

Flutter: Giao diện đáp ứng, package & plugin, tùy biến và phát hành ứng dụng Android

Bùi Võ Quốc Bảo

Khoa CNTT | Trường CNTT-TT | Đại học Cần Thơ

Giao diện đáp ứng

Tài liệu tham khảo

- <https://docs.flutter.dev/development/ui/layout/adaptive-responsive>
- https://github.com/sbis04/responsive_design

Giao diện đáp ứng vs thích ứng

- Giao diện thích ứng (adaptive UI): giao diện hỗ trợ cho các *loại thiết bị* khác nhau (di động/desktop, Android/iOS,...)

```
import 'dart:io' show Platform;

if (Platform.isAndroid) {
  // Android-specific code
} else if (Platform.isIOS) {
  // iOS-specific code
}
```

```
const CircularProgressIndicator.adaptive(),
Slider.adaptive(onChanged: (value) {}, value: 1),
SwitchListTile.adaptive(value: true, onChanged: (value) {}),
Switch.adaptive(value: true, onChanged: (value) {}),
Icon(Icons.adaptive.share),
```

- Giao diện đáp ứng (responsive UI): giao diện hỗ trợ nhiều *kích thước màn hình hiển thị* khác nhau

Xây dựng giao diện đáp ứng với Flutter

- “Khóa” ứng dụng vào một chế độ xoay màn hình

```
void main() {  
  WidgetsFlutterBinding.ensureInitialized();  
  
  SystemChrome.setPreferredOrientations([  
    DeviceOrientation.portraitUp,  
    DeviceOrientation.portraitDown,  
  ]).then((_) {  
    runApp(const MyApp());  
  });  
  
  // runApp(const MyApp());  
}
```

import 'package:flutter/services.dart';

Xây dựng giao diện đáp ứng với Flutter

- Sử dụng widget [LayoutBuilder](#)
 - Từ hàm **builder** có thể truy xuất đối tượng **BoxConstraints** (cho biết ràng buộc **maxWidth**, **maxHeight**)
 - Khi ràng buộc thay đổi (ví dụ như người dùng xoay màn hình), builder sẽ thực thi builder để tái tạo lại UI
- Sử dụng widget [OrientationBuilder](#)
 - Tương tự LayoutBuilder nhưng cho chế độ xoay của widget cha

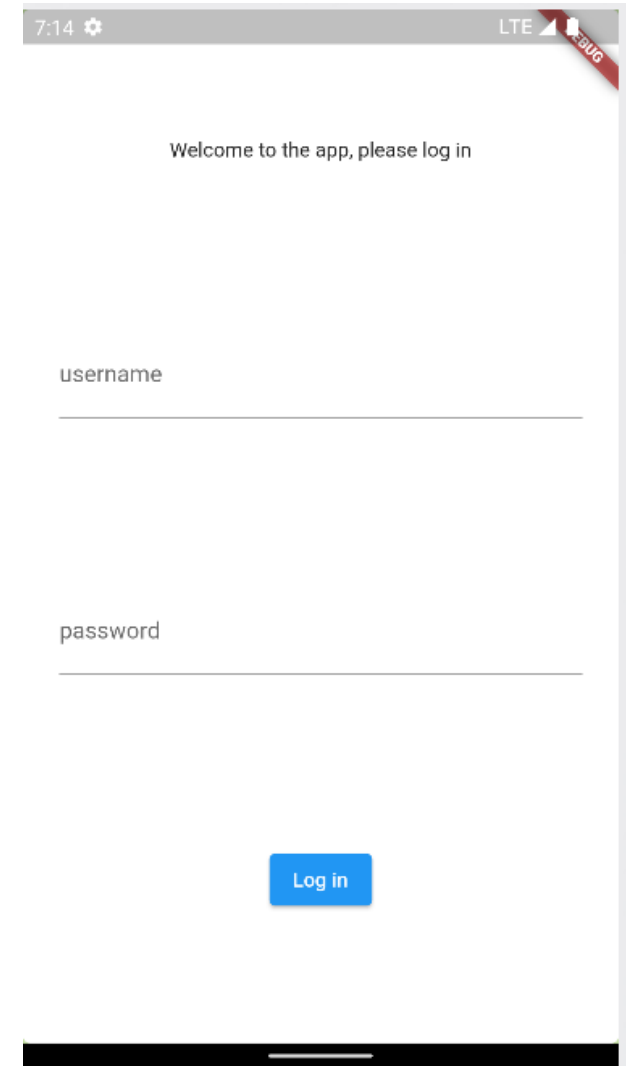
Xây dựng giao diện đáp ứng với Flutter

- Sử dụng [MediaQuery.of\(\)](#) để lấy thông tin về *kích thước, chế độ xoay (orientation)* của thiết bị trong phương thức build
 - Phương thức build của các widget có sử dụng `MediaQuery.of()` sẽ chạy nếu các thuộc tính trả về từ `MediaQuery.of()` có thay đổi
- Một số widget hỗ trợ khác: [AspectRatio](#), [FittedBox](#), [FractionallySizedBox](#), ...

Xây dựng giao diện đáp ứng với Flutter

- Ví dụ ứng dụng hỗ trợ ba trang: login, home, profile

```
class MyApp extends StatelessWidget {  
  const MyApp({Key? key}) : super(key: key);  
  
  @override  
  Widget build(BuildContext) {  
    return MaterialApp(  
      initialRoute: '/login',  
      routes: {  
        '/profile': (context) => const ProfilePage(),  
        '/login': (context) => const LoginPage(),  
        '/home': (context) => const HomePage()  
      },  
    ); // MaterialApp  
  }  
}
```



Xây dựng giao diện đáp ứng với Flutter

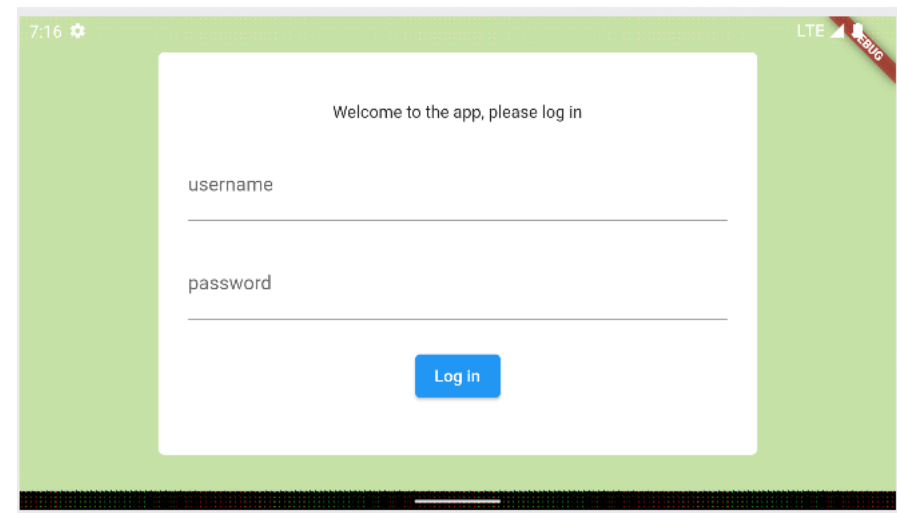
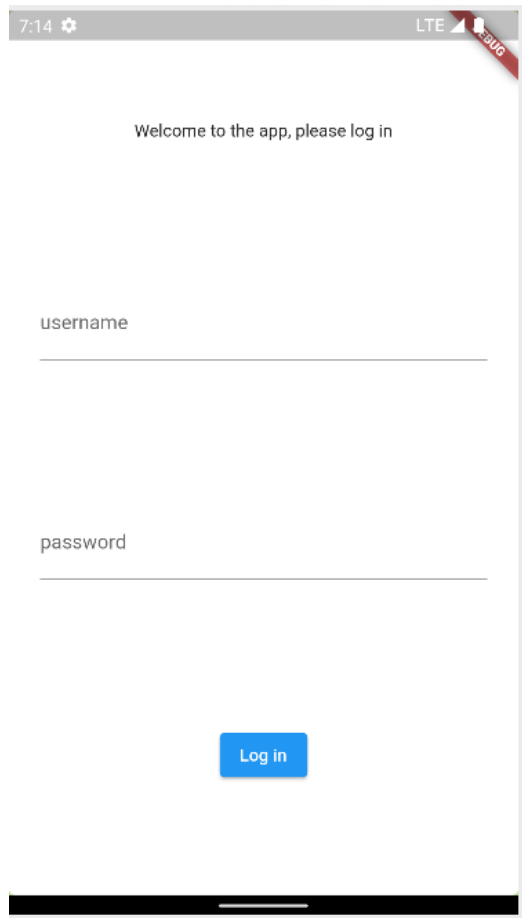
- Trang login sử dụng **LayoutBuilder** để tùy chỉnh padding

