MOJADIPRO

MEMBANGUN FLUTTER MENGGUNAKAN STATE

MOJADIPRO

Materi yang akan dipelajari

01 Expanded Widget

02 Button Widget

03 Variabel Dart

04 Tipe Data Dart

05 Stateful dan Stateless Widget

MOJADIPRO

Materi yang akan dipelajari

06 SetState()

07 Dart Library

08 Fungsi Operator

09 Fungsi Dart Input Output

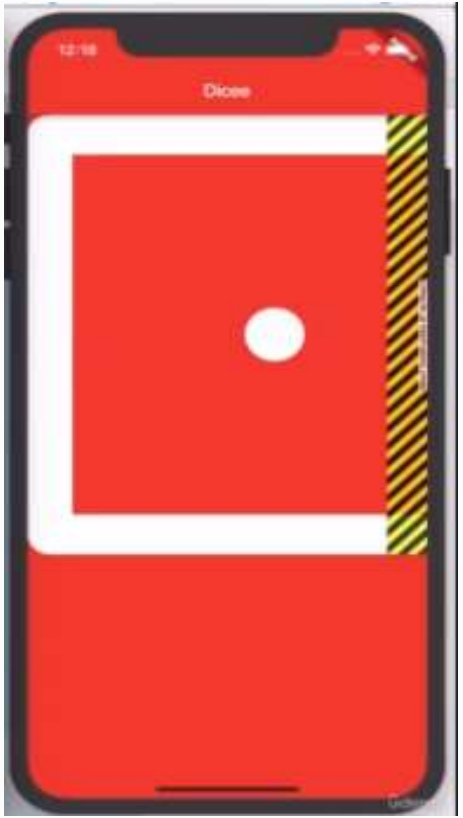
10 Dart Packages Dart Function

11 Audioplayer Packages

MOJADIPRO

Overflow

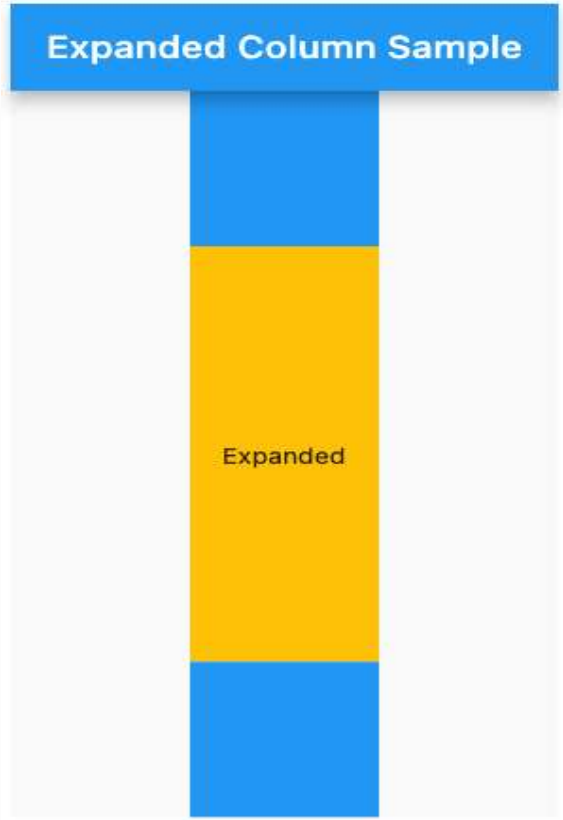
Overflow pada Flutter merujuk pada situasi di mana konten yang ditampilkan di dalam widget tidak cukup untuk mengakomodasi seluruh isi atau elemen yang diberikan. Ini dapat terjadi ketika ukuran widget atau ruang tampilan yang tersedia terlalu kecil untuk memuat semua elemen dengan benar.
Flutter menyediakan solusi salah satunya yaitu dengan menggunakan expanded widget.



