

Giới thiệu về Flutter

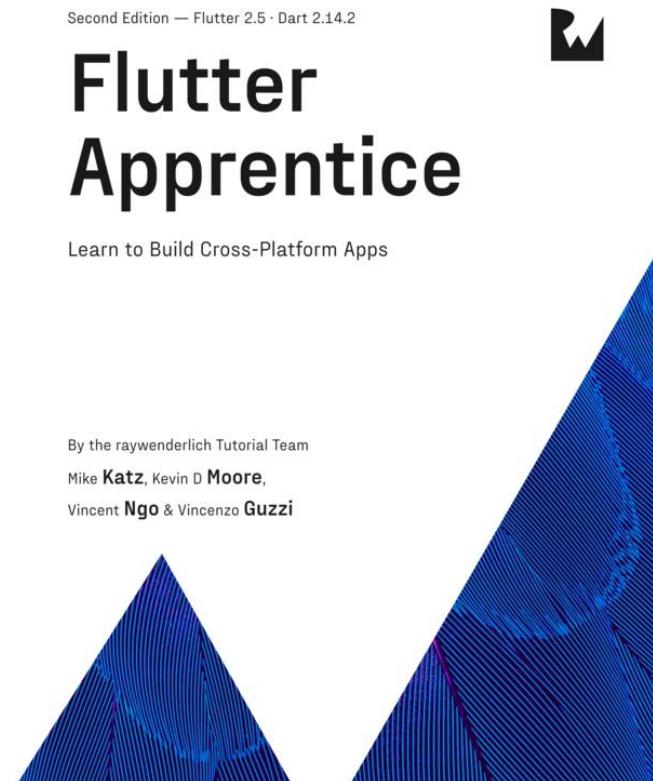
Bùi Võ Quốc Bảo

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



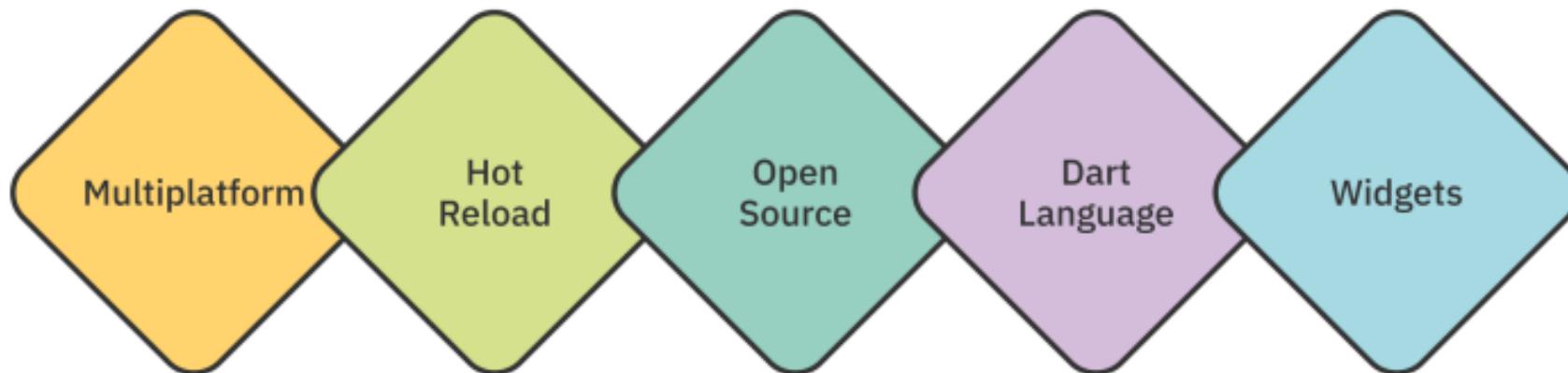
Tài liệu tham khảo

- Chương 1-2, **Flutter Apprentice** by Vincenzo Guzzi, Kevin D Moore, Vincent Ngo and Michael Katz

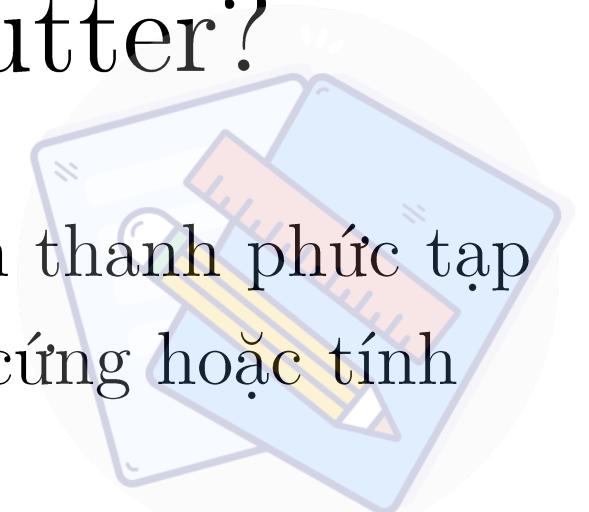


Flutter là gì?

- Bộ công cụ phát triển phần mềm từ Google cho việc phát triển ứng dụng đa nền tảng (cross-platform)
 - Các nền tảng hỗ trợ (Flutter 3.x): Di động (Android/iOS), Web, Desktop (Windows, MacOS, Linux), Nhúng

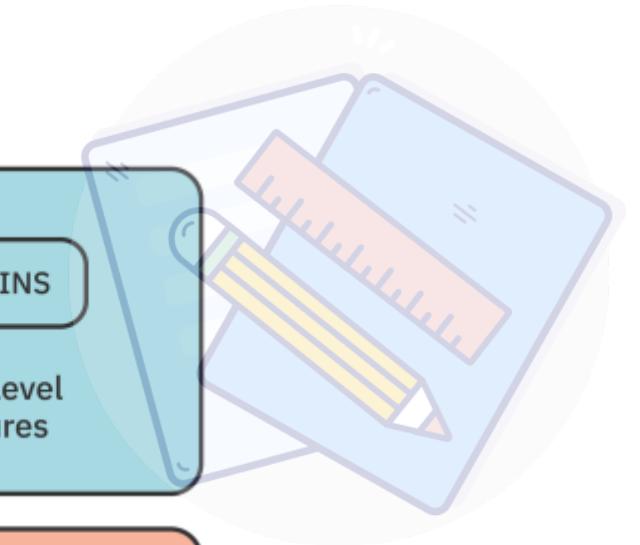
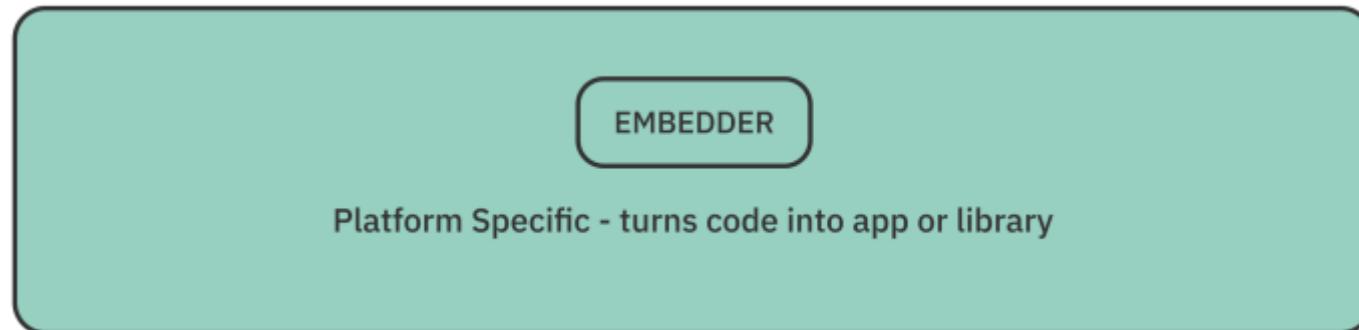
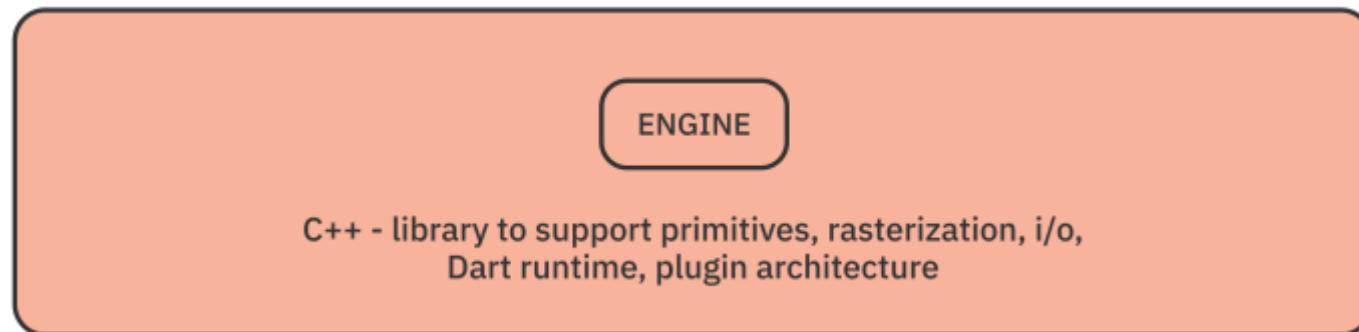
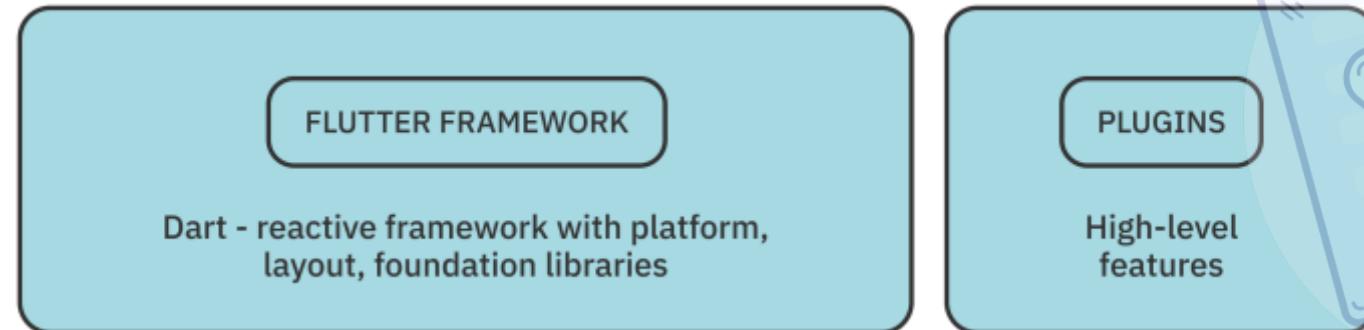