[AnimatedContainer](#) tạo hiệu ứng khi thuộc tính của nó thay đổi

```
@override
Widget build(context) {
  return Scaffold(
    body: LayoutBuilder(
      builder: (context, constraints) {
        return AnimatedContainer(
          duration: const Duration(milliseconds: 500),
          color: Colors.lightGreen[200],
          padding: constraints.maxWidth < 500
            ? EdgeInsets.zero
            : const EdgeInsets.all(30.0),
          child: Center(
            child: Container(
              padding: const EdgeInsets.symmetric(
                vertical: 30.0,
                horizontal: 25.0,
              ), // EdgeInsets.symmetric
              constraints: const BoxConstraints(
                maxWidth: 500,
              ), // BoxConstraints
            ),
          ),
        ),
  ),
)
```

Xây dựng giao diện đáp ứng với Flutter

- Trang login sử dụng **LayoutBuilder** để tùy chỉnh padding



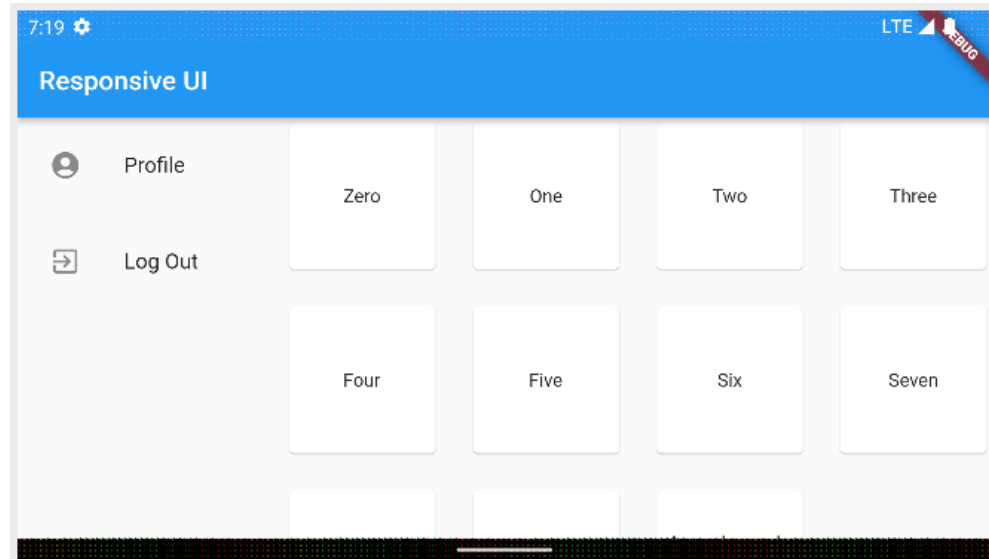
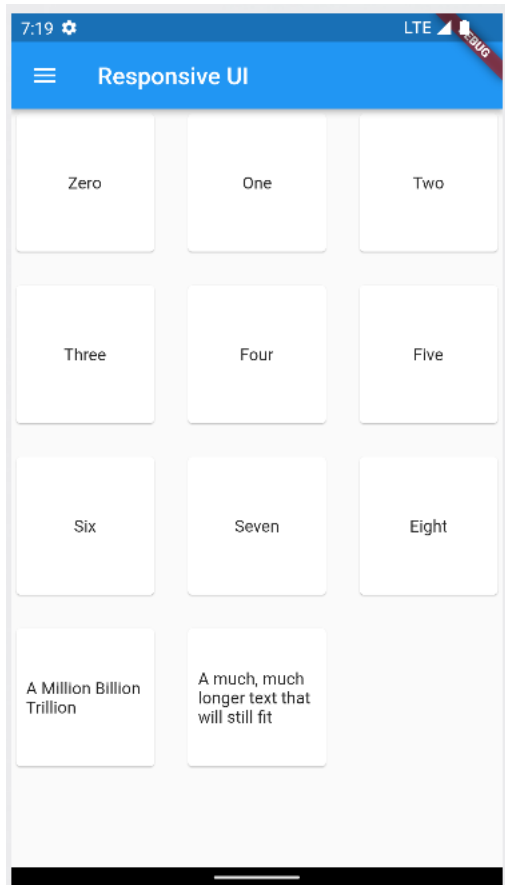
Xây dựng giao diện đáp ứng với Flutter

- Trang home sử dụng **MediaQuery.of()** để tùy chỉnh bố cục

```
@override
Widget build(context) {
  final deviceWidth = MediaQuery.of(context).size.width;
  final isSmallScreen = deviceWidth < 500;
  return Scaffold(
    appBar: AppBar(title: const Text('Responsive UI')),
    drawer: isSmallScreen ? const Drawer(child: Menu()) : null,
    body: SafeArea(
      child: Center(
        child: isSmallScreen
          ? Content()
          : Row(
              children: [
                const SizedBox(width: 200.0, child: Menu()),
                SizedBox(
                  width: deviceWidth - 200.0,
                  child: Content(),
                ) // SizedBox
              ],
            ), // Row
      ), // Center
    ), // SafeArea
  ); // Scaffold
}
```

Xây dựng giao diện đáp ứng với Flutter

- Trang home sử dụng `MediaQuery.of()` để tùy chỉnh bố cục



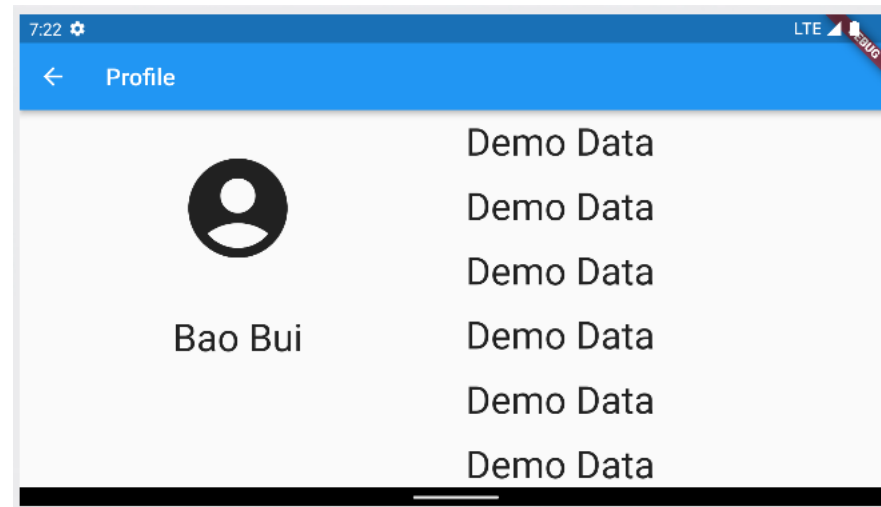
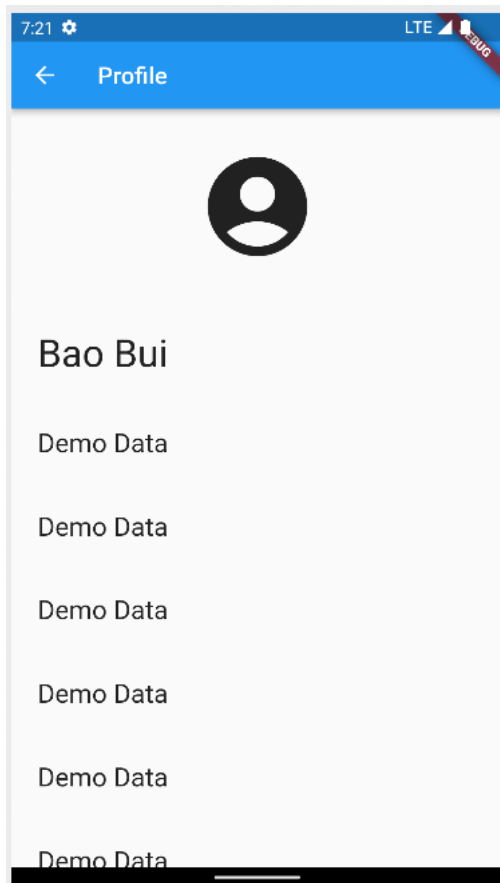
Xây dựng giao diện đáp ứng với Flutter

