Distribusi Aplikasi

Mendistribusikan aplikasi - flutter

Ahmad Adivar, S.Kom. M.Kom
ahmad.adivar@unsulbar.ac.id

Distribusi Aplikasi ?

Distribusi aplikasi adalah **tahapan akhir** dalam siklus pengembangan aplikasi ketika developer:

- Membuat build aplikasi versi release
- Menandatangani aplikasi (signing) menggunakan keystore (Android) atau certificate (iOS)
- Mengoptimasi ukuran dan performa aplikasi
- Mengunggah build ke Google Play/App Store
- Membagikan aplikasi ke user (publik atau internal)

"Distribusi aplikasi = mempersiapkan aplikasi agar dapat digunakan oleh pengguna akhir."

Tujuan Distribusi Aplikasi

1. Aplikasi dapat diinstal dan digunakan oleh user
2. Menjamin aplikasi aman dan terverifikasi
3. Melakukan deployment ke Play Store atau App Store
4. Membuat versi final dari aplikasi yang stabil (release build)

Jenis Distribusi Aplikasi – Flutter (1)

1. Distribusi Android
   - Membangun file:
     - APK → cocok untuk distribusi manual
     - AAB → format wajib untuk Play Store
   - Menandatangani aplikasi dengan keystore
   - Mengunggah ke Google Play Console
2. Distribusi iOS
   - Membangun **IPA file** → proses pengemasan aplikasi dengan ekstensi .ipa (*iOS App Store Package*)
   - Mengatur **Provisioning Profile** & **Certificates**
   - Upload via **Xcode** → **App Store Connect**

Jenis Distribusi Aplikasi – Flutter (2)

3. Web
   - Flutter bisa dibuild menjadi website: “flutter build web”
   - Deploy ke hosting seperti:
     - Firebase Hosting
     - Vercel
     - Netlify
4. Desktop (Windows / macOS / Linux), dengan format:
   - .exe (Windows)
   - .app (macOS)
   - .deb atau .rpm (Linux)

Praktik Terbaik Sebelum Distribusi

- Gunakan ikon aplikasi yang konsisten
- Update app_name, package name, bundle ID
- Testing pada berbagai device
- Gunakan release mode untuk uji performa
- Pastikan tidak ada API key yang terbuka

Manfaat Proses Distribusi Aplikasi

- Aplikasi menjadi layak digunakan public
- Mendapatkan umpan balik pengguna
- Aplikasi dapat dipelihara melalui update versi
- Pengembang dapat mengontrol versi dan rilis

Release Build & Debug

Release build adalah versi aplikasi yang:

- Sudah dioptimasi performanya
- Tidak menampilkan debug banner
- Tidak menyertakan debugging tools
- Ukuran lebih kecil dan lebih cepat
- Siap dipublikasikan

Versi debug adalah mode pembangunan aplikasi yang digunakan khusus untuk proses pengembangan dan pengujian. Di mode debug, aplikasi berjalan dengan alat bantu khusus yang memungkinkan pengembang melakukan debugging secara mudah dan cepat.

Perbedaan Debug vs Release Mode

Aspect	Debug	Release
Hot Reload	✓	
Performance	Rendah	Optimal
Logging	Banyak log	Minim log
Digunakan untuk	Pengembangan	Publik / Deployment

Langkah-langkah Menyiapkan Release Build Android (1)

1. Pastikan Aplikasi dalam Mode Release

   Secara default, Flutter melakukan build dalam mode debug. Untuk membuat release build, gunakan:

   APK Release: flutter build apk –release

   AAB Release (untuk Play Store): flutter build appbundle –release

   Output file akan ditemukan di:

   - build/app/outputs/flutter-apk/app-release.apk
   - build/app/outputs/bundle/release/app-release.aab

Langkah-langkah Menyiapkan Release Build Android (2)

2. Menambahkan App Icon dan App Name

- Tahapan penambahan icon:
  - Siapkan gambar icon aplikasi dalam format .png, ukuran minimal 512x512 pixel agar hasilnya tajam di berbagai resolusi.
  - Tambahkan package **flutter_launcher_icons** pada bagian dev_dependencies di file pubspec.yaml:

```
dev_dependencies:
  flutter_launcher_icons:
```

  - Buat folder untuk icon, misal: assets/icon/, lalu simpan gambar icon di folder tersebut
  - Tambahkan konfigurasi berikut pada bagian bawah file pubspec.yaml:

```
flutter_icons:
  android: "launcher_icon”
  ios: true
  image_path: "assets/icon/app_icon.png”
```

  - Jalankan perintah berikut di terminal:

```
flutter pub run flutter_launcher_icons:main
```

- Tahapan mengganti App Name:
  - Tambahkan pada “android/app/src/main/AndroidManifest.xml”:

```
android:label="Nama Aplikasi”
```

- Mengubah applicationId (package)
  - Sesuaikan identitas pada “android/app/build.gradle”

```
defaultConfig {
    applicationId "com.example.namaaplikasi"
}
```

Langkah-langkah Menyiapkan Release Build Android (3)

3. Menyiapkan Signing (Penandatanganan Aplikasi)

Android mewajibkan setiap aplikasi ditandatangani digital sebelum dipublikasikan.

