Flutter.id

Petualangan 10 Hari Menjadi Jagoan Flutter

10 Hari Jago Flutter

Hari #1 - Kenalan dengan Flutter

Hari 1

Kenalan Dulu Sama Flutter

1.1 Apa itu Flutter?

Hai **Jagoan Flutter**, sudah kenalan belum nih sama yang mau dipelajari. Tak kenal maka tak sayang, untuk menjadi jagoan Flutter, kamu harus mengenal, mempelajari dan mempraktekan-nya setiap hari.

Flutter adalah sebuah *framework* dari bahasa *Dart* yang diciptakan oleh *Google*. Flutter merupakan teknologi yang digunakan untuk membuat aplikasi antar *platform* (*Android, iOS, Web, Desktop*) dengan menggunakan satu bahasa pemrograman yaitu *Dart*.

Gambar 1.1 Framework Flutter dan Bahasa Dart

Dart merupakan bahasa pemrograman yang dikembangkan oleh *Google* untuk kebutuhan umum (*general-purpose programming language*). Dart diperuntukan untuk berbagai macam platform diantaranya web, aplikasi mobile, server dan perangkai IoT (Internet of Think).

Gambar 1.2 Platform Bahasa Dart

Dart merupakan bahasa berorientasi objek dengan menggunakan syntax bahasa C. Dart diperkenalkan Google sebagai bahasa pengganti Javascript.

Versi pertama Flutter dikenal sebagai "Sky" dan berjalan pada sistem operasi Android. Diresmikan pada perhelatan Dart Developer Summit 2015, dengan tujuan untuk mampu merender grafis secara konsisten pada 120 bingkai per detik.

Versi Flutter sampai saat ini masih terus di upgrade, berikut versi SDK Flutter untuk versi stabil.

Version	Ref	Release Date
v1.12.13+hotfix.5	27321eb	12/11/2019
v1.9.1+hotfix.6	68587a0	10/24/2019
v1.9.1+hotfix.5	1aedbb1	10/18/2019
v1.9.1+hotfix.4	cc949a8	10/1/2019
v1.9.1+hotfix.2	2d2a1ff	9/10/2019
v1.7.8+hotfix.4	20e5931	7/24/2019

Gambar 1.3 SDK Flutter Versi Stabil

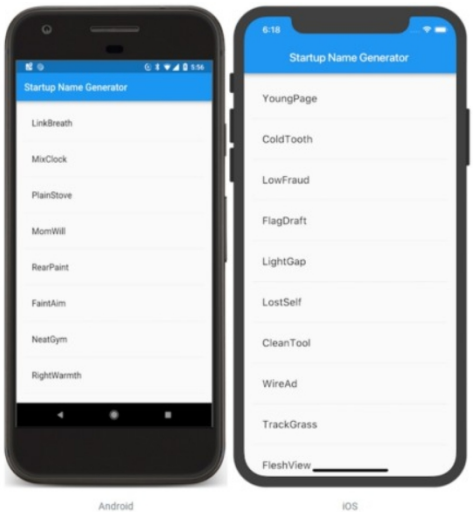
1.2 Kenapa harus pakai Flutter?

Selain Flutter masih ada teknologi lain yang bisa digunakan untuk membuat aplikasi mobile seperti : react native, xamarin, ionic, java, kotlin. Kenapa penulis sarankan untuk memakai Flutter ?

1. Mengembangkan Aplikasi Cross Platform dengan satu codebase

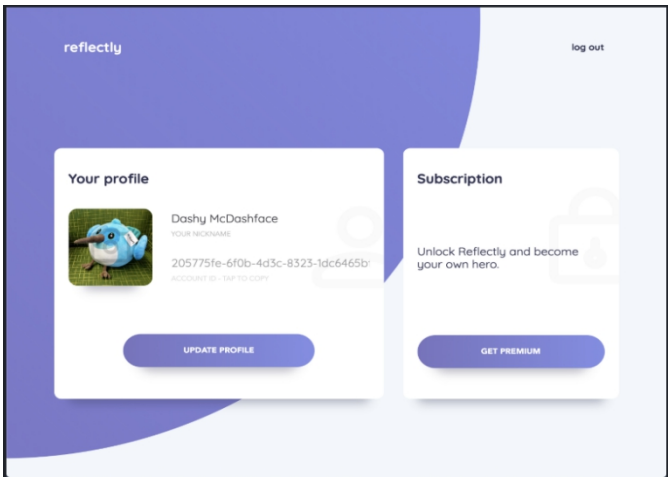
Dengan menggunakan Flutter, developer dapat membuat aplikasi dengan satu bahasa yaitu Dart. Untuk hasilnya bisa dijalankan pada beberapa platform seperti :

Dijalankan pada device Android dan iOS



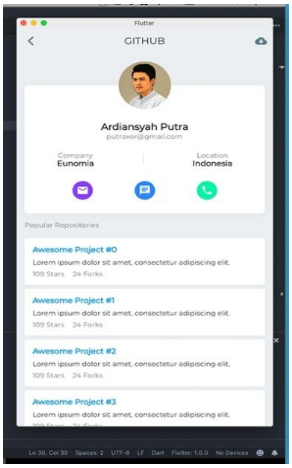
Gambar 1.4 Flutter Android dan iOS

Dijalankan pada web browser



Gambar 1.5 Flutter Web

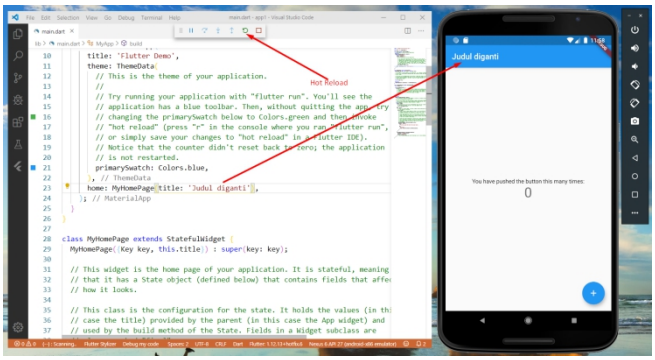
Dijalankan pada Desktop



Gambar 1.6 Flutter Deskstop

2. Fitur Hot Reload

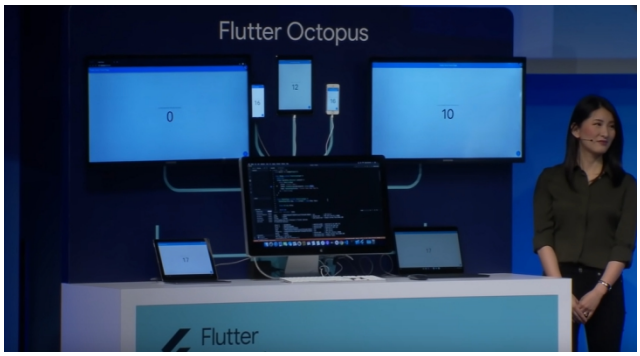
Dengan adanya fitur hot reload meungkinkan kita untuk mengubah kode saat aplikasi berjalan dan dapat melihat perubahannya secara langsung, keren kan !!! yang basic web developer seneng banget nih, terasa seperti ngoding web.



Gambar 1.7 Fitur Hot Reload

2. Flutter Octopus

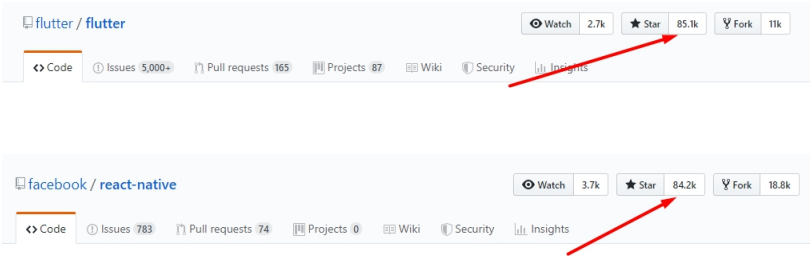
Octopus apa nih, seperti nama hewan? Yup betul sekali, di versi terbarunya Flutter memiliki fungsi seperti hewan octopus / gurita yang memiliki banyak tentakel, Flutter pun meniru fungsi tersebut sehingga dalam satu project bisa di running ke 7 device.



