

```
ldren: [  
  icon(icon, color: color  
  container(  
    margin: const EdgeInsets  
    child:  
      label  
      style
```



Google Developer Student Clubs

Institut Teknologi Telkom Surabaya

Intro to Flutter Development

Developing Mobile Apps on Multiplatform Technology



Ahmad Mu'min Faisal
Core Team, GDSC ITTS



Ahmad Mu'min Faisal

Core Team at



“A dedicated Flutter development enthusiast with a profound love for Linux and Free/Libre and Open-Source Software (FLOSS)”

- Division Head of PKM Center – Punggawa Inspiratif
- Lecturer Assistant of DSA, Comp. System Programming, and OOPs
- Head of Public Relation – Kelompok Linux Arek Suroboyo
- 2nd Place – IT Convert 2023
- 1st Place – PLAYBOX Season 4
- 3x Funded – PKM 2022-2023

ahmadmfaisal

fzl-22

contact@ahmadfaisal.online

fzl-22

Ask me :)



bit.ly/ask-mobile-dev



Get to Know About Mobile Devices

```
Lookup.KeyValue
f.constant(['en
=tf.constant([
.lookup.StaticV
_buckets=5)
```

A mobile device is a **portable computing device** such as a smartphone or tablet computer

On each mobile device, there is an **operating system** embedded as an **interface** between **device hardware** and **application** on top of it.

OS	Vendor	Description
Android	Google	<ul style="list-style-type: none">• using Linux kernel• open source (Android Open Source Project)• embedded on most mobile devices• Can be forked by other vendor, such as MIUI by Xiaomi.
iOS	Apple	<ul style="list-style-type: none">• using XNU kernel• closed source• only embedded on Apple's iPhone and iPad





How to Build Application on Mobile Devices?

```
Lookup.KeyValue
f.constant(['en'])
=tf.constant([0])
.lookup.StaticV
_buckets=5)
```

Native Development

By default, mobile apps are developed **natively**.

Native development means creating application for each **specific platform**, such as Android and iOS.



Native Development

The consequences is we must use **different** programming languages and tools for **each platform**.

Android

Java

Kotlin

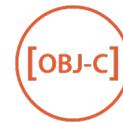
iOS

Objective-C

SwiftUI



or



or

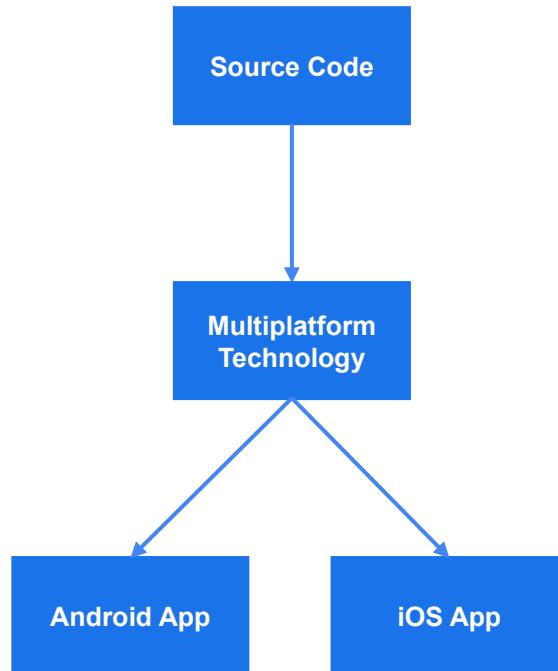


Multiplatform Development

Alternatively, mobile apps can be developed using **multiplatform technology**.

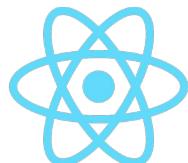
Application developed by this approach is expected to be **compatible with multiple operating system**.

The magic is, **it is.**



Multiplatform Development

Application developed this way requires a **framework** to do so. Currently, the most popular and widely-used multiplatform frameworks are **Flutter** and **React Native**.



Flutter

Using Dart

Created by Google in 2017

Can build Android, iOS, Linux, MacOS, Windows, and Web Application. Even on embedded device also

Compiled into target platform's native code

Near-native performance

github.com/flutter/flutter

React Native

Using Javascript

Created by Facebook in 2015

Can build iOS and Android application.

