DevFest

Bandung 2024

Unlocking the Power of Dart Macros

Revolutionizing Code Generation

Erico Darmawan Handoyo

Google Developer Groups

Erico Darmawan Handoyo

- Lecturer at Maranatha Christian University
- Flutter Trainer
- Content creator:
  - Flutter Tutorial YouTube Channel
  - Online Courses at Udemy

Flutter Tutorial YouTube Channel
provides **300++** fundamental & tutorial videos
https://youtube.com/@ericodarmawan

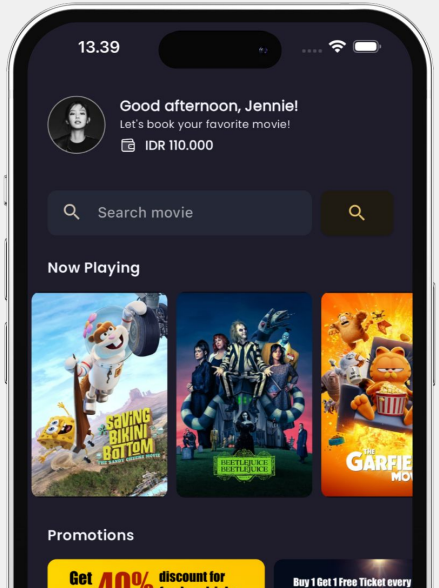
Google Developer Groups

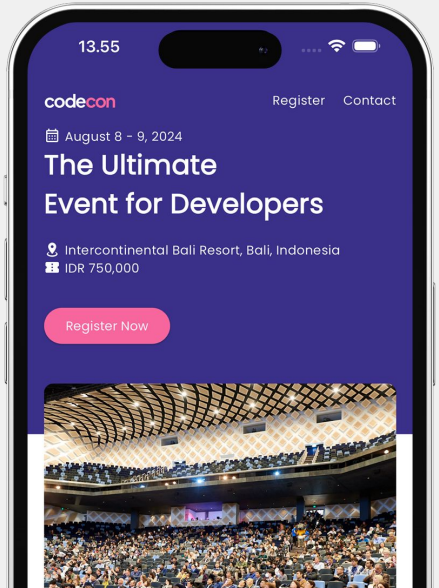
s.id/komunitas-flutter
s.id/flixid
s.id/ig-ericodh
ericodarmawan.com
s.id/fb-ericodh
s.id/in-ericodh