Khi nào không nên sử dụng Flutter?

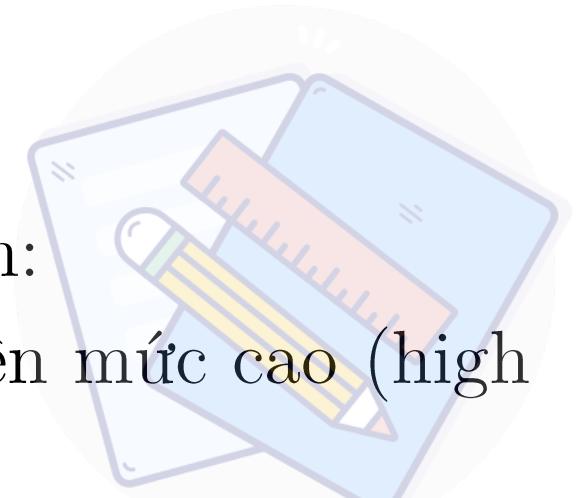


- Xây dựng game đồ họa nặng, ứng dụng xử lý âm thanh phức tạp
- Các ứng dụng có nhiều yêu cầu tính năng phần cứng hoặc tính năng native đặc biệt
- Ứng dụng trên một vài nền tảng như watchOS, tvOS (Flutter chưa hỗ trợ chính thức)

Kiến trúc của Flutter



Kiến trúc của Flutter

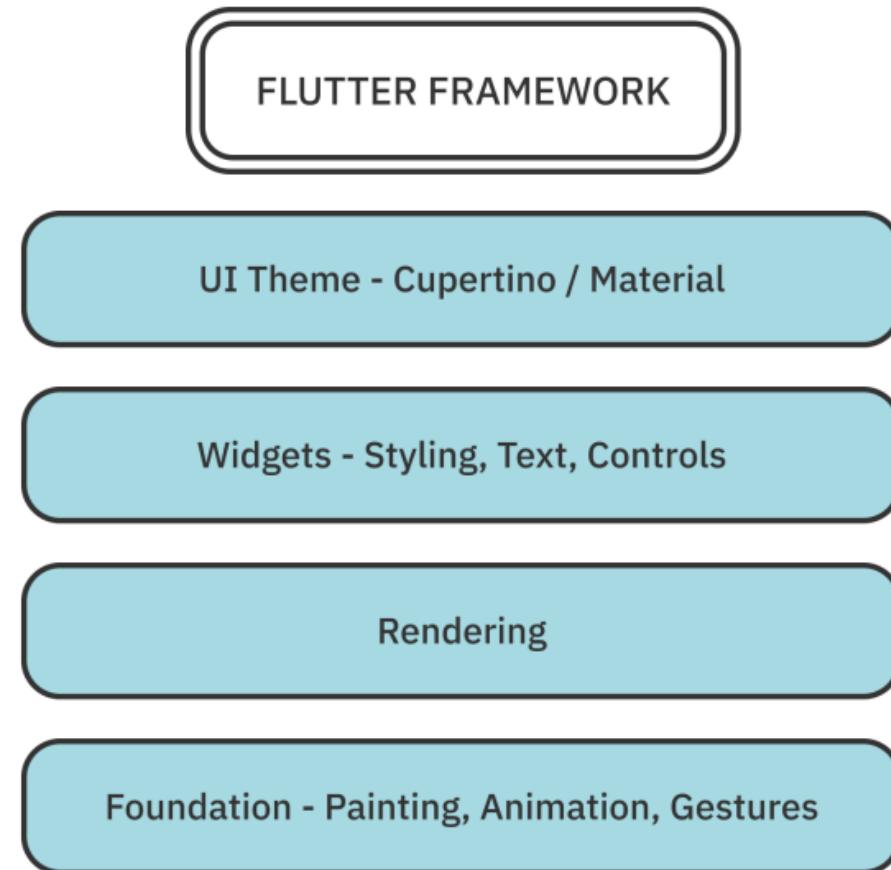


Kiến trúc của Flutter bao gồm ba lớp (layer) chính:

- Lớp **framework** viết bằng Dart chứa các thư viện mức cao (high level) được sử dụng để xây dựng ứng dụng
 - Hỗ trợ các **plugin**: các tính năng mức cao như định vị (geolocation), truy cập máy ảnh, thanh toán trong ứng dụng,...
- Lớp **engine** chứa các thư viện lõi C++ là nền tảng cho các ứng dụng Flutter
- Lớp **embedder** khác biệt cho từng nền tảng, xử lý đóng gói mã lệnh thành ứng dụng hoặc mô-đun nhúng

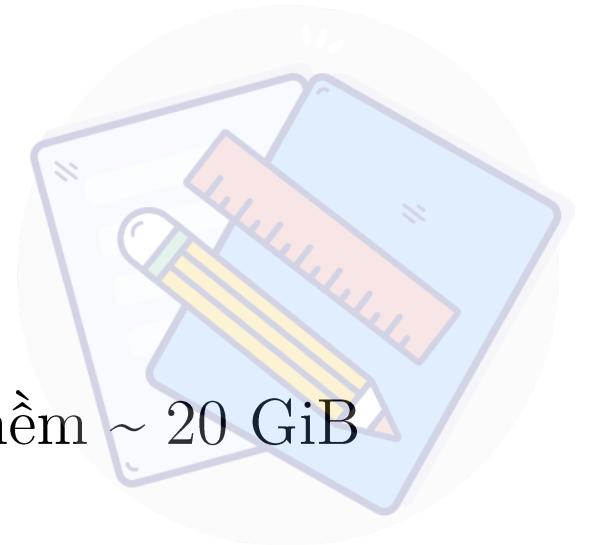
Kiến trúc của Flutter

Mỗi lớp trong kiến trúc bao gồm các lớp con và các mô-đun:



Bạn cần những gì?

- Máy tính chạy Windows/MacOS/Linux
 - Cần MacOS nếu muốn xây dựng ứng dụng cho iOS
 - Dung lượng đĩa trống cho cài đặt các công cụ phần mềm ~ 20 GiB
- Flutter SDK phiên bản mới nhất:
<https://docs.flutter.dev/development/tools/sdk/releases>
- Android Studio (<https://developer.android.com/studio>)
- Một IDE: Visual Studio Code hoặc Android Studio
- Một thiết bị di động (nếu muốn chạy trên thiết bị thật)
- Hướng dẫn cài đặt: <https://docs.flutter.dev/get-started/install>



Bạn cần những gì?

Tạo thiết bị di động giả lập: <https://docs.flutter.dev/get-started/install/windows>



Set up the Android emulator

To prepare to run and test your Flutter app on the Android emulator, follow these steps:

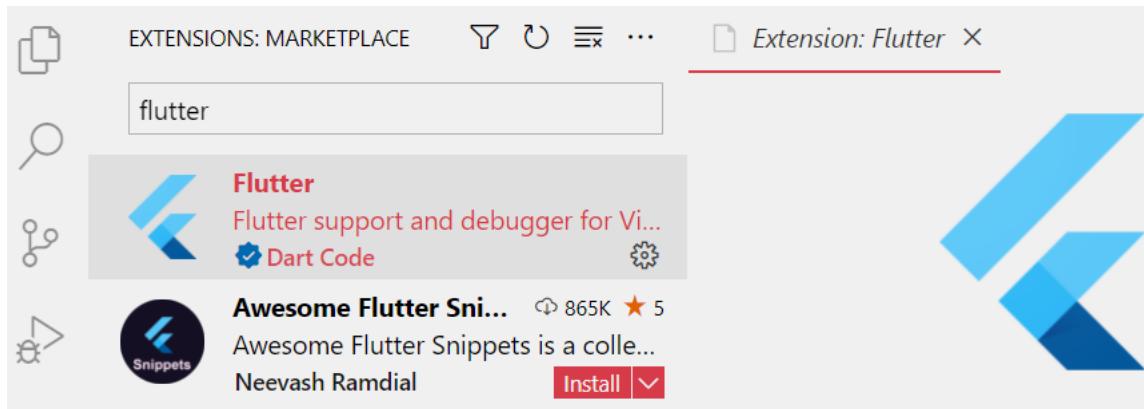
1. Enable [VM acceleration](#) on your machine.
2. Launch **Android Studio**, click the **AVD Manager** icon, and select **Create Virtual Device...**
In older versions of Android Studio, you should instead launch **Android Studio > Tools > Android > AVD Manager** and select **Create Virtual Device....** (The **Android** submenu is only present when inside an Android project.)
If you do not have a project open, you can choose **Configure > AVD Manager** and select **Create Virtual Device...**
3. Choose a device definition and select **Next**.
4. Select one or more system images for the Android versions you want to emulate, and select **Next**. An *x86* or *x86_64* image is recommended.
5. Under Emulated Performance, select **Hardware - GLES 2.0** to enable [hardware acceleration](#).
6. Verify the AVD configuration is correct, and select **Finish**.

