

Modul 15

Testing, Debugger and Deployment.

Performance profiling, Inspecting Widget Tree and Deployment

Module Overview

Profil kinerja aplikasi Flutter

Memeriksa hierarki widget Flutter

Mempersiapkan aplikasi untuk disebarakan

Module Objectives

Setelah mempelajari dan mempraktikkan modul ini, mahasiswa diharapkan dapat memahami bagaimana menghasilkan aplikasi mobile dari Flutter yang siap untuk disebarakan.

Profiling Flutter

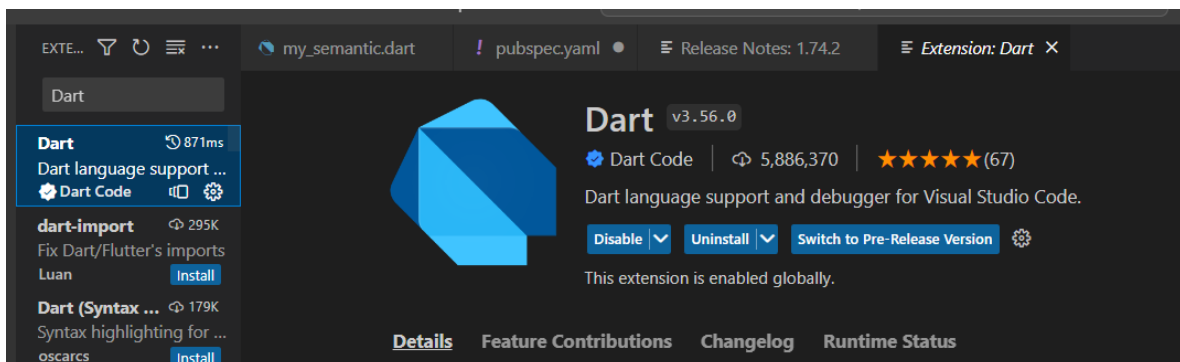
Dengan menggunakan Flutter, anda dapat menyediakan aplikasi berperforma tinggi dengan frekuensi gambar tinggi dan dapat berjalan dengan mulus. Sama seperti debugging yang dapat membantu menemukan bug, *profiling* adalah alat yang dapat membantu pengembang menemukan *bottlenecks* dalam aplikasi, mencegah *memory leaks*, atau meningkatkan kinerja aplikasi.

Alat yang dapat kita gunakan adalah *DevTools* yang merupakan salah satu alat yang dapat kita gunakan untuk memeriksa kinerja aplikasi Flutter.

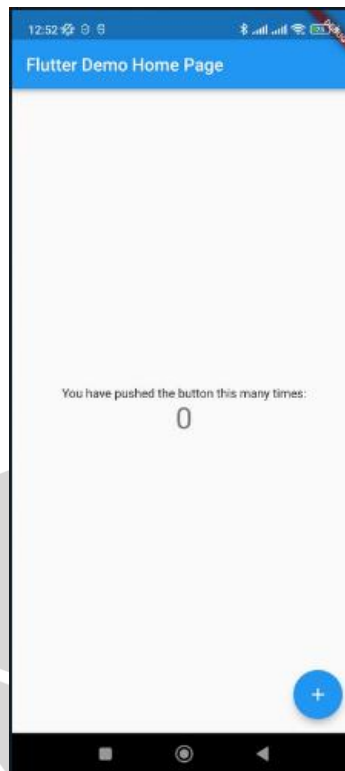
Berikut adalah beberapa hal yang dapat Anda lakukan dengan *DevTools*:

1. Memeriksa tata letak UI dan status aplikasi Flutter.
2. Mendiagnosis masalah kinerja UI di aplikasi Flutter.
3. *Profiling* CPU untuk aplikasi Flutter atau Dart.
4. Pembuatan *profiling* jaringan untuk aplikasi Flutter.
5. Debug tingkat sumber dari aplikasi Flutter atau Dart.
6. Debug masalah memori di aplikasi baris perintah Flutter atau Dart.
7. Lihat informasi log dan diagnostik umum tentang aplikasi baris perintah Flutter atau Dart yang sedang berjalan.
8. Analisis kode dan ukuran aplikasi.