- Trang profile sử dụng **OrientationBuilder** để tùy chỉnh bố cục

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Profile'),
    ), // AppBar
    body: OrientationBuilder(
      builder: (context, orientation) {
        return orientation == Orientation.portrait
          ? _buildVerticalLayout()
          : _buildHorizontalLayout();
      },
    ), // OrientationBuilder
  ); // Scaffold
}
```

Xây dựng giao diện đáp ứng với Flutter

- Trang profile sử dụng **OrientationBuilder** để tùy chỉnh bố cục



Package & Plugin

Sử dụng package & plugin

- Hệ sinh thái Dart/Flutter sử dụng khái niệm package để quản lý các thư viện, công cụ được sử dụng
- Các gói thư viện sẵn dùng công khai được liệt kê trên trang web <https://pub.dev/>
- Sử dụng công cụ quản lý gói **pub** để quản lý các phụ thuộc của ứng dụng Dart/Flutter

<code>flutter pub get</code>	# Tải về các gói phụ thuộc
<code>flutter pub add <package></code>	# Thêm một gói phụ thuộc
<code>flutter pub remove <package></code>	# Xóa một gói phụ thuộc

Sử dụng package & plugin

- **pub** dựa vào một tập tin tên *pubspec.yaml* để liệt kê thông tin các gói phụ thuộc
- Danh sách các gói thư viện phát triển/bảo trì bởi đội ngũ phát triển chính của Flutter có thể được tìm thấy tại:
 - <https://github.com/flutter/packages>

```
name: myshop
description: A new Flutter project.

publish_to: 'none'

version: 1.0.0+1

environment:
  sdk: "≥2.17.3 <3.0.0"

dependencies:
  flutter:
    sdk: flutter
  intl: ^0.17.0
  provider: ^6.0.3
  http: ^0.13.4
  shared_preferences: ^2.0.15
  flutter_dotenv: ^5.0.2