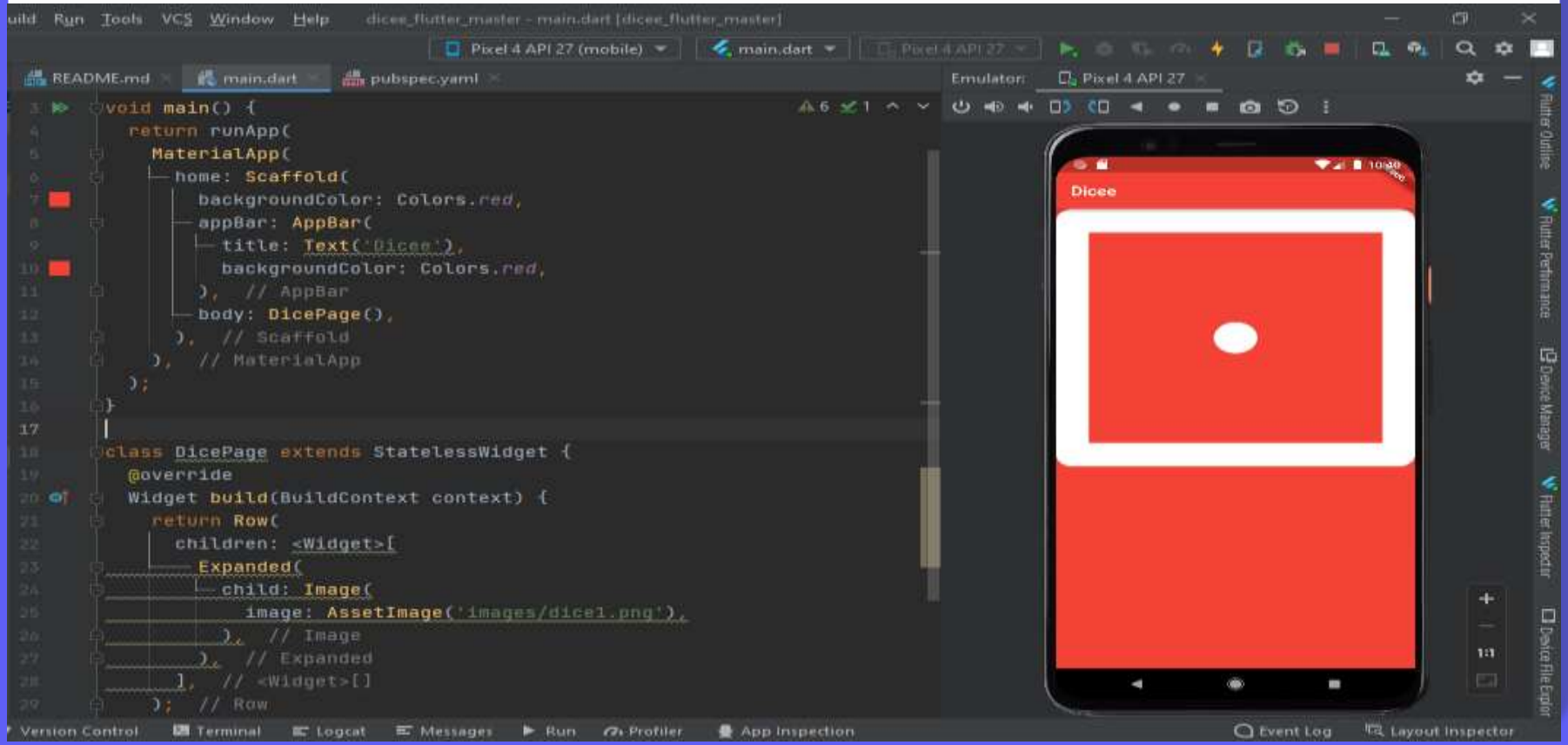
MOJADIPRO

Expanded Widget

Expanded adalah Widget yang digunakan untuk mengatur bagaimana widget harus mengisi ruang yang tersedia di sepanjang sumbu utama (secara horizontal untuk Baris atau vertikal untuk Kolom).



Contoh Penggunaan Kode Pada Expanded Widget

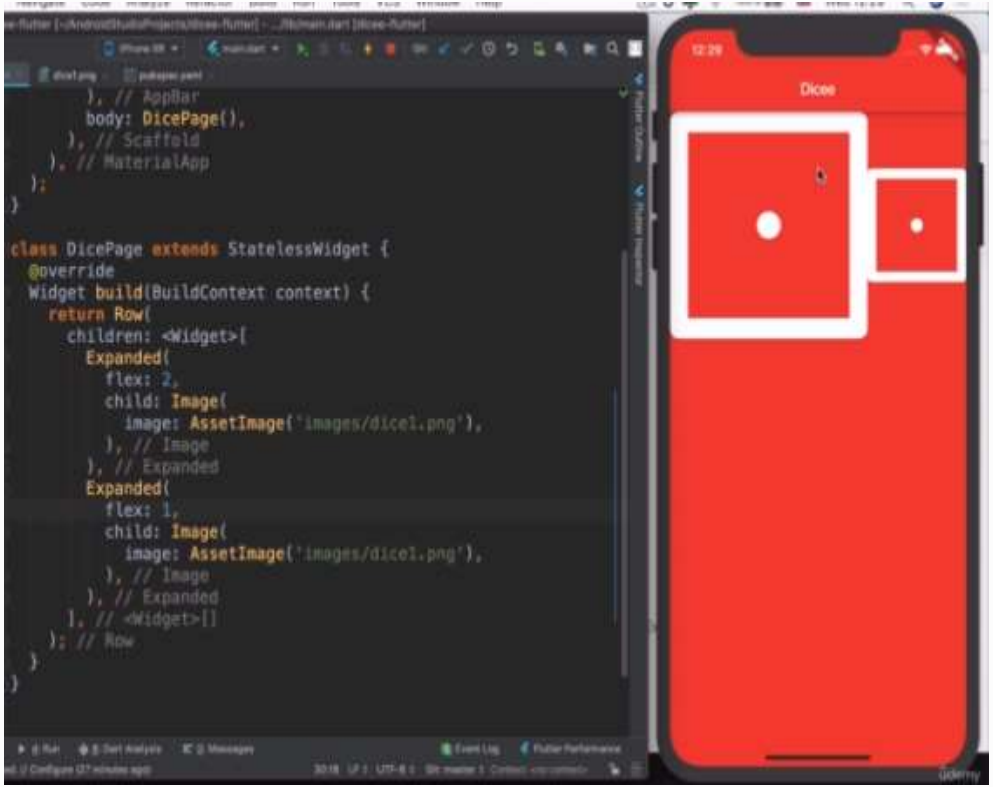


MOJADIPRO

Expanded Widget

Properti pada Expanded Widget :

Flex: Nilai flex menggambarkan perbandingan ukuran antara widget Expanded dengan widget Expanded lainnya dalam container yang sama.

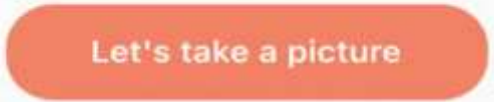


MOJADIPRO

Button Widget

1. **TextButton:** pembaruan dari tombol FlatButton yang digunakan untuk melakukan navigasi ke *page* yang lain.

```
TextButton(
    onPressed: () {},
    child: Text( "Let's take a picture"),
),
```



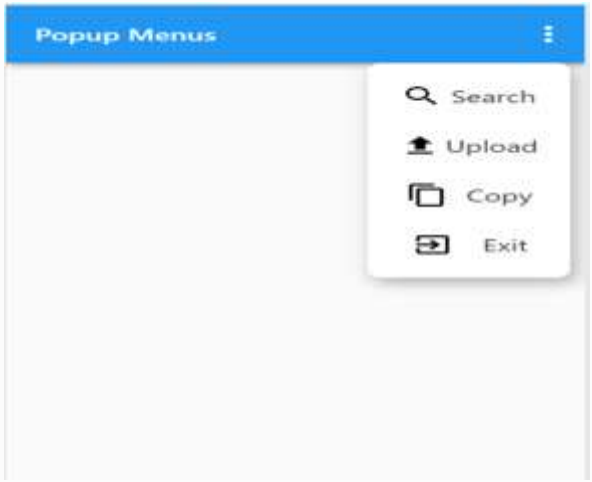
2. **ElevatedButton:** pembaruan dari tombol Raised Button, yaitu tombol dengan latar belakang terisi yang menonjol.

3. **IconButton:** Tombol yang disisipi *icon* sehingga menjadi *Widget Icon* yang bisa diklik.

