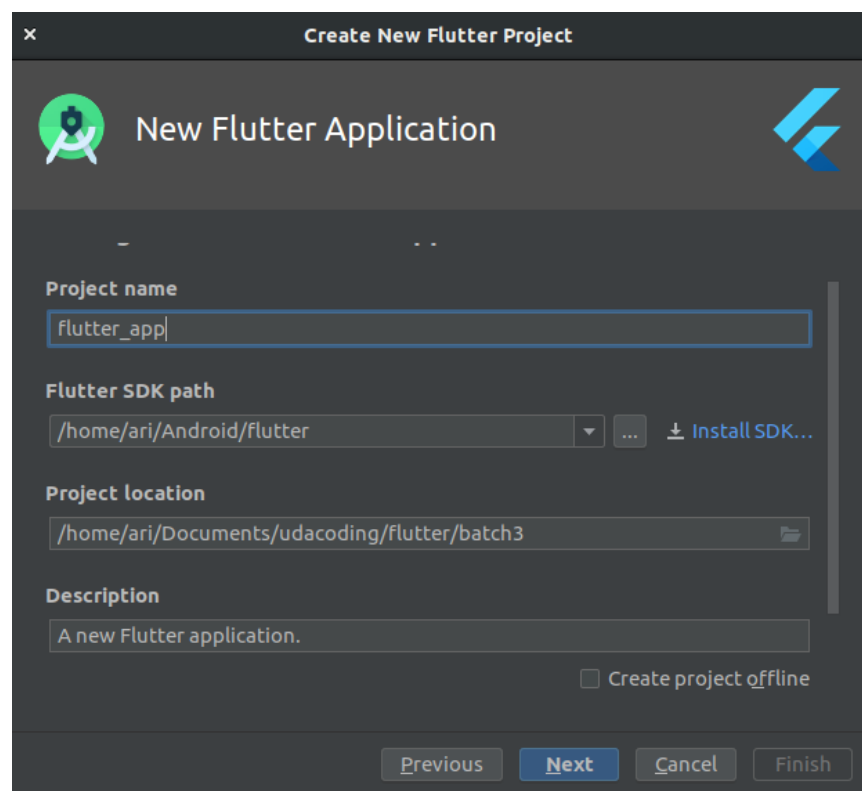
2. **Android Studio** ada beberapa langkah untuk membuat project flutter .



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



### 3.1. 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 mereturn sebuah Scaffold, scaffold adalah Widget yang bisa kita isi dengan tampilan-tampilan umum pada aplikasi android seperti dukungan AppBar, bagian body, ataupun floating action button.

### 3.2. 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 data-data yang dinamis atau datanya mengalami perubahan, seperti data dari database atau semisalnya.
- **StalelessWidget** 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

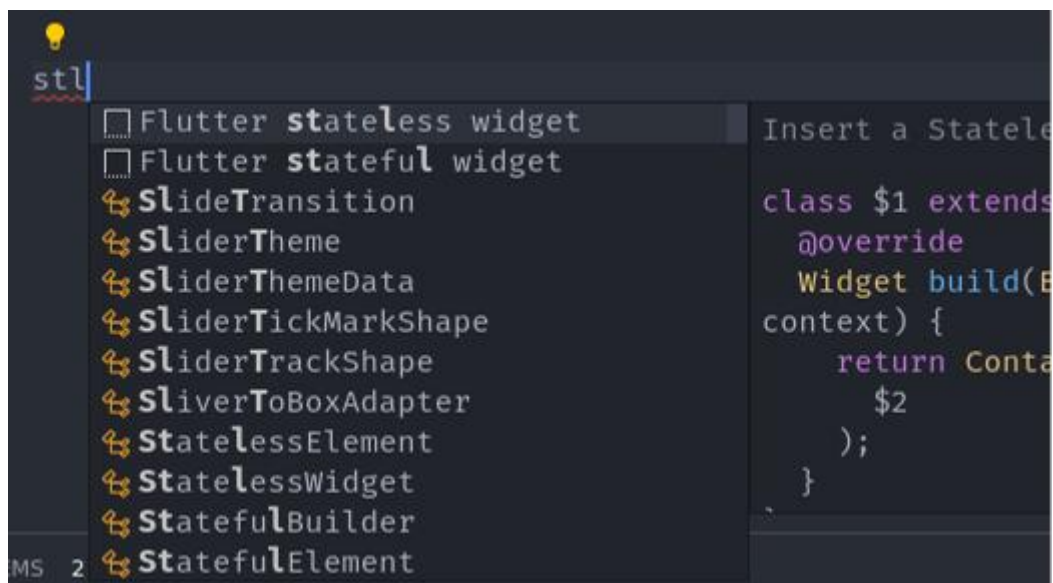
Pada aplikasi flutter tersebut terdapat 1 buah tombol dan terdapat text yang akan berubah dan melakukan penambahan angkanya pada saat mengklik tombolnya, hal ini bisa dilakukan apabila kita menggunakan Stateful widget, pada stateful widget terdapat fungsi setState yang artinya kita mengupdate sebuah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