- Buat Keystore

  keytool -genkey -v -keystore ~/my-key.jks -keyalg RSA -keysize 2048 -validity 10000 -alias my-key

  Maka akan muncul file : my-key.jks

- Simpan Keystore di folder Android

  Copy ke folder project: /project-name/android/app/my-key.jks

- Buat file konfigurasi keystore: “android/key.properties”

  storePassword=yourpassword

  keyPassword=yourpassword

  keyAlias=my-key

  storeFile=../app/my-key.jks

Langkah-langkah Menyiapkan Release Build Android (3.1)

- Daftarkan keystore di Gradle

Buka/edit file : android/app/build.gradle

```
def keystoreProperties = new Properties()
def keystorePropertiesFile = rootProject.file('key.properties')
if (keystorePropertiesFile.exists()) {
    keystoreProperties.load(new FileInputStream(keystorePropertiesFile))
}

android {
    ...
    signingConfigs {
        release {
            keyAlias keystoreProperties['keyAlias']
            keyPassword keystoreProperties['keyPassword']
            storeFile file(keystoreProperties['storeFile'])
            storePassword keystoreProperties['storePassword']
        }
    }

    buildTypes {
        release {
            signingConfig signingConfigs.release
            minifyEnabled false
        }
    }
}
```

Langkah-langkah Menyiapkan Release Build Android (4)

4. Mengaktifkan ProGuard (opsional untuk optimasi ukuran)

   Jika ingin memperkecil ukuran aplikasi:

```
buildTypes {
    release {
        signingConfig signingConfigs.release
        minifyEnabled true
        shrinkResources true
        proguardFiles getDefaultProguardFile('proguard-android-optimize.
        txt'), 'proguard-rules.pro'
    }
}
```

Langkah-langkah Menyiapkan Release Build Android (5)

5. Build Release

   Setelah semua konfigurasi signing dan setting selesai, jalankan:

   - Build APK: flutter build apk --release
   - Build AAB (Play Store): flutter build appbundle -release

   Output berada pada folder: build/app/outputs/

6. Test Release Build di Device

   Untuk install APK ke HP: flutter install

   atau manual klik APK pada device Android.

   Jika aplikasi berjalan normal tanpa debug banner → release build sukses.

Siap Upload ke Play Store Jika sudah build App Bundle:

app-release.aab → file inilah yang diupload ke Google Play Console.

Langkah-langkah Menyiapkan Release Build iOS

1. Sebelum membuat release build, pastikan:

   Akun Apple Developer

   - Gratis → hanya bisa testing di perangkat nyata
   - Berbayar/tahun → diperlukan untuk upload ke App Store

   Xcode Terinstal → Minimal versi sesuai SDK iOS terbaru.

   Flutter SDK Terupdate: flutter upgrade

2. Konfigurasi Project iOS

   Masuk ke folder iOS: cd ios

   Lalu buka proyek di Xcode: open Runner.xcworkspace

3. Set App Name, Bundle Identifier & Version

   Buka: Xcode → Runner → Targets → Runner → General

   - Untuk mengatur **Display Name** & **Bundle Identifier,** contoh : com.kampus.namaaplikasi
   - Masih di bagian tab General:
     - Version → versi publik (misal: 1.0.0)
     - Build → versi internal (misal: 1)

Langkah-langkah Menyiapkan Release Build iOS (2)

4. Mengatur Signing & Capabilities

   Pada bagian **Runner** → **Signing** & **Capabilities**

   - **Team** → pilih akun Apple Developer Anda
   - Provisioning Profile → pilih otomatis (Automatically manage signing)
   - Signing Certificate → Apple Development / Apple Distribution

   "*Jika ingin upload ke App Store, pilih mode **Release** + **Distribution certificate***"

5. Konfigurasi Mode Release di Flutter

   Pastikan build iOS menggunakan mode release: flutter build ios –release

   Perintah ini:

   - Mengoptimalkan kode
   - Membangun file .ipa sementara
   - Menyiapkan project untuk Xcode Archive

6. Build Release dari Xcode (Archive)

   Ini wajib untuk upload ke App Store. Langkah:

   - Buka Xcode
   - Pilih **Product** → **Scheme** → **Runner**
   - Pilih device: Any iOS Device (arm64)
   - Klik **Product** → **Archive**

   Tunggu proses hingga muncul jendela Organizer.

Langkah-langkah Menyiapkan Release Build iOS (3)

7. Export atau Upload Build

   Setelah selesai archive, Anda akan melihat list archive di Xcode Organizer. Terdapat 2 opsi :

   - Opsi A — Upload langsung ke App Store

     Pada bagian Organizer klik: **Distribute App** → **App Store Connect** → **Upload**, kemudian atur:

     - Include bitcode: optional
     - Strip Swift symbols: optional
     - Upload

     Setelah sukses, build akan muncul di App Store Connect (TestFlight atau App Store Review).

   - Opsi B — Export file IPA

     Jika ingin mengirim IPA secara manual:

     Distribute App → **Ad Hoc** / **Development** / **Enterprise** → **Export**

     Xcode akan menghasilkan file: Runner.ipa

8. Verifikasi Build di App Store Connect

   Masuk ke: https://appstoreconnect.apple.com/

   Pada menu : **My Apps** → **Pilih App** → **TestFlight / App Store Version**

   Biasanya butuh 10–30 menit sebelum build muncul.

Terima Kasih