#03 | Building and Running Flutter

| **Summary** | - Mahasiswa mampu melakukan mode debug, release, profile, dan running sesuai target platform  - Mahasiswa mampu menggunakan dan menjelaskan konsep dari file pubspec.yaml termasuk mengelola aset, library dependencies |
| --- | --- |
| **URL** | 03-building-and-running-flutter |
| **Category** | Mobile Flutter |
| **Environment** | Flutter, mobile, android, iOS, web, desktop |
| **Status** | Published |
| **Feedback Link** | https://github.com/jti-polinema/flutter-codelab/issues |
| **Author** | Tim Pengajar Mobile TSA 2022 |
| **Author LDAP** | hbb-polinema |
| **Analytics Account** | G-H7507PFEJB |

[Building and running Flutter](#_ok7k5uux6)

[Apa yang Anda akan pelajari](#_21yzqg98x7h6)

[Apa yang Anda perlu persiapkan](#_hs6hf8vnv2d6)

[Pengetahuan yang Anda harus dimiliki](#_319h1e1sjzdz)

[Debug mode](#_bwo0af2iwk90)

[Release mode](#_jdmzwkcs72o)

[Profile mode](#_ot43tirxvs6w)

[Supported platforms](#_3tnx3743juyf)

[File pubspec.yaml](#_li2h8jf4xyp6)

[Running the generated project](#_jz06ckvuvhfu)

[File lib/main.dart](#_tvxi8fggv8e)

[Flutter run](#_u3nt87aj8yq6)

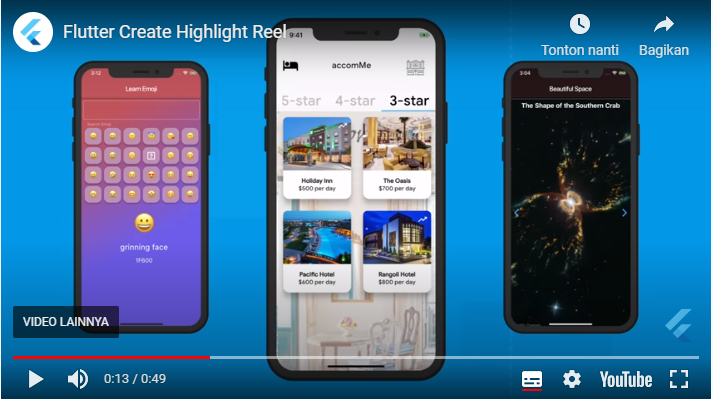
[Tugas Praktikum](#_dz35nqtk0wjb)

[Selamat](#_s22qmcj4zu61)

[Apa selanjutnya?](#_9mqifffit2ew)

[Referensi](#_an95hoqxrlhg)

# Building and running Flutter

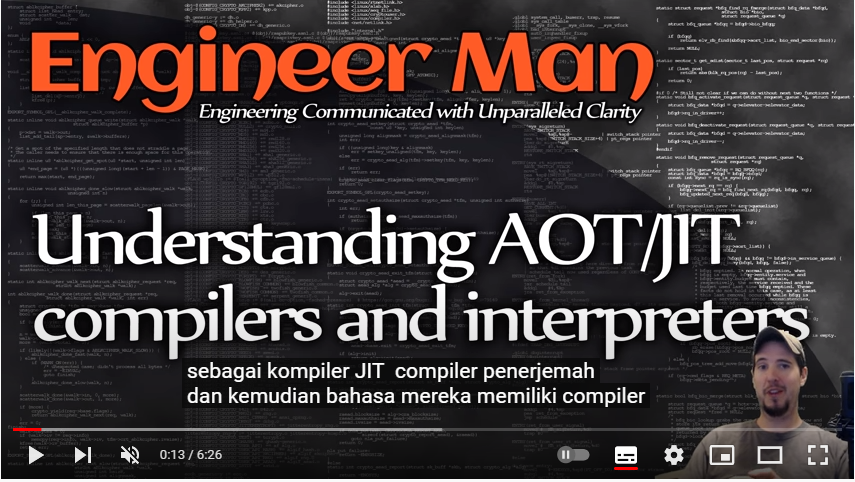


**Terakhir diperbarui:** 10 Agustus 2022

**Penulis:** Tim Pengajar Mobile Flutter

Cara sebuah aplikasi dibangun adalah hal mendasar yang sangat penting untuk diperhatikan karena terkait dengan kinerjanya terhadap target platform yang diinginkan. Ini merupakan bagian langkah yang penting terkait kinerja pembangunan aplikasi. Meskipun Anda tidak perlu mengetahui hal ini untuk setiap jenis aplikasi. Dengan mengetahui bagaimana aplikasi tersebut dibangun, maka dapat membantu Anda untuk memahami dan mengukur kemungkinan terjadinya peningkatan/pengembangan fitur-fitur selanjutnya.