For details on the above steps, see [Managing AVDs](#).

7. In **Android Virtual Device Manager**, click **Run** in the toolbar. The emulator starts up and displays the default canvas for your selected OS version and device.

Bạn cần những gì?

Cài đặt phần mở rộng Flutter trên VSCode



The screenshot shows the VSCode Extension Marketplace interface. A search bar at the top contains the text "flutter". Below it, two extensions are listed:

- Flutter** by Dart Code (v3.40.0) - This extension is highlighted with a gray background. It provides support and debugging for Visual Studio Code. It has 4,277,843 installs and a 5-star rating from 61 reviews.
- Awesome Flutter Snippets** by Neevash Ramdial (v3.40.0) - This extension offers snippets for Flutter development. It has 865K installs and a 5-star rating from 5 reviews.

On the right side of the screenshot, there is a large, semi-transparent watermark featuring a pencil and a ruler on a light blue background, with the text "Học lập trình mobile với flutter" faintly visible.

Bạn cần những gì?

Sau khi cài đặt các công cụ cần thiết, kiểm tra các cài đặt bằng lệnh: *flutter doctor*



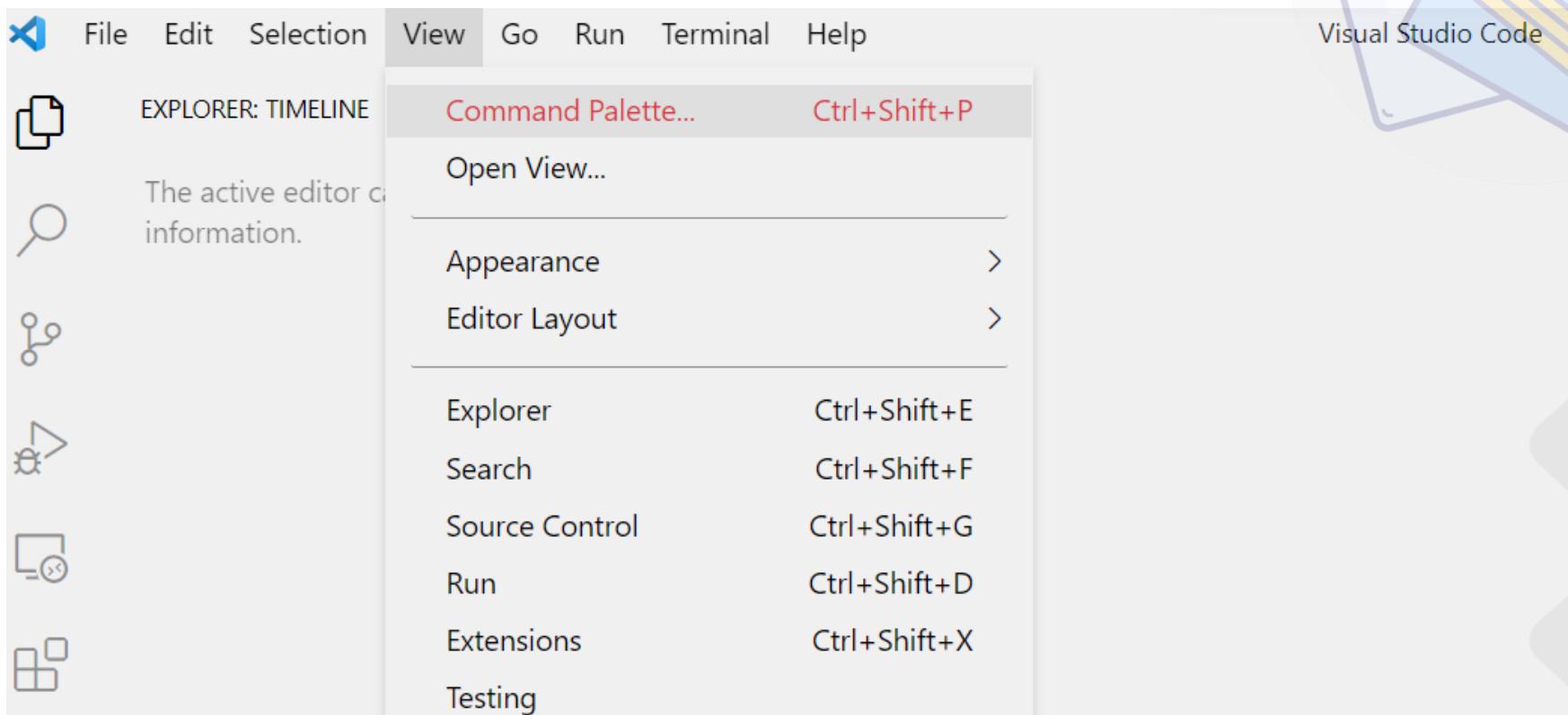
```
flutter> flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.0.0, on Microsoft Windows [Version 10.0.22000.675], locale vi-VN)
[✓] Android toolchain - develop for Android devices (Android SDK version 32.1.0-rc1)
[✓] Chrome - develop for the web
[✓] Visual Studio - develop for Windows (Visual Studio Community 2022 17.1.1)
[✓] Android Studio (version 2021.1)
[✓] VS Code (version 1.67.1)
[✓] Connected device (3 available)
[✓] HTTP Host Availability

• No issues found!
```

(Chỉ cần cài đặt các công cụ phát triển ứng dụng cho Android)

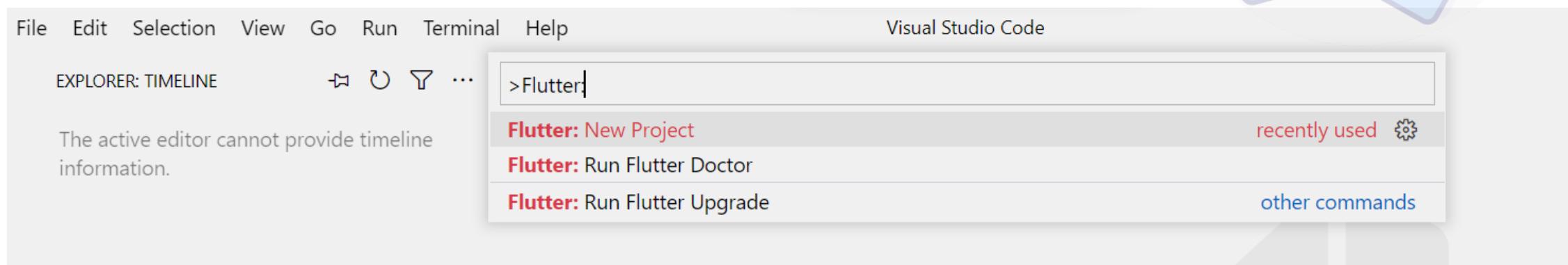
Tạo ứng dụng Flutter (VSCode)

Mở Visual Studio Code, View > Command Palette...



Tạo ứng dụng Flutter (VSCode)

Mở Visual Studio Code, View > Command Palette... > Flutter: New Project và làm theo hướng dẫn



Tạo ứng dụng Flutter (Terminal)

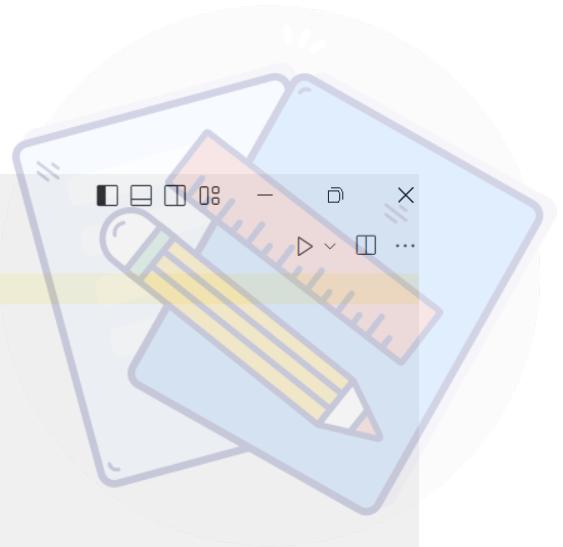
Tạo dự án dùng lệnh trên Terminal/Command Prompt:
`flutter create <project_name>`



Chỉ định các nền tảng được hỗ trợ trong dự án:
`flutter create --platforms=android,ios <project_name>`