Gambar 1.8 Fitur Flutter Octopus

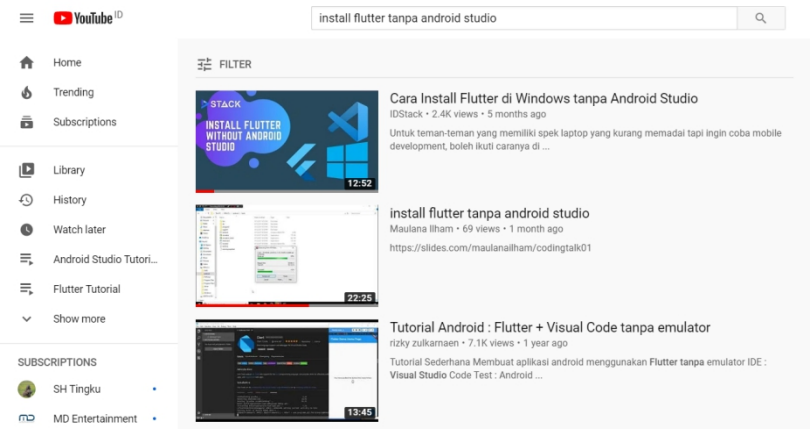
3. Komunitas semakin meningkat

Pada Januari 2020 ini Flutter telah mendapatkan 85.000 lebih star pada github melebihi react native.



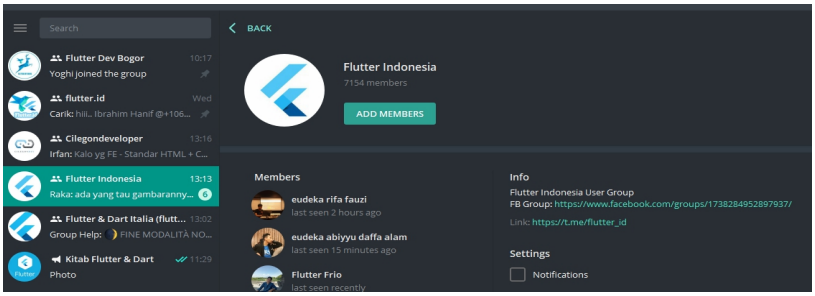
Gambar 1.9 Github start Flutter vs React Native

Selain github, banyak tutorial gratis di youtube salah satunya channel erico darmawan dan idr corner. Tinggal siapkan laptop, secangkir kopi, dan kuota untuk streaming youtube.



Gambar 1.10 Tutorial sudah banyak di youtube

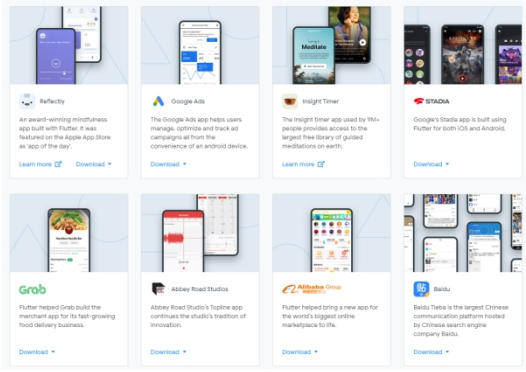
Selain Whatsapp, Telegram merupakan media pembelajaran yang efektif. Telegram grup bisa menampung lebih banyak user bila dibandingkan Whatsapp. Contohnya Group Flutter Indonesia yang sudah mencapai 7.154 members, jika ada kendala dalam ngoding Flutter, **Jagoan Flutter** bisa tanya-tanya disini, tapi sebelumnya bisa tanya mbah Google dulu yach…



Gambar 1.11 Telegram group Flutter indonesia sudah mencapai 7.154 members

4. Kini sudah banyak digunakan oleh perusahaan besar

Saat ini Flutter sudah banyak digunakan oleh perusahaan besar, khususnya untuk aplikasi mobile (Android, iOS) diantaranya jagoan Flutter bisa lihat di https://Flutter.dev/showcase


Gambar 1.12 Perusahaan pengguna Flutter

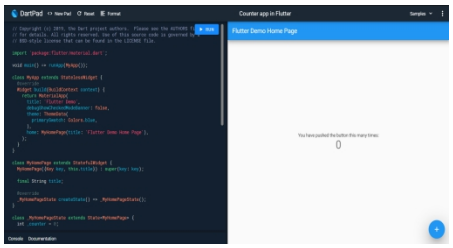
5. Teknologi Masa Depan

Karena Flutter dibuat oleh Google, Google senantiasa membuat event-event untuk Flutter diantaranya Flutter interact 2019.

https://developers.Google.com/events/Flutter-interact

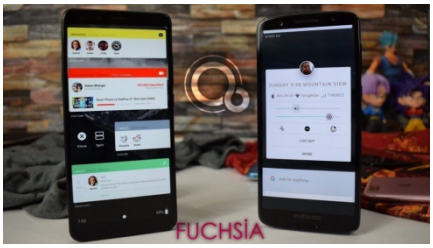
Beberapa update yang didapatkan adalah :

- Update Flutter 1.12 (Google Font, Dark Mode, AndroidX support)
- Flutter Dekstop versi Alpha di MacOs
- Flutter Web satu codebase dengan Mobile.
- Desain UI dengan Adobe XD atau Supernova
- Upgrade Animasi Flare menjadi https://rive.app/
- Dartpad, editor online untuk Flutter
  https://Dartpad.dev/



Gambar 1.13 Dartpad Editor Online Flutter

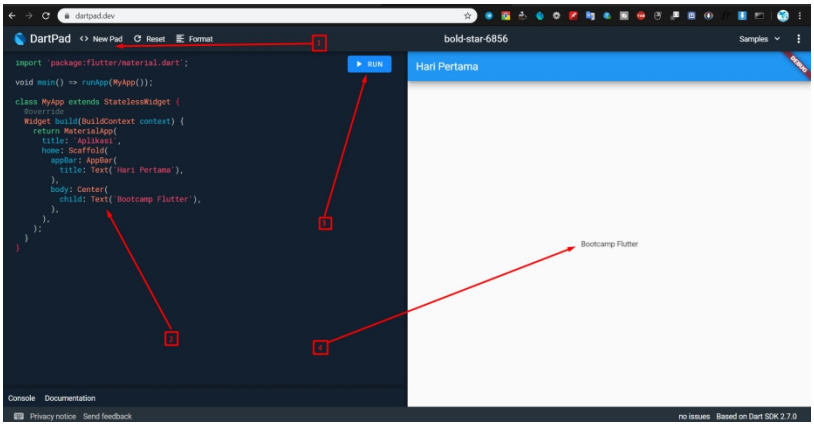
Akan munculnya OS Fuchsia di tahun mendatang merupakan alasan utama penulis pilih Flutter.