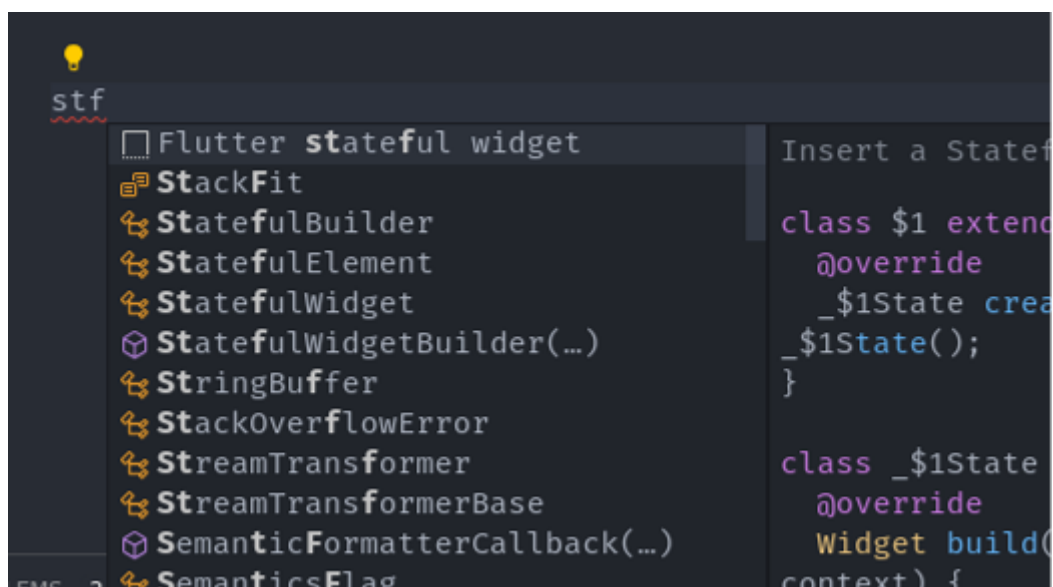
Ada cara yang mudah untuk membuat stateless dan statefull, caranya cukup mudah yaitu dengan menggunakan autocomplete yang telah disediakan yaitu dengan mengetikkan:



**stl** = untuk stateless lalu enter



**stf** = untuk statefull lalu enter



### 3.3. Property Child dan Children

#### 3.3.1. 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.

### 3.3.2. 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



### 3.4. Widget Layout

Untuk mengatur tata letak Terdapat dua widget yaitu Row dan Column

#### 3.4.1. 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

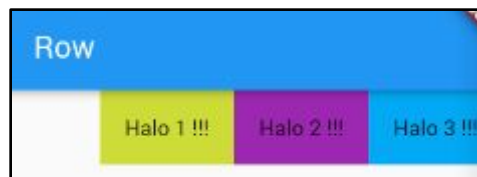


Kita juga bisa menambahkan pengaturan posisi menggunakan mainAxisAlignment, ada beberapa opsi diantaranya :

1. MainAxisAlignment.spaceBetween = akan tampil seperti diatas.
2. MainAxisAlignment.spaceEvenly = akan seperti berikut :



3. mainAxisAlignment: MainAxisAlignment.end = akan seperti berikut :