dev_dependencies:
  flutter_test:
    sdk: flutter
  flutter_lints: ^2.0.0

flutter:
  uses-material-design: true
```

Sử dụng package & plugin

- Các gói thư viện được phát triển có sử dụng đến các API / tính năng của ít nhất một nền tảng cục thể (Android, iOS, Web, ...) gọi là các **plugin**
 - Ví dụ như một plugin có thể cung cấp cho ứng dụng Flutter khả năng sử dụng camera của thiết bị
- Danh sách các plugin phát triển/bảo trì bởi đội ngũ phát triển chính của Flutter có thể được tìm thấy tại:
 - <https://github.com/flutter/plugins>

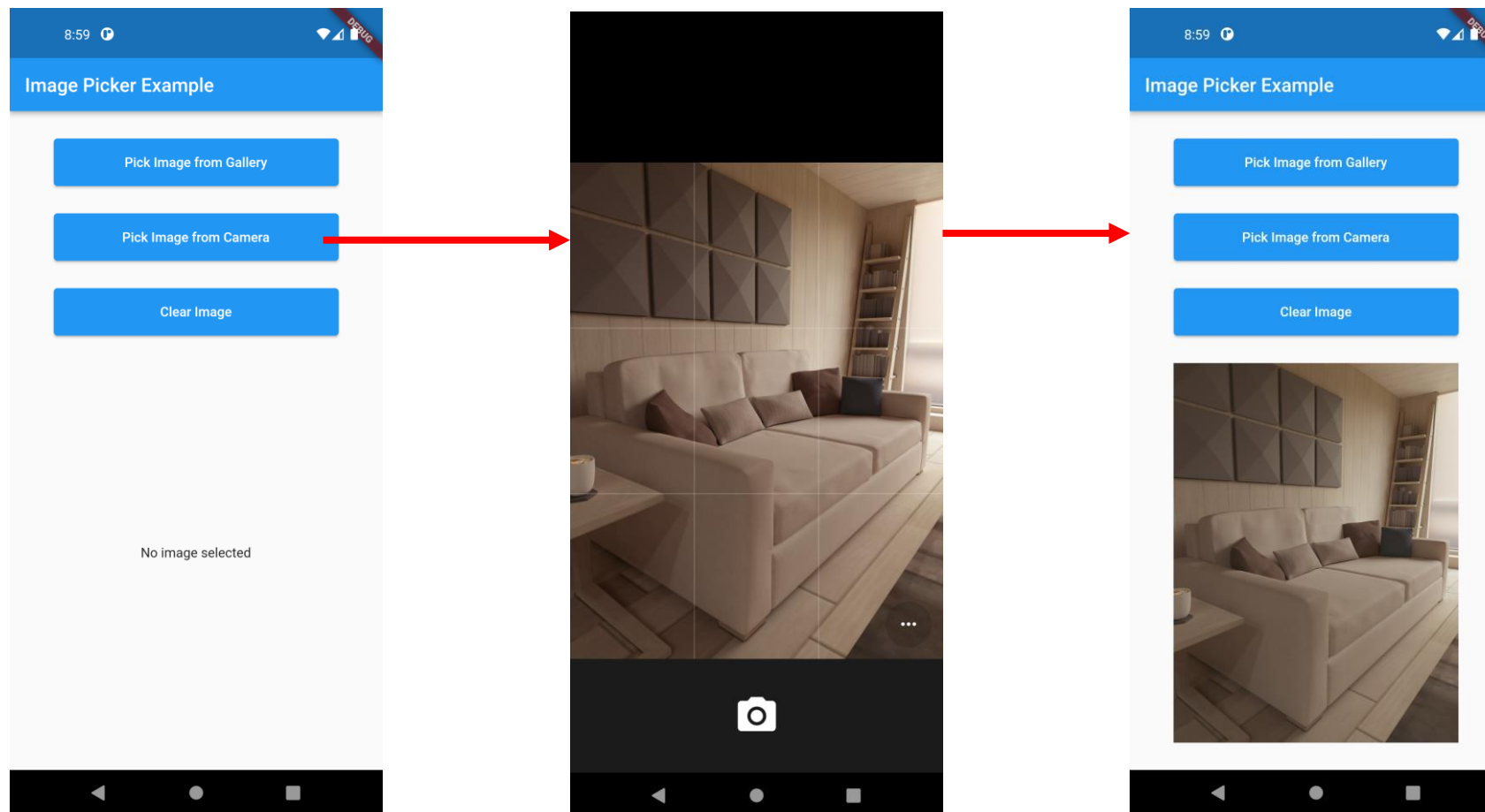
Sử dụng package & plugin

Ví dụ về package: [flutter_animate](#)



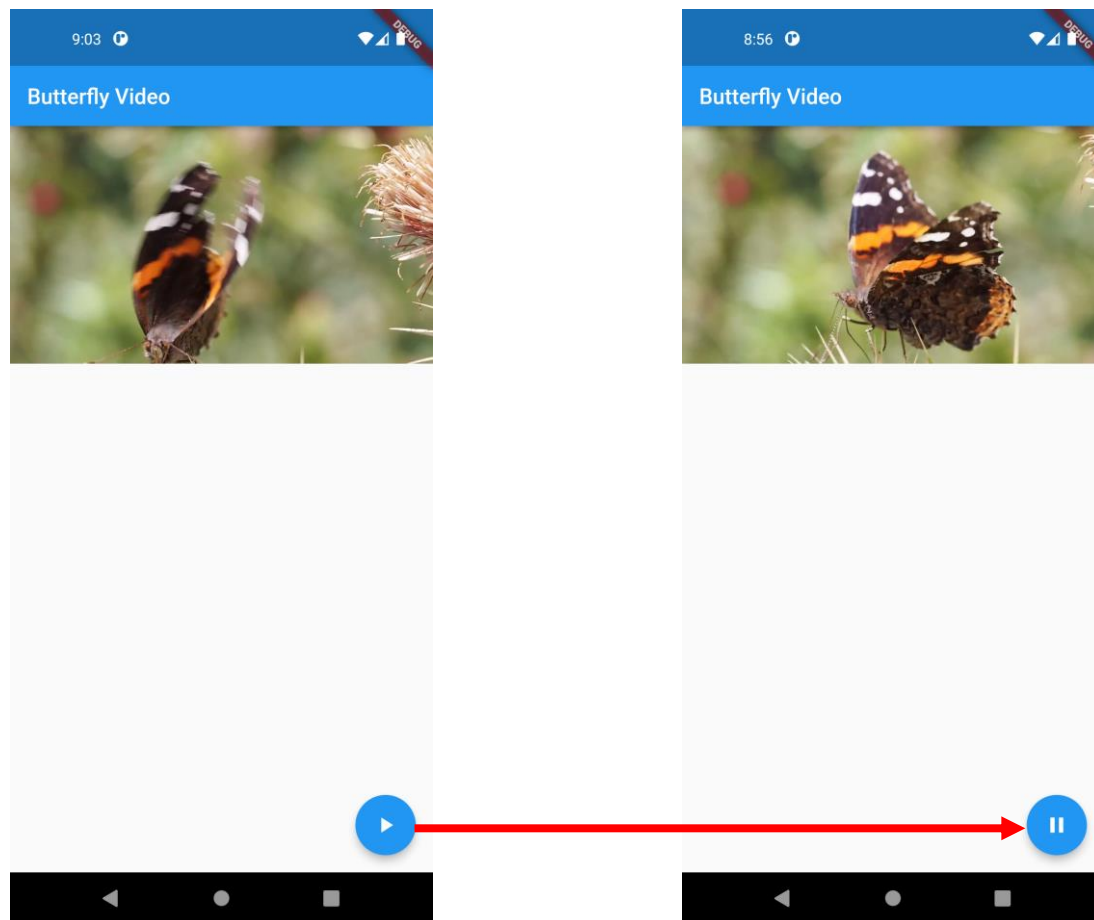
Sử dụng package & plugin

Ví dụ về plugin: [image_picker](#)



Sử dụng package & plugin

Ví dụ về plugin: [video_player](#)



Sử dụng package & plugin

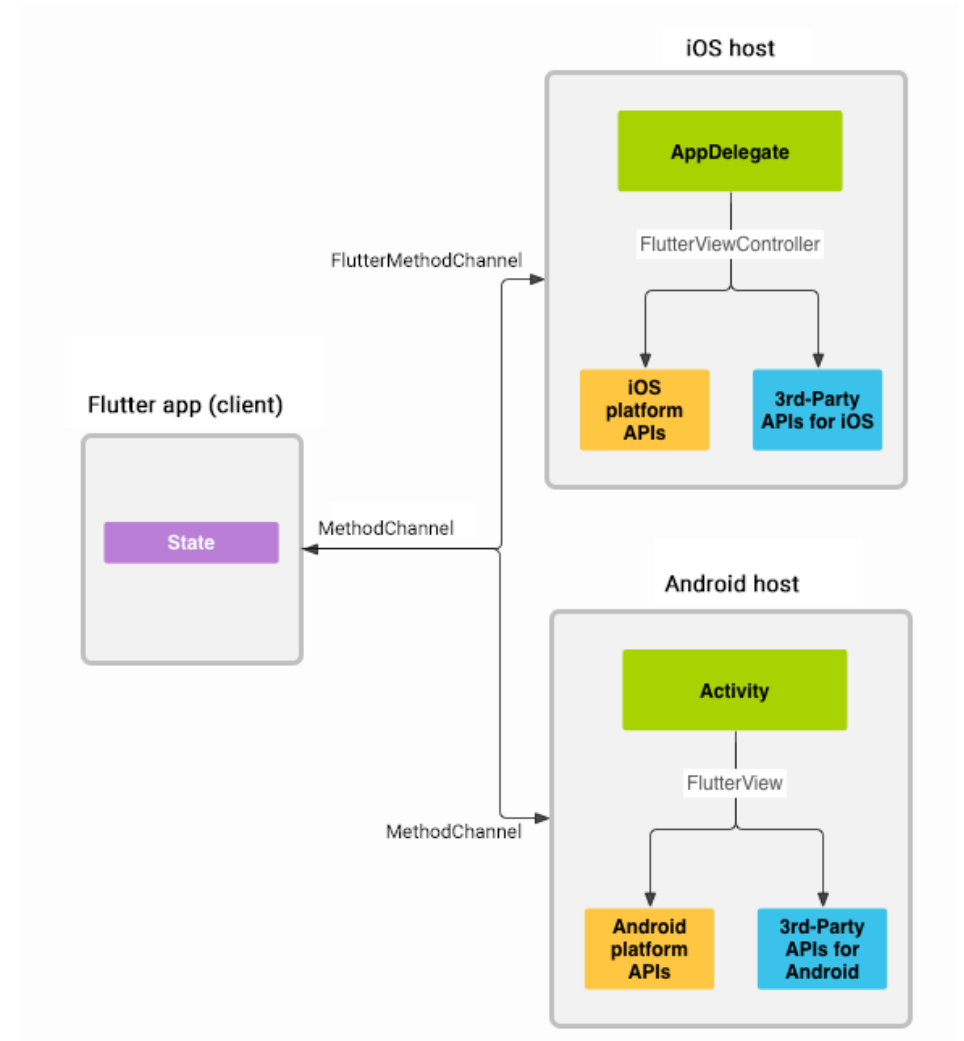
- Đơn giản hóa cài đặt so sánh bằng:
<https://pub.dev/packages/equatable>
- Thư viện sinh RFC4122 UUID
<https://pub.dev/packages/uuid>
- Sử dụng hoặc tự xây dựng các theme Material cho Flutter:
https://pub.dev/packages/flex_color_scheme
- Cache lại ảnh từ mạng:
https://pub.dev/packages/cached_network_image

Sử dụng package & plugin

- Tạo màn hình splash cho ứng dụng:
https://pub.dev/packages/flutter_native_splash
- Tạo biểu tượng launch ứng dụng:
https://pub.dev/packages/flutter_launcher_icons
- Một số hoạt ảnh/hiệu ứng xây dựng sẵn cho Flutter:
<https://pub.dev/packages/animations>
- Game engine đơn giản cho Flutter:
<https://pub.dev/packages/flame>
- ...

Tự phát triển các plugin

- Flutter trên hết là một UI framework đa nền tảng, các plugin phụ thuộc rất lớn vào cộng đồng phát triển
- Trong một số trường hợp, không có sẵn plugin hoặc các plugin đã có không đáp ứng được nhu cầu ứng dụng, nhà phát triển có thể cần phải tự [phát triển các plugin](#)



Tùy biến và phát hành ứng dụng Android

Tài liệu tham khảo

- Chương 16-17, **Flutter Apprentice** by Vincenzo Guzzi, Kevin D Moore, Vincent Ngo and Michael Katz
- <https://docs.flutter.dev/deployment/android>

Đổi biểu tượng ứng dụng

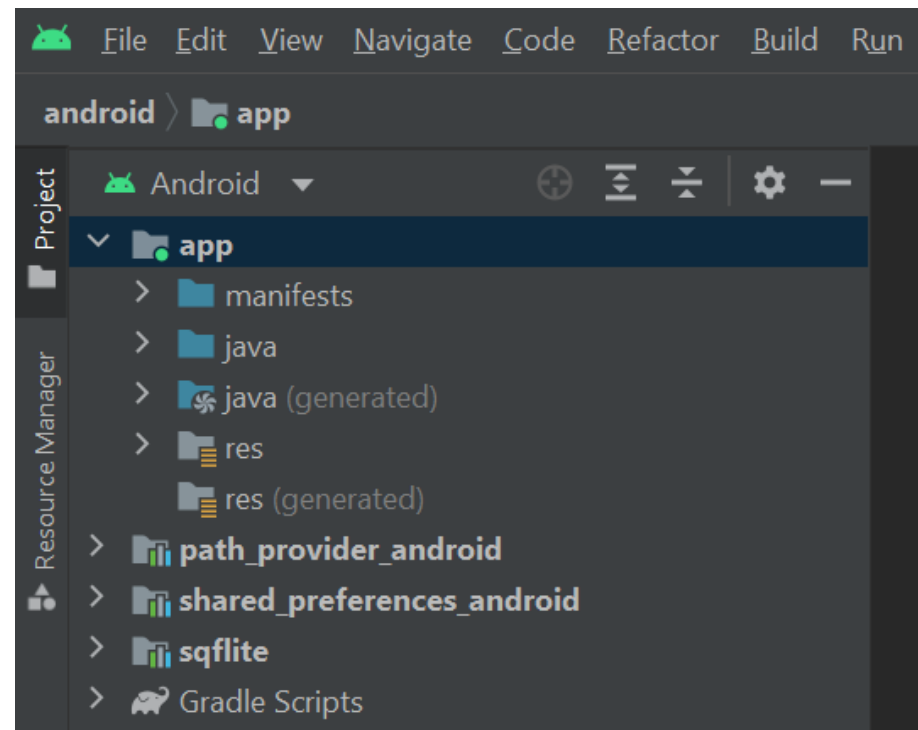
- Tập tin **android/app/src/main/AndroidManifest.xml** định nghĩa các thuộc tính của ứng dụng liên quan đến quá trình khởi động, quyền hạn, Play Store và hệ thống Android
- Biểu tượng của ứng dụng được định nghĩa tại