Gambar 1.14 OS Google Fuchsia

1.3 Ayo coba Flutter?

Sebelum menginstall Flutter, Jagoan Flutter bisa mencoba ngoding Flutter secara online melalui web : https://Dartpad.dev/

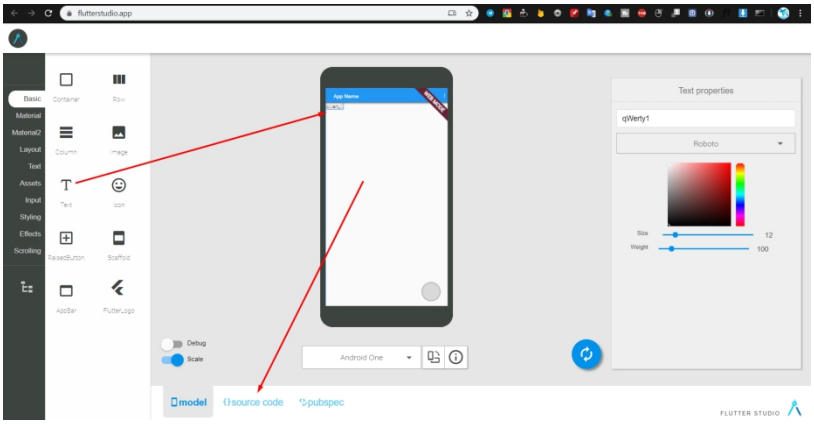


Gambar 1.15 DartPad Flutter

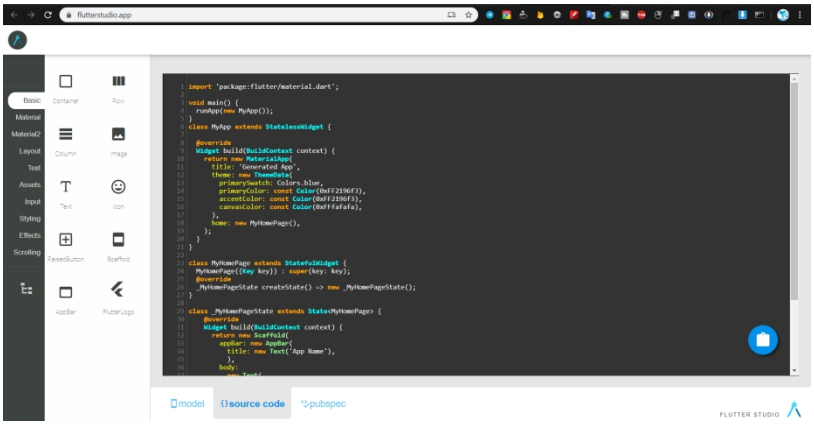
Klik New Pad, tuliskan code, dan run

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Aplikasi',
      home: Scaffold(
        appBar: AppBar(
          title: Text('Hari Pertama'),
        ),
        body: Center(
          child: Text('Bootcamp Flutter'),
        ),
      ),
    );
  }
}
```

Ada satu lagi aplikasi online untuk membuat aplikasi Flutter dengan cara drag n drop yaitu https://Flutterstudio.app/



Gambar 1.16 Model Flutterstudio.app



Gambar 1.17 Source code Flutterstudio.app

1.4 Install Flutter

Karena kemudahan **Flutter** untuk membuat aplikasi **Android** dan **iOS** dalam satu single codebase membuat para developer merasa terbantu untuk hal deployment, dan kita bisa memakai Flutter tidak hanya pada satu sistem operasi saja, melainkan Flutter SDK dapat dijalankan di multi platform seperti Windows, Linux dan MacOS.

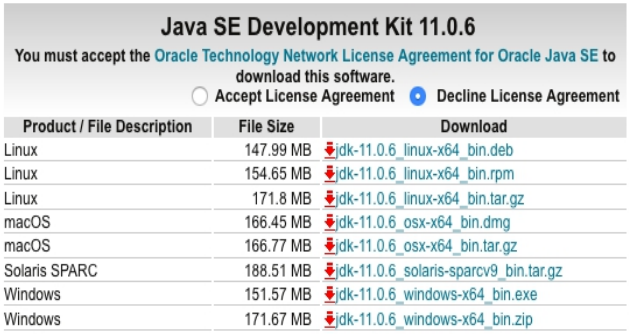
Komponen yang akan kita pakai adalah sebagai berikut:

1. Android Studio
2. Flutter SDK
3. Visual Studio Code

- **Android Studio** – Untuk proses development Flutter kita memerlukan **Android SDK** dan **Google USB Driver** yang berfungsi agar aplikasi dapat di deploy ke real device Android, daripada kita menginstall secara manual kita bisa menggunakan Android Studio untuk mengunduh komponen-komponen yang disebutkan tadi.
- **Flutter SDK** – Pastinya untuk memulai development Flutter kita harus punya Flutter SDK itu sendiri agar project kita bisa berjalan dengan normal
- **Visual Studio Code** – Text Editor yang akan kita pakai untuk membuat program dengan Flutter.

Step 1: Installing Android Studio

- Pastikan kalian sudah menginstall Java JDK terlebih dahulu dan dapat diunduh pada link berikut https://www.oracle.com/technetwork/java/javase/downloads/jdk11-downloads-5066655.html (Sesuaikan dengan sistem operasi yang dipakai)



Java SE Development Kit 11.0.6

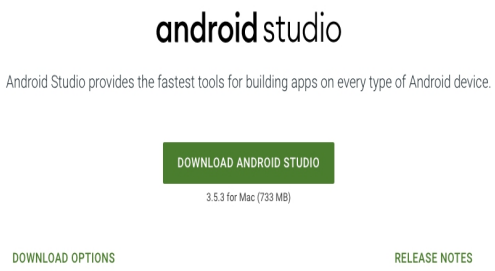
You must accept the Oracle Technology Network License Agreement for Oracle Java SE to download this software.

○ Accept License Agreement ● Decline License Agreement

Product / File Description	File Size	Download
Linux	147.99 MB	jdk-11.0.6_linux-x64_bin.deb
Linux	154.65 MB	jdk-11.0.6_linux-x64_bin.rpm
Linux	171.8 MB	jdk-11.0.6_linux-x64_bin.tar.gz
macOS	166.45 MB	jdk-11.0.6_osx-x64_bin.dmg
macOS	166.77 MB	jdk-11.0.6_osx-x64_bin.tar.gz
Solaris SPARC	188.51 MB	jdk-11.0.6_solaris-sparcv9_bin.tar.gz
Windows	151.57 MB	jdk-11.0.6_windows-x64_bin.exe
Windows	171.67 MB	jdk-11.0.6_windows-x64_bin.zip

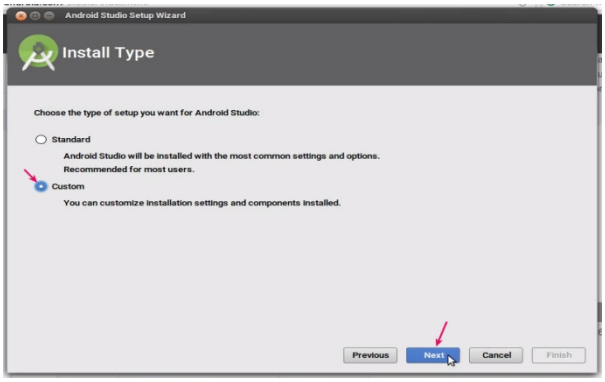
Gambar 1.18 Java JDK

- Sekarang kalian perlu mengunduh Android Studio pada link berikut https://developer.android.com/studio



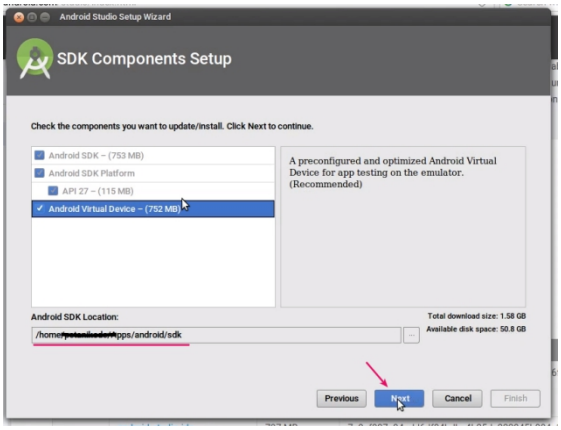
Gambar 1.19 Android Studio

- Setelah proses download selesai silahkan buka dan mulai menginstall Android Studionya. Untuk type instalasinya kalian bisa pilih Custom jika ingin menggunakan tema kustom.



Gambar 1.20 InstallType

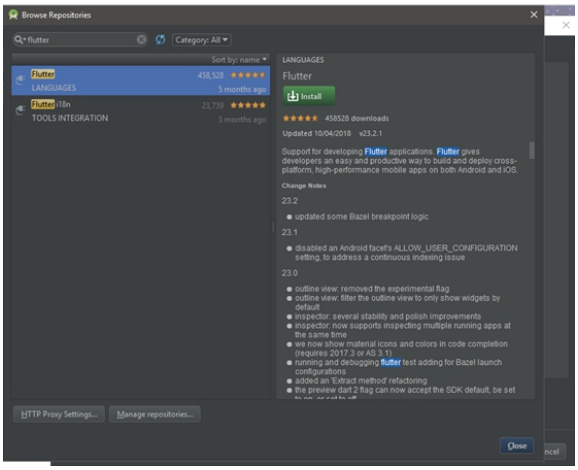
- Kemudian pada proses install komponen pastikan sudah mencentang **Android Virtual Device** dan **Android SDK Platform**.



Gambar 1.21 SDK Components Setup

- Setelah instalasi berhasil jangan lupa untuk mengaktifkan Flutter extension di Android Studio. Proses ini bisa di skip jika kalian tidak ingin menggunakan Android Studio sebagai tempat untuk ngoding Flutter.