MOJADIPRO

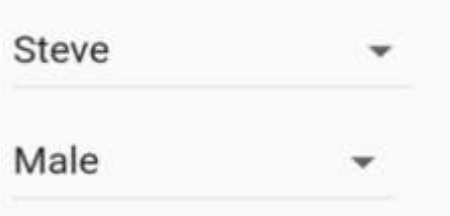
Button Widget

4. **Popup Menu Button:** Menampilkan menu saat ditekan.



5. **FloatingButton:** icon mengambang berbentuk lingkaran yang digunakan untuk melakukan *action* atau menambahkan sesuatu pada halaman aplikasi.

6. **Drop-down:** Tombol yang memungkinkan pengguna dapat memilih satu nilai dari daftar.

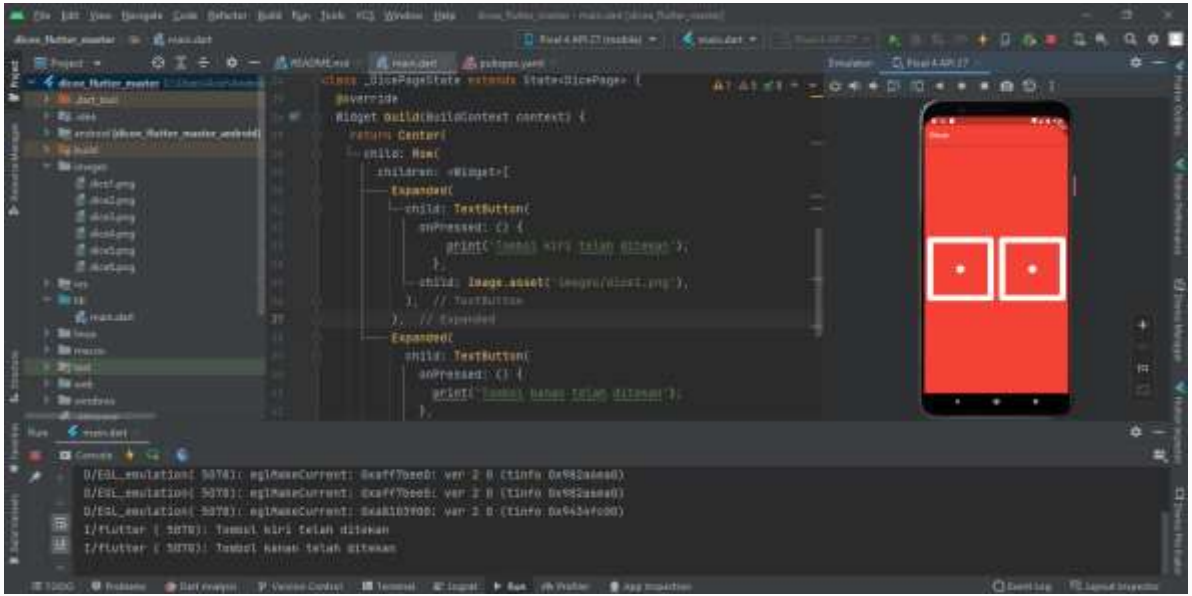


MOJADIPRO

Penggunaan callback pada Button Widget Flutter

Callback adalah mekanisme yang digunakan untuk mengirim fungsi sebagai parameter ke suatu objek atau widget. Callback digunakan untuk memanggil kembali (call back) fungsi yang diberikan ketika tindakan tertentu selesai dilakukan.

```
void onPressButton() {
    // Lakukan tindakan yang diinginkan saat tombol ditekan
    print('Tombol ditekan');
}
```



MOJADIPRO

Variabel Dart

Dart adalah bahasa pemrograman yang menggunakan tipe statis,
yang berarti tipe variabel harus dideklarasikan sebelum digunakan.

Variabel adalah wadah untuk menyimpan suatu nilai yang nantinya bisa dipanggil.

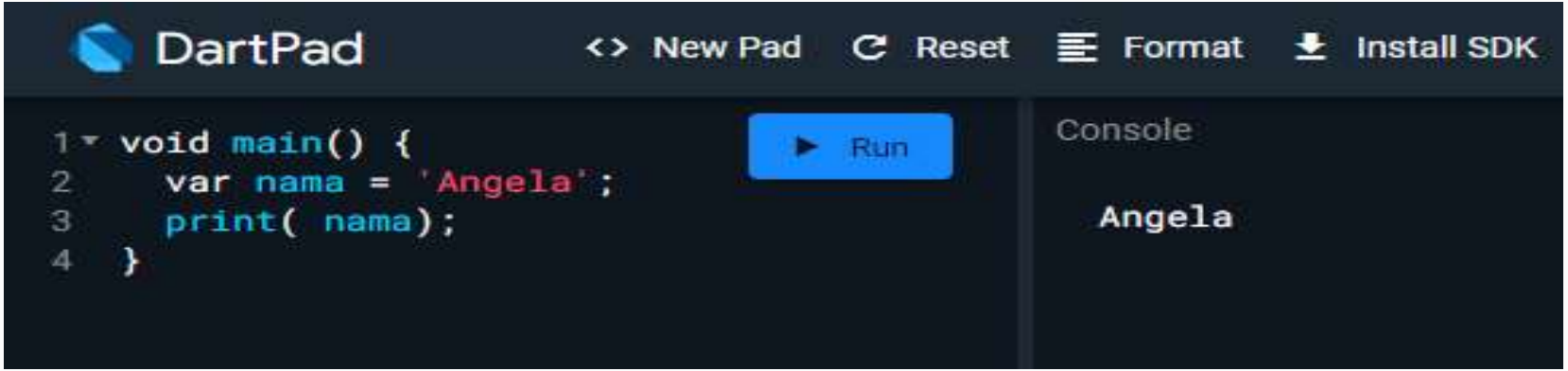
Contoh variabel dart:

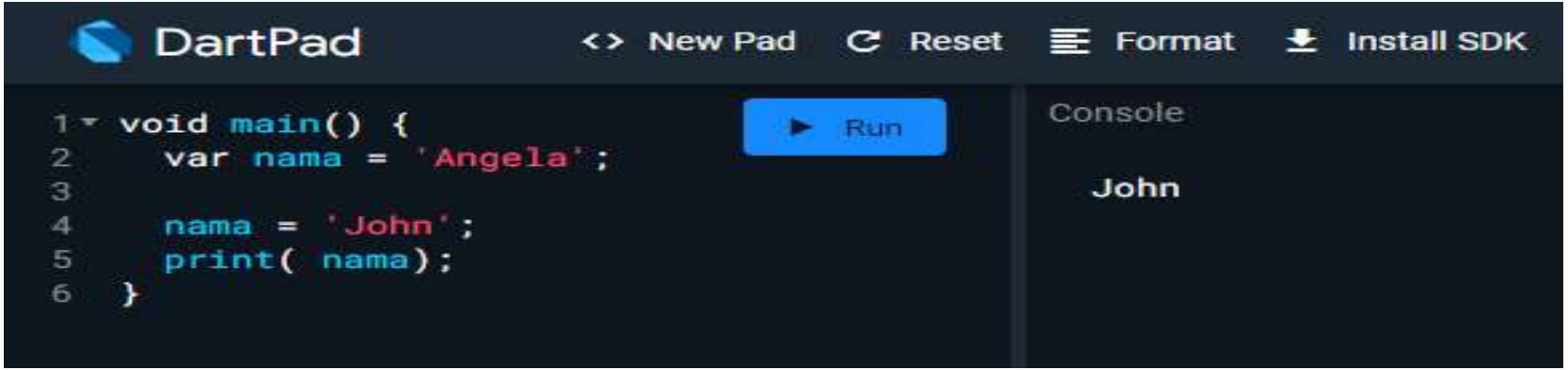
```
var nama = 'John';
String nama = 'John';
```

Kita bisa menggunakan kata kunci (var) apabila tidak ingin mendefinisikan secara eksplisit apa tipe datanya

MOJADIPRO

Contoh Penggunaan Kode Pada Variabel Dart





MOJADIPRO

Tipe Data Dart

1. String : menyimpan teks atau karakter. Didalamnya terdapat tanda petik.

```
var nama = 'John';
String nama = 'John';
```

2. Numbers (int/integer dan double).
   Integer : menyimpan bilangan bulat.
   Double : menyimpan bilangan desimal.

```
var age = 21;
int age = 21;
double weight = 77.5;
```

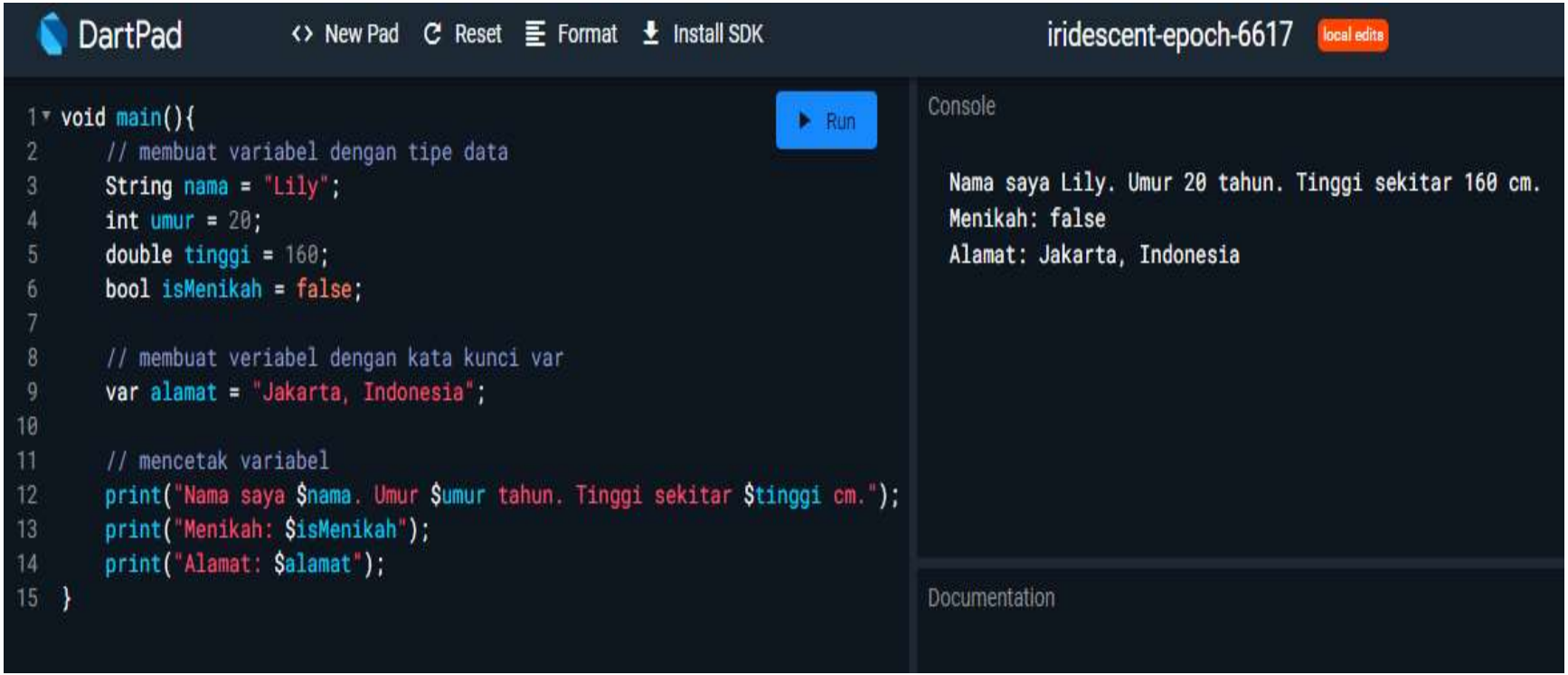
3. Booleans : menyimpan dua nilai yaitu true dan false.

```
var isTrue = true;
bool isTrue = false;
```

4. Dynamic : menyimpan nilai dari jenis apa pun, dan memungkinkan perubahan tipe data saat runtime.

```
dynamic data = 'Hello World';
data = 21; //SUCCESS

var data = 'Hello World';
data = 21; //ERROR
```