Flutter mengandalkan kompilasi AOT (***Ahead of Time***) dari fitur Dart untuk mode rilis dan kompilasi JIT (***Just in Time***) dari fitur Dart untuk mode pengembangan/debug. Dart adalah satu dari beberapa bahasa pemrograman yang mampu mengompilasi kode ke AOT dan JIT, dan Flutter memanfaatkan kelebihan ini dengan baik.



## Apa yang Anda akan pelajari

Di codelab ini Anda akan mempelajari tentang:

* Building and running Flutter
* Debug mode
* Release mode
* Profile mode
* Supported platforms
* File pubspec.yaml
* Running the generated project

| **Catatan:** Materi ini diadaptasi dari Buku Flutter for Beginners dan Buku Ajar Pemrograman Mobile Flutter JTI Polinema. |
| --- |

## Apa yang Anda perlu persiapkan

* PC atau Laptop dengan spesifikasi minimum RAM 4GB Processor Core i3
* Koneksi internet
* Chrome Browser
* Android Device atau iOS iPhone

## Pengetahuan yang Anda harus dimiliki

* Algoritma dan Pemrograman Dasar
* Pemrograman Berorientasi Objek

# Debug mode

Duration: 12:00

Selama pengembangan, Flutter menggunakan kompilasi JIT dalam mode debug. Kompilasi mode debug dioptimalkan untuk memberikan umpan balik dengan cepat, namun kecepatan eksekusi menjadi tidak terlalu cepat serta ukuran binary menjadi lebih besar. Kelebihan yang dimiliki kompiler Dart membuat interaksi antara kode dan simulator/perangkat masih dirasa sangat cepat, dan alat debugging memungkinkan untuk membantu pengembang melakukan analisis kode program dan tata letak widget.