- Jika sudah selesai pilih menu **Configure -> Plugins->Browse Repositories**, kemudian cari dengan kata kunci **Flutter**. Maka pencarian Flutter akan muncul pada urutan nomor 1. Tekan tombol install untuk memasang Flutter di Android Studio.

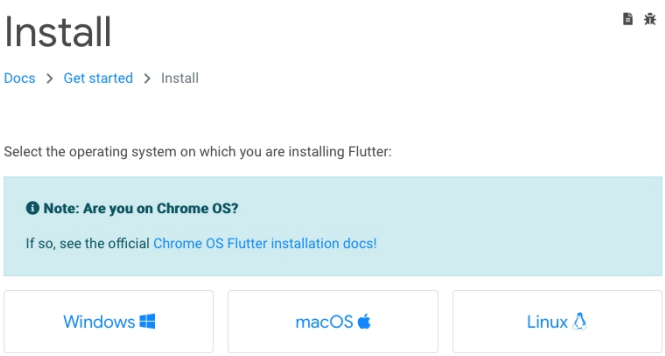


Gambar 1.22 Plugin Flutter

Step 2: Installing Flutter

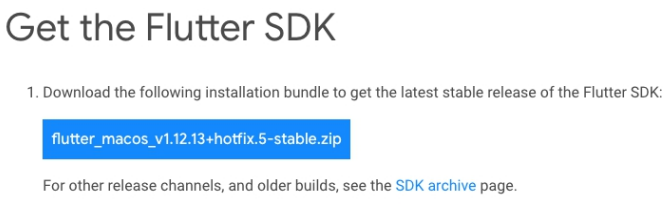
- Pertama kita harus mengunduh Flutter SDK dari situs resminya pada link berikut https://Flutter.dev/docs/get-started/install

- Kita akan disuguhkan 3 tipe pilihan sistem operasi untuk mengunduh SDK, pilih sesuai dengan sistem operasi yang dipakai



Gambar 1.23 Install Flutter

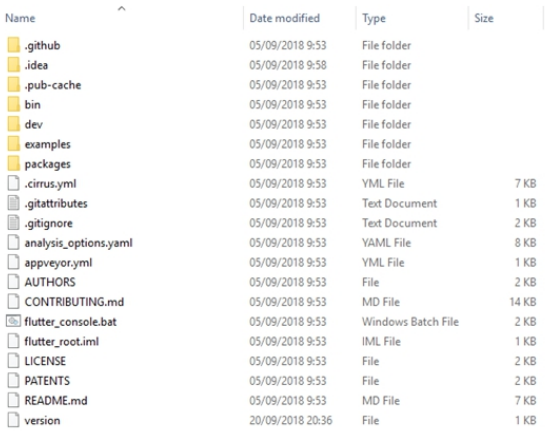
- Unduh file Flutter SDK versi stable dengan menekan tombol biru pada step pertama di situs resminya



Gambar 1.24 Flutter SDK

- Setelah proses instalasi selesai silahkan extract file tersebut di lokasi folder yang diinginkan. Perlu diperhatikan karena lokasi tersebut akan kita masukkan ke dalam Environment Variable, jadi perlu diingat kembali dimana kita menaruhnya.

- Kira-kira isi foldernya adalah sebagai berikut:



Name	Date modified	Type	Size
.github	05/09/2018 9:53	File folder	
.idea	05/09/2018 9:58	File folder	
.pub-cache	05/09/2018 9:53	File folder	
bin	05/09/2018 9:53	File folder	
dev	05/09/2018 9:53	File folder	
examples	05/09/2018 9:53	File folder	
packages	05/09/2018 9:53	File folder	
.cirrus.yml	05/09/2018 9:53	YML File	7 KB
.gitattributes	05/09/2018 9:53	Text Document	1 KB
.gitignore	05/09/2018 9:53	Text Document	2 KB
analysis_options.yaml	05/09/2018 9:53	YAML File	8 KB
appveyor.yml	05/09/2018 9:53	YML File	1 KB
AUTHORS	05/09/2018 9:53	File	2 KB
CONTRIBUTING.md	05/09/2018 9:53	MD File	14 KB
flutter_console.bat	05/09/2018 9:53	Windows Batch File	2 KB
flutter_root.iml	05/09/2018 9:53	IML File	1 KB
LICENSE	05/09/2018 9:53	File	2 KB
PATENTS	05/09/2018 9:53	File	2 KB
README.md	05/09/2018 9:53	MD File	7 KB
version	20/09/2018 20:36	File	1 KB

Gambar 1.25 Folder SDK Flutter

Step 3: Setting Environment Variable

Sebelum Flutter dapat digunakan melalui terminal kita perlu melakukan setting environment variable terlebih dahulu, karena proses ini berfungsi sebagai shortcut path agar bisa menggunakan perintah Flutter pada terminal.

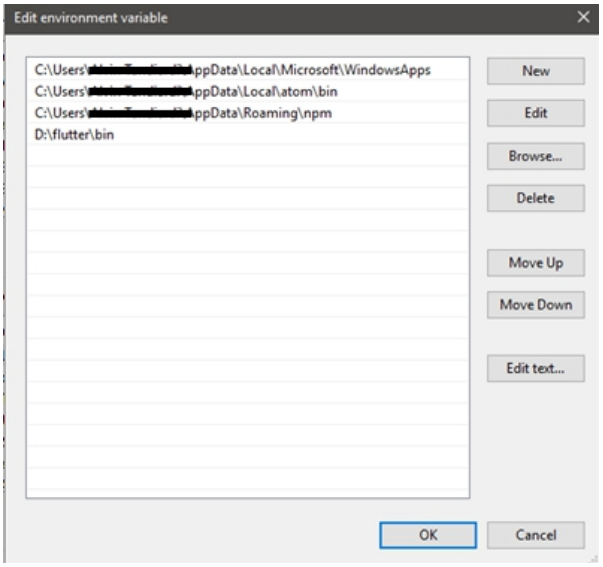
Untuk cara penyetingannya mungkin berbeda pada tiap sistem operasi, saya akan memberikan panduan satu persatu sesuai sistem operasi yang ada.

Windows

- Pada Start Search di windows cari dengan kata kunci "env" maka akan muncul pilihan **Setting Environment Variable**->**Environment Variable**.

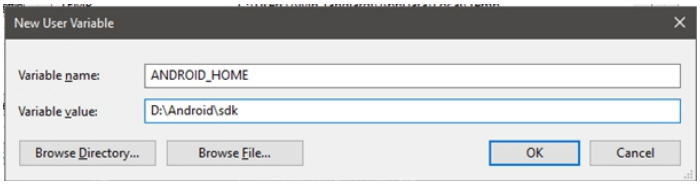
- Dibagian Users Variable dan System Variables klik 2x item PATH, maka akan muncul form edit environment variable seperti berikut:

- Silahkan klik **New** dan masukkan PATH lokasi dimana kita menginstall Flutter.



Gambar 1.26 Path di windows

- Selanjutnya pada User Variable tekan tombol New dan tambahkan PATH untuk ANDROID_HOME sebagai berikut:



Gambar 1.27 User Variabel

Linux

- Buka terminal lalu masukkan perintah berikut
  - sudo nano ~/.bashrc
- Kemudian pada baris paling akhir tambahkan konfigurasi berikut:
  - export ANDROID_HOME=$HOME/Android/Sdk
  - export PATH=$PATH:$ANDROID_HOME/tools
  - export PATH=$PATH:<lokasi install Flutter>/bin
- Tutup dan simpan editor kemudian refresh ~/.bashrc nya dengan perintah
  - source ~/.bashrc

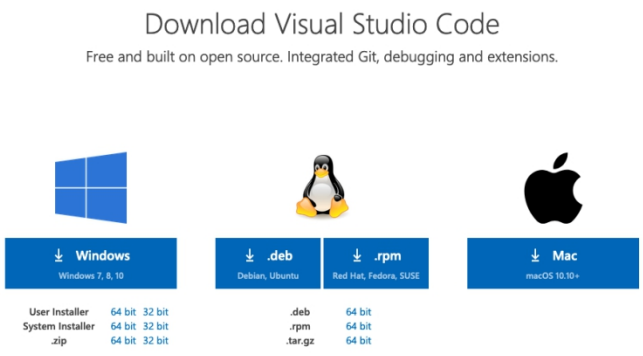
MacOS

- Buka terminal lalu masukkan perintah berikut
  - sudo nano ~/.bash_profile