Rely on bridging with native modules

Good, but not as good as Flutter

github.com/facebook/react-native

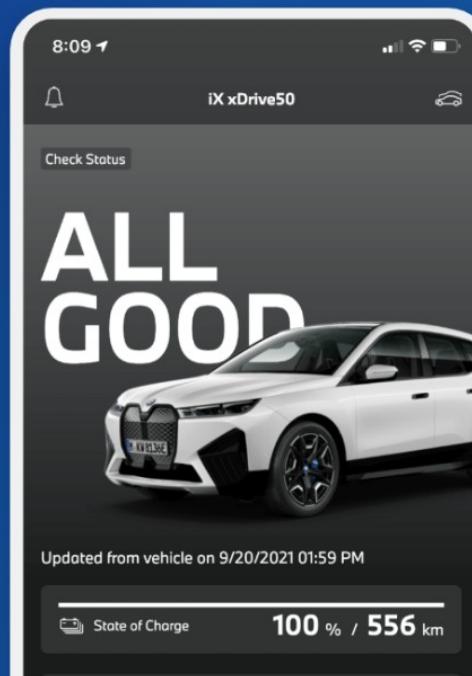
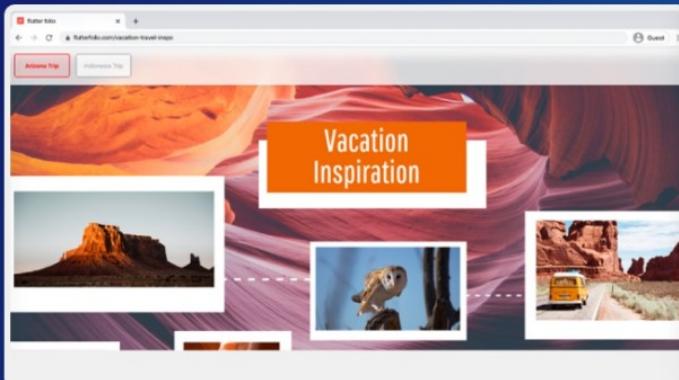


What is Flutter?

“Flutter is an open source framework by Google for building **beautiful, natively compiled, multi-platform applications** from a **single codebase.**”

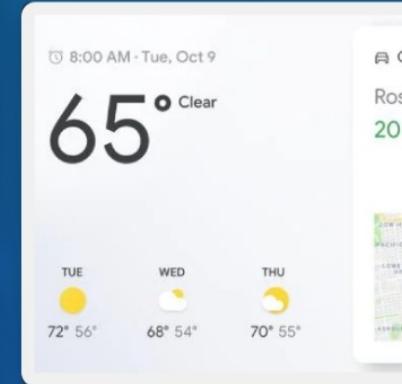
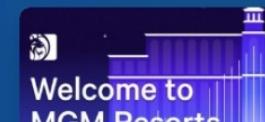
You can visit Flutter’s webpage at
flutter.dev