Chỉ định tên tổ chức (mặc định là `com.example`):
`flutter create --org net.baobui <project_name>`

Tạo ứng dụng Flutter (VSCode)



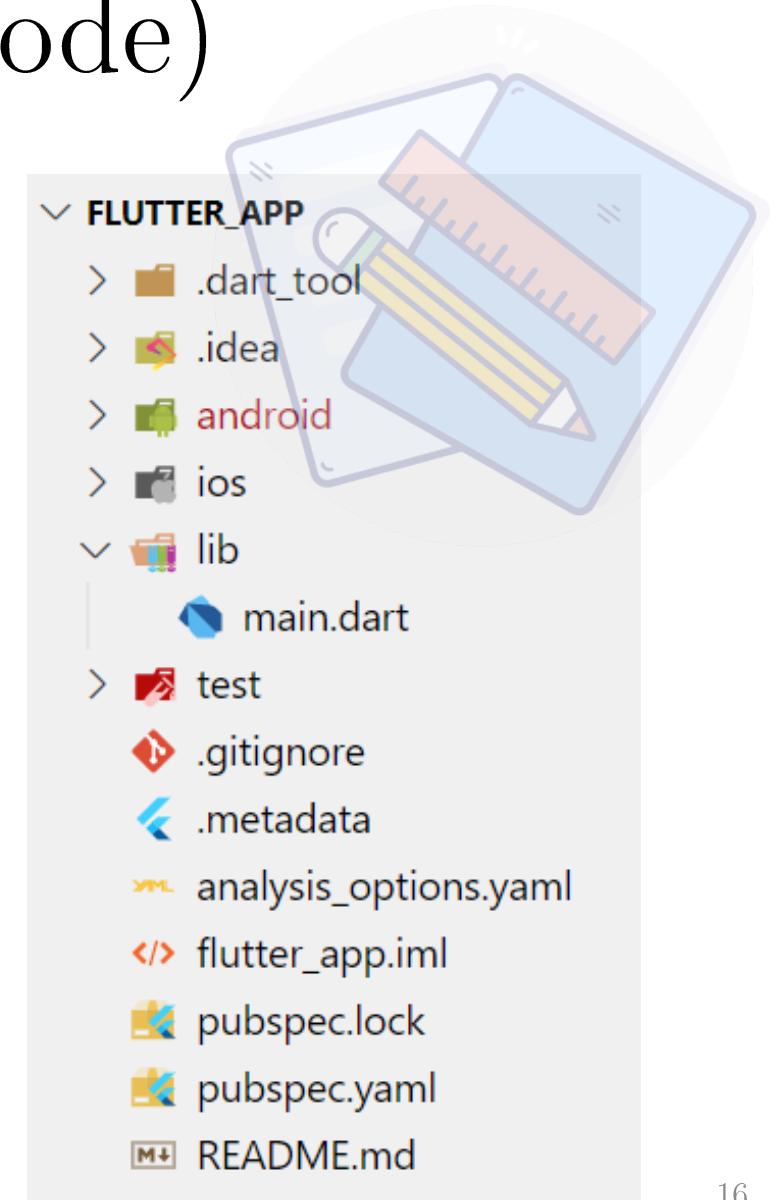
A screenshot of the Visual Studio Code (VSCode) interface showing a Flutter application setup. The left sidebar displays the project structure under 'EXPLORER' with a tree view of files and folders. The main editor area shows the 'main.dart' file with Dart code. The status bar at the bottom provides build information and other developer tools.

```
File Edit Selection View Go Run Terminal Help
EXPLORER
FLUTTER_APP
  .dart_tool
  .idea
  android
  build
  ios
  lib
    main.dart
  test
    .gitignore
    .metadata
    analysis_options.yaml
    flutter_app.iml
    pubspec.lock
    pubspec.yaml
  README.md
...
main.dart - flutter_app - Visual Studio Code
1 import 'package:flutter/material.dart';
2
3 void main() {
4   runApp(const MyApp());
5 }
6
7 class MyApp extends StatelessWidget {
8   const MyApp({super.key});
9
10 // This widget is the root of your application.
11 @override
12 Widget build(BuildContext context) {
13   return MaterialApp(
14     title: 'Flutter Demo',
15     theme: ThemeData(
16       // This is the theme of your application.
17       //
18       // Try running your application with "flutter run". You'll see the
19       // application has a blue toolbar. Then, without quitting the app, try
20       // changing the primarySwatch below to Colors.green and then invoke
21       // "hot reload" (press "r" in the console where you ran "flutter run",
22       // or simply save your changes to "hot reload" in a Flutter IDE).
23       // Notice that the counter didn't reset back to zero; the application
24       // is not restarted.
25       primarySwatch: Colors.blue,
26     ), // ThemeData
27     home: const MyHomePage(title: 'Flutter Demo Home Page'),
28   ); // MaterialApp
}
Ln 1, Col 40  Spaces: 2  UTF-8  CRLF  {} Dart  ⚡ Go Live  No Device  Colorize: 0 variables  ⚡ Colorize  ⚡ Prettier  🔍  🔍
```

Live Share

Tạo ứng dụng Flutter (VSCode)

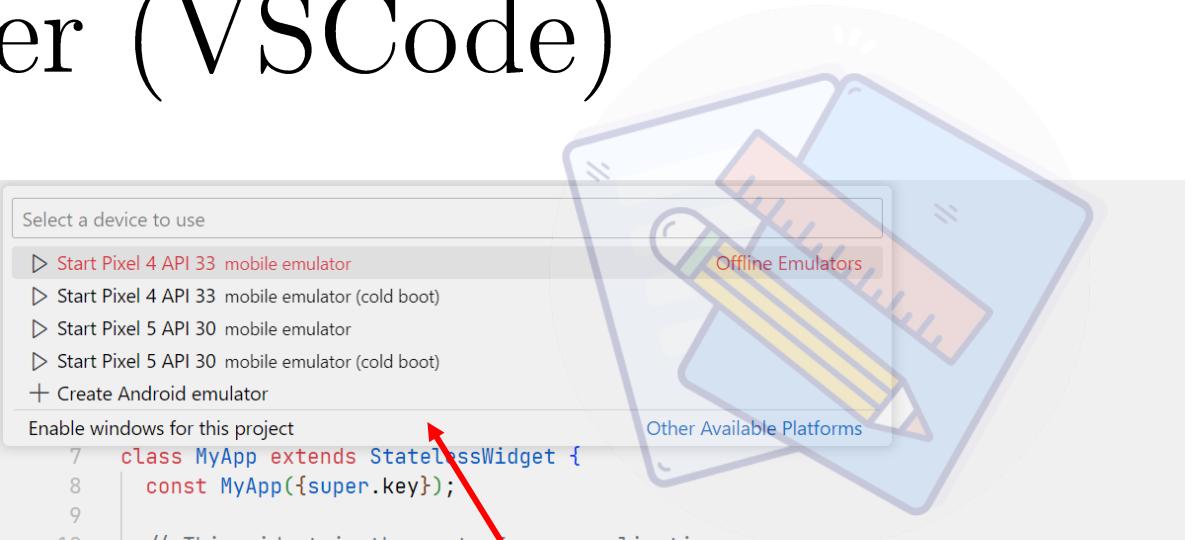
Chủ yếu làm việc trong thư mục **lib/** và tập tin **pubspec.yaml**



Chạy ứng dụng Flutter (VSCode)

Chọn nền tảng/thiết bị để chạy ứng dụng

- Chọn một trong các thiết bị giả lập được liệt kê
- Thiết bị thật kết nối với máy tính ở chế độ “Developer Options” cũng sẽ hiển thị tại đây