- Kemudian tambahkan baris konfigurasi berikut:
  - export ANDROID_HOME=<lokasi install android sdk>
  - export PATH=$PATH:$ANDROID_HOME/tools
  - export PATH=$PATH:<lokasi install Flutter>/bin
- Tutup dan simpan editor kemudian refresh ~/.bash_profile nya dengan perintah
  - source ~/.bash_profile

Step 3: Install Visual Studio Code

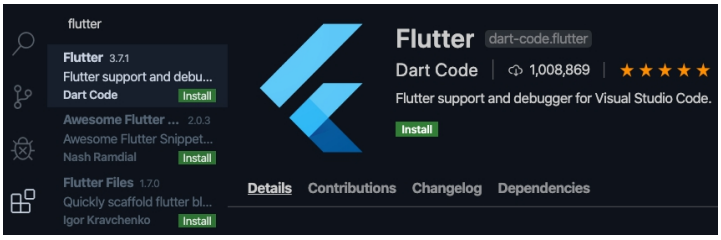
Karena kita nanti serempak tidak menggunakan Android Studio dan lebih fokus dengan Visual Studio Code, maka diwajibkan untuk mengunduhnya terlebih dahulu

- Kita dapat mengunduh Visual Studio Code pada link berikut https://code.visualstudio.com/download (Pilih sesuai sistem operasi yang dipakai)



Gambar 1.28 Download vscode

- Jika proses instalasi sudah berhasil sekarang waktunya menginstall extension Flutter di Visual Studio Code. Klik pada menu extension di sebelah kiri, kemudian cari dengan kata kunci "Flutter", maka kita akan mendapatkannya di urutan pertama. Tekan install dan restart jika sudah selesai



Gambar 1.29 Extension Flutter

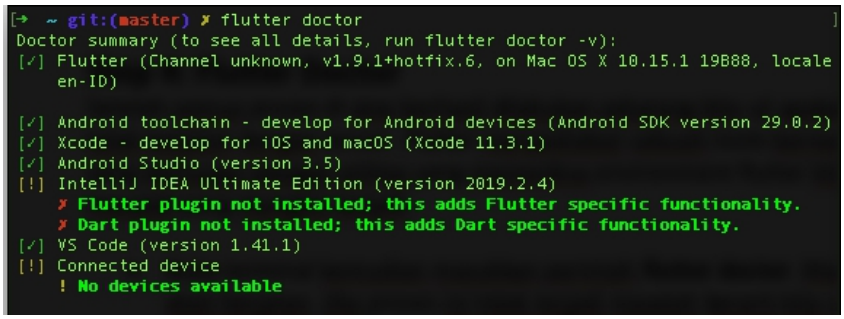
Step 4: Flutter Doctor

Setelah semua proses di atas berhasil dilakukan sekarang kita uji apakah proses instalasi kita telah berjalan dengan lancar. Flutter menyediakan sebuah tools bernama **Flutter Doctor** yang berfungsi sebagai utilitas yang memeriksa environment Flutter kita terhadap masalah-masalah yang mungkin terjadi.

- Buka terminal kemudian masukkan perintah **Flutter doctor**. Maka proses diagnosa akan berjalan. Jika proses ini tidak terjadi masalah berarti kita sudah bisa mulai develop Flutter menggunakan Visual Studio Code.

Note: Jika kalian menemukan error mengenai android-license yang belum terpasang, silahkan masukkan perintah berikut untuk memperbaikinya

- Flutter doctor –android-licenses



Gambar 1.30 Flutter Doctor

Merah tersebut karena saya tidak menginstall Flutter di IntelliJ, penulis tidak pakai itu untuk ngoding Flutter melainkan VSCode.

1.5 Flutter CLI (Command Line Interface)

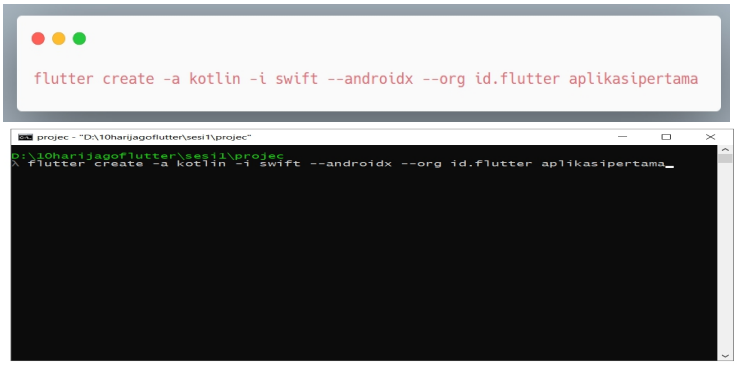
Flutter CLI adalah sebuah perintah untuk melakukan tugas seperti pada IDE. Dengan menggunakan perintah di terminal kita bisa men-debug, me-load ulang, clean package, dll. Dengan adanya Flutter CLI membuat kita lebih mudah untuk mengembangkan aplikasi Flutter dengan menggunakan terminal tanpa menggerakkan mouse di sekitar layar.

1. Membuat Project Baru

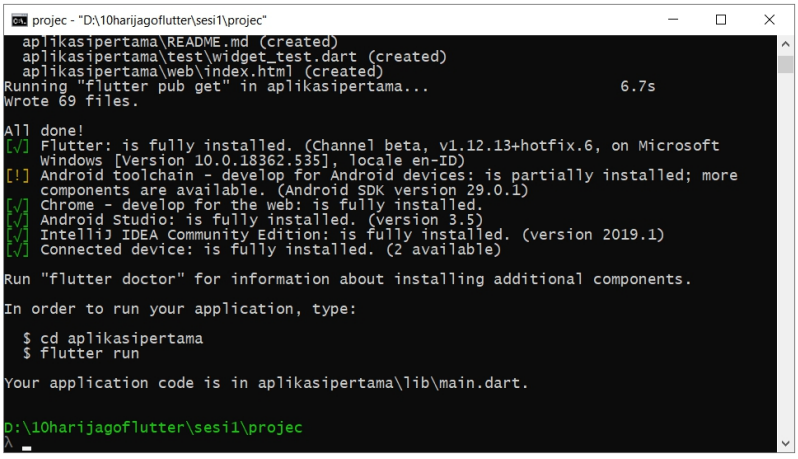
Silahkan buka terminal dan masukkan perintah berikut untuk membuat project baru

Flutter create -a kotlin -i swift --androidx --org com.yourdomain appname

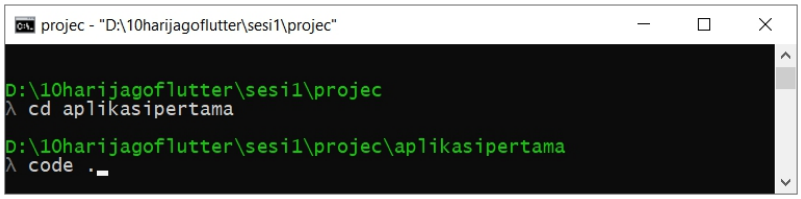
- -a – Kita akan menggunakan kotlin sebagai Bahasa Androidnya
- -i – Kita akan menggunakan Swift sebagai Bahasa iOS nya
- --androidx – Kita akan otomatis migrate project ke android-X
- --org – Isikan dengan package name kita
- appname – Isikan dengan nama project kita



Gambar 1.31 Membuat Project baru

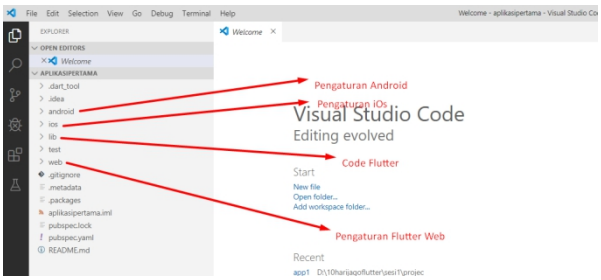


Gambar 1.32 Aplikasi berhasil dibuat



Gambar 1.33 masuk ke project Flutter

cd aplikasi pertama -- masuk ke folder project Flutter
code . -- membuka project Flutter

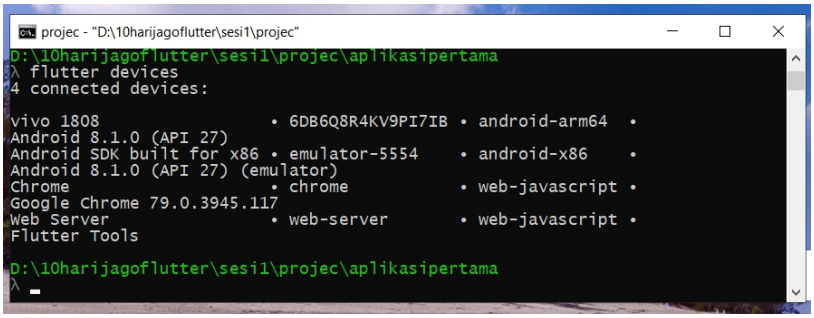


Gambar 1.34 Struktur Flutter

Keunggulan membuat project dengan CLI adalah project sudah termasuk dengan Flutter Web. Silahkan ubah file **lib/main.Dart**

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Aplikasi',
      home: Scaffold(
        appBar: AppBar(
          title: Text('Hari Pertama'),
        ),
        body: Center(
          child: Text('Bootcamp Flutter'),
        ),
      ),
    );
  }
}
```