Build apps for any screen

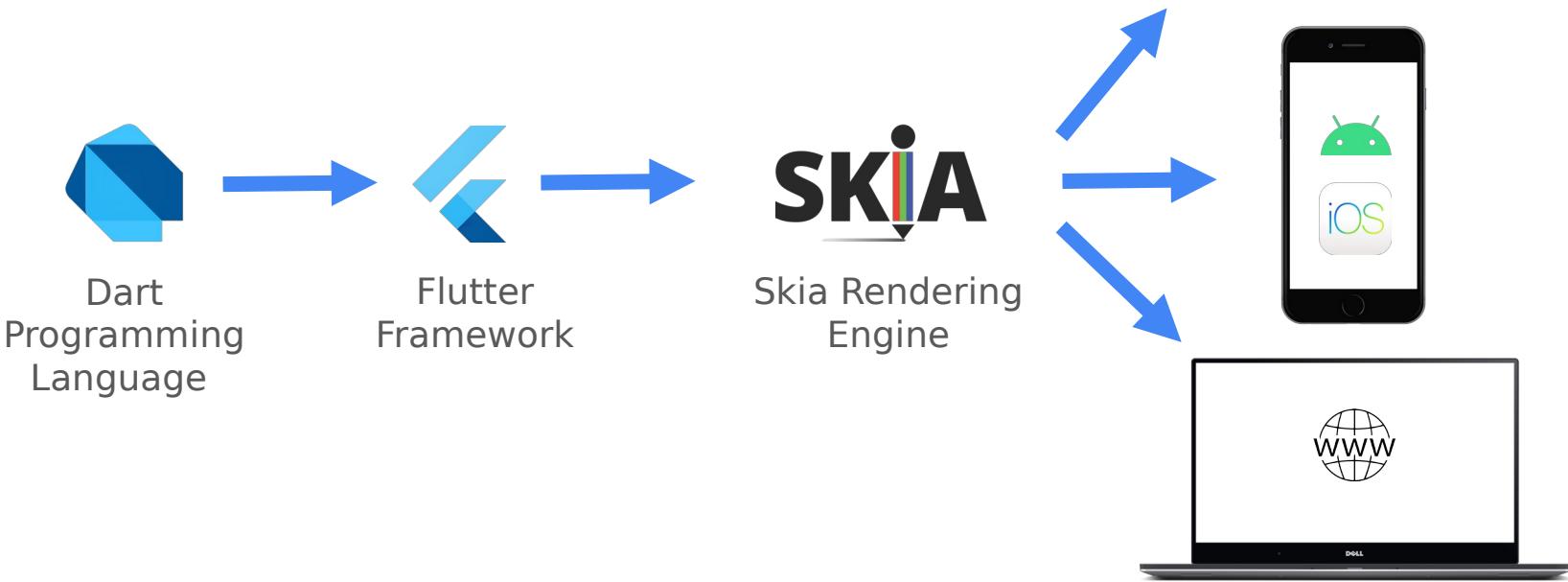


Updated from vehicle on 9/20/2021 01:59 PM

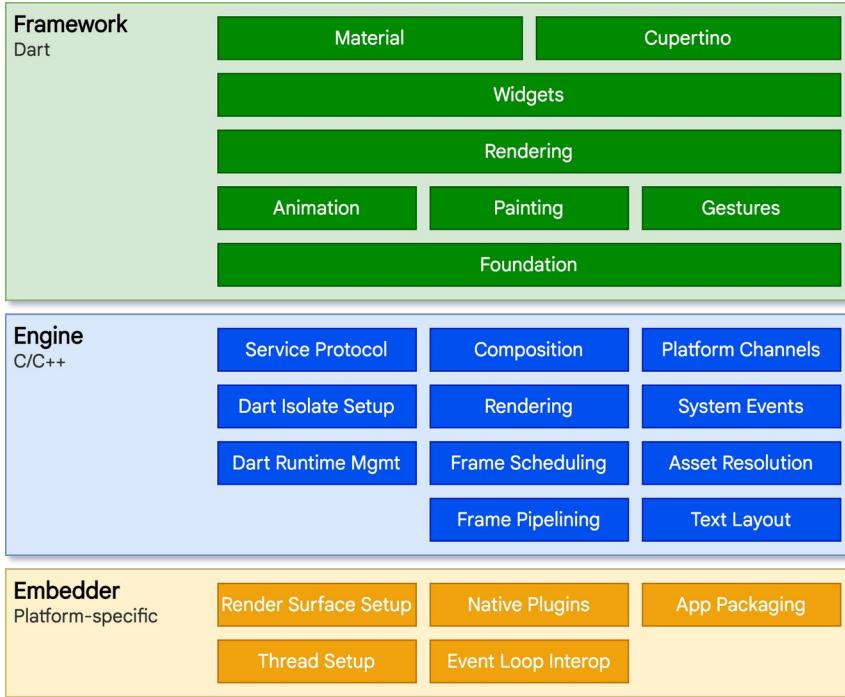
100 % / 556 km



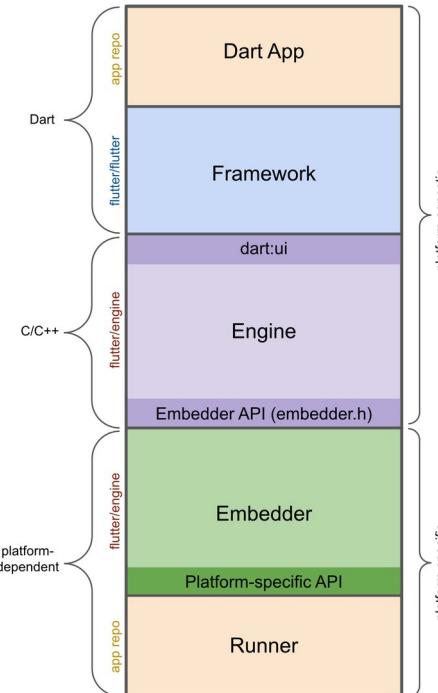
How Flutter Works



Flutter Architectural Layer



Flutter App Anatomy





Let's Get Our Hand Dirty!

```
Lookup.KeyValue
f.constant(['em
=tf.constant([
.lookup.StaticV
_buckets=5)
```