```
android:icon="@mipmap/ic_launcher"
```

- **@mipmap/ic_launcher** tương ứng với tập tin biểu tượng tên **ic_launcher** nằm trong một thư mục **res/mipmap-{resolution}** (tùy vào màn hình thiết bị)

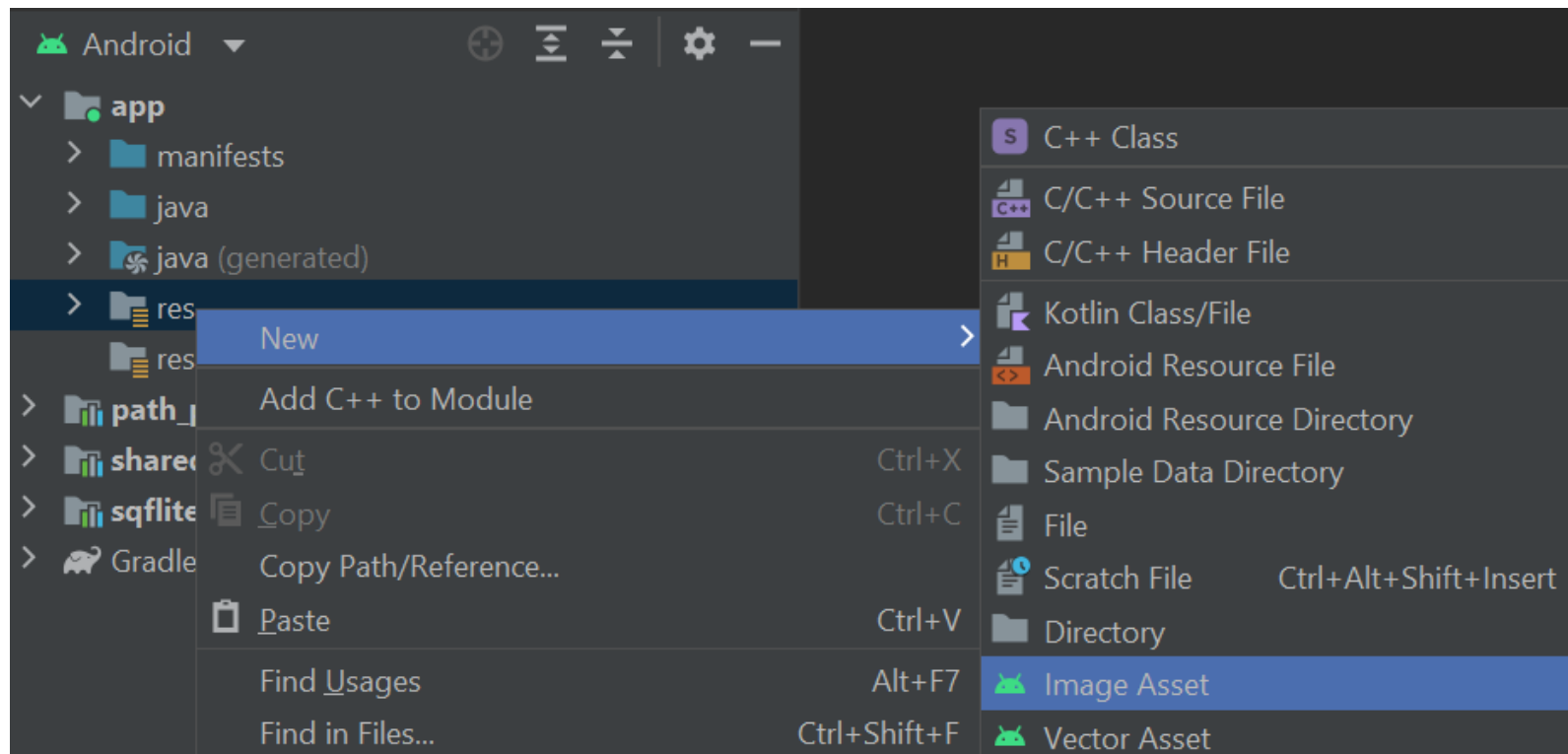
Đổi biểu tượng ứng dụng

- Tạo biểu tượng ứng dụng từ tập tin ảnh
 - Mở thư mục android dùng Android Studio



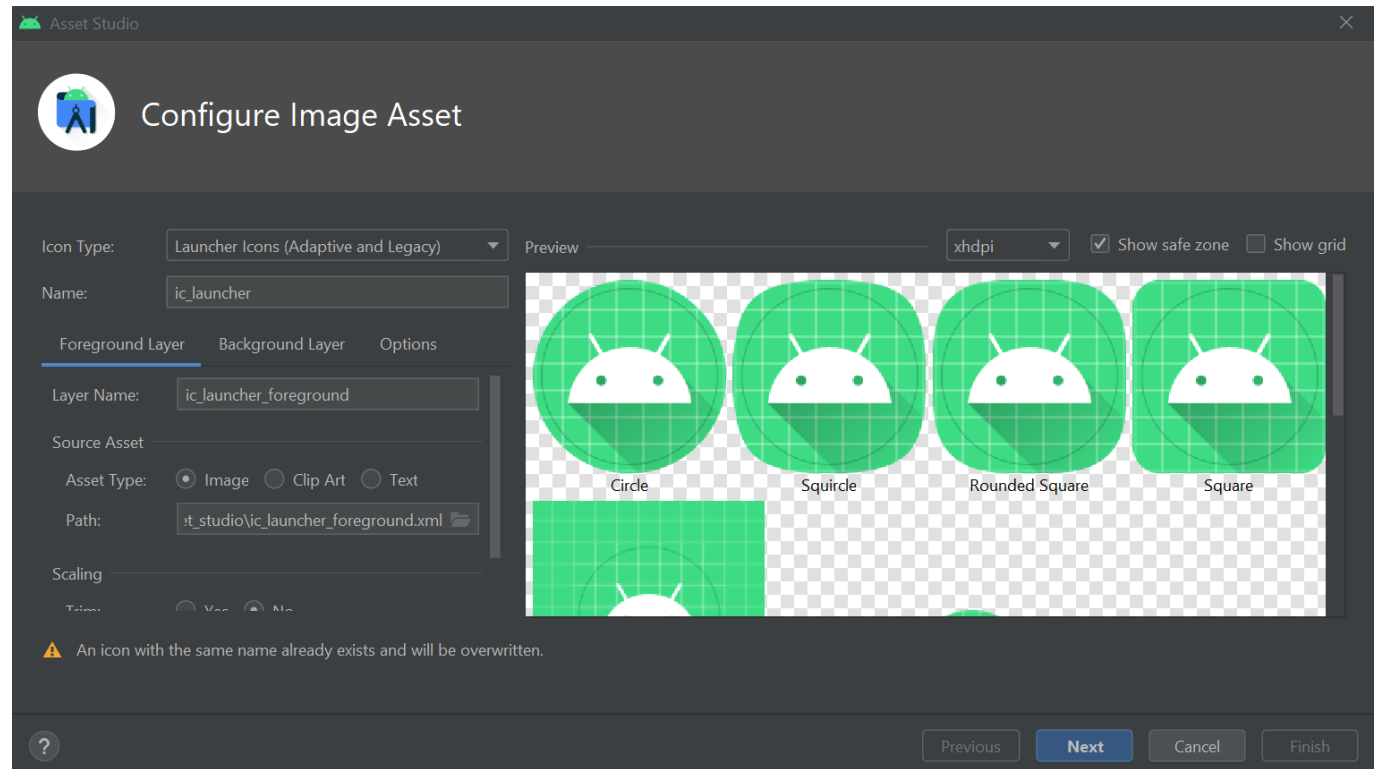
Đổi biểu tượng ứng dụng

- Tạo biểu tượng ứng dụng từ tập tin ảnh
 - Chuột phải lên thư mục res > New > Image Asset



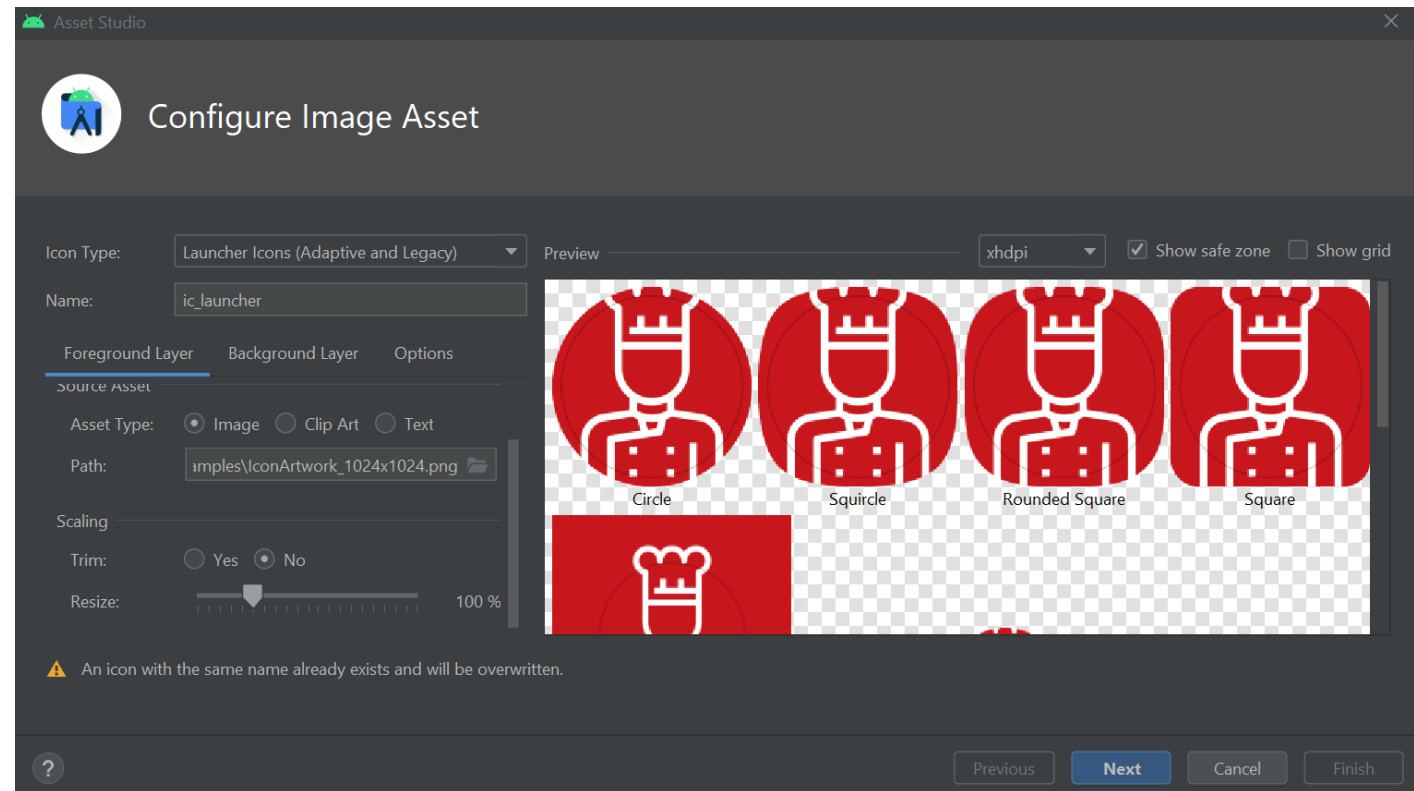
Đổi biểu tượng ứng dụng

- Tạo biểu tượng ứng dụng từ tập tin ảnh
 - Trong cửa sổ Configure Image Asset, chọn tập tin ảnh để tạo biểu tượng tại Source Asset > Path