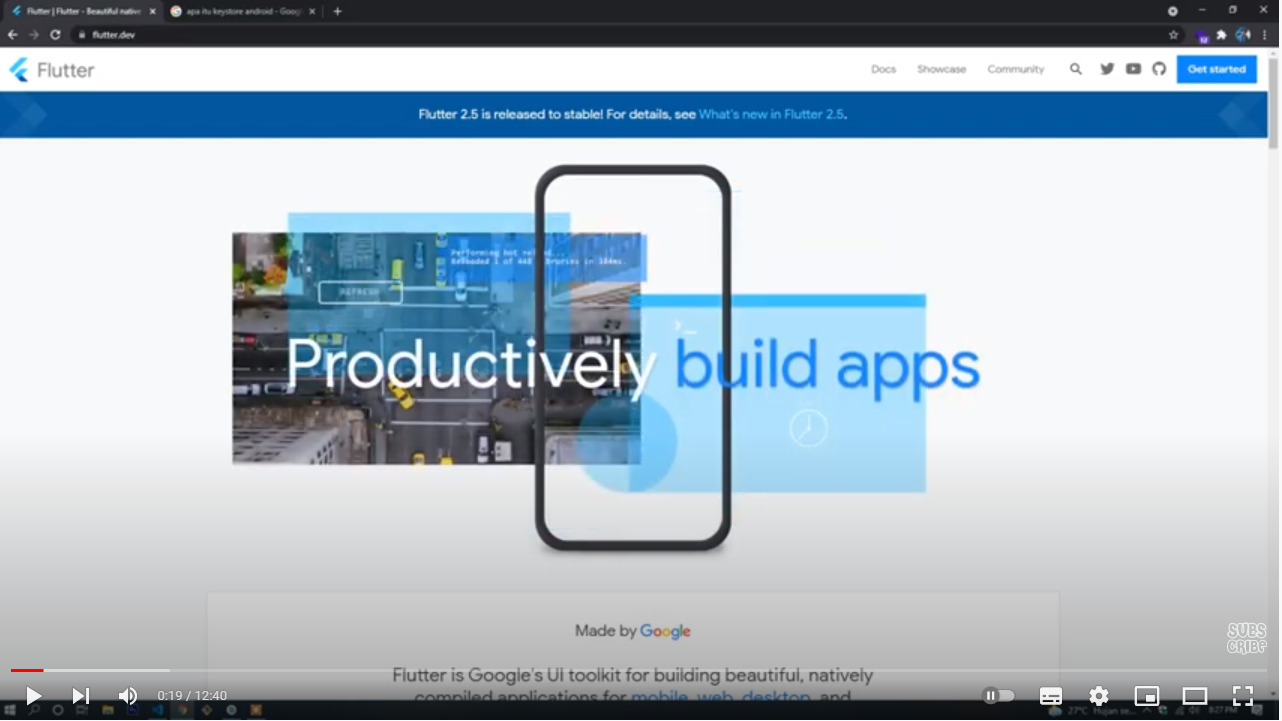


# Release mode

Duration: 10:30

Dalam mode rilis, informasi debug tidak diperlukan, sehingga cukup fokus pada kinerja aplikasi. Flutter menggunakan teknik yang umum digunakan pada *game engine*. Dengan menggunakan mode AOT, kode Dart dikompilasi ke *native code*, dan aplikasi memuat library Flutter dan melakukan proses rendering, input, dan *event handling* dengan Skia engine.

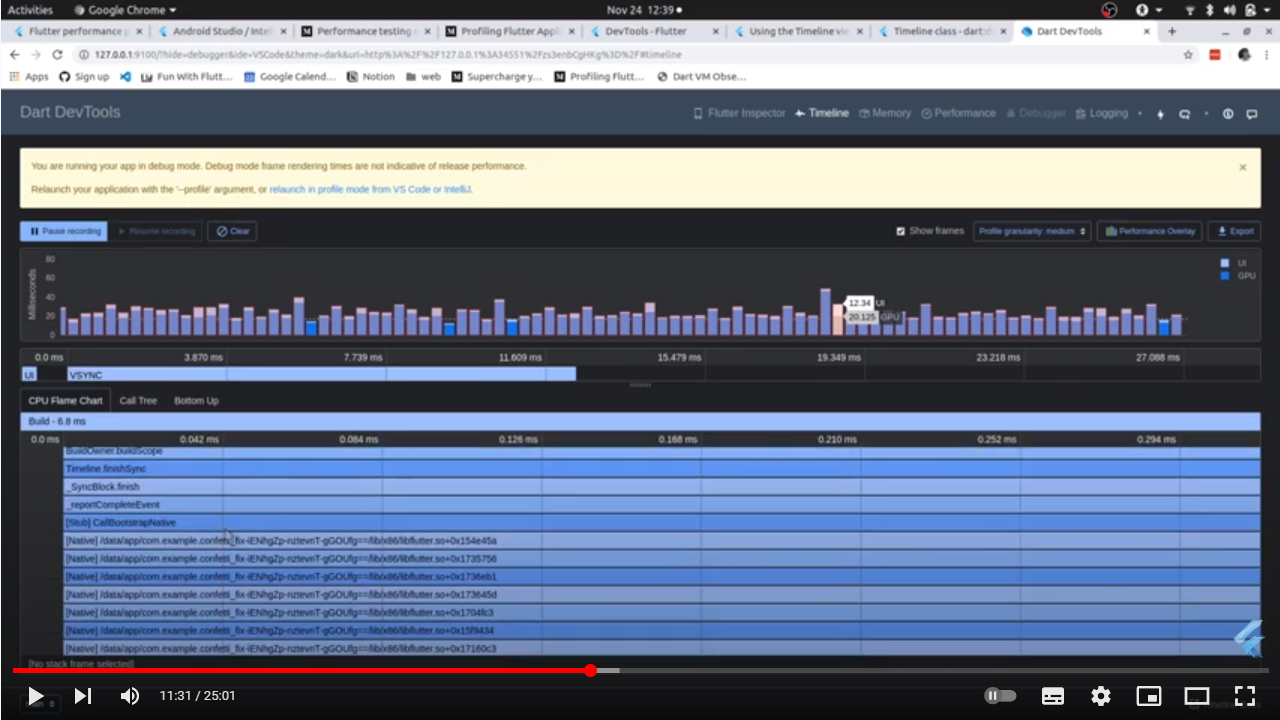
| **Skia graphics library**  Skia adalah open source library yang menyediakan API untuk grafik 2D. Ini digunakan dalam Flutter termasuk aplikasi lain seperti Google Chrome, Android, Firefox, dan banyak lainnya. Itu juga diciptakan oleh Google, seperti Dart dan Flutter. |
| --- |