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

### 3.4.2. Column

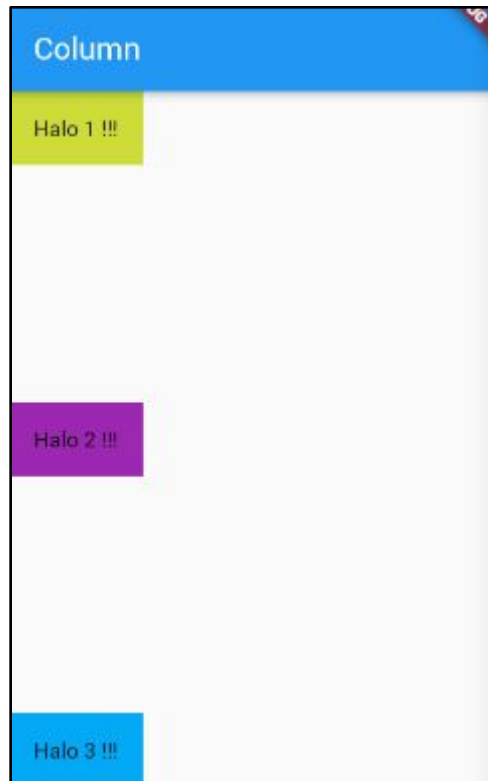
Sedangkan pada Column berlaku sebaliknya widget akan mengarah secara vertikal atau lurus kebawah.

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.

## BAB 4 : Widget Umum

### 4.1. Center

Center berfungsi untuk memposisikan widget yang kita buat berada ditengah.

```
...  
  body: Center(  
    child: Text("Text di tengah",  
      style: TextStyle(  
        fontSize: 30.0,  
        fontWeight: FontWeight.bold),  
      ),  
  )  
...
```

Hasilnya seperti berikut

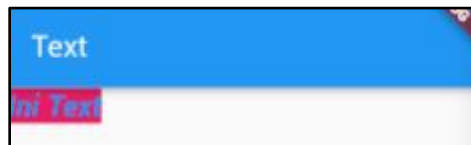


## 4.2. Text

Text berfungsi untuk menampilkan sebuah teks, atau bisa kita berikan style dengan menambahkan property style.

```
body: Text('Ini Text', style: TextStyle(  
    color: Colors.blue,  
    backgroundColor: Colors.pink,  
    fontSize: 20.0,  
    fontStyle: FontStyle.italic,  
    fontWeight: FontWeight.bold  
),)
```

Hasilnya seperti berikut



## 4.3. Icon

Widget ini untuk menggunakan icon yang telah disediakan, berikut adalah contohnya.

```
body: Container(  
  padding: EdgeInsets.all(16.0),  
  child: Row(  
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
    children: <Widget>[  
      Column(  
        children: <Widget>[  
          Icon(Icons.access_alarm),  
          Text('Alarm')  
        ],  
      ),  
      Column(  
        children: <Widget>[  
          Icon(Icons.phone),  
          Text('Phone')  
        ],  
      ),  
      Column(  
        children: <Widget>[
```

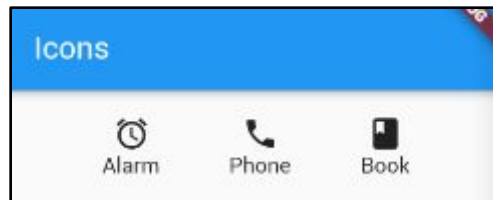


```

        Icon(Icons.book),
        Text('Book')
      ],
    ),
  ],
),
)

```

Hasilnya seperti berikut



#### 4.4. Container

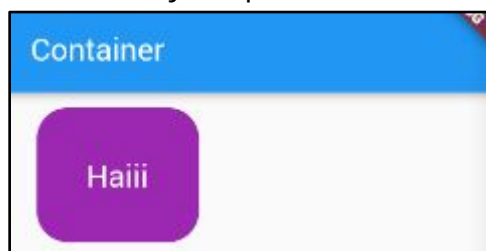
Container merupakan widget yang fungsinya untuk membungkus widget lain sehingga dapat diberikan nilai seperti margin, padding, warna background, atau dekorasi.

```

...
body: Container(
  padding: EdgeInsets.all(32.0),
  margin: EdgeInsets.fromLTRB(20.0, 10.0, 20.0, 0),
  decoration: BoxDecoration(
    borderRadius: BorderRadius.circular(20.0),
    color: Colors.purple,
    // color: Colors.purple,
    child: Text('Haiii', style: TextStyle(color:
Colors.white,fontSize: 20.0),)
  )
...