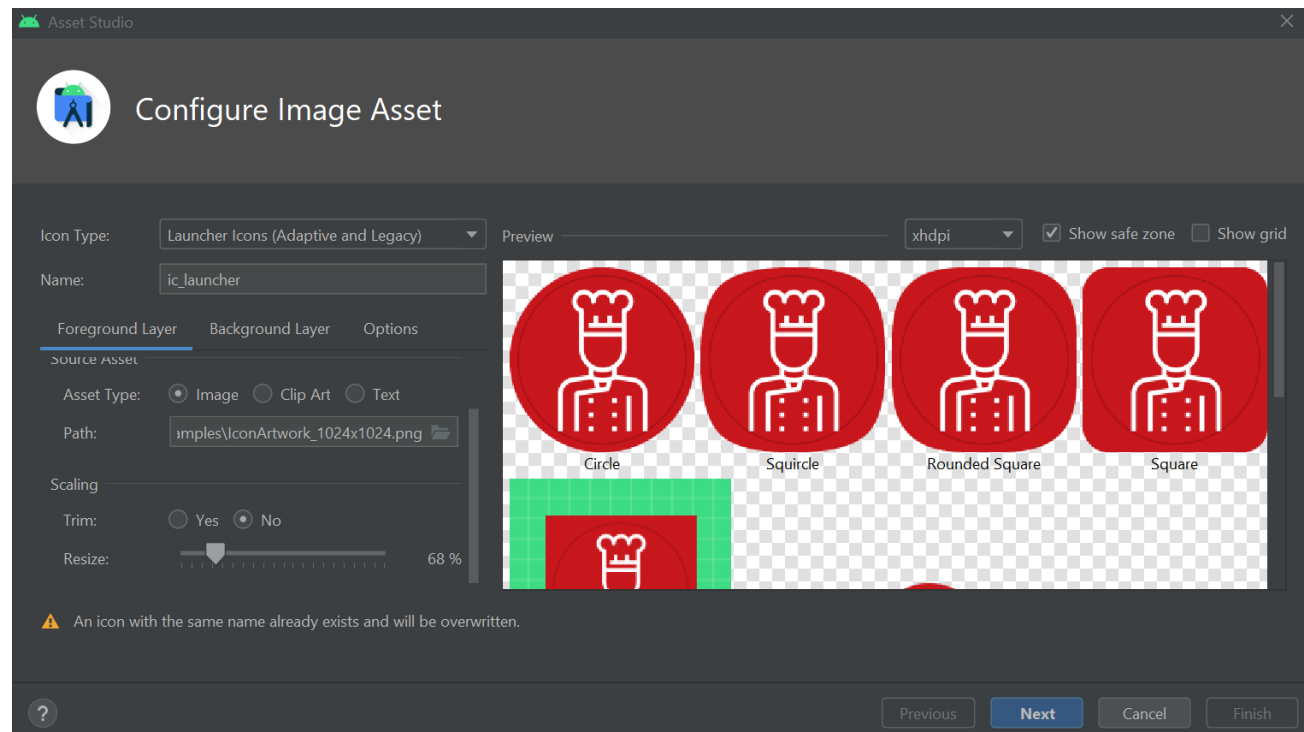
Đổi biểu tượng ứng dụng

- Tạo biểu tượng ứng dụng từ tập tin ảnh
 - Trong cửa sổ Configure Image Asset, chọn tập tin ảnh để tạo biểu tượng tại Source Asset > Path



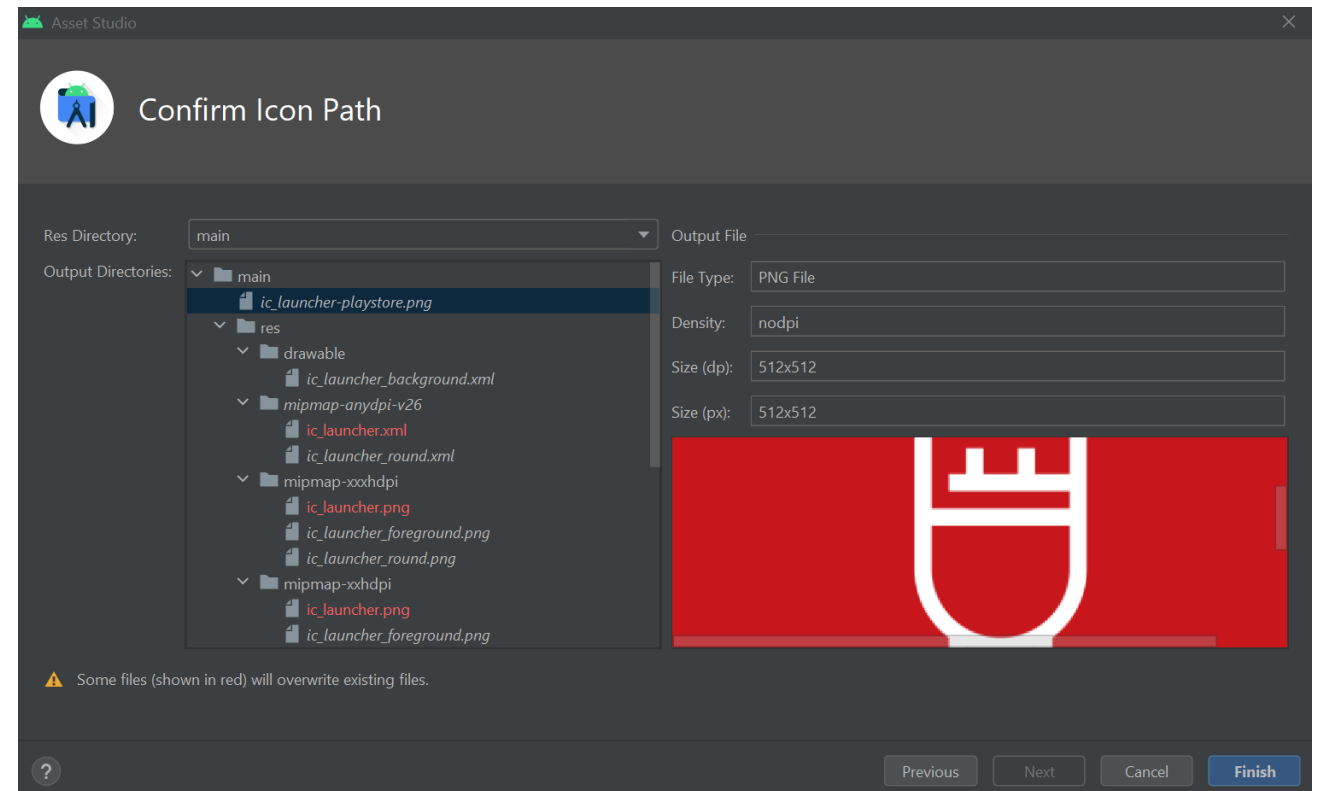
Đổi biểu tượng ứng dụng

- Tạo biểu tượng ứng dụng từ tập tin ảnh
 - Tùy chỉnh kích thước (scaling) sao cho nội dung chính của biểu tượng nằm trong vòng tròn vùng an toàn



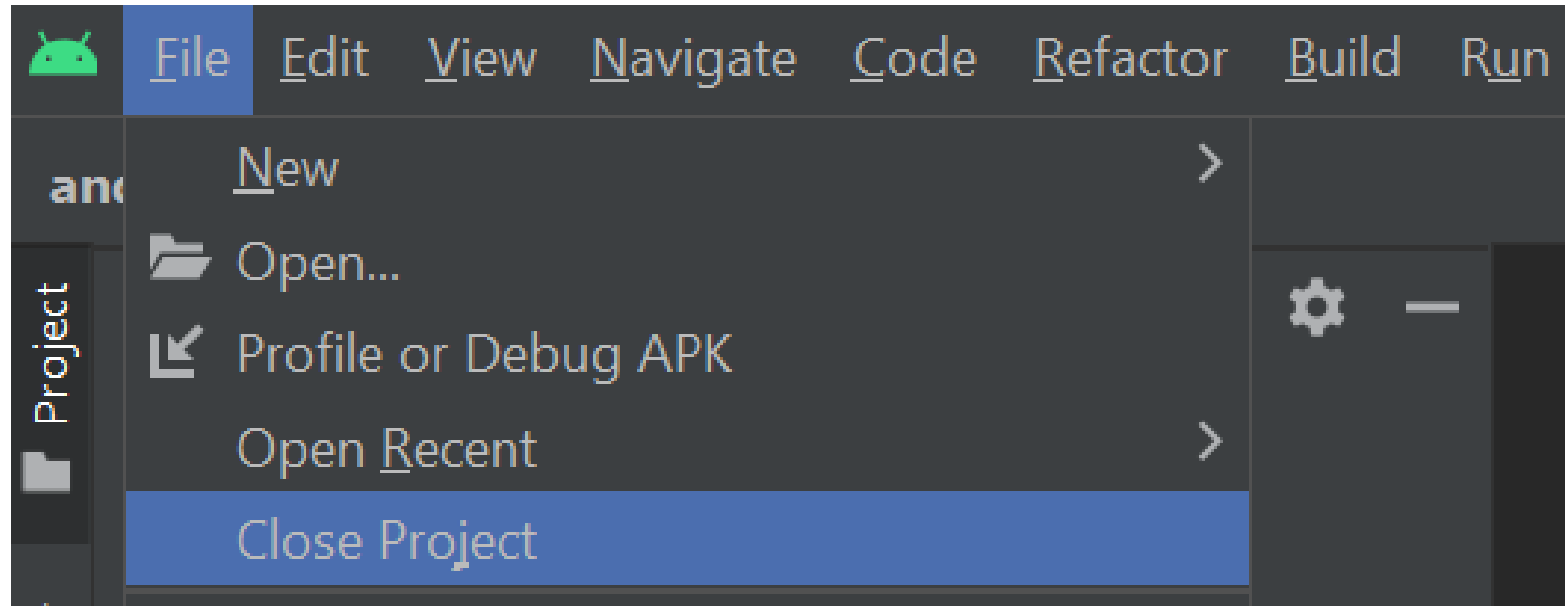
Đổi biểu tượng ứng dụng

- Tạo biểu tượng ứng dụng từ tập tin ảnh
 - Tùy chỉnh kích thước xong chọn Next
 - Tại Res Directory đặt là main rồi chọn Finish



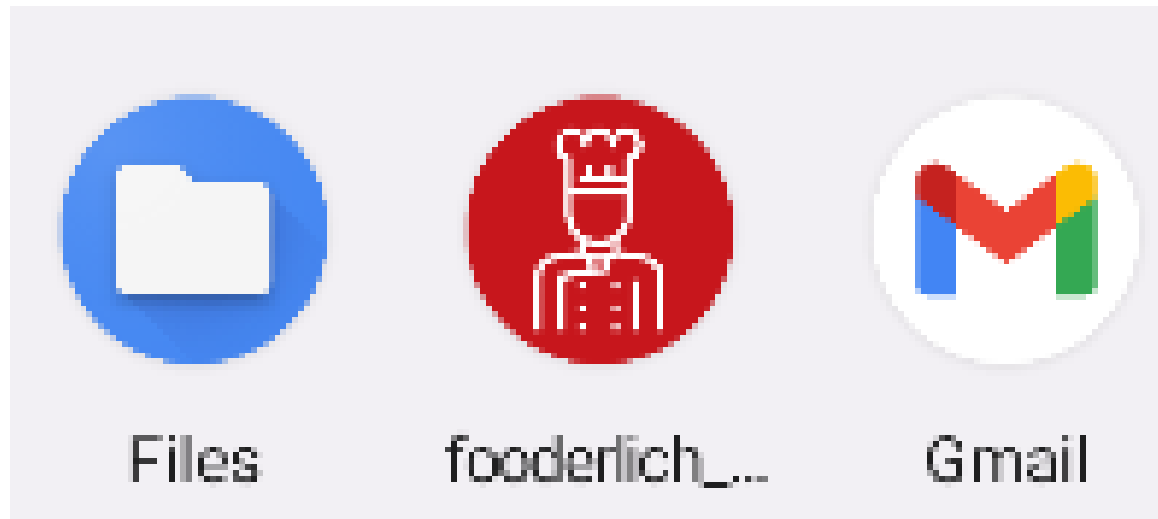
Đổi biểu tượng ứng dụng