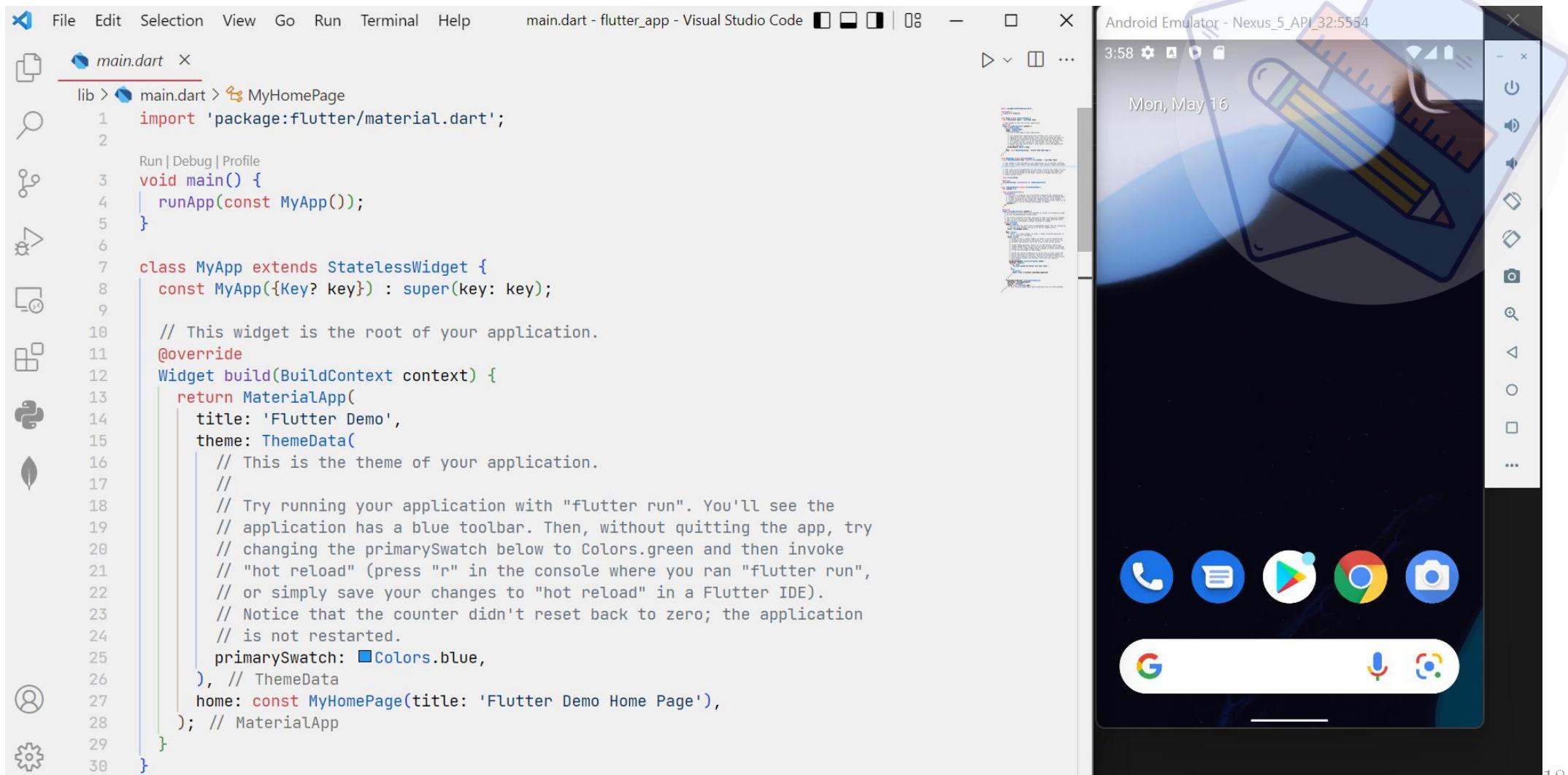


The screenshot shows the Visual Studio Code interface. A dropdown menu titled "Select a device to use" is open, listing several emulator options. A red arrow points from the "No Device" status in the bottom right corner of the code editor towards the "Select a device to use" dropdown. A floating window titled "Offline Emulators" displays two stylized smartphone icons with the text "Offline Emulators" and "Other Available Platforms". The code editor contains a Dart file named "main.dart" with the following content:

```
7 class MyApp extends StatelessWidget {  
8   const MyApp({super.key});  
9  
10  // This widget is the root of your application.  
11  @override  
12  Widget build(BuildContext context) {  
13    return MaterialApp(  
14      title: 'Flutter Demo',  
15      theme: ThemeData(  
16        // This is the theme of your application.  
17        //  
18        // Try running your application with "flutter run". You'll see the  
19        // application has a blue toolbar. Then, without quitting the app, try  
20        // changing the primarySwatch below to Colors.green and then invoke  
21        // "hot reload" (press "r" in the console where you ran "flutter run",  
22        // or simply save your changes to "hot reload" in a Flutter IDE).  
23        // Notice that the counter didn't reset back to zero; the application  
24        // is not restarted.  
25        primarySwatch: Colors.blue,  
26      ), // ThemeData  
27      home: const MyHomePage(title: 'Flutter Demo Home Page'),  
28    ); // MaterialApp
```

At the bottom right of the code editor, there is a status bar with the following information: "Ln 51, Col 20 Spaces: 2 UTF-8 CRLF {} Dart ⚡ Go Live No Device Colorize: 0 variables ⚡ Colorize". The "No Device" link in the status bar is circled with a red oval.

Chạy ứng dụng Flutter (VSCode)



The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Editor:** The main editor window displays the file `main.dart` from the `lib` directory of a `flutter_app` project. The code implements a simple Flutter application structure with a `MaterialApp` widget.
- Output Panel:** A sidebar panel shows the output of the build process, indicating successful compilation and hot reload.
- Android Emulator:** A floating window titled "Android Emulator - Nexus_5_API_32:5554" shows a dark-themed mobile interface with a blue toolbar at the top. The date "Mon, May 16" is visible. The home screen includes icons for Phone, Messages, Google Play Store, Google Chrome, and Camera, along with a Google search bar.

```
main.dart - flutter_app - Visual Studio Code
File Edit Selection View Go Run Terminal Help
main.dart x
lib > main.dart > MyHomePage
import 'package:flutter/material.dart';

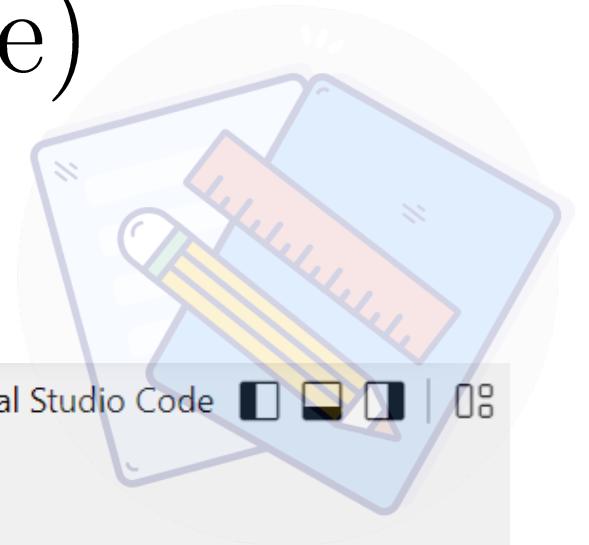
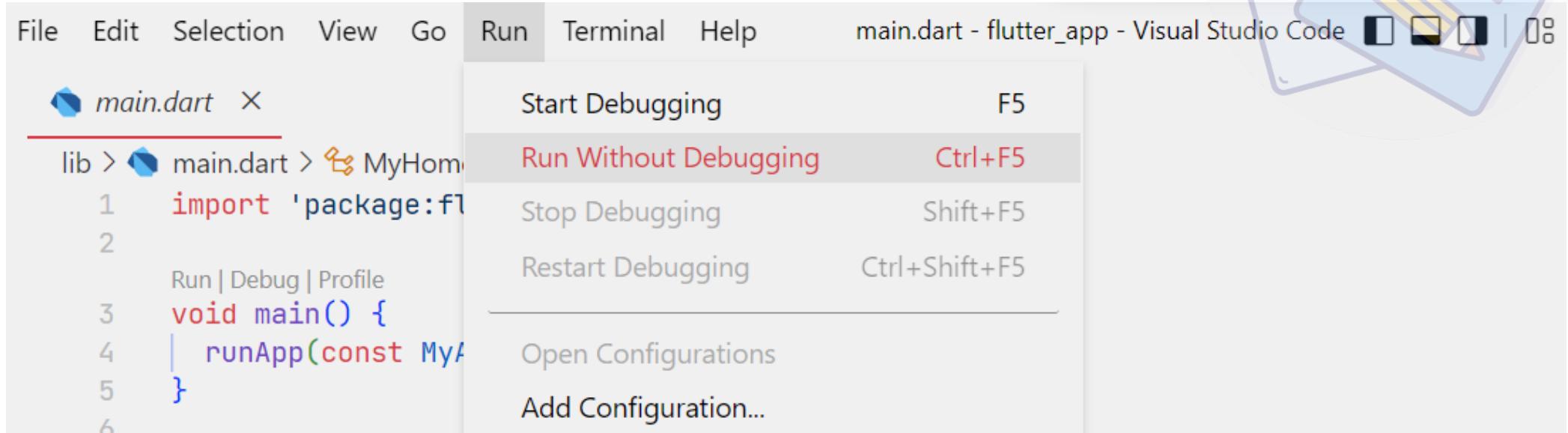
void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

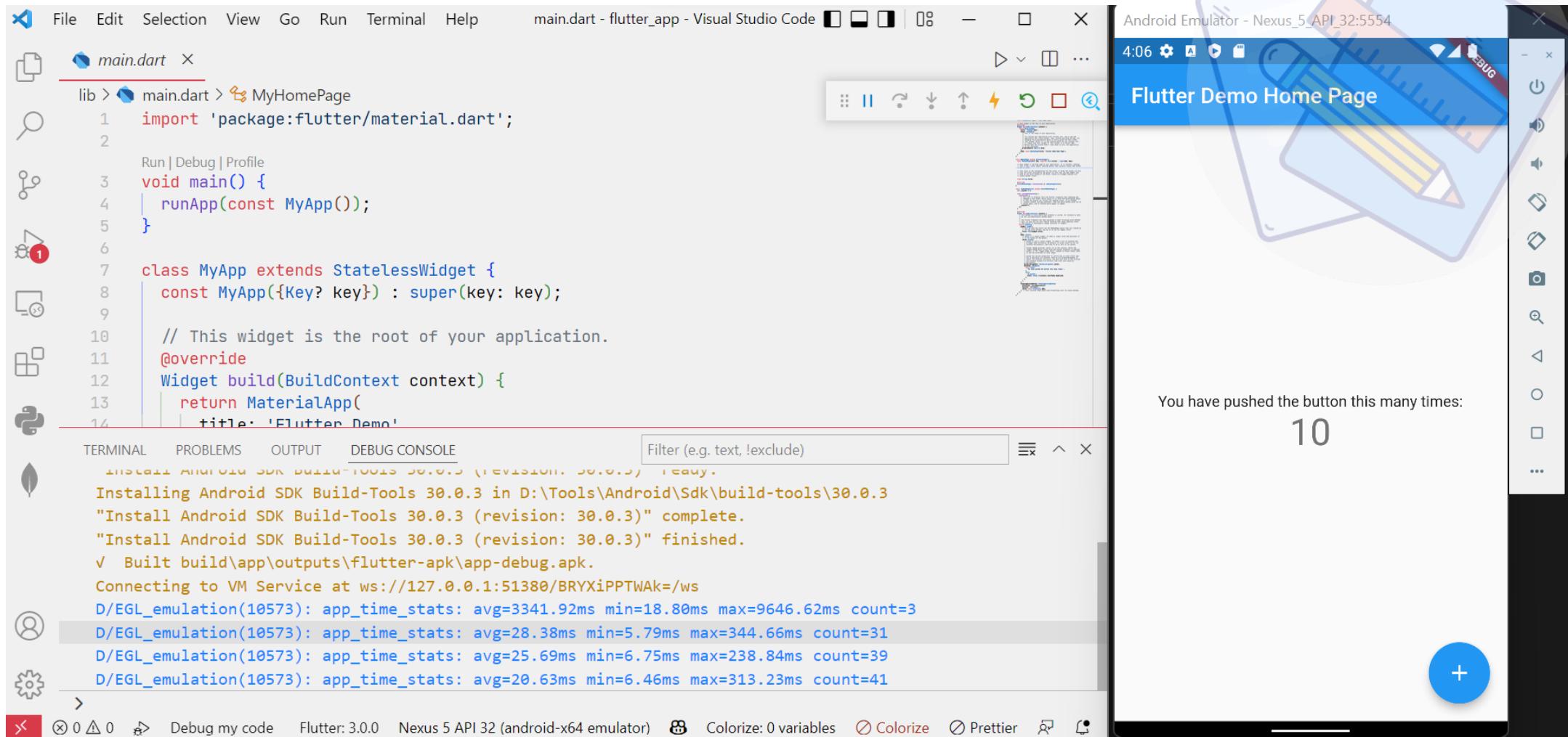
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // Try running your application with "flutter run". You'll see the
        // application has a blue toolbar. Then, without quitting the app, try
        // changing the primarySwatch below to Colors.green and then invoke
        // "hot reload" (press "r" in the console where you ran "flutter run",
        // or simply save your changes to "hot reload" in a Flutter IDE).
        // Notice that the counter didn't reset back to zero; the application
        // is not restarted.
        primarySwatch: Colors.blue,
      ), // ThemeData
      home: const MyHomePage(title: 'Flutter Demo Home Page'),
    ); // MaterialApp
}
```