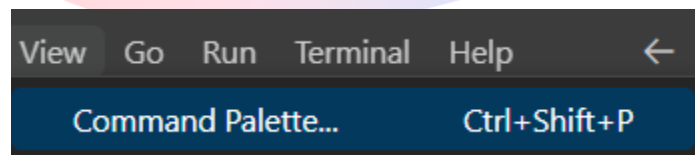
Untuk menggunakan *DevTools* pada VS Code, pastikan anda telah memasang *extension dart* sebagai tools dalam proyek VS Code anda.



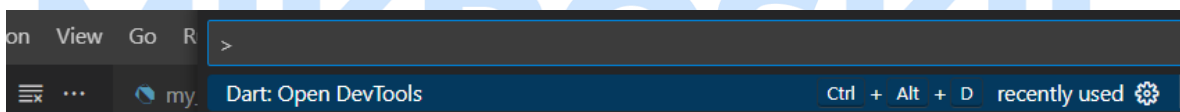
Buatlah sebuah **proyek baru**, dan jalankan aplikasi anda dalam **Debug (F5)**.



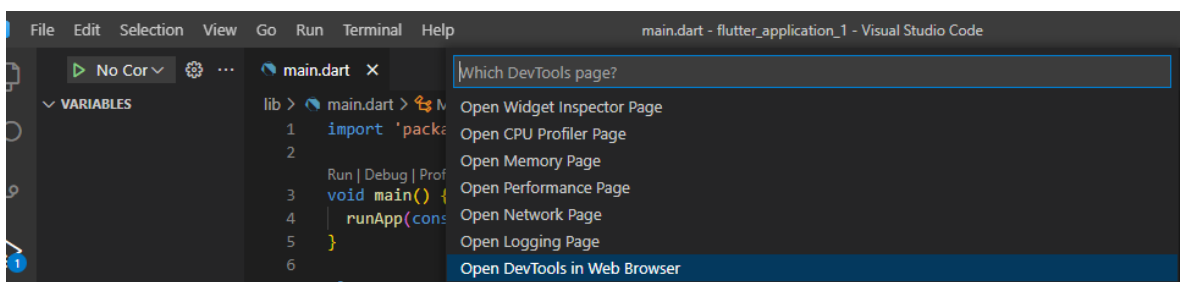
Anda dapat mencoba menjalankan DevTools dengan menjalankan perintah pada **view >> command palette**



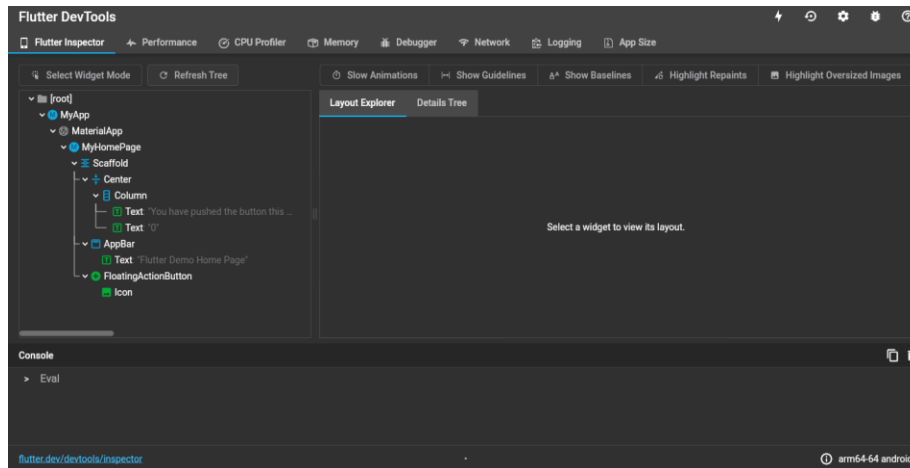
dan jalankan DevTools pada perintah **Dart: Open DevTools**.