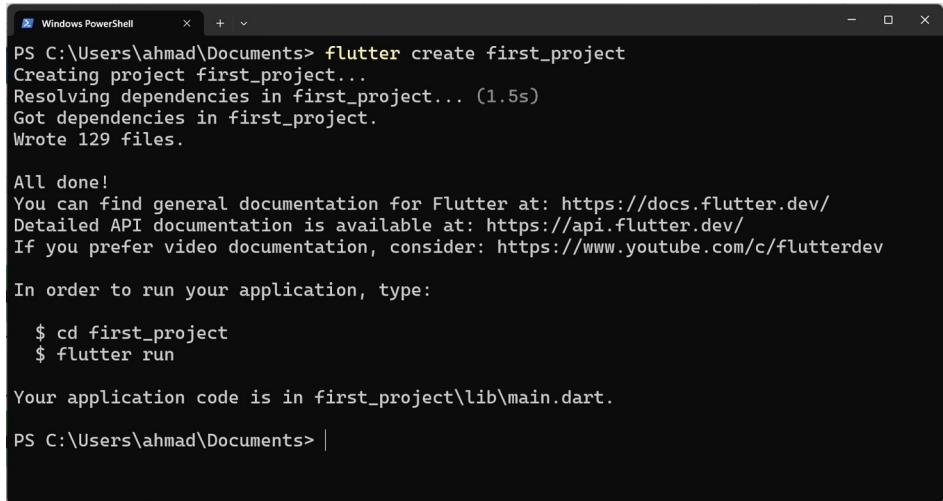
Create a Flutter Project

Flutter project can be easily created by running this following command on terminal:

```
flutter create {project_name}
```

For example,

```
flutter create first_project
```



```
PS C:\Users\ahmad\Documents> flutter create first_project
Creating project first_project...
Resolving dependencies in first_project... (1.5s)
Got dependencies in first_project.
Wrote 129 files.

All done!
You can find general documentation for Flutter at: https://docs.flutter.dev/
Detailed API documentation is available at: https://api.flutter.dev/
If you prefer video documentation, consider: https://www.youtube.com/c/flutterdev

In order to run your application, type:

$ cd first_project
$ flutter run

Your application code is in first_project\lib\main.dart.

PS C:\Users\ahmad\Documents> |
```

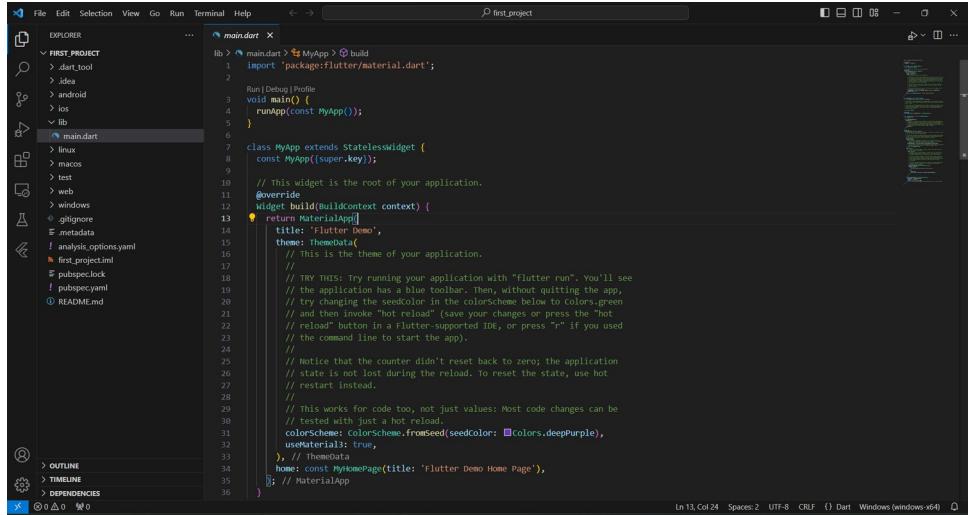
Now, Flutter initial project is successfully created.

Open Flutter Project

Open the project in Visual Studio Code by running this command:

```
code first_project
```

Make sure that **Dart** and **Flutter** official extensions on VS Code is already installed.



The screenshot shows the Visual Studio Code interface with a Flutter project named "first_project". The Explorer sidebar on the left lists the project structure, including "FIRST_PROJECT", "lib" (which contains "main.dart"), and various platform-specific folders like "linux", "macos", "ios", and "windows". The "main.dart" file is selected in the Explorer and is displayed in the central code editor. The code in "main.dart" is as follows:

```
lib/main.dart
1 import 'package:flutter/material.dart';
2
3 void main() {
4   runApp(const MyApp());
5 }
6
7 class MyApp extends StatelessWidget {
8   const MyApp({super.key});
9
10  // This widget is the root of your application.
11  @override
12  Widget build(BuildContext context) {
13    return MaterialApp(
14      title: 'Flutter Demo',
15      theme: ThemeData(
16        // This is the theme of your application.
17        //
18        // TRY THIS: Try running your application with "flutter run". You'll see
19        // the application has a blue toolbar. Then, without quitting the app,
20        // try changing the seed color in the colorscheme below to Colors.green
21        // and then press "hot reload" (save your changes or press the "hot
22        // reload" button in a Flutter-supported IDE, or press "F5" if you used
23        // the command line to start the app).
24        //
25        // Notice that the counter didn't reset back to zero; the application
26        // state is not lost during the reload. To reset the state, use hot
27        // restart instead.
28        //
29        // This works for code too, not just values: Most code changes can be
30        // tested with just a hot reload.
31        colorscheme: Colorscheme.fromSeed(seedColor: Colors.deepPurple),
32        useMaterial3: true,
33      ), // These two
34      home: const MyHomePage(title: 'Flutter Demo Home Page'),
35    );
36  }
37}
```

The status bar at the bottom of the code editor shows "Ln 13, Col 24" and other standard IDE information.