Chạy ứng dụng Flutter (VSCode)

Run > Run Without Debugging



Chạy ứng dụng Flutter (VSCode)



The screenshot shows the Visual Studio Code interface with a Flutter project open. The left side displays the code editor with `main.dart` file content:

```
File Edit Selection View Go Run Terminal Help
main.dart - flutter_app - Visual Studio Code
main.dart x
lib > main.dart > MyHomePage
1 import 'package:flutter/material.dart';
2
3 void main() {
4   runApp(const MyApp());
5 }
6
7 class MyApp extends StatelessWidget {
8   const MyApp({Key? key}) : super(key: key);
9
10 // This widget is the root of your application.
11 @override
12 Widget build(BuildContext context) {
13   return MaterialApp(
14     title: 'Flutter Demo'
```

The terminal below shows the output of the build process:

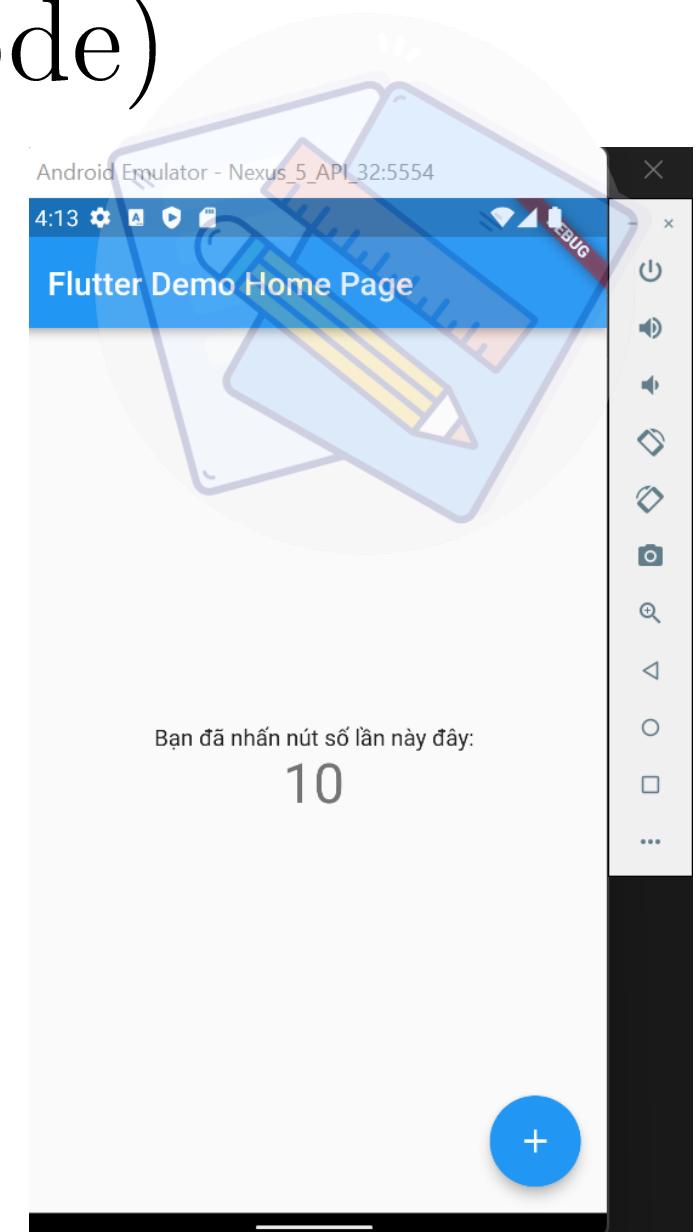
```
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE
Filter (e.g. text, !exclude)
Install Android SDK Build-Tools 30.0.3 (revision: 30.0.3) ready.
Installing Android SDK Build-Tools 30.0.3 in D:\Tools\Android\Sdk\build-tools\30.0.3
"Install Android SDK Build-Tools 30.0.3 (revision: 30.0.3)" complete.
"Install Android SDK Build-Tools 30.0.3 (revision: 30.0.3)" finished.
v Built build\app\outputs\flutter-apk\app-debug.apk.
Connecting to VM Service at ws://127.0.0.1:51380/BRYXipPTWAK=/ws
D/EGL_emulation(10573): app_time_stats: avg=3341.92ms min=18.80ms max=9646.62ms count=3
D/EGL_emulation(10573): app_time_stats: avg=28.38ms min=5.79ms max=344.66ms count=31
D/EGL_emulation(10573): app_time_stats: avg=25.69ms min=6.75ms max=238.84ms count=39
D/EGL_emulation(10573): app_time_stats: avg=20.63ms min=6.46ms max=313.23ms count=41
```

The right side of the interface shows the Android emulator running the `Flutter Demo Home Page`. The screen displays a blue header with the text "Flutter Demo Home Page" and a message below it stating "You have pushed the button this many times: 10".

Chạy ứng dụng Flutter (VSCode)