- Tạo biểu tượng ứng dụng từ tập tin ảnh
 - Đóng dự án Android



Đổi biểu tượng ứng dụng

- Tạo biểu tượng ứng dụng từ tập tin ảnh
 - Chạy lại ứng dụng Android sẽ thấy biểu tượng mới của ứng dụng



- Một giải pháp khác là dùng package: [flutter_launcher_icons](#)

Đổi tên ứng dụng

- Tên ứng dụng được định nghĩa bởi thuộc tính **android:label** trong tập tin **AndroidManifest.xml**

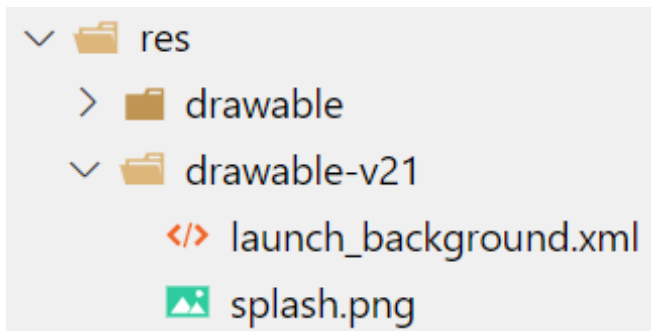
```
<application  
    android:label="Fooderlich"
```

- Build lại ứng dụng sẽ thấy tên thay đổi



Thêm màn hình khởi động

- Mặc định, cấu hình màn hình khởi động (splash screen) nằm trong tập tin **res/drawable-v21/launch_background.xml** (cho API level ≥ 21)