# Profile mode

Duration: 10:00

Terkadang Anda perlu menganalisis performa aplikasi Anda. Mode profile cukup untuk melakukan proses debugging agar performa aplikasi Anda dapat dianalisis, yang seolah-olah aplikasi Anda digunakan oleh pengguna di dunia. Mode ini hanya tersedia pada perangkat fisik karena jika menggunakan emulator, itu tidak akan merepresentasikan yang sebenarnya.



# Supported platforms

Duration: 15:00

Saat ini, Flutter mendukung perangkat Android dengan arsitektur prosesor ARM yang berjalan minimal pada API 19 (Android 4.4 atau KitKat), dan perangkat iOS di iOS 9 atau lebih baru (termasuk iPhone 4S dan lebih baru). Seperti yang Anda harapkan, aplikasi Flutter dapat dijalankan di emulator, dan debugging berfungsi dengan baik seperti pada perangkat fisik.

Selain itu, Flutter Web masih versi beta, dan dukungan untuk aplikasi desktop (Windows, macOS, dan Linux) tersedia pada versi Alpha. Seperti yang Anda lihat, visi Flutter adalah untuk pengembang dapat membuat satu kode untuk multi platform seperti mobile, web, dan desktop.

| Pada codelab ini tidak akan membahas lebih detail tentang aspek kompilasi Flutter. Untuk informasi lebih lanjut, Anda dapat membaca tautan berikut ini:  <https://docs.flutter.dev/resources/faq#run-android>  <https://docs.flutter.dev/resources/faq#run-ios> |
| --- |

# File pubspec.yaml

Duration: 15:00

File pubspec.yaml di Flutter sebenarnya adalah file yang digunakan untuk mendefinisikan library yang ada pada kode Dart aplikasi kita. Selain itu, juga berisi beberapa bagian tambahan untuk konfigurasi khusus pada aplikasi Flutter kita. Mari kita lihat isi file pubspec.yaml secara detail:

| name: konversi\_suhu  description: A new Flutter project.  version: 1.0.0+1 |
| --- |

Awal file masih sederhana. Seperti yang sudah kita ketahui, properti name didefinisikan ketika kita menjalankan perintahpub create. Berikutnya secara default terdapat bagian description*;* Anda bebas untuk mengubah bagian deskripsi ini menjadi sesuatu yang lebih menarik. Perhatikan bahwa ketika Anda melakukan perubahan dan telah tersimpan pada file pubspec.yaml, IDE Anda mungkin tiba-tiba akan menjalankan perintah flutter pub get

| **Penjelasan ketika membuat project baru**  Seperti pada beberapa bagian dari file pubspec.yaml, Anda dapat menentukan isi atribut deskripsi ketika membuat project baru dengan perintah flutter createditambah argumen –description |
| --- |

Bagian atribut version ini mengikuti konvensi dari paket Dart: nomor versi, ditambah nomor versi build opsional yang dipisahkan oleh tanda +. Selain itu, Flutter memungkinkan Anda untuk mengubah nilai ini selama proses build. Kita akan mempelajari ini nanti ketika proses rilis aplikasi ke versi production atau ke Play Store.

Kemudian terdapat bagian dependensi dari file pubspec:

| environment:  sdk: ">=2.16.1 <3.0.0"  dependencies:  flutter:  sdk: flutter  cupertino\_icons: ^1.0.2  dev\_dependencies:  flutter\_test:  sdk: flutter |
| --- |

Kita mulai pada atribut environment. Bagian ini menentukan versi kode Dart yang akan kita gunakan. Jika dilihat dari gambar di atas, kode akan memerlukan versi Dart 2.12.0 atau yang lebih tinggi, tetapi tidak akan berjalan di versi Dart 3.0.0. Sesuai standar versi, Anda berharap bahwa jika Dart 3.0.0 dirilis, akan terdapat beberapa perubahan yang tidak kompatibel sehingga dapat menghentikan kompilasi kode kita. Hal ini dapat terjadi ketika Dart diperbarui dari 1.x.x hingga 2.x.x. Dengan membatasi versi Dart, berarti kode Anda tidak perlu mendukung versi Dart 3.x.x sampai Anda siap melakukan perubahan pada kode aplikasi kita. Perhatikan bahwa sejak versi Dart 2.12 merupakan versi yang memperkenalkan konsep *null safety*. Nanti akan dibahas lebih rinci pada pertemuan yang akan datang terkait pemrograman Dart.