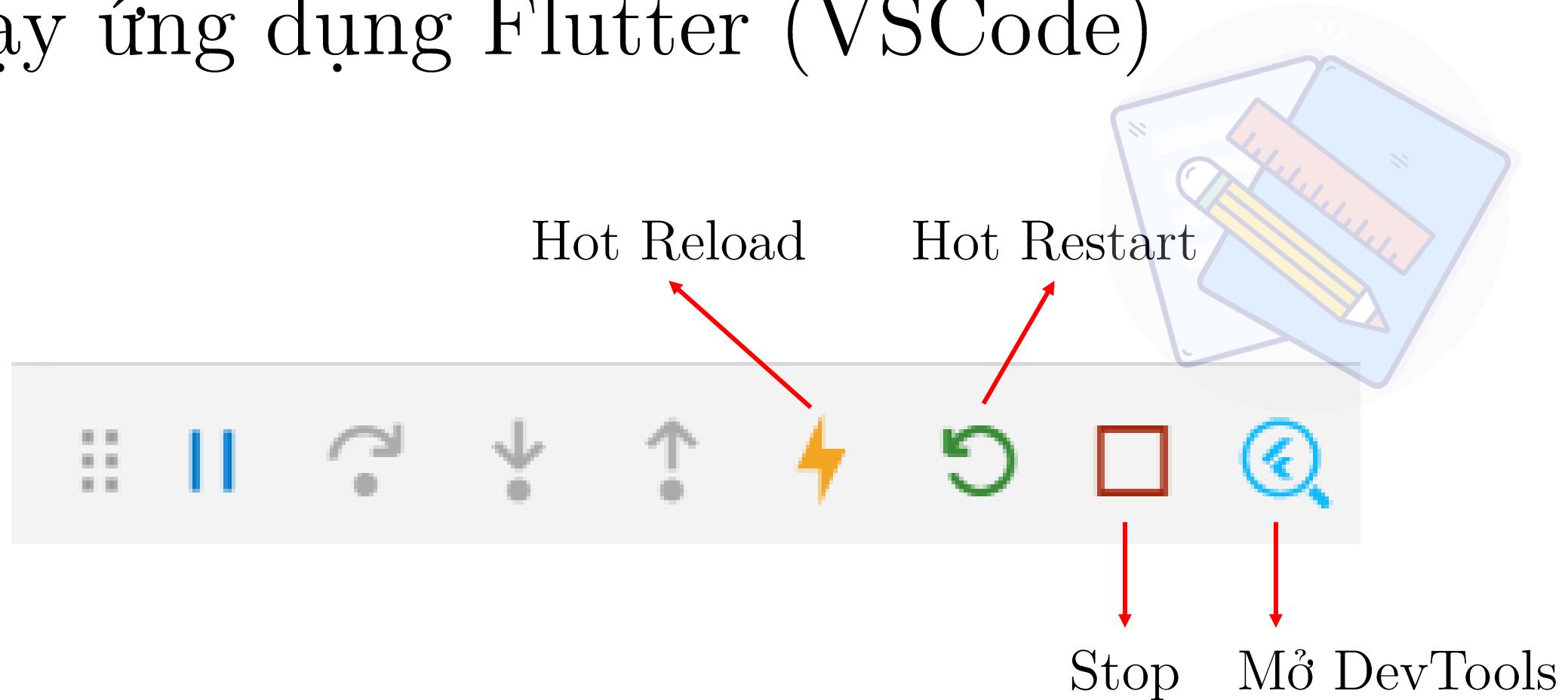
Thử tính năng Hot Reload: hiệu chỉnh đoạn text trong file **main.dart** như sau và tiến hành lưu:

```
97     |   |   |   | children: <Widget>[  
98     |   |   |   |   const Text(  
99     |   |   |   |   |   'Bạn đã nhấn nút số lần này đây:',  
100    |   |   |   |   ), // Text
```



Quan sát sẽ thấy ứng dụng trên thiết bị giả lập thay đổi ngay lập tức!

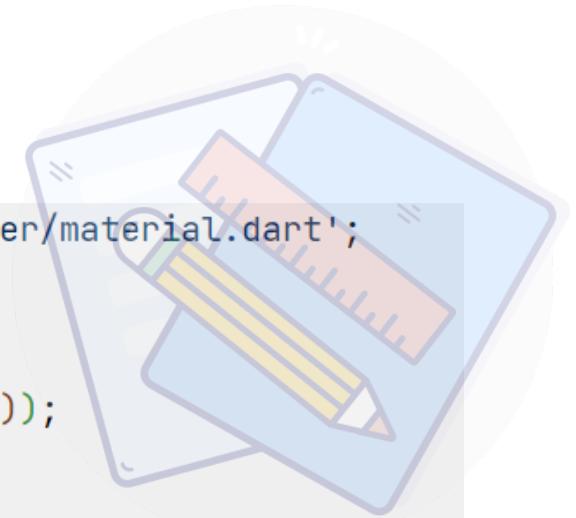
Chạy ứng dụng Flutter (VSCode)



Hello, Flutter!

- Dòng 1: khai báo sử dụng thư viện material của Flutter
- Dòng 3-5: điểm đầu vào của ứng dụng
 - Ứng dụng Flutter được cấu thành từ các widget
 - Bản thân ứng dụng Flutter cũng là một widget
- Dòng 7-21: định nghĩa widget **MyApp**
 - Mỗi widget phải định nghĩa phương thức **build** khai báo cách tạo widget

```
1 import 'package:flutter/material.dart';
2
3 Run | Debug | Profile
4 void main() {
5   runApp(const MyApp());
6 }
7
7 class MyApp extends StatelessWidget {
8   const MyApp({Key? key}) : super(key: key);
9
10 @override
11 Widget build(BuildContext context) {
12   return MaterialApp(
13     debugShowCheckedModeBanner: false,
14     title: 'Flutter Demo',
15     theme: ThemeData(
16       primarySwatch: Colors.green,
17     ), // ThemeData
18     home: const MyHomePage(),
19   ); // MaterialApp
20 }
21 }
```



Hello, Flutter!

- Dòng 7-21: định nghĩa widget **MyApp**
 - Mỗi widget phải định nghĩa phương thức **build** khai báo cách tạo widget
 - Một số tham số để tạo một widget **MaterialApp**
 - + **debugShowCheckedModeBanner**: ẩn/hiện biểu ngữ debug trên giao diện ứng dụng
 - + **title**: tên ứng dụng
 - + **theme**: theme của ứng dụng
 - + **home**: trang chủ (trang mặc định) của ứng dụng

```
1 import 'package:flutter/material.dart';
2
3 Run | Debug | Profile
4 void main() {
5   runApp(const MyApp());
6 }
7
7 class MyApp extends StatelessWidget {
8   const MyApp({Key? key}) : super(key: key);
9
10
11 @override
12 Widget build(BuildContext context) {
13   return MaterialApp(
14     debugShowCheckedModeBanner: false,
15     title: 'Flutter Demo',
16     theme: ThemeData(
17       primarySwatch: Colors.green,
18     ), // ThemeData
19     home: const MyHomePage(),
20   ); // MaterialApp
21 }
```