```
<layer-list xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:drawable="@android:color/black" />

    <item>
        <bitmap
            android:gravity="center"
            android:src="@drawable/splash" />
    </item>
</layer-list>
```

- Một giải pháp khác là dùng package: [flutter_native_splash](#)

Tạo build phát hành cho ứng dụng

- Hai tùy chọn định dạng cho build phát hành
 - App bundle (khuyến nghị bởi Google Play Store)
 - APK
- Tạo app bundle

```
cd [project]  
flutter build appbundle
```

- [project]/build/app/outputs/bundle/release/app-release.aab
- Mặc định, app bundle hỗ trợ kiến trúc ARM 32-bit/64-bit, x86 64-bit
- Có thể kiểm tra app bundle trên thiết bị với [bundletool](#) hoặc tải lên [Google Play Store](#)

Tạo build phát hành cho ứng dụng

- Hai tùy chọn định dạng cho build phát hành
 - App bundle (khuyến nghị bởi Google Play Store)
 - APK
- Tạo APK

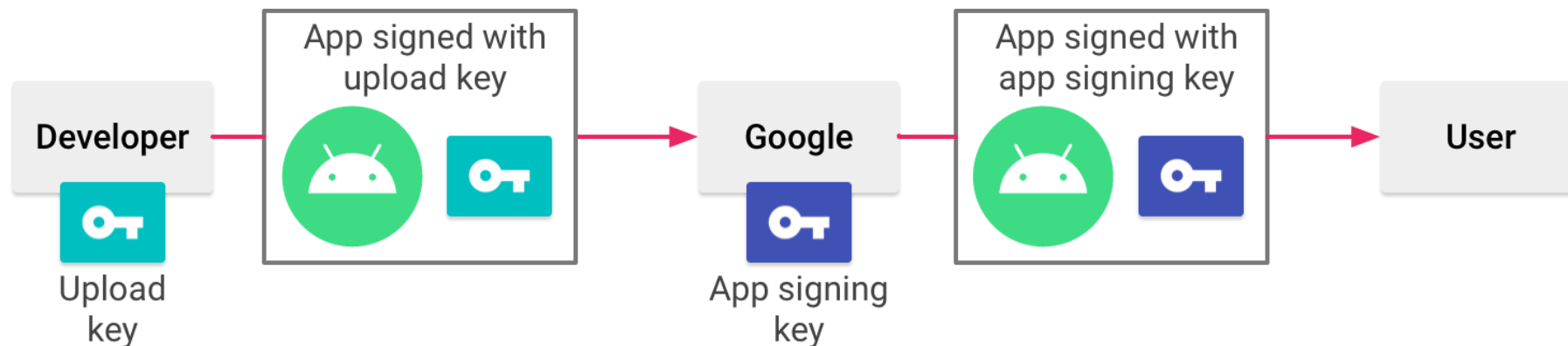
```
cd [project]
```

```
flutter build apk --split-per-abi
```

- [project]/build/app/outputs/apk/release/app-*-release.apk
- Mặc định, APK được tạo cho kiến trúc ARM 32-bit/64-bit, x86 64-bit
- Cài đặt lên thiết bị: kết nối thiết bị, `cd [project]`, `flutter install`

Phát hành lên Google Play Store

- Đăng ký tài khoản nhà phát triển trên [Google Play Console](#)
 - Phí đăng ký tài khoản là \$25
 - Xác nhận danh tính (cần CCCD/Hộ chiếu, ...)
- Tạo khóa upload và dùng nó ký số vào build phát hành
- Dùng Play App Signing trên Play Console để tải ứng dụng lên



Phát hành lên Google Play Store

- <https://docs.flutter.dev/deployment/android>
- <https://developer.android.com/studio/publish>
- Tham khảo Chương 17 Flutter Apprentice

Câu hỏi?