Kemudian ada bagian dependencies. Bagian ini dimulai dengan dependensi utama aplikasi Flutter, kemudian SDK Flutter itu sendiri, yang berisi banyak paket inti dari framework Flutter.

Sebagai dependensi tambahan, generator menambahkan paket cupertino\_icons, yang berisi aset ikon yang digunakan oleh widget Flutter Cupertino.

Saat Anda menambahkan dependensi lain (dan Anda pasti akan menambahkan banyak dependensi pada bagian ini), maka library-library tersebut juga akan muncul di sini.

Bagian dev\_dependencies hanya berisi paket flutter\_test yaitu dependensi milik Flutter SDK itu sendiri, yang mana berisi ekstensi khusus untuk Dart test package. Kita akan eksplorasi lebih jauh pada materi tentang *testing dan debugging*.

Di bagian blok terakhir pada file pubspec.yaml, ada bagian khusus untuk flutter:

| flutter:  # The following line ensures that the Material Icons font is  # included with your application, so that you can use the icons in  # the material Icons class.  uses-material-design: true  # To add assets to your application, add an assets section, like this:  # assets:  # - images/a\_dot\_burr.jpeg  # - images/a\_dot\_ham.jpeg  # An image asset can refer to one or more resolution-specific "variants", see  # https://flutter.dev/assets-and-images/#resolution-aware.  # For details regarding adding assets from package dependencies, see  # https://flutter.dev/assets-and-images/#from-packages  # To add custom fonts to your application, add a fonts section here,  # in this "flutter" section. Each entry in this list should have a  # "family" key with the font family name, and a "fonts" key with a  # list giving the asset and other descriptors for the font. For  # example:  # fonts:  # - family: Schyler  # fonts:  # - asset: fonts/Schyler-Regular.ttf  # - asset: fonts/Schyler-Italic.ttf  # style: italic  # - family: Trajan Pro  # fonts:  # - asset: fonts/TrajanPro.ttf  # - asset: fonts/TrajanPro\_Bold.ttf  # weight: 700  #  # For details regarding fonts from package dependencies,  # see https://flutter.dev/custom-fonts/#from-packages |
| --- |

Bagian flutter ini memungkinkan kita untuk mengonfigurasi *resources* yang dibundel dalam aplikasi kita untuk digunakan selama runtime, seperti gambar, font, musik, efek suara, dan video.

Mari kita lihat lebih dekat:

* use-material-design: Kita akan membahas widget Material yang disediakan oleh Flutter di pertemuan berikutnya. Selain itu, kita juga dapat menggunakan ikon yang disediakan oleh Material Desain (<https://fonts.google.com/icons?selected=Material+Icons>), yang ada di bagian khusus format font. Agar ini dapat berfungsi dengan baik, kita perlu mengaktifkan bagian properti ini (atur ke nilai true) sehingga ikon dapat digunakan dalam aplikasi.
* assets: bagian properti ini digunakan untuk membuat daftar alamat/lokasi *resource* yang akan dibungkus dengan aplikasi kita. File dan folder assets dapat diatur dengan cara apa pun; yang penting bagi Flutter adalah alamat/lokasi file sudah benar. Anda dapat menentukan alamat/lokasi file relatif ke file project root. Ini dapat digunakan nanti ketika dalam kode Dart Anda perlu merujuk ke file asset tersebut. Berikut ini contoh menambahkan satu gambar:

| assets:  images/home\_background.jpeg |
| --- |

Seringkali Anda ingin menambahkan banyak gambar, dan mencantumkannya satu per satu. Alternatifnya adalah memasukkan sebagai seluruh folder:

| assets:  images/ |
| --- |

Anda dapat menambahkan karakter / di akhir path yang digunakan untuk memasukkan semua file pada folder tersebut. Perhatikan bahwa ini tidak termasuk subfolder; Anda perlu mencantumkannya juga secara manual:

| assets:  images/  images/icons/ |
| --- |

* fonts: bagian properti ini memungkinkan kita untuk menambahkan font kustom ke aplikasi kita.