Anda akan diberikan beberapa pilihan untuk melakukan profiling, pilih lah, **Open DevTool in Web Browser**.




Jika berhasil, maka akan ada tampilan sebagai berikut pada browser anda:

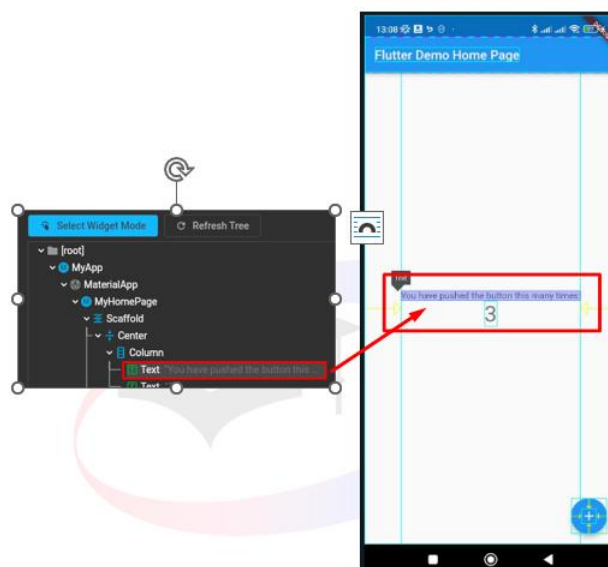



Pada bagian ini anda akan melihat **Flutter Inspector** yang merupakan alat untuk memvisualisasikan dan menjelajahi pohon widget dari aplikasi yang dibangun melalui Flutter. Pada bagian ini anda dapat gunakan untuk hal-hal berikut:


- memahami tata letak yang ada
- mendiagnosis masalah tata letak

Adapun beberapa fitur yang dapat anda gunakan pada Flutter Inspector adalah:

1.  **Select Widget Mode** : untuk memberikan **highlight** pada widget perangkat fisik yang sedang diperiksa.




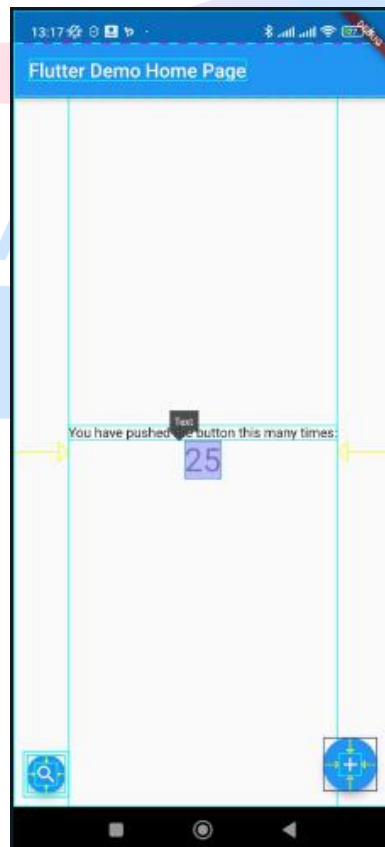
2.  : Muat ulang info widget terbaru. Perubahan pada widget tidak akan secara otomatis ditampilkan pada Inspector, sehingga anda harus melakukan Refresh Tree, untuk mendapatkan info terbaru.

3.  : memperlambat animasi 5x dari waktu normal. Anda dapat mencoba memasukkan gambar gif berikut:

```
Image.network('https://docs.flutter.dev/assets/images/dash/dash_fainting.gif')
```

perhatikan ketika anda mengaktifkan fitur ini, animasi gif akan berjalan lebih lambat.