Jalankan emulator atau hubungkan ke real device, untuk mengecek device yang aktif jalankan perintah : **Flutter devices**

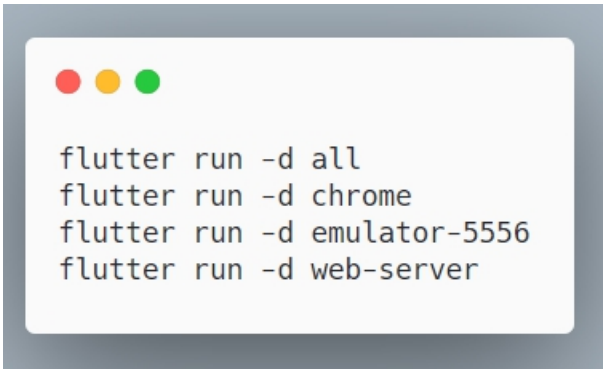


Gambar 1.35 ada 4 connected devices

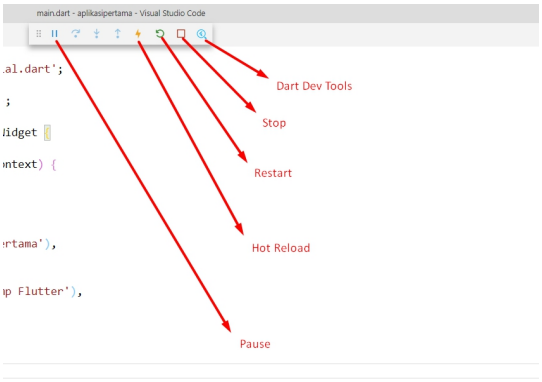
2. Menjalankan Aplikasi

Saatnya kita mencoba Flutter Octopus untuk menjalankan ke tiga devices yaitu, emulator android, real devices android dan chrome

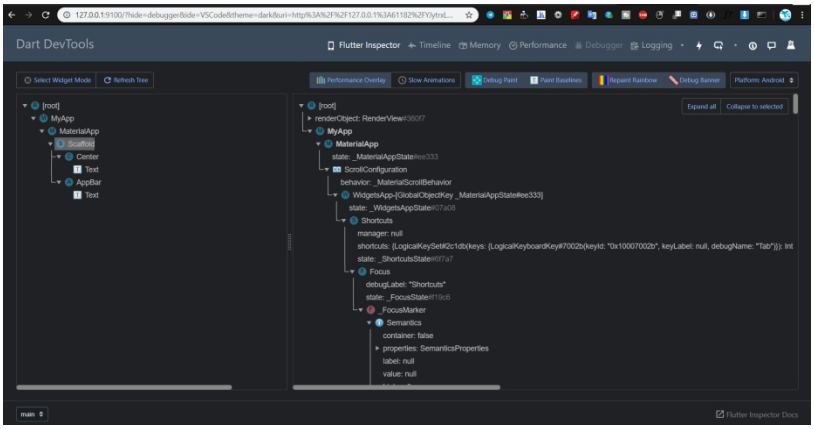
Flutter run -d nama_devices



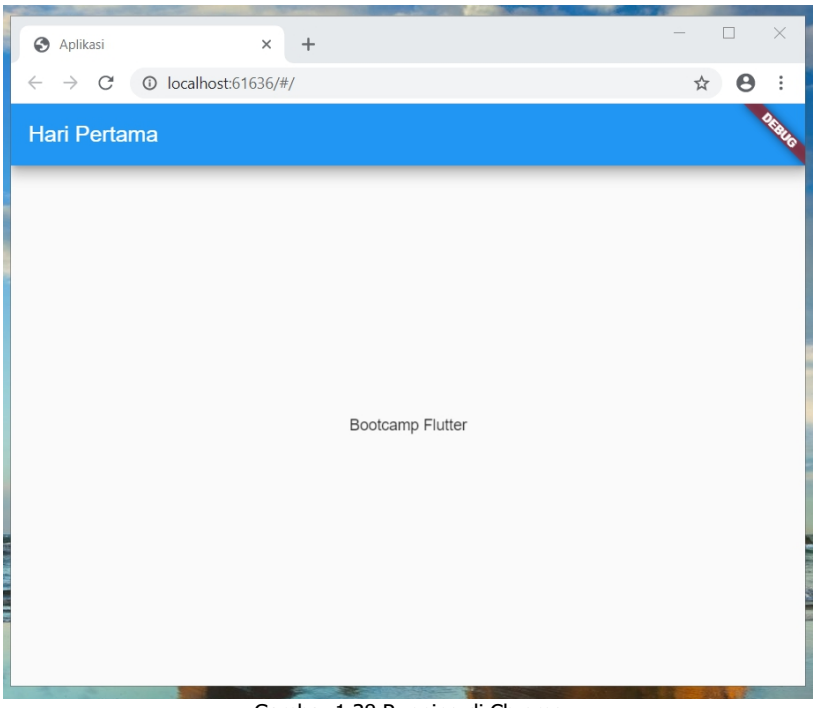
Cara lain untuk running aplikasi adalah dengan menekan F5, maka Flutter akan memilih devices yang aktif