Kita akan periksa bagaimana cara memuat aset dengan cara yang berbeda. Selain itu, Anda dapat membaca selengkapnya tentang detail spesifikasi aset di web Flutter docs: <https://docs.flutter.dev/development/ui/assets-and-images>

# Running the generated project

Duration: 30:00

Aplikasi secara default yang kita buat sebelumnya memiliki gaya pemrograman React di Flutter. Kita akan melihat lebih detail pada kode Dart di pertemuan yang akan datang, tapi mari kita lihat file main.dart lebih rinci sebelum kita mencoba menjalankan aplikasi.

## File lib/main.dart

Kita eksplorasi file main.dart terlebih dahulu untuk melihat sebuah widget. File ini juga merupakan entri poin dari aplikasi Flutter kita:

| void main() {  runApp(const MyApp());  } |
| --- |

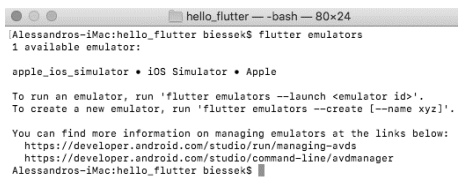
Fungsi main adalah bagian utama pada kode Dart untuk mulai menjalankan aplikasi kita; di sinilah eksekusi aplikasi Anda akan dimulai. Flutter kemudian mengambil alih eksekusi pada fungsi runApp, yaitu dipanggil dengan meneruskan widget tingkat atas (atau root) Anda sebagai sebuah parameter. Ini adalah widget yang kita lihat sebelumnya, yaitu widget MyApp.

## Flutter run

Untuk mengeksekusi aplikasi Flutter, kita harus memiliki perangkat atau simulator yang terhubung. Untuk memeriksanya dapat dilakukan dengan menggunakan flutter doctor yang telah kita bahas di pertemuan sebelumnya, dan flutter emulators tool, yang akan menjalankan emulator/simulator di sistem Anda. Berikut ini adalah perintah yang perlu Anda ketahui untuk melihat ketersediaan emulator Android dan iOS yang dapat digunakan untuk mengeksekusi file project aplikasi kita:

| flutter emulators |
| --- |

Anda akan melihat tampilan yang mirip seperti berikut



Anda dapat memeriksa bagaimana cara mengelola emulator Android Anda di <https://developer.android.com/studio/run/managing-avds> . Untuk simulator perangkat iOS, Anda harus menggunakan alat pengembang Simulator Xcode.

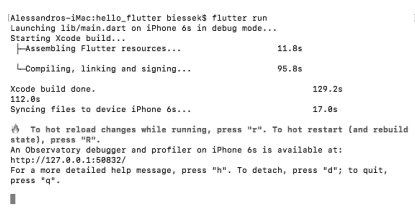
| **Emulator versus simulator**  Anda akan melihat bahwa Android memiliki emulator dan iOS memiliki simulator. Emulator Android merupakan perangkat lunak dan perangkat keras tiruan seperti perangkat Android yang sebenarnya. Sebaliknya, simulator iOS hanya meniru perangkat lunak dari perangkat iOS. Oleh karena itu sangat disarankan agar Anda menguji aplikasi Anda di perangkat iOS yang sebenarnya sebelum merilisnya ke App Store untuk memastikan tidak ada masalah pada perangkat keras yang sebenarnya seperti konsumsi memori yang berlebihan. |
| --- |

Sebagai alternatif, Anda dapat memilih untuk mengeksekusi aplikasi di perangkat fisik. Anda perlu mengatur perangkat Anda ke mode pengembangan (Developer Mode), jadi untuk saat ini mungkin lebih mudah menggunakan emulator atau simulator.