flixid

codecon

SCAN ME





for the Presentation Slide

SPECIAL OFFER 50% OFF

~~IDR 399.000~~

IDR 199.000

7 - 11 Desember 2024

s.id/flixid

s.id/codecon

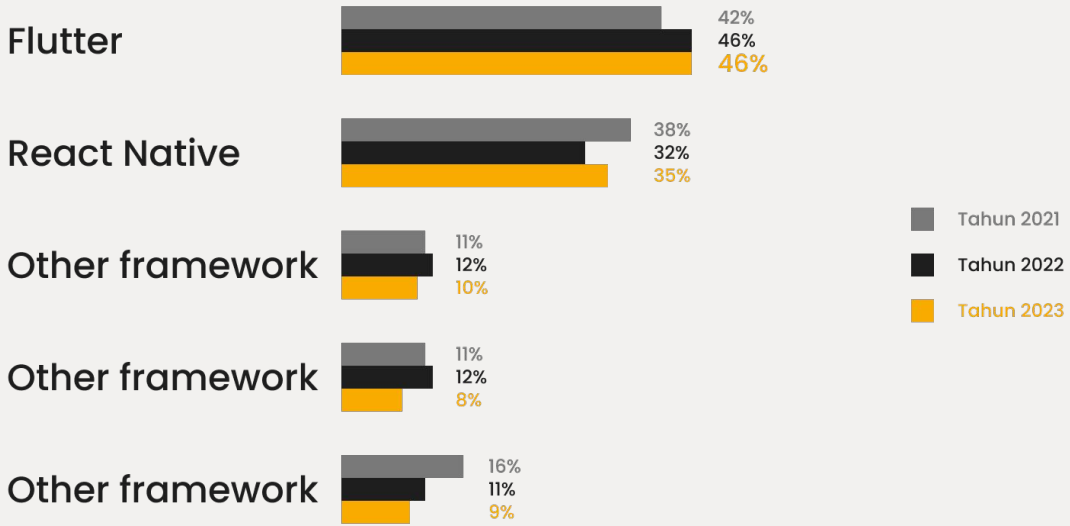
Discount Code: **DEVFESTBANDUNG24**

Flutter is The Most Popular Cross- Platform Framework

Google Developer Groups

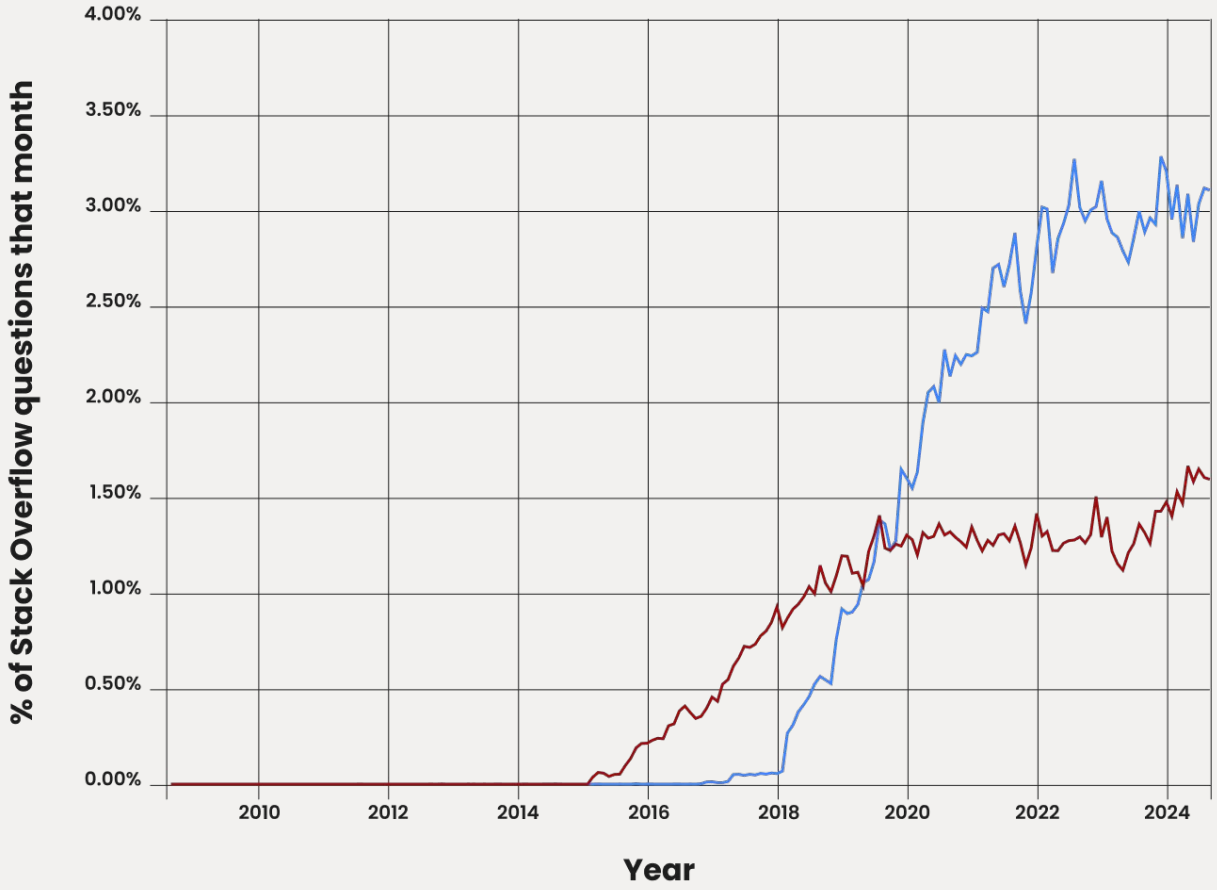
Erico Darmawan Handoyo
Private Training & Online School

Which cross-platform mobile frameworks do you use?