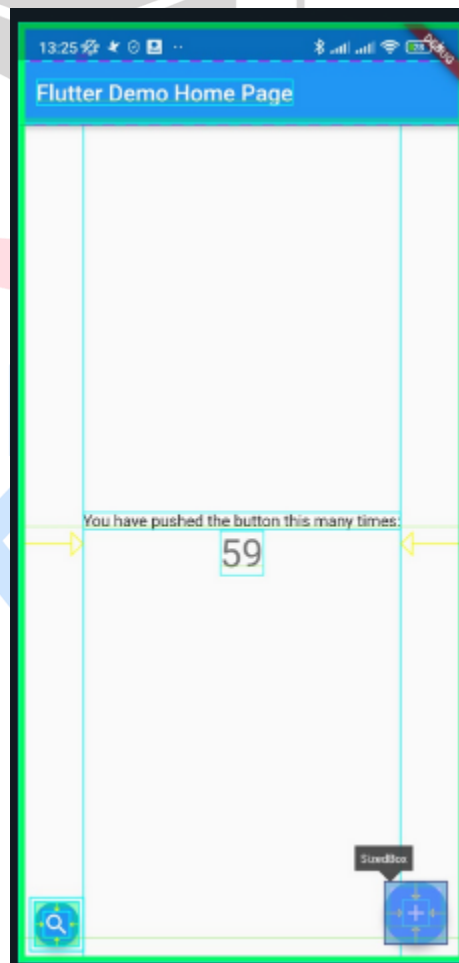
4.  : Fitur ini memberikan panduan pada aplikasi Anda yang menampilkan kotak render, perataan, padding, tampilan gulir, klipning, dan spacer.



5. **Show Baselines** : Opsi ini membuat semua garis dasar/garis horizontal terlihat yang digunakan untuk memposisikan teks.



6. **Highlight Repaints** : Opsi ini memberikan warna batas di sekitar semua kotak render yang akan berubah warna setiap kali kotak mengalami perubahan (update widget).



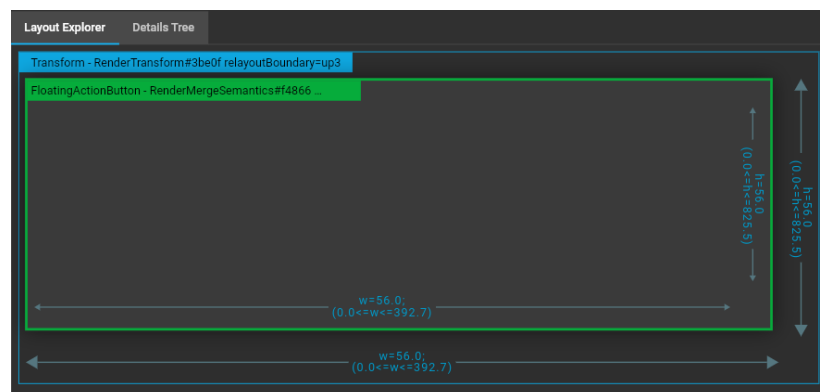
7. **Highlight Oversized Images** : menghighlight gambar yang terlalu besar dengan membalikkan warna gambar dan membalikkan gambar secara vertical. Anda dapat mencoba menambahkan gambar berikut:

```
Image.network(
  'https://img.freepik.com/free-vector/big-sale-banner-design-vector-illustration_157027-488.jpg?w=2000'),
```

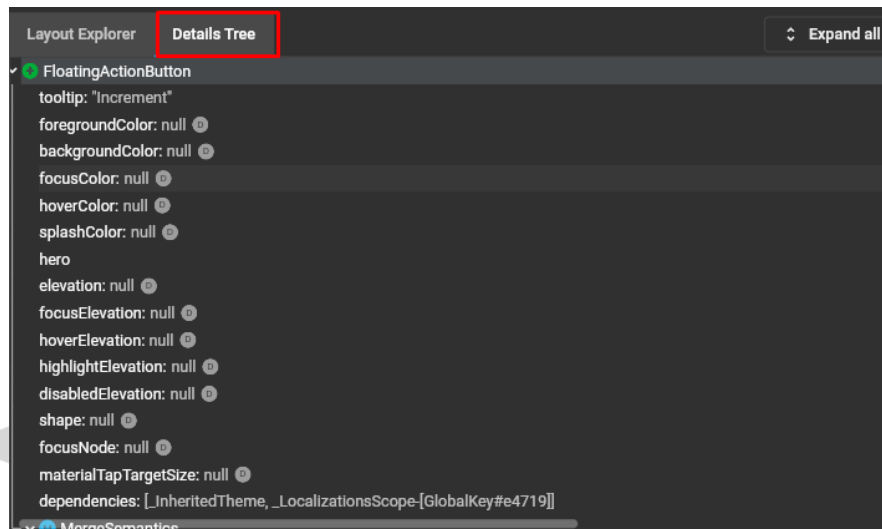
Maka, gambar akan ditampilkan sebagai berikut:



8. **Layout Explorer** : Digunakan untuk menampilkan blue print dari widget



9. Details Tree : Digunakan untuk menampilkan isi detail setiap properti yang di miliki widget



Adapun beberapa fungsi DevTools lainnya yang dapat anda gunakan adalah:

1. **Performance view** : Tampilan kinerja menawarkan informasi waktu dan kinerja untuk aktivitas di aplikasi Anda. Ini terdiri dari tiga bagian, masing-masing meningkat dalam perincian.
2. **CPU Profiler View** : memungkinkan Anda merekam dan membuat profil sesi dari aplikasi Dart atau Flutter Anda.
3. **Memory View** : memberikan wawasan tentang detail alokasi memori aplikasi dan alat untuk mendeteksi dan men-debug masalah tertentu.
4. **Network View** : memungkinkan Anda memeriksa lalu lintas HTTP, HTTPS, dan soket web dari aplikasi Dart atau Flutter Anda.
5. **Debugger**: Memungkinkan Anda untuk melakukan debug aplikasi. Adapun fitur ini tidak jauh berbeda dengan fungsi debugger yang ada dalam visual studio code.
6. **Logging View** : menampilkan peristiwa dari waktu proses Dart, kerangka kerja aplikasi (seperti Flutter), dan peristiwa logging tingkat aplikasi.

7. **App Size Tool** : memungkinkan Anda untuk menganalisis ukuran total aplikasi Anda.

Deployment

Merilis aplikasi di Google Play Store memerlukan akun penerbit yang valid. Anda harus memahami dokumentasi untuk platform PlayStore sehingga mengetahui cara memublikasikan aplikasi setelah membuat versi rilis aplikasi Anda. Google menggunakan cara pembayaran **one-time registration fee** untuk mendaftar sebagai penerbit, dan anda dapat mempublish setiap aplikasi yang anda inginkan kedalam akun penerbit.

Aplikasi yang dipublish, haruslah aplikasi yang berada dalam **release mode**, bukan **debug mode**. Dalam mode rilis, informasi debug dihapus dari aplikasi dan hanya dapat dijalankan pada perangkat fisik. Jalankan perintah **flutter run -release** sebagai flag untuk menghasilkan aplikasi dalam **release mode**.

Dalam Android, **.apk** merupakan format yang digunakan dalam **Play Store**. Jalankan perintah **flutter build apk** atau **flutter build**, untuk menghasilkan file yang siap untuk dipublikasikan. Akan tetapi sebelum anda mempublikasikan aplikasi ke dalam **app store**, pastikan:

1. Melengkapi meta informasi pada AndroidManifest.xml dan build.gradle. Adapun yang perlu dilengkapi adalah:
 - a. Permission request

Pastikan anda melengkapi permisson pada AndroidManifest.xml anda sesuai dengan kebutuhan aplikasi, permintaan ijin yang berlebihan dapat menyebabkan aplikasi anda ditolak oleh publiser

Contoh :

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.WRITE_CONTACTS"/>
<uses-permission android:name="android.permission.CAMERA" />
```

```
<uses-feature  
  android:name="android.hardware.camera"  
  android:required="false" />
```

b. Meta data

Meta tag merupakan informasi tambahan untuk menandakan service yang anda gunakan di dalam aplikasi, seperti AdMob, atau GoogleMap

```
<meta-data  
  android:name="com.google.android.gms.ads.APPLICATION_ID"  
  android:value="ADMOB-KEY"/>
```

c. Nama Aplikasi dan Icon

Hingga saat ini, anda meluncurkan aplikasi hanya menggunakan Logo flutter. Untuk rilis, anda perlu menukarnya dengan ikon unik yang dapat membedakan aplikasi anda dengan jutaan aplikasi lainnya dan menentukan nama aplikasi, Pemberian nama dan Icon aplikasi ditentukan dalam tag aplikasi manifes.