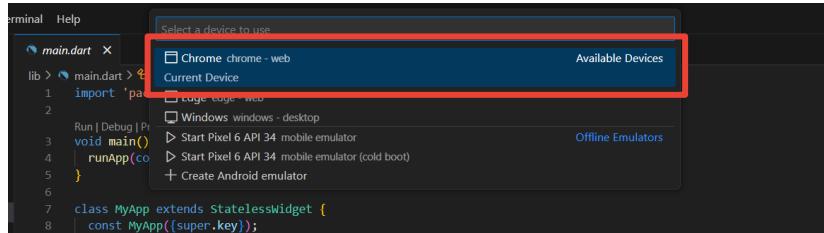
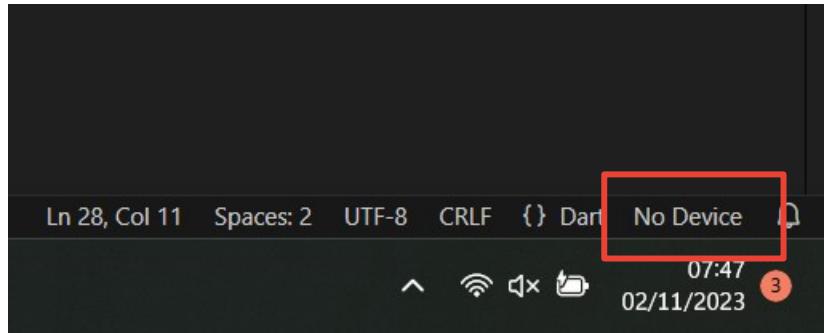
Note: your application code is in **lib/main.dart**

Select Target Platform (Web)

You can select target platform in bottom-right corner.

The pop up will appear to let you choose desired target platform.

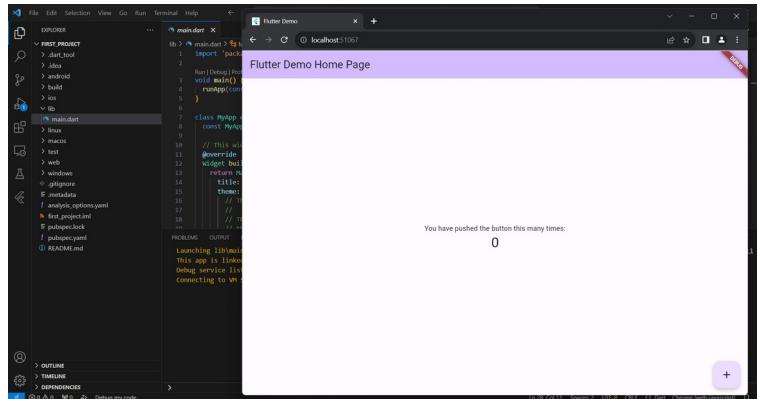
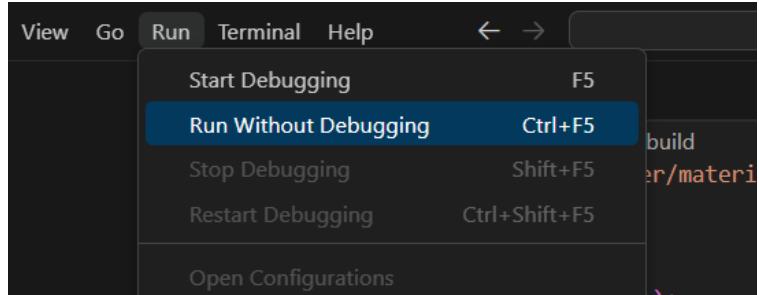
Select **Chrome** or **Edge** to run Flutter on the web.



Run Flutter Project (Web)

It's quite easy to run Flutter project.
Just go to top bar and select **Run > Run Without Debugging**.

Just wait a little bit, and Flutter will serve you a web in a browser.



Select Target Platform (Desktop)

You can only select desktop OS depending on your **current host device**.

For example, your current OS need to be a GNU/Linux to develop Linux desktop application.

And, you are **required** to install current platform's **toolchain** before you develop desktop app there.