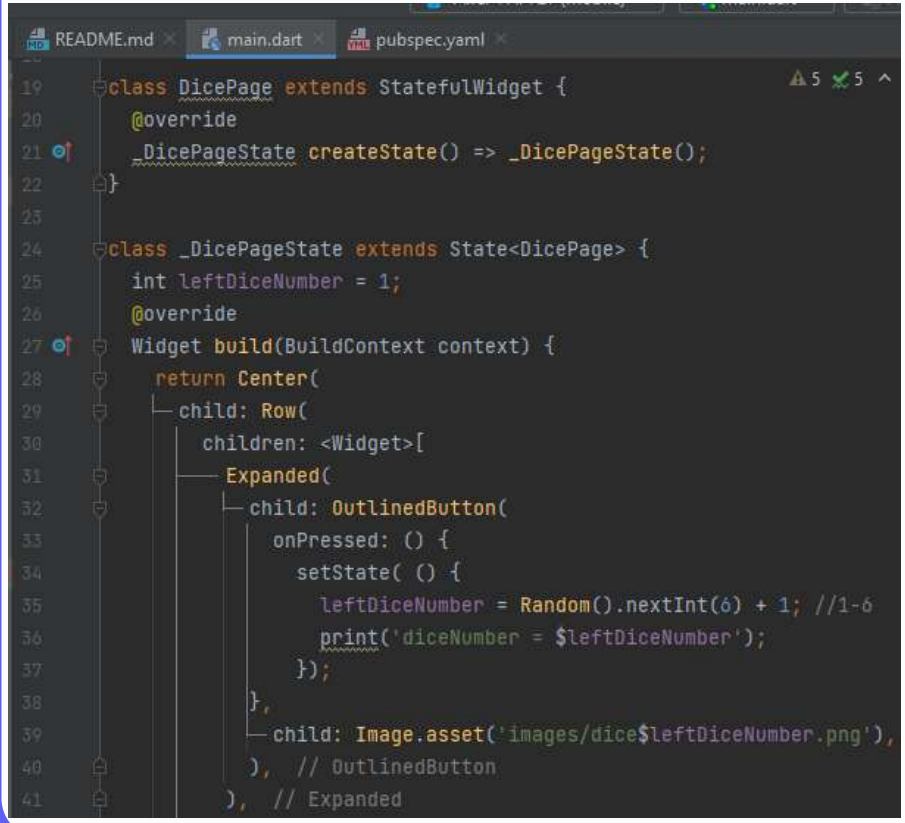
Contoh Penggunaan Kode Pada Tipe Data Dart



Stateless & Stateful Widget

Jika anda ingin membuat antarmuka pengguna dimana status widget tidak akan berubah, maka anda akan menggunakan stateless widget.

Tetapi jika anda ingin membuat sesuatu yang akan berubah, seperti pengguna mengetuk tombol, atau menarik beberapa data dari internet, maka anda akan menggunakan stateful widget.



setState()

setState() adalah metode yang digunakan dalam kelas StatefulWidget untuk memberi tahu Flutter bahwa ada perubahan pada State yang mempengaruhi tampilan widget.

Anda memberikan fungsi pembaruan yang berisi perubahan yang ingin Anda terapkan pada State object.

Stateful Widget akan memanggil metode build() untuk membangun ulang antarmuka pengguna dengan menggunakan nilai yang diperbarui.

MOJADIPRO

Dart Math Library

Dart Math Library adalah Library yang menyediakan berbagai fungsi matematika dan konstanta.
Library ini mencakup fungsi untuk operasi aritmatika dasar, trigonometri, logaritma, eksponen, pembulatan, bilangan acak, dan lain-lain.

dart:math library

Mathematical constants and functions, plus a random number generator.

To use this library in your code:

```
import 'dart:math';
```

Random

Random is a generator of bool, int or double values.

```
var intValue = Random().nextInt(10); // Value is >= 0 and < 10.
var doubleValue = Random().nextDouble(); // Value is >= 0.0 and < 1.0.
var boolValue = Random().nextBool(); // true or false, with equal chance.
```

MOJADIPRO

Fungsi Operasi Dart Math Library

Konstanta

“pi” (π)
“e” (basis logaritma alami).

Aritmatika

“sqrt” untuk menghitung akar kuadrat,
“pow” untuk perpangkatan,
“abs” untuk nilai absolut,
“min” dan “max” untuk menemukan nilai minimum dan maksimum

Logaritma

“log(x)” yang mengembalikan logaritma alami dari suatu bilangan,
“logBase(x, base)” yang menghitung logaritma bilangan x dengan dasar tertentu.

MOJADIPRO

Fungsi Operasi Dart Math Library

Pembulatan

seperti ceil(x) untuk membulatkan ke atas, floor(x) untuk membulatkan ke bawah, round(x) untuk membulatkan ke bilangan bulat terdekat.

Trigonometri

sin,
cos,
tan,
asin,
acos,
atan

Bilangan Acak

Random
Metodenya:
- nextInt,
- nextDouble,
- nextBool.

MOJADIPRO

Random Class

Kelas "Random" digunakan untuk menghasilkan bilangan acak.

1. Metode nextInt(int max): menghasilkan sebuah bilangan bulat acak antara 0 (inklusif) dan max (eksklusif).
   Misalnya, Random().nextInt(10) akan menghasilkan bilangan bulat acak antara 0 dan 9.

```
import 'dart:math';

void main() {
 Random random = Random();
 int randomNumber = random.nextInt(10);
 print(randomNumber);
}
```

MOJADIPRO

Random Class

MOJADIPRO

Random Class

2. Metode nextDouble(): menghasilkan sebuah bilangan desimal acak antara 0.0 (inklusif) dan 1.0 (eksklusif).
   Misalnya, Random().nextDouble() akan menghasilkan bilangan desimal acak antara 0.0 dan 0.9999999999999999.

```
import 'dart:math';

void main() {
  Random random = Random();
  double randomDouble = random.nextDouble();
  print(randomDouble);
}
```

3. Metode nextBool(): menghasilkan sebuah nilai boolean acak, yaitu true atau false. Misalnya, Random().nextBool() akan menghasilkan true atau false secara acak.

```
import 'dart:math';

void main() {
  Random random = Random();
  bool randomBool = random.nextBool();
  print(randomBool);
}
```