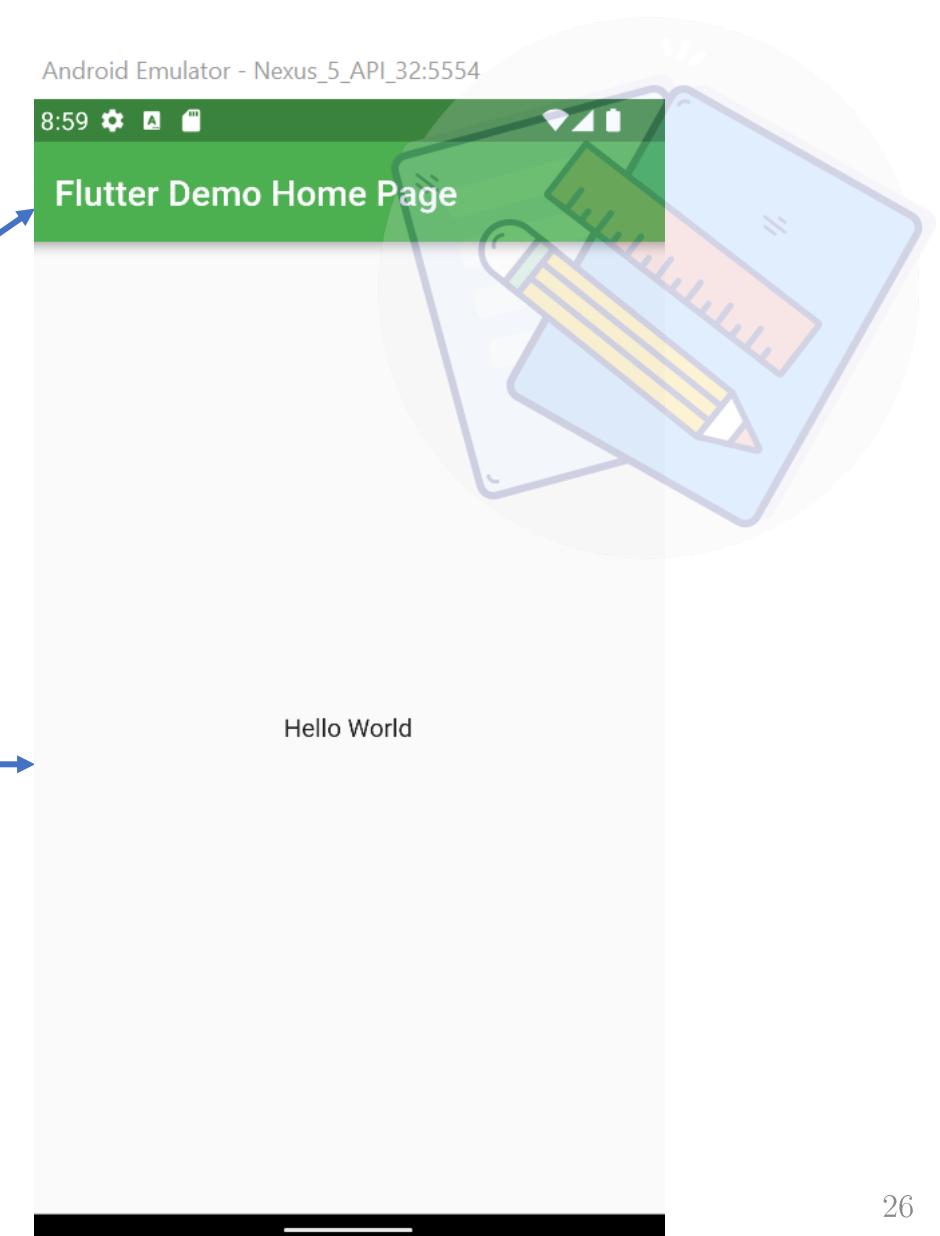
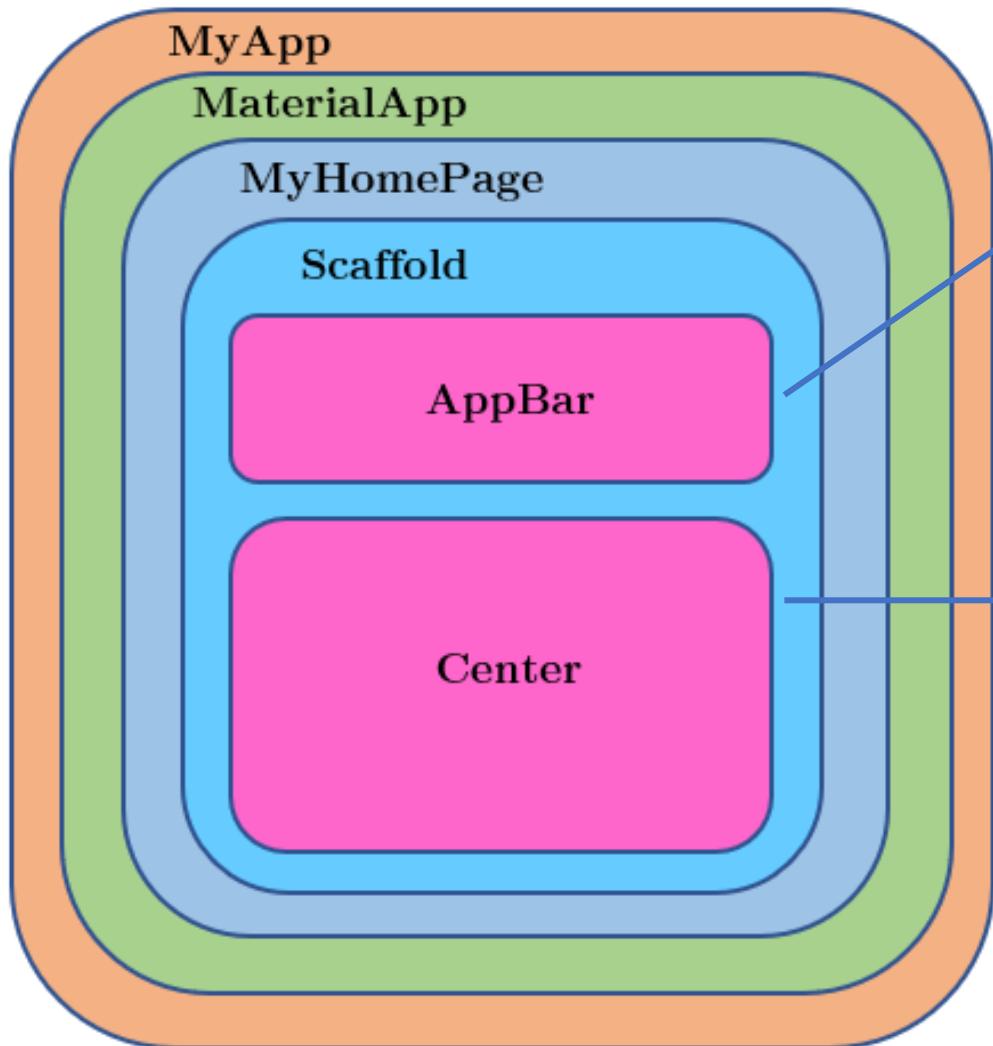
Hello, Flutter!

- Dòng 23-37: định nghĩa widget **MyHomePage**
 - Scaffold: widget khung sườn ứng dụng material
 - Center: widget đặt widget con của nó nằm giữa



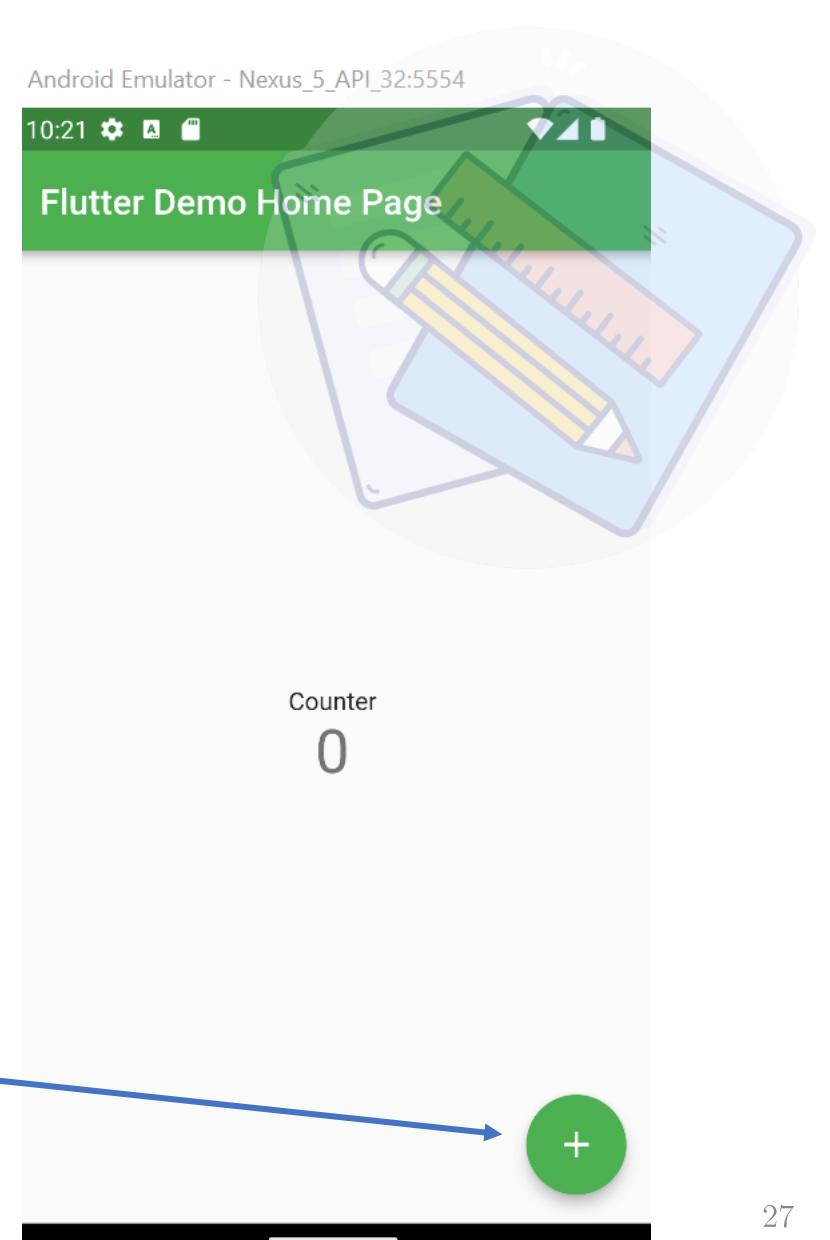
```
23 class MyHomePage extends StatelessWidget {  
24   const MyHomePage({Key? key}) : super(key: key);  
25  
26   @override  
27   Widget build(BuildContext context) {  
28     return Scaffold(  
29       appBar: AppBar(  
30         title: const Text('Flutter Demo Home Page'),  
31       ), // AppBar  
32       body: const Center(  
33         child: Text('Hello World'),  
34       ), // Center  
35     ); // Scaffold  
36   }  
37 }
```

Hello, Flutter!



Hello, Flutter!

```
class MyHomePage extends StatelessWidget {  
  ...  
  int _count = 0;  
  void _increaseCounter() { _count++; }  
  
  @override  
  Widget build(BuildContext context) {  
    return Scaffold(  
      ...  
      body: Center(  
        child: Column(  
          mainAxisAlignment: MainAxisAlignment.center,  
          children: <Widget>[  
            const Text('Counter'),  
            Text(  
              '_$count',  
              style: Theme.of(context).textTheme.headline4,  
            ),  
            ...  
          ],  
        ),  
        floatingActionButton: FloatingActionButton(  
          onPressed: _increaseCounter,  
          child: const Icon(Icons.add),  
        ),  
      );  
  }  
}
```



Hello, Flutter!

- Tuy nhiên, giá trị của counter trên ứng dụng không thay đổi khi nhấn nút +. Tại sao?
- **MyHomePage** hiện là một *StatelessWidget*, không thể thay đổi trạng thái (`_count`) một khi đã được tạo ra
- Nhấp chuột phải vào dòng lệnh định nghĩa **MyHomePage**, Refactor... > Convert to *StatefulWidget*



```
class MyHomePage extends StatelessWidget {  
  MyHo (key: key);  
  Convert to StatefulWidget  
  int _count = 0;  
  void _increaseCounter() {  
    _count++;  
  }  
}
```

Hello, Flutter!

- Để báo cho Flutter biết khi nào trạng thái của một StatefulWidget thay đổi và cần cập nhật lại giao diện, đặt các thay đổi trạng thái trong phương thức `setState`



```
class MyHomePage extends StatefulWidget {
  const MyHomePage({Key? key}) : super(key: key);