Gambar 1.36 Running Flutter

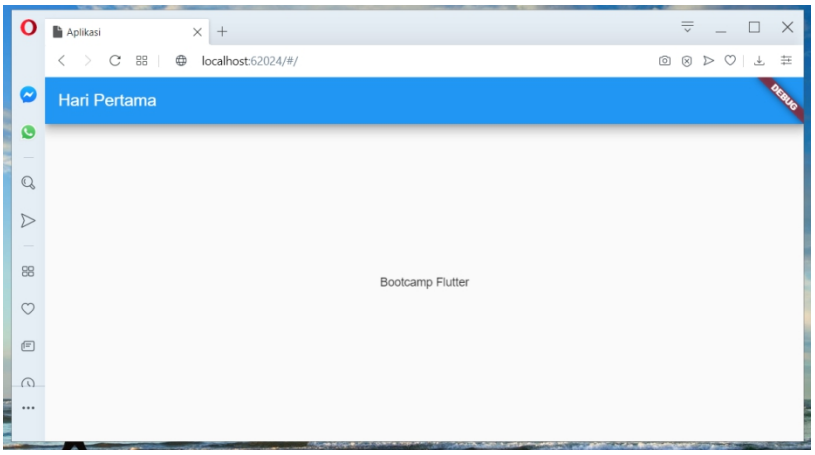


Gambar 1.37 Dart Dev Tools



Gambar 1.38 Running di Chrome

Gambar 1.39 Running di Android



Gambar 1.40 Running di Web Server

3. Perintah CLI Lainnya

Selain perintah diatas, berikut daftar perintah yang sering digunakan

Perintah CLI	Keterangan
Flutter help	Melihat perintah CLI Flutter
Flutter doctor	Memeriksa keadaan instalasi Flutter
Flutter doctor -v	Memeriksa lebih detail
Flutter doctor --android-licenses	Validasi android licenses
Flutter create	Membuat project baru
Flutter create .	Menambahkan Flutter web ke project yang sudah ada
code .	Membuka VScode
Flutter run	Menjalankan 1 device aktif
Flutter run -d all	Menjalankan semua device mobile
Flutter run -d chrome	Menjalankan Flutter di chrome

Flutter run -d web-server	Menjalankan Flutter di Web Server
Flutter run -d namadevice	Menjalankan sesuai nama device
Flutter devices	Melihat devices yang aktif
Flutter upgrade	Upgrade Flutter SDK
Flutter pub get	Menambahkan Package di pubspec.yaml
Flutter pub upgrade	Upgrade versi baru package
Flutter clean	Membersihkan folder build dan Dart_tools folder
Flutter version	Mengecek semua versi Flutter
Flutter channel	Melihat versi Flutter
Flutter emulator	Cek daftar emulator
Flutter emulator --launch nama emulator	Untuk menjalankan emulator
Flutter logs	Melihat log aplikasi yang dijalankan
Flutter build apk	Membuat apk android
Flutter **build** apk --target-platform **andro arm,android-arm64,android-x64** --sp per-ab	Membuat apk android per platform

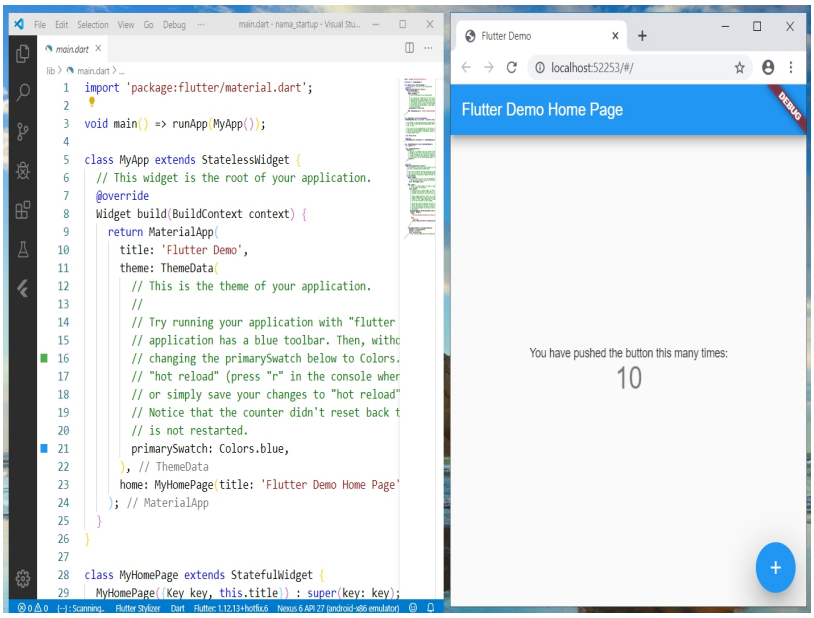
1.6 Latihan Hari 1

1. Membuat Project

Buatlah project baru dengan nama : nama_startup

Ketikan perintah dibawah satu-satu pada terminal.

```
flutter create -a kotlin -i swift --androidx --org id.flutter nama_startup

cd nama_startup

code .

flutter run -d chrome
```



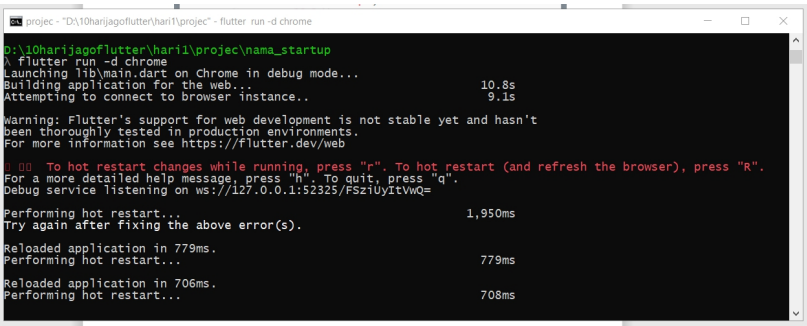
Gambar 1.41 Tampilan awal running aplikasi

Penulis mencoba menjalankan aplikasi pertama kita ke browser chrome. Jagoan Flutter bisa mencoba ke devices lainnya.

Ubah kode pada **lib/main.Dart** menjadi seperti dibawah ini

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Nama Startup',
      home: Scaffold(
        appBar: AppBar(title: Text('Nama Startup'),),
        body: Center(child: Text('Nama Random'),),
      ),
    );
  }
}
```

Untuk **hot reload** silahkan tekan huruf **r** pada terminal, atau huruf **R** untuk **restart** aplikasi.



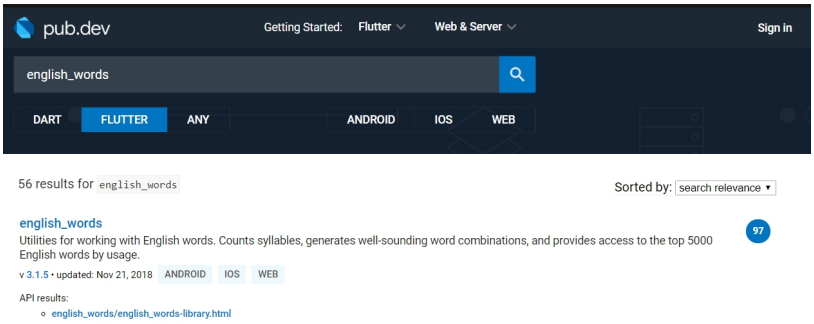
Gambar 1.42 Hot Reload

2. Menggunakan Package External

Flutter menyediakan paket-paket yang dapat digunakan untuk mempermudah dalam pembuatan aplikasi.

Silahkan buka web : https://pub.dev/

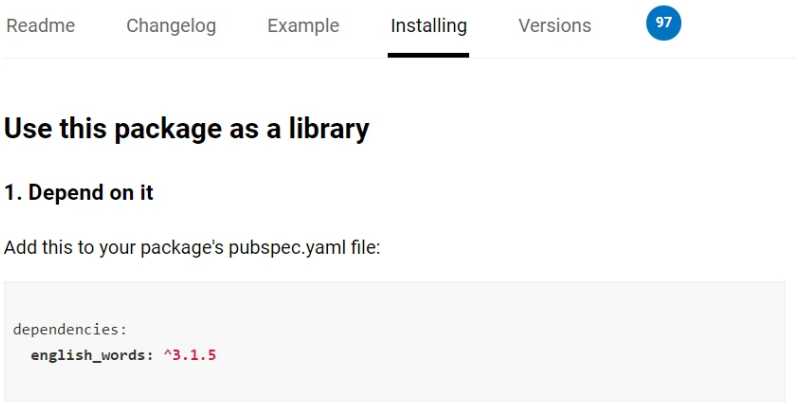
Cari paket : **english_words**



Gambar 1.43 Package English Words

Pilih english_words -> kemudian pilih Installing -> cari nama depedenciesnya.

Gambar 1.44 Installing

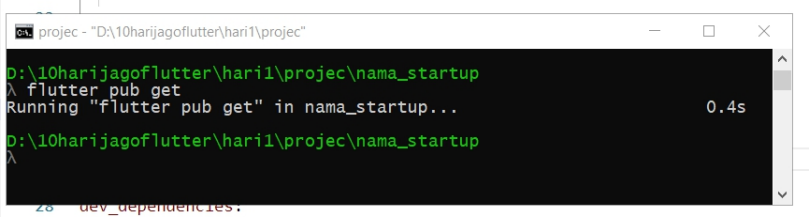


Buka file **pubspec.yaml** dan tambahkan package pada baris 26