```

Hasilnya seperti berikut

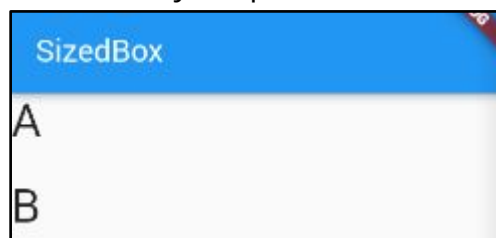


## 4.5. SizedBox

fungsi ini yaitu untuk menambahkan jarak baik secara vertikal atau horizontal tergantung nilai yang kita tentukan.

```
...
body: Column(
  children: <Widget>[
    Text("A", style: TextStyle(fontSize: 30.0),),
    SizedBox(height: 20.0,),
    Text("B", style: TextStyle(fontSize: 30.0),)
  ],
)
...
```

Hasilnya seperti berikut



## 4.6. Button

Terdapat 3 widget Button yang umumnya dipakai yaitu RaisedButton, MaterialButton, dan FlatButton

- Pada raised button dan Material tombol akan sedikit menonjol.
- Pada flat button tombolnya akan lebih datar tanpa adanya efek-efek seperti bayangan dan lain-lain.

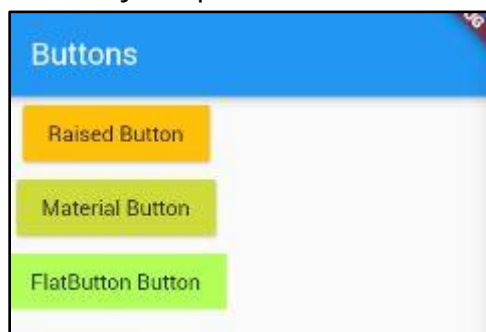
```
...
body: Column(
  children: <Widget>[
    RaisedButton(
      color: Colors.amber,
      child: Text("Raised Button"),
      onPressed: () {},
    ),
    MaterialButton(
      color: Colors.lime,
      child: Text("Material Button"),
      onPressed: () {},
    ),
  ],
)
```

```

    ),
    FlatButton(
      color: Colors.lightGreenAccent,
      child: Text("FlatButton Button"),
      onPressed: () {},
    ),
  ],
)
...

```

Hasilnya seperti berikut



#### 4.7. TextFormField

TextFormField merupakan widget yang berguna untuk membuat form untuk diisi user.

```

...
body: Padding(
  padding: const EdgeInsets.all(8.0),
  child: Form(
    child: Column(
      children: <Widget>[
        TextFormField(
          decoration: InputDecoration(hintText: "Username"),
        ),
        TextFormField(
          obscureText: true,
          decoration: InputDecoration(hintText: "Password"),
        ),
        RaisedButton(
          child: Text("Login"),
          onPressed: () {},

```

```

    ),
  ],
),
),
...

```

Hasilnya seperti berikut

#### 4.8. InkWell

Inkwell berguna untuk menambahkan action pada widget seperti action onTap misalnya :

```

body: Padding(
  padding: const EdgeInsets.all(8.0),
  child: InkWell(
    onTap: () {
      print("Ditekan");
    },
  ),
); // Scaffold

```

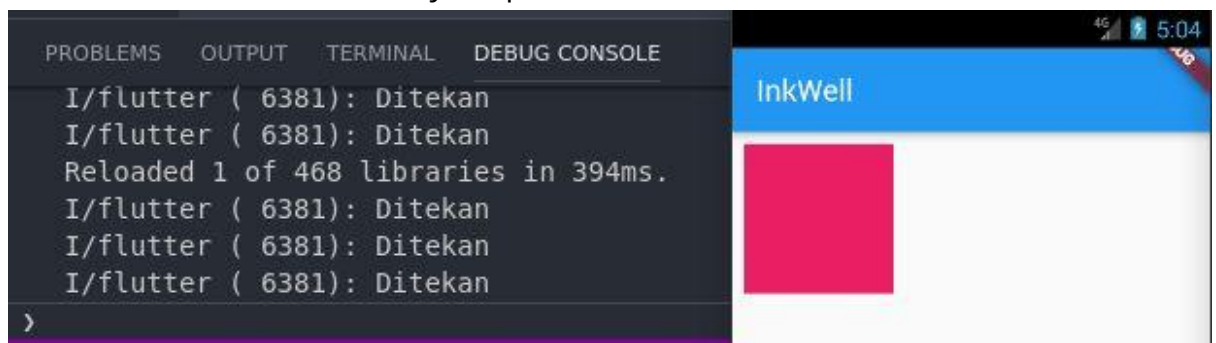
```

...
body: Padding(
  padding: const EdgeInsets.all(8.0),
  child: InkWell(
    onTap: () {
      print("Ditekan");
    },
  ),
);