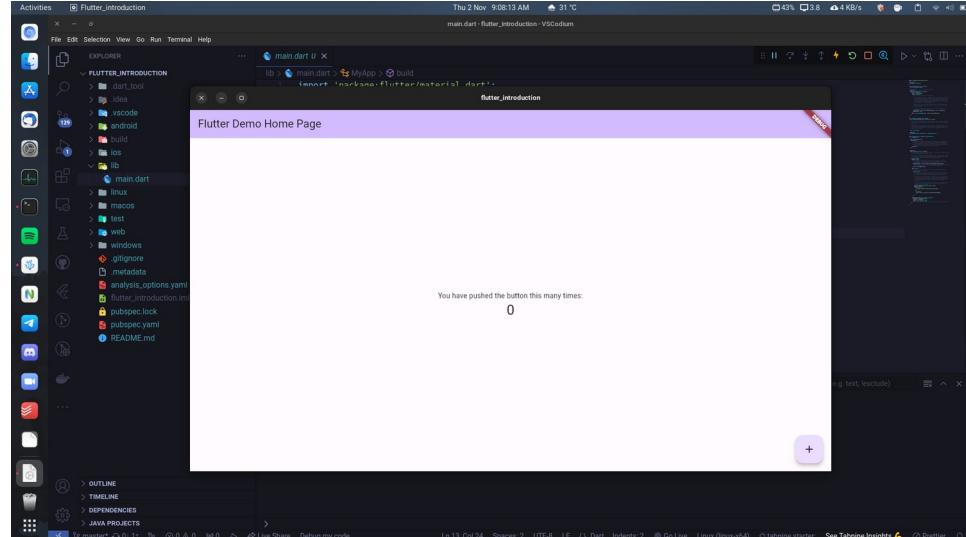
```
flutter_introduction on ✚ master [:$?] is 1.0.0+1 ...
+ flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.13.4, on Ubuntu 22.04.3 LTS 5.19.0-43-generic, locale en_US.UTF-8)
[✓] Android toolchain - develop for Android devices (Android SDK version 33.0.2)
[✓] Chrome - develop for the web
[✓] Linux toolchain - develop for Linux desktop [✓]
[✓] Android Studio (version 2022.3)
[✓] IntelliJ IDEA Community Edition (version 2023.2)
[✓] Connected device (2 available)
[✓] Network resources

• No issues found!
```

Run Flutter Project (Desktop)

Then, you can run the project by selecting desktop platform (Linux, Windows, or MacOS) and click **Run > Run Without Debugging** in the top bar.

Then, you will have your flutter application as a Windows/Linux/MacOS desktop application.





Mobile Development using Flutter

```
Lookup.KeyValue
f.constant(['en
=tf.constant([
.lookup.StaticV
_buckets=5)
```

What'll We Build in Mobile Development Track?

We will build **Android application**, why not iOS?

Easy, because I **don't have** any Apple's product (MacOS,
iPhone, and etc) 🎉

You can build iOS application **on your own** if you want to.

Requirements

Software Requirements:

-  Android SDK & Android Emulator
(from Android Studio)
-  Visual Studio Code (including Dart and Flutter official extensions)

Hardware Requirements:

-  x86_64 CPU architecture with support for a [Windows Hypervisor](#)
-  8 GB RAM or more
-  8 GB of available disk space minimum (IDE + Android SDK + Android Emulator)
-  1280 x 800 minimum screen resolution

Create Android Emulator

You can create Android Emulator from Android Studio.

1. Open Android Studio
2. Select **More Actions > Virtual Device Manager > Create Device.**
3. Choose emulator profile with selected options in the table on the right
4. Click “**Finish**”
5. Run the newly created Android emulator

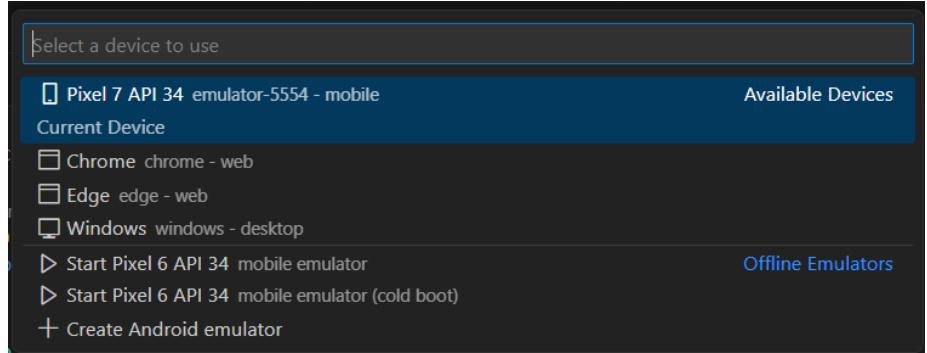
Configuration	Selected Option
Hardware	Phone, Pixel 7
System Image	API 34 (Google APIs)
Additional Configuration	Default