```
dependencies:
  Search flutter in Dart Packages
  flutter:
    sdk: flutter

  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  Search cupertino_icons in Dart Packages
  cupertino_icons: ^0.1.2
  Search english_words in Dart Packages
  english_words: ^3.1.5

dev_dependencies:
  Search flutter_test in Dart Packages
  flutter_test:
    sdk: flutter
```

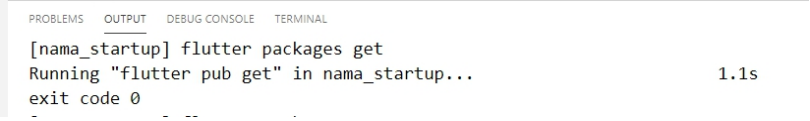
Gambar 1.45 pubspec.yaml

Jalankan **Flutter pub get** untuk menambahkan package.



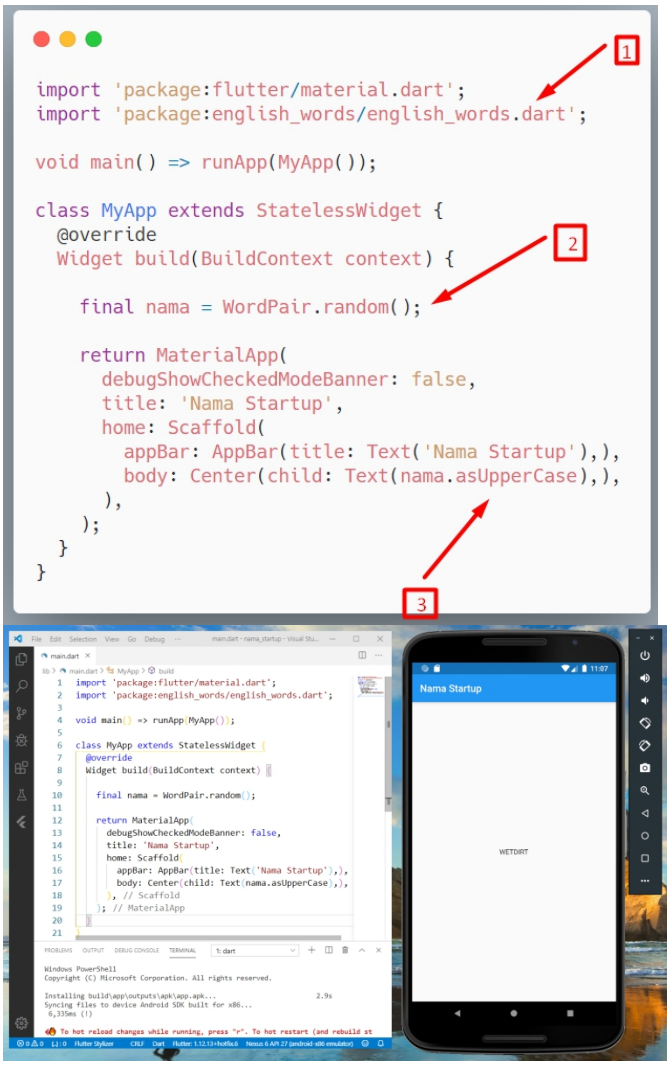
Gambar 1.45 menambahkan package

Atau bisa juga dengan tekan **ctrl + s** untuk menambahkan package.



Gambar 1.46 Menambahkan package

Untuk mencoba package english_words, silahkan ubah file **lib/main.Dart**



Gambar 1.47 Hasil nama random

3. Stateful Widget

Untuk membuat aplikasi menjadi dinamis diperlukan stateful widget.

Ubah **lib/main.Dart** menjadi

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Nama Startup',
      home: Scaffold(
        appBar: AppBar(title: Text('Nama Startup'),),
        body: AcakNama(),
      ),
    );
  }
}

class AcakNama extends StatefulWidget {
  @override
  _AcakNamaState createState() => _AcakNamaState();
}

class _AcakNamaState extends State<AcakNama> {
  @override
  Widget build(BuildContext context) {
    final nama = WordPair.random();
    return Container(
      child: Center(
        child: Text(nama.asUpperCase),
      ),
    );
  }
}
```

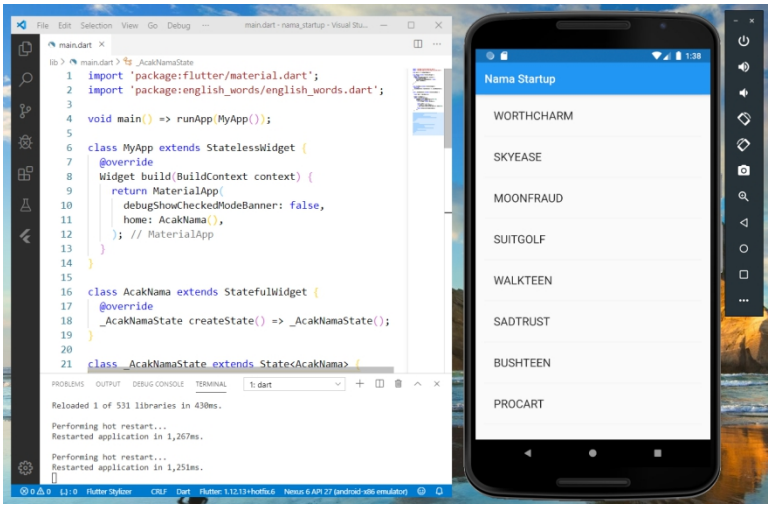
4. Listview

Untuk menampilkan daftar nama startup kita bisa gunakan listview.

Ubah **lib/main.Dart** menjadi

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: AcakNama(),
    );
  }
}

class AcakNama extends StatefulWidget {
  @override
  _AcakNamaState createState() => _AcakNamaState();
}

class _AcakNamaState extends State<AcakNama> {

  final nama = <WordPair>[];

  Widget listNama() {
  return ListView.builder(
      padding: const EdgeInsets.all(16.0),
      itemBuilder:(context, i) {
        if (i.isOdd) return Divider();

        final index = i ~/ 2;
        if (index >= nama.length) {
          nama.addAll(generateWordPairs().take(10));
        }
        return detailList(nama[index]);
      });
  }
```

```dart
  Widget detailList(WordPair detailNama) {
    return ListTile(
      title: Text(
        detailNama.asUpperCase,
        style: TextStyle(fontSize: 20.0),
      ),
    );
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Nama Startup'),
      ),
      body: listNama(),
    );
  }

}
```



Gambar 1.48 Infinity Listview