```

```
child: Container(  
  width: 100.0,  
  height: 100.0,  
  color: Colors.pink,  
),  
)  
...  
)
```

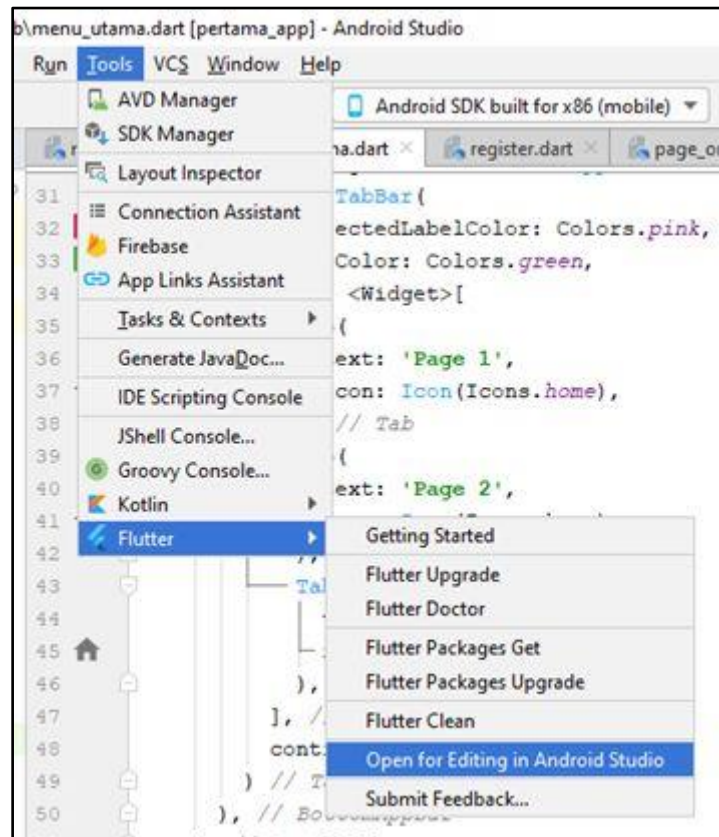
Hasilnya seperti berikut



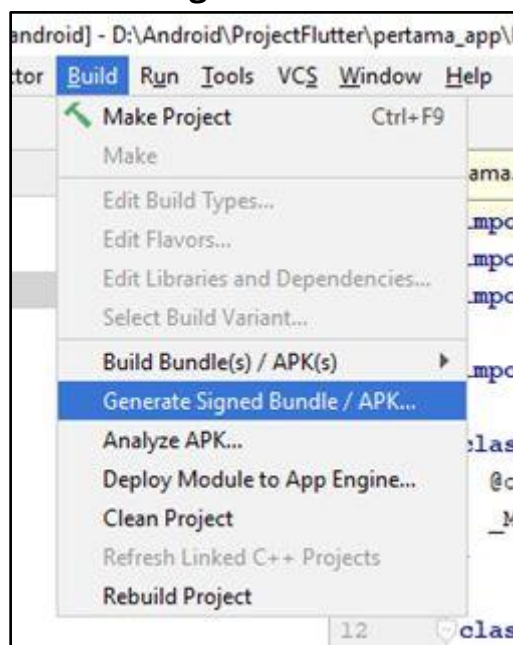
## BAB 5. MEMBUAT APK DARI PROJECT FLUTTER

Adapun langkah kerja yang dilakukan untuk membuat apk dari project flutter, yaitu sebagai berikut:

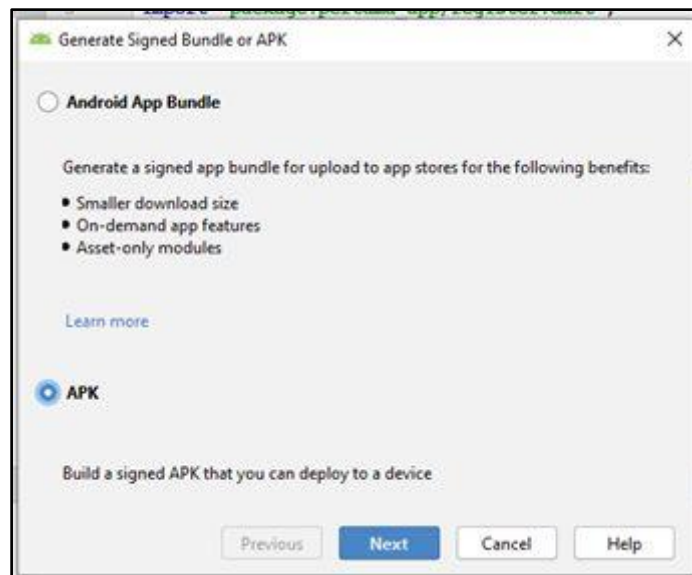
1. Buka project flutter di android studio, klik **Tools** kemudian pilih **flutter** dan pilih **Open For Editing In Android Studio**.