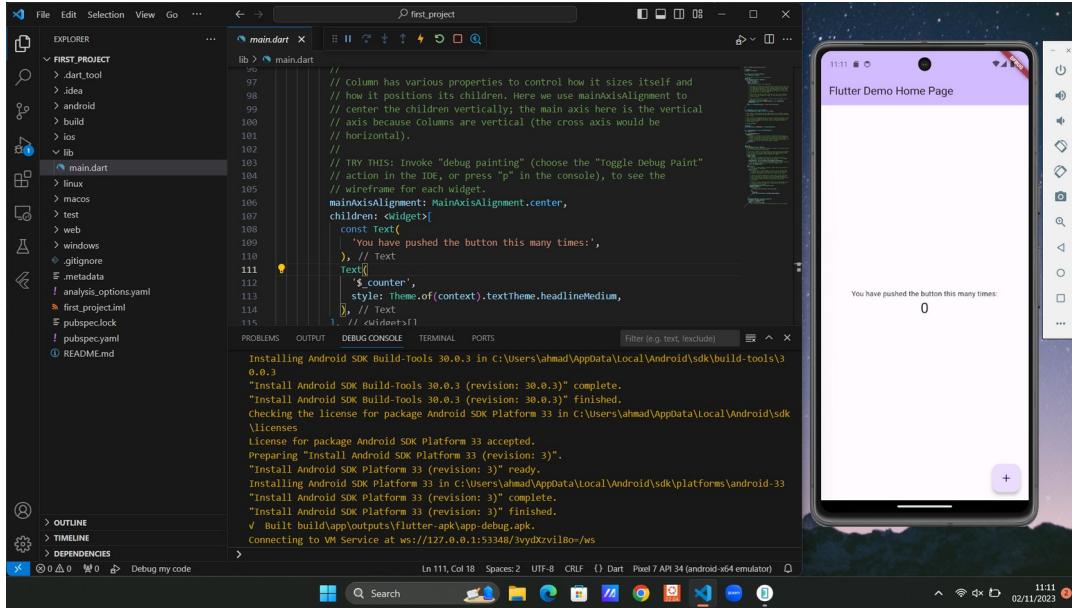
Run Flutter Project

To run Flutter project on Android emulator, do these following instructions:

1. Select your newly created emulator as target device
2. Click **Run > Run Without Debugging**
3. Wait a little bit, then your Flutter project is compiled on your emulator.



Run Flutter Project



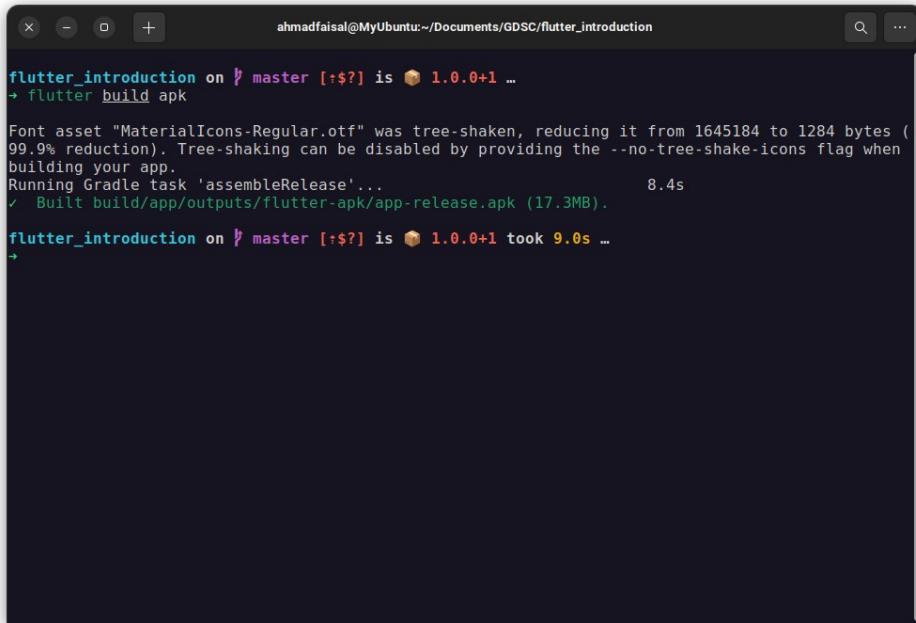
Let's **CODE** and exploring Hot Reload and
Hot Restart!