Setelah memiliki perangkat yang terhubung untuk mengeksekusi aplikasi kita, Anda dapat menggunakan perintah berikut:

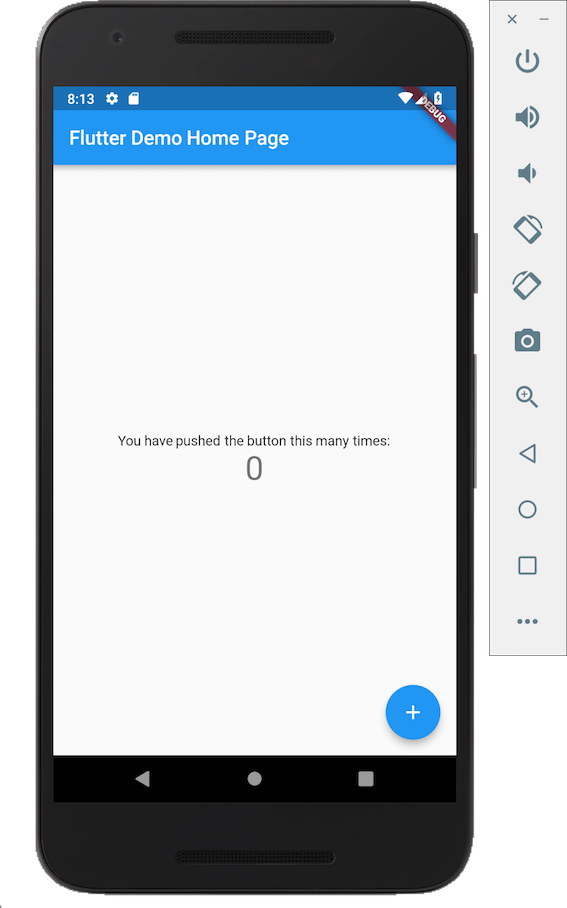
| flutter run |
| --- |

Anda akan melihat output yang mirip seperti pada gambar berikut:



Perintah ini akan mengeksekusi debugger dan membuat fungsi **hot reload** aktif, seperti yang dapat Anda lihat. Eksekusi aplikasi pertama kali biasanya akan memakan waktu sedikit lebih lama dari eksekusi selanjutnya.

Emulator atau simulator seharusnya sudah mulai tampil, setelah jeda untuk memuat sistem operasi, baru kemudian mengeksekusi aplikasi Flutter Anda. Jika Anda melihat layar seperti gambar berikut, maka selamat. Anda baru saja menjalankan aplikasi Flutter pertama Anda!



Aplikasi Anda telah aktif dan berjalan; Anda dapat melihat label debug berwarna merah di sudut kanan atas. Itu berarti aplikasi Anda saat ini bukan versi rilis yang sedang berjalan; aplikasi dalam mode debug, yang berarti Anda memiliki semua fitur pada mode debug yang tersedia untuk Anda, seperti *hot reload* dan fasilitas debug kode.

Contoh gambar di atas menggunakan emulator Android, atau **Android virtual device (AVD)**.

# Tugas Praktikum

Duration: 90:00

Silakan Anda mencoba mengeksekusi aplikasi Flutter pertama Anda boleh menggunakan emulator, simulator, atau device yang sebenarnya.

Kemudian screenshot hasil running aplikasi Anda dan kirimkan ke LMS sebagai tugas mandiri.

(**Opsional**) Anda dapat mencoba melakukan tiga mode seperti yang telah dijelaskan dalam codelab ini, yaitu mode debug, release, dan profile.

# Selamat

Duration: 0:00

Selamat Anda telah menyelesaikan Codelab ini sebagai langkah awal untuk menjalankan aplikasi flutter pertama Anda.

Pada codelab berikutnya, Anda akan mempelajari tentang teknologi pengembangan aplikasi mobile secara native, teknologi pengembangan aplikasi mobile dengan cross-platform framework, lokasi sumber daya kebutuhan dalam pengembangan aplikasi flutter serta kelebihan dan kelemahan dari framework flutter.

## Apa selanjutnya?

Silakan cek beberapa sumber belajar lainnya…

* [FlutLab is a modern Flutter online IDE](https://flutlab.io/)
* [Flutter System Architecture](https://docs.google.com/presentation/d/1cw7A4HbvM_Abv320rVgPVGiUP2msVs7tfGbkgdrTy0I/edit#slide=id.p)
* <https://docs.flutter.dev/create>
* <https://docs.flutter.dev/testing/build-modes>
* <https://docs.flutter.dev/deployment/android>

## Referensi

* [Flutter for Beginners Second Edition](https://drive.google.com/file/d/1hJbctx-_O3B6kUeG70bJD5xfbnsWwBXY/view?usp=sharing)
* <https://www.youtube.com/flutterdev>
* Buku Ajar Pemrograman Mobile Dengan Flutter, Jurusan Teknologi Informasi, Politeknik Negeri Malang. 2021.