2. Pilih Build dan pilih **Generate Signed Bundle/APK**



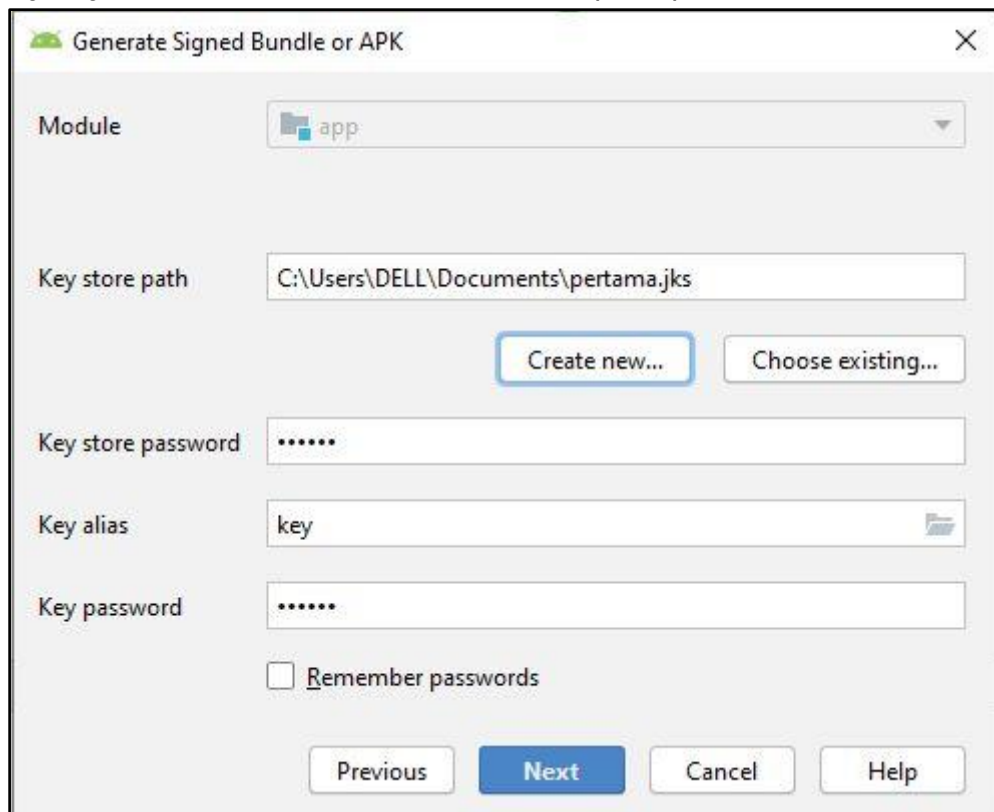
3. Selanjutnya pilih **APK**



4. Isi form dengan benar

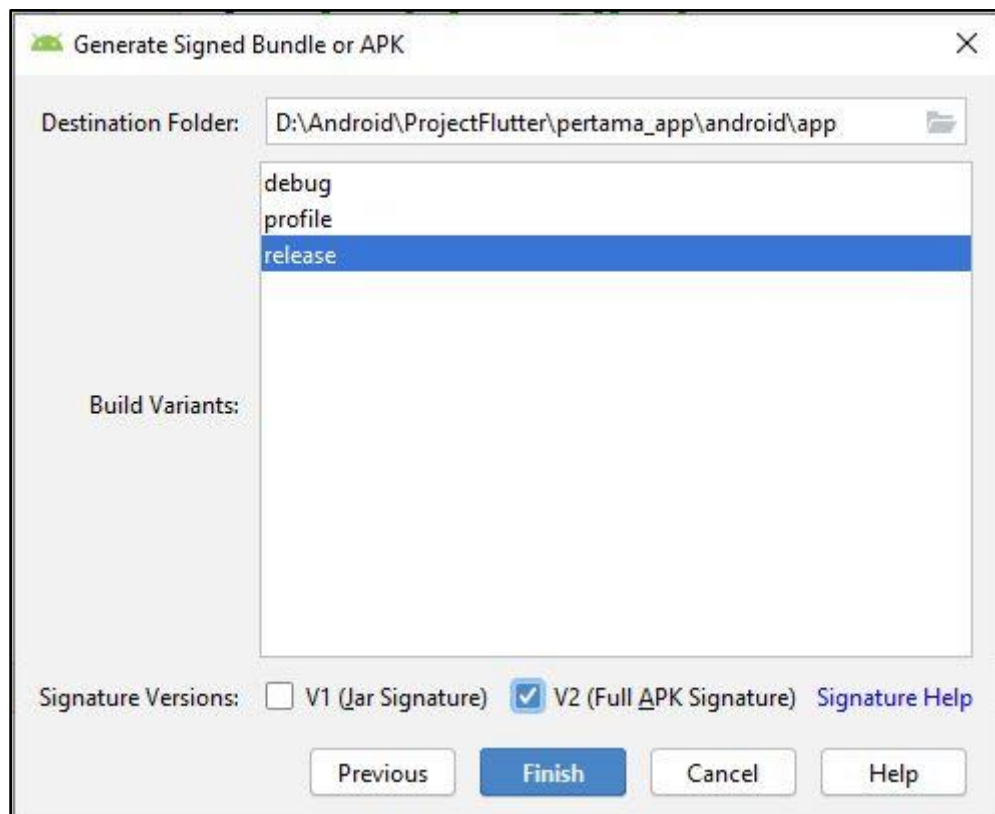
A screenshot of the 'New Key Store' dialog box. The title bar says 'New Key Store'. The form contains the following fields and values: 'Key store path:' with the value 'C:\Users\DELL\Documents\pertama.jks'; 'Password:' and 'Confirm:' fields both containing '\*\*\*\*\*'; 'Key' section with 'Alias:' set to 'key', 'Password:' and 'Confirm:' fields both containing '\*\*\*\*\*', and 'Validity (years):' set to '125'; 'Certificate' section with 'First and Last Name:' set to 'siti', 'Organizational Unit:' set to 'udacoding', and empty fields for 'Organization:', 'City or Locality:', 'State or Province:', and 'Country Code (XX):'. At the bottom right, there are 'OK' and 'Cancel' buttons.

5. Selanjutnya klik OK maka akan muncul tampil seperti berikut ini:

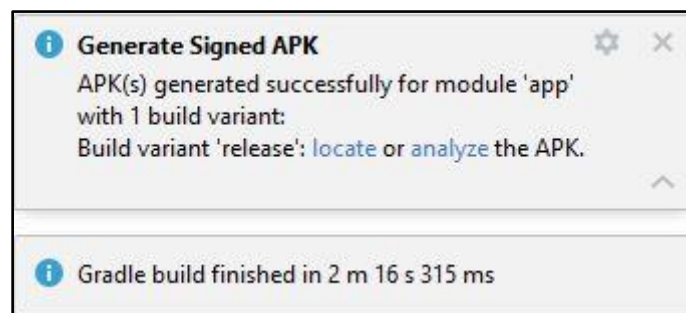


6. Klik Next. Selanjutnya pilih **Release**, hal ini bertujuan untuk release APK yang akan diupload pada playstore. dan klik salah satu Signature Versionnya.





7. Selanjutnya pilih locate untuk melihat APK release



8. Berikut adalah APK sudah release