Let's Package Your Flutter App

You can package your Flutter app to the form of **.apk** by running this command:

flutter build apk

Your built **.apk** file will be stored in **build/app/outputs/flutter-apk/app-release.apk** so that you can install it on your own physical device.



```
ahmadfaisal@MyUbuntu:~/Documents/GDSC/flutter_introduction
flutter_introduction on ✘ master [?:?] is 📦 1.0.0+1 ...
→ flutter build apk

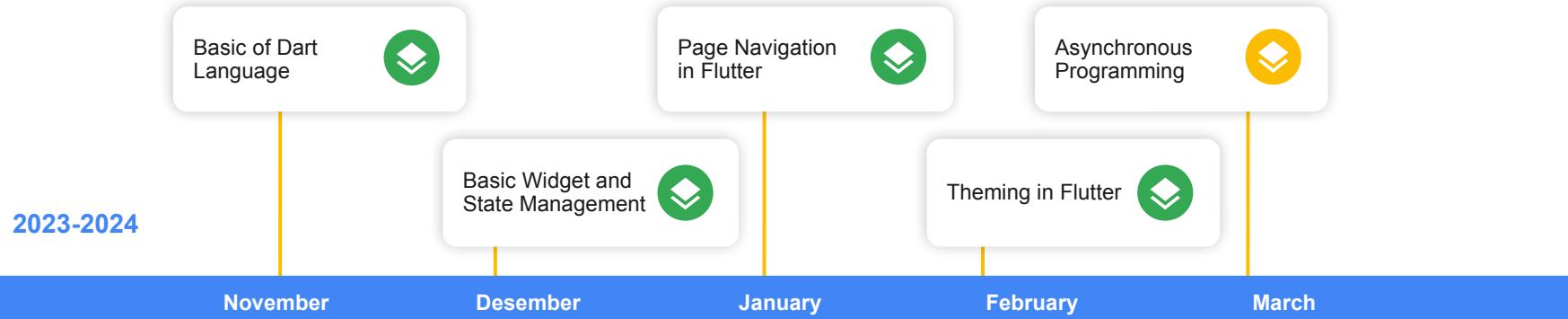
Font asset "MaterialIcons-Regular.otf" was tree-shaken, reducing it from 1645184 to 1284 bytes (99.9% reduction). Tree-shaking can be disabled by providing the --no-tree-shake-icons flag when building your app.
Running Gradle task 'assembleRelease'...                                8.4s
✓ Built build/app/outputs/flutter-apk/app-release.apk (17.3MB).

flutter_introduction on ✘ master [?:?] is 📦 1.0.0+1 took 9.0s ...
→
```

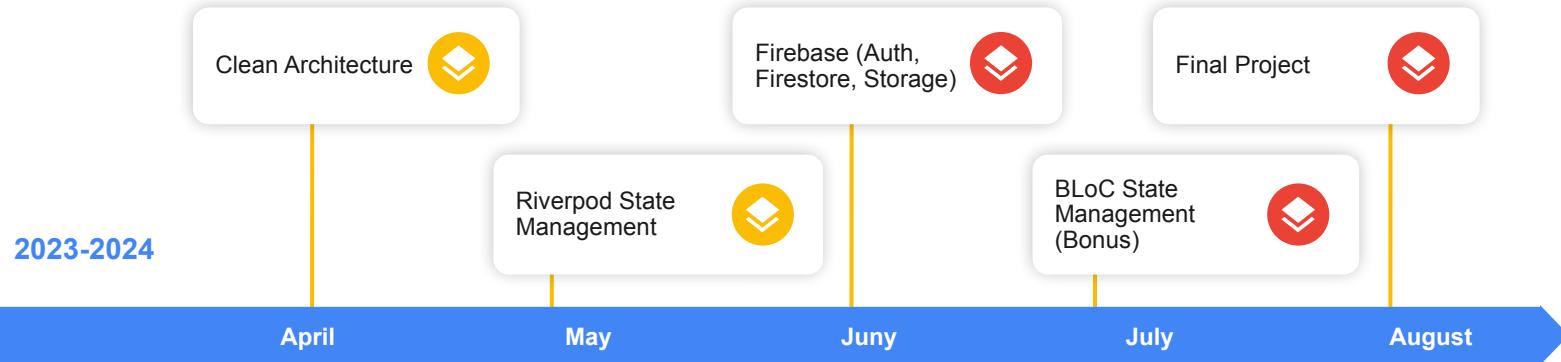


So, What's Next?

Learning Curriculum of Mobile Development Track



Learning Curriculum of Mobile Development Track





 Google Developer Student Clubs

*“You don’t have to be great to start,
but you have to start to be great”*

Zig Ziglar

```
1*/  
child: Column(  
crossAxisAlignment: CrossAxisAlignment.  
children: [  
/*2*/  
Conta  
pad  
chi  
'  
s  
)  
,  
Text(  
'Ka  
sty  
,  
,  
),
```

Thank you!

Any Questions?



<https://bit.ly/ask-mobile-dev>