MOJADIPRO

Fungsi Operator

1. Operator Aritmatika :
- Penjumlahan (+)
- Pengurangan (-)
- Pembagian (/) yang menghasilkan nilai double.
- Pembagian (~/) yang menghasilkan nilai integer.
- Perkalian (*)
- Sisa bagi atau modulo (%)
- Increment (++)
- Derement (--)

2. Operator Relasi :
- Sama dengan (==)
- Tidak samadengan (!=)
- Lebih besar (>)
- Lebih kecil (<)
- Lebih besar sama dengan (>=)
- Lebih kecil sama dengan (<=)

3. Operator Penugasan
Pengisian (=)
Penjumlahan (+=)
Pengurangan (-=)
Pembagian (/=)
Perkalian (*=)
Sisa Bagi (%=)

MOJADIPRO

Fungsi Operator

4. Operator Logika
- And (&&)
- Or (||)
- Not (!)

5. Operator Bitwise
- And (&)
- Or (|)
- Xor (^)
- Not (~)
- Left Shift (<<)
- Right Shift (>>)

6. Operator Ternary (?)

MOJADIPRO

Fungsi Dart Input & Output

1. Tentukan tipe data argumen masukan (input) dan tipe data yang akan dikembalikan (output) dari fungsi.

```
tipeDataOutput namaFungsi(tipeDataInput
argumen1, tipeDataInput argumen2) {
  // Logika atau operasi yang ingin Anda lakukan
  // pada argumen masukan

  // Mengembalikan output
  return output;
}
```

MOJADIPRO

Fungsi Dart Input & Output

2. Tentukan nama fungsi yang sesuai dengan fungsionalitasnya. Misalnya, jika Anda ingin membuat fungsi penjumlahan, Anda dapat menggunakan nama "sum" atau "add".

3. Isi logika atau operasi pada argumen masukan di dalam fungsi. Misalnya, untuk fungsi penjumlahan. Anda dapat menambahkan dua argumen masukan dan menyimpan hasilnya ke dalam variabel output.

4. Mengembalikan output menggunakan pernyataan “return output”.

MOJADIPRO

Fungsi Dart Input & Output

DartPad <> New Pad | Reset | Format | Install SDK

```dart
int sum(int number1, int number2) {
  int result = number1 + number2;
  return result;
}

void main() {
  int a = 5;
  int b = 3;
  int sumResult = sum(a, b);
  print("Hasil penjumlahan: $sumResult");
}
```

Run

Console

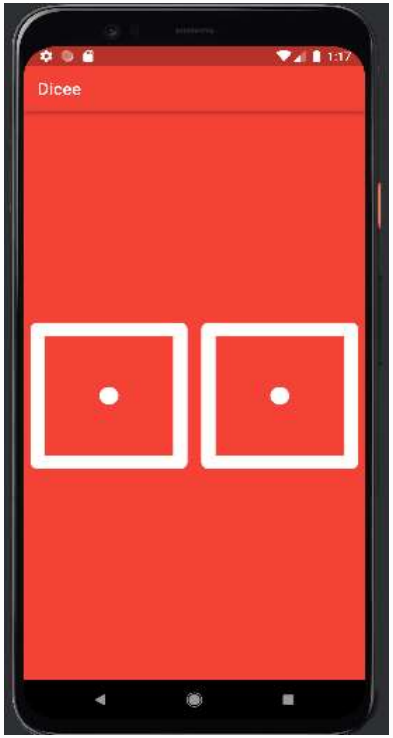
```
Hasil penjumlahan: 8
```

MOJADIPRO

Aplikasi Yang Telah Dibuat

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

void main() {
  return runApp(
    MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        backgroundColor: Colors.red,
        appBar: AppBar(
          title: Text('Dicee'),
          backgroundColor: Colors.red,
        ),
        body: DicePage(),
      ),
    ),
  );
}

class DicePage extends StatefulWidget {
  @override
  _DicePageState createState() => _DicePageState();
}

class _DicePageState extends State<DicePage> {
  int leftDiceNumber = 1;
  int rightDiceNumber = 1;

  void changeDiceFace() {
    setState(() {
      leftDiceNumber = Random().nextInt(6) + 1;
      rightDiceNumber = Random().nextInt(6) + 1;
    });
  }
```

```dart
@override
Widget build(BuildContext context) {
  return Center(
    child: Row(
      children: <Widget>[
        Expanded(
          child: TextButton(
            child: Image.asset(
              'images/dice$leftDiceNumber.png',
            ),
            onPressed: () {
              changeDiceFace();
            },
          ),
        ),
        //Get students to create the second die as a challenge
        Expanded(
          child: TextButton(
            child: Image.asset(
              'images/dice$rightDiceNumber.png',
            ),
            onPressed: () {
              changeDiceFace();
            },
          ),
        ),
      ],
    ),
  );
}
}
```



MOJADIPRO

Dart Function

Dalam bahasa pemrograman Dart, terdapat dua cara untuk mendefinisikan fungsi :

- Fungsi Reguler (Dart Function)
- Fungsi Tanda Panah (Dart Arrow Function).

Dart Function

Fungsi reguler dalam Dart didefinisikan dengan menggunakan kata kunci "void" diikuti dengan nama fungsi, argumen, dan blok kode di dalamnya.

Berikut adalah contoh penggunaan dart function :

```
void greet(String name) {
  print('Hello, $name!');
}

void main() {
  greet('John');
}
```

MOJADIPRO

Dart Arrow Function

Dart Arrow Function

Fungsi tanda panah atau dart arrow function merupakan sintaksis pendek untuk mendefinisikan fungsi dengan hanya satu ekspresi. Mereka sangat berguna saat ingin mendefinisikan fungsi sederhana.

Berikut adalah contoh penggunaan dart arrow function:

```
void main() {
 Function sayHello = (String name) => print('Hello, $name!');
 sayHello('John');
}
```

Dart arrow function juga dapat digunakan untuk mengembalikan nilai.

Berikut adalah contoh penggunaannya:

```
void main() {
 String capitalize(String text) => text.toUpperCase();

 print(capitalize('hello'));
}
```

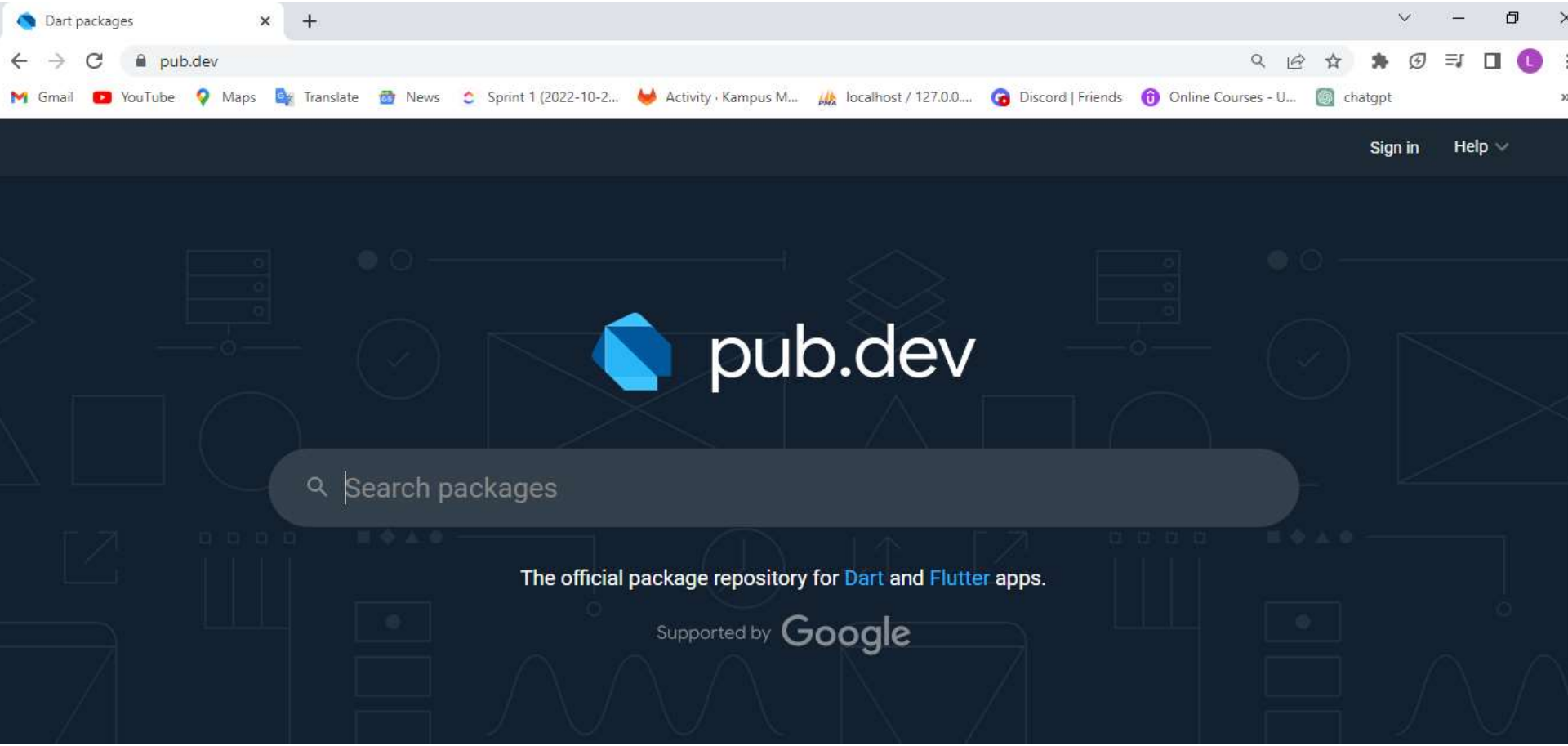
MOJADIPRO

Dart Packages

Dart Packages, atau sering disebut juga dengan pub packages, adalah kumpulan kode dan sumber daya yang telah dikemas secara terstruktur dalam format tertentu.
library, framework, atau alat bantu lainnya yang dapat membantu dalam mengembangkan aplikasi dengan lebih cepat dan efisien.

MOJADIPRO

Cara Menggunakan Dart Package



Buka laman https://pub.dev/ di google, lalu ketik di pencarian “audio player”

Sign in Help

audio player

Platforms

- Android
- iOS
- Linux
- macOS
- Web
- Windows

SDKs

License

Advanced

RESULTS 237 packages

SORT BY SEARCH RELEVANCE

audioplayers

2169 LIKES | 140 PUB POINTS | 99% POPULARITY

A Flutter plugin to play multiple audio files simultaneously

v 4.0.1 (42 days ago) blue-fire.xyz MIT Dart 3 compatible

SDK FLUTTER PLATFORM ANDROID IOS LINUX MACOS WEB WINDOWS

API results: ▸ audioplayers/AudioPlayer-class.html

assets_audio_player

968 LIKES | 120 PUB POINTS | 98% POPULARITY

Play music/audio stored in assets files directly from Flutter & Network, Radio, LiveStream, Local files. Compatible with Android, iOS, web and macOS.

v 3.0.6 (6 months ago) Apache-2.0 Dart 3 compatible

SDK FLUTTER PLATFORM ANDROID IOS MACOS WEB

API results: ▸ assets_audio_player/assets_audio_player-library.html

audioplayers 4.0.1

Published 42 days ago • blue-fire.xyz Dart 3 compatible

SDK FLUTTER PLATFORM ANDROID IOS LINUX MACOS WEB WINDOWS

2.1K

Readme Changelog Example Installing Versions Scores

MOJADIPRO

Audio PLAYERS

A Flutter plugin to play multiple simultaneously audio files, works for Android, iOS, Linux, macOS, Windows, and web.

pub v4.0.1 build passing chat 528 online maintained with melos

2169 LIKES | 140 PUB POINTS | 99% POPULARITY

Publisher

blue-fire.xyz

Metadata

A Flutter plugin to play multiple audio files simultaneously

Homepage

Repository (GitHub)

Documentation

API reference

License

MIT (LICENSE)

Dependencies

Use this package as a library

Depend on it

Run this command:

With Flutter:

```
$ flutter pub add audioplayers
```

This will add a line like this to your package's pubspec.yaml (and run an implicit `flutter pub get`):

```
dependencies:
  audioplayers: ^4.0.1
```

Alternatively, your editor might support `flutter pub get`. Check the docs for your editor to learn more.

Import it

Now in your Dart code, you can use:

```
import 'package:audioplayers/audioplayers.dart';
```

2169 LIKES 140 PUB POINTS 99% POPULARITY

Publisher

blue-fire.xyz

Metadata

A Flutter plugin to play multiple audio files simultaneously

Homepage

Repository (GitHub)

Documentation

API reference

License

MIT (LICENSE)

Dependencies

audioplayers_android, audioplayers_darwin, audioplayers_linux, audioplayers_platform_interface, audioplayers_web, audioplayers_windows, file, flutter, http, meta,

example/lib/main.dart

```dart
import 'dart:async';

import 'package:audioplayers/audioplayers.dart';
import 'package:audioplayers_example/components/indexed_stack.dart';
import 'package:audioplayers_example/components/tabs.dart';
import 'package:audioplayers_example/components/tgl.dart';
import 'package:audioplayers_example/tabs/audio_context.dart';
import 'package:audioplayers_example/tabs/controls.dart';
import 'package:audioplayers_example/tabs/logger.dart';
import 'package:audioplayers_example/tabs/sources.dart';
import 'package:audioplayers_example/tabs/streams.dart';
import 'package:audioplayers_example/utils.dart';
import 'package:flutter/material.dart';

const defaultPlayerCount = 4;

typedef OnError = void Function(Exception exception);

void main() {
  runApp(const MaterialApp(home: ExampleApp()));
}

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

  @override
  _ExampleAppState createState() => _ExampleAppState();
}

class _ExampleAppState extends State<ExampleApp> {
  List<AudioPlayer> audioPlayers = List.generate(
    defaultPlayerCount,
    (_) => AudioPlayer()..setReleaseMode(ReleaseMode.stop),
  );
```

MOJADIPRO

Memasukkan audioplayers package

1. Pastikan Anda telah menambahkan dependensi audio_players ke dalam file pubspec.yaml
(dan menjalankan implisit flutter pub get)

```
dependencies:
    flutter:
      sdk: flutter
    audioplayers: ^4.0.1
```

2. Import package flutter pada main.dart

```
Import 'package:audioplayers/audioplayers.dart';
```

3. Menambahkan file audio ke dalam folder assets

```
flutter:
 assets:
  - assets/note1.wav
  - assets/note2.wav
  - assets/note3.wav
  - assets/note4.wav
  - assets/note5.wav
  - assets/note6.wav
  - assets/note7.wav
```

README.md | main.dart | pubspec.yaml

Flutter commands | Pub get | Pub upgrade | Pub outdated | Flutter doctor

```yaml
dependencies:
  flutter:
    sdk: flutter

  cupertino_icons: ^1.0.2
  audioplayers: ^4.0.1

dev_dependencies:
  flutter_test:
    sdk: flutter

  flutter_lints: ^2.0.0

flutter:
  uses-material-design: true

  assets:
    - assets/note1.wav
    - assets/note2.wav
    - assets/note3.wav
    - assets/note4.wav
    - assets/note5.wav
    - assets/note6.wav
```

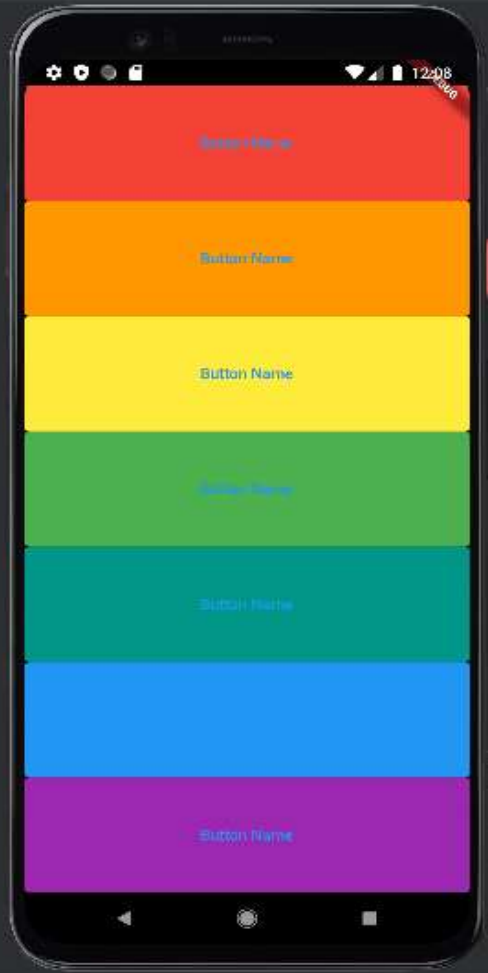
README.md | main.dart | pubspec.yaml

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

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

class XylophoneApp extends StatelessWidget {
  get audioPlayer => null;

  void playSound(int soundNumber) {
    final player = AudioCache();
    audioPlayer.play('note$soundNumber.wav');
  }

  Expanded buildKey({required Color color, required int soundNumber}) {
    return Expanded(
      child: TextButton(
        onPressed: () {
          playSound(soundNumber);
        },
        style: ButtonStyle(backgroundColor: MaterialStateProperty.all(color)),
        child: Text('Button Name'),
      ),  // TextButton
    );  // Expanded
  }
```

README.md | main.dart | pubspec.yaml

```dart
@override
Widget build(BuildContext context) {
  return MaterialApp(
    home: Scaffold(
      backgroundColor: Colors.black,
      body: SafeArea(
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.stretch,
          children: <Widget>[
            buildKey(color: Colors.red, soundNumber: 1),
            buildKey(color: Colors.orange, soundNumber: 2),
            buildKey(color: Colors.yellow, soundNumber: 3),
            buildKey(color: Colors.green, soundNumber: 4),
            buildKey(color: Colors.teal, soundNumber: 5),
            buildKey(color: Colors.blue, soundNumber: 6),
            buildKey(color: Colors.purple, soundNumber: 7),
          ],  // <Widget>[]
        ),  // Column
      ),  // SafeArea
    ),  // Scaffold
  );  // MaterialApp
}
}
```

Emulator: Pixel 4 API 27



Version Control | Terminal | Logcat | Run | Profiler | App Inspection | Event Log | Layout Inspec

MOJADIPRO

TERIMA KASIH