```
<application  
  android:name="io.flutter.app.FlutterApplication"  
  android:label="Hands On: Favors app"  
  android:icon="@mipmap/ic_launcher">
```

d. Application Id dan Versi

ID aplikasi inilah yang membuat aplikasi unik di Play Store dan sistem Android. Praktik yang baik adalah menggunakan domain organisasi, nama organisasi dan diikuti dengan nama aplikasi. Pastikan anda mengingat nilai ini, karena tidak dapat diubah setelah Anda mengunggah aplikasi ke play store.

Anda dapat menemukan pengaturan kode ini di file android/app/build.gradle, di dalam bagian defaultConfig:

```
defaultConfig {  
  applicationId "com.example.handson"  
  minSdkVersion 16  
  targetSdkVersion 28  
  multiDexEnabled true  
  versionCode flutterVersionCode.toInteger()  
  versionName flutterVersionName  
  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  
}
```

e. Signing app

Langkah penandatanganan adalah langkah terakhir namun paling penting sebelum merilis aplikasi ke publik, bahkan jika Anda tidak ingin memublikasikan di Google Play Store. Penandatanganan merupakan penanda kepemilikan aplikasi — singkatnya, siapa pun yang memiliki tanda tangan adalah pemilik aplikasi tersebut. Anda memerlukan ini agar Anda dapat memublikasikan aplikasi dan melakukan pembaruan ke dalam aplikasi yang telah anda publikasi.

Adapun langkah-langkahnya adalah:

1. Bentuk keystore anda dengan menjalankan perintah java berikut ini:
`keytool -genkey -v -keystore D:/Nim Anda/my-key.keystore -keyalg RSA -keysize 2048 -validity 10000 -alias key`
Pastikan alamat penyimpanan yang anda berikan benar
2. Masukkan password, unit organisasi, nama organisasi, lokasi organisasi, code lokasi, dan ketikkan yes. Maka anda akan dibuatkan sebuah keystore, sesuai dengan lokasi yang anda buat. **PASTIKAN ANDA TIDAK MELUPAKAN LETAK KEYSTORE DAN PASSWORD KEYSTORE ANDA.** Keystore ini adalah tanda tangan digital yang diperlukan untuk mengakses kembali aplikasi yang akan dan telah anda publish.

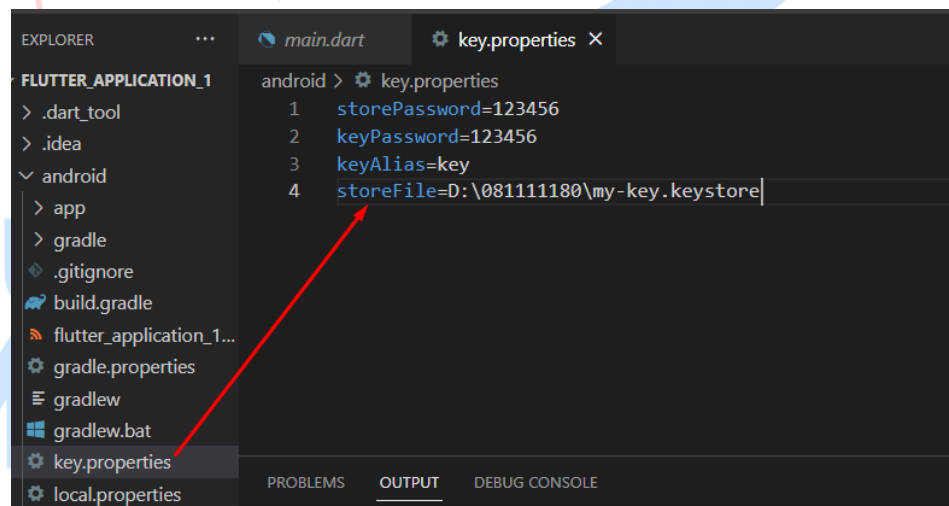
```
Enter keystore password:  
Re-enter new password:  
What is your first and last name?  
[Unknown]: Sunaryo  
What is the name of your organizational unit?  
[Unknown]: Mikroskil  
What is the name of your organization?  
[Unknown]: Universitas  
What is the name of your City or Locality?  
[Unknown]: Medan  
What is the name of your State or Province?  
[Unknown]: Sumatra Utara  
What is the two-letter country code for this unit?  
[Unknown]: ID  
Is CN=Sunaryo, OU=Mikroskil, O=Universitas, L=Medan, ST=Sumatra Utara, C=ID correct?  
[no]: yes  
  
Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 10,000 days  
for: CN=Sunaryo, OU=Mikroskil, O=Universitas, L=Medan, ST=Sumatra Utara, C=ID  
[Storing D:/08111180/my-key.keystore]  
  
C:\Program Files\Android\Android Studio\jre\bin>
```

C > Data (D:) > 081111180

Search 081111180

Name	Date modified	Type
M01	9/19/2022 03:25 PM	File folder
M02	9/23/2022 02:06 PM	File folder
M03	10/1/2022 10:33 AM	File folder
M04	9/27/2022 02:00 PM	File folder
M05	10/24/2022 10:22 AM	File folder
M06	9/29/2022 04:12 PM	File folder
M07	10/31/2022 12:54 PM	File folder
M09	10/6/2022 05:30 PM	File folder
M11	10/20/2022 01:22 PM	File folder
M13	10/27/2022 05:10 PM	File folder
M14	1/4/2023 02:10 PM	File folder
M15	11/15/2022 03:16 PM	File folder
my-key.keystore	1/4/2023 03:38 PM	KEYSTORE File

3. Buat file baru pada folder android anda, dengan nama key.properties dan isikan dengan:



4. Tambahkan code berikut sebelum block android pada app/build.gradle:

```

28  def keystoreProperties = new Properties()
29  def keystorePropertiesFile = rootProject.file('key.properties')
30  if (keystorePropertiesFile.exists()) {
31      keystoreProperties.load(new FileInputStream(keystorePropertiesFile))
32  }
33  android {
34      compileSdkVersion flutter.compileSdkVersion
35      ndkVersion flutter.ndkVersion

```

5. Ubah **buildTypes** pada build.gradle file dari:

```
61 buildTypes {  
62     release {  
63         // TODO: Add your own signing config for the release build.  
64         // Signing with the debug keys for now, so `flutter run --release` works.  
65         signingConfig signingConfigs.debug  
66     }  
67 }
```

Menjadi

```
61 buildTypes {  
62     release {  
63         // TODO: Add your own signing config for the release build.  
64         // Signing with the debug keys for now, so `flutter run --release` works.  
65         signingConfig signingConfigs.release  
66     }  
67 }
```

6. Tambahkan code berikut sebelum buildTypes anda:

```
signingConfigs {  
    release {  
        keyAlias keystoreProperties['keyAlias']  
        keyPassword keystoreProperties['keyPassword']  
        storeFile keystoreProperties['storeFile'] ? file(keystoreProperties['storeFile']) : null  
        storePassword keystoreProperties['storePassword']  
    }  
}  
buildTypes {  
    release {  
        signingConfig signingConfigs.release  
    }  
}
```

Sekarang, ketika kita menggunakan perintah **flutter build apk** atau **flutter run --release**, file aplikasi akan ditandatangani dengan kunci yang telah anda buat. File apk yang dihasilkan flutter build apk inilah yang dapat anda gunakan untuk diterbitkan di Play Store, yang biasanya tersedia di: build/app/outputs/apk/app.apk.

Latihan

1. Berikan hasil Inspecting Widget Tree dari pertemuan 1 sampai dengan pertemuan 13 yang telah kita buat sebelumnya.