Google Developer Groups

Source: https://www.jetbrains.com/lp/devecosystem-2023/development
https://www.jetbrains.com/lp/devecosystem-2022/miscellaneous



Google Developer Groups

Source: https://trends.stackoverflow.co/?tags=flutter,react-native

Erico Darmawan Handoyo
Private Training & Online School

number 5

Top 10 public projects by contributors on GitHub	
Project	**Contributor count**
home-assistant/core	>21K
microsoft/vscode	>20K
ProvableHQ/leo	>20K
firstcontributions/first-contributions	>13k
flutter/flutter	>10K
NixOS/nixpkgs	>9K
vercel/next.js	>9K
langchain-ai/langchain	>8K
godotengine/godot	>7K
ollama/ollama	>7K

Source: https://github.blog/news-insights/octoverse/octoverse-2024/#the-most-popular-programming-languages

appfigures

The Most Popular Development SDKs

Type: Development Category: Apps

App Store		Google Play	
Flutter	**13%**	Flutter	**22%**
React Native	13%	React Native	17%

Source: https://appfigures.com/top-sdks/development/apps

Erico Darmawan Handoyo
Private Training & Online School

Is Flutter Stopped by Google

Flutter Dihentikan oleh Google?
https://youtube.com/watch?v=mIlgG7xUXxY

Google Developer Groups

#FlutterInProduction

Tune in at 11am PT on December 17, 2024

Register:
https://flutter.dev/events/flutter-in-production

=

Flutter+

Flock is Flutter+ • Flock is a fork of Flutter.
Flock stays up to date with Flutter, and
also adds new community features.

Google Developer Groups

Flock = Flutter+
https://youtu.be/jAHH33e3Nd8

Why do Mobile Developers Love Flutter?

- Flutter is platform-agnostic
- Flutter apps offer an excellent user interface and experience
  - Flutter's widget system simplifies creating great-looking interfaces and works well across platforms.
  - It doesn't need platform-specific UI components.
- Numerous useful open-source packages.

Google Developer Groups

Erico Darmawan Handoyo
Private Training & Online School

Why do Mobile Developers Love Flutter?

- Flutter is easy to learn
  - Learning Flutter & Dart is easy for Java, Python, or C# devs due to its familiar syntax.
  - Excellent documentation.

```java
public class Person {
    public String name;
    public int age;

    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
}

Person person = new Person("Alice", 30);
```

```python
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

person = Person("Alice", 30)
```

```dart
class Person {
  final String name;
  final int age;

  Person({required this.name, required this.age});
}

Person person = Person(name: 'Alice', age: 30);
```

Flutter
@DevFest

Google
Developer
Groups

Evolving Dart

Erico Darmawan Handoyo
Private Training & Online School

Evolving Dart

- Null Safety: prevents null errors by ensuring variables can't be null unless specified.
- Enhanced enum: allow enum to have additional features such as fields, methods, and constructors.
- Dart 3 new class modifiers: mixin, final, sealed, base, & interface class.

```dart
enum Role {
  admin('Administrator', 'Has full access'),
  user('User', 'Has limited access'),
  guest('Guest', 'Has minimal access');

  final String name;
  final String description;

  const Role(this.name, this.description);

  void describe() {
    print('$name: $description');
  }
}
```

Google Developer Groups

Erico Darmawan Handoyo
Private Training & Online School

Flutter
@DevFest

Google
Developer
Groups

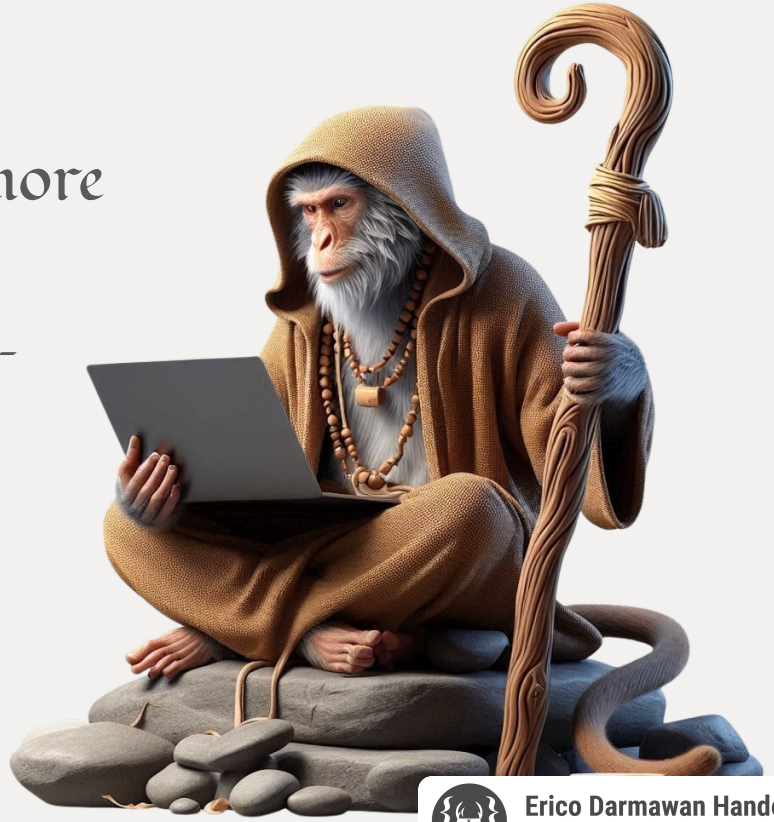
Macros

Erico Darmawan Handoyo
Private Training & Online School

Remember this my child..

“The more you craft your code, the more errors may whisper in.”

– Mysterious Programming Guru –



Google Developer Groups

Erico Darmawan Handoyo
Private Training & Online School

Build Runner Package

Provides a concrete way of generating files using Dart code.

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

part 'person.g.dart';

@JsonSerializable()
class Person {
  final String name;
  final int age;

  Person({required this.name, required this.age});

  Map<String, dynamic> toJson() => _$PersonToJson(this);
  factory Person.fromJson(Map<String, dynamic> json) =>
    _$PersonFromJson(json);
}
```

```
> dart run build_runner build
```

```dart
// // GENERATED CODE - DO NOT MODIFY BY HAND

part of 'person.dart';


Person _$PersonFromJson(Map<String, dynamic> json) => Person(
      name: json['name'] as String,
      age: (json['age'] as num).toInt(),
    );

Map<String, dynamic> _$PersonToJson(Person instance) => <String, dynamic>{
      'name': instance.name,
      'age': instance.age,
    };
```

Google Developer Groups

Erico Darmawan Handoyo
Private Training & Online School

What is Dart Macro?

Dart macro is a user-definable piece of code that takes in other code as parameters and operates on it in real-time to create, modify, or add declarations.

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

@JsonCodable()
class Person {
  final String name;
  final int age;

  Person({required this.name, required this.age});
}
```

```dart
augment library 'file:///Users/ericodh/Dart%20Project/macros_demo/bin/macros_demo.dart';

import 'dart:core' as prefix0;

augment class Person {
  external Person.fromJson(prefix0.Map<prefix0.String, prefix0.Object?> json);
  external prefix0.Map<prefix0.String, prefix0.Object?> toJson();
  augment Person.fromJson(prefix0.Map<prefix0.String, prefix0.Object?> json, )
      : this.name = json[r'name'] as prefix0.String,
        this.age = json[r'age'] as prefix0.int;
  augment prefix0.Map<prefix0.String, prefix0.Object?> toJson() {
    final json = <prefix0.String, prefix0.Object?>{};
    json[r'name'] = this.name;
    json[r'age'] = this.age;
    return json;
  }
}
```

Google Developer Groups

Erico Darmawan Handoyo
Private Training & Online School

Another Example

Manually

```dart
class Person {
  const Person({this.name, this.age});

  final String name;
  final int age;

  @override
  bool operator ==(Object other) => identical(this, other) ||
    other is Person && runtimeType == other.runtimeType &&
    name == other.name && age == other.age;

  @override
  int get hashCode => Object.hash(runtimeType, name, age);

  Person copyWith(String? name, int? age) =>
    Person(name: name ?? this.name, age: age ?? this.age);

  @override
  String toString() => 'Person(name: $name, age: $age)';
}
```

Using **freezed** package

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

part 'main.freezed.dart';

@freezed
class Person with _$Person {
  const factory Person({
    required String name,
    required int age,
  }) = _Person;
}
```

Using **data_class** macro

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

@Data()
class Person {
  const Person({this.name, this.age});

  final String name;
  final int age;
}
```

Google Developer Groups

Erico Darmawan Handoyo
Private Training & Online School

How to Use Macro?

1. Switch to the Flutter master channel.
2. Run dart --version and make sure you have Dart version 3.5.0-152 or later.
3. Edit the SDK constraint in your pubspec to require the Dart version: sdk: ^3.5.0-152.
4. Add the package json to dependencies: dart pub add json.

How to Use Macro?

5. Enable the experiment in your package's analysis_options.yaml file. file at the root of your project:

6. Run your project with the experimental flag:

   dart run --enable-experiment=macros your_file.dart

Flutter
@DevFest

Google
Developer
Groups

How to Create Macro (Demo)

Erico Darmawan Handoyo
Private Training & Online School

DevFest

Bandung 2024

Thank You

Google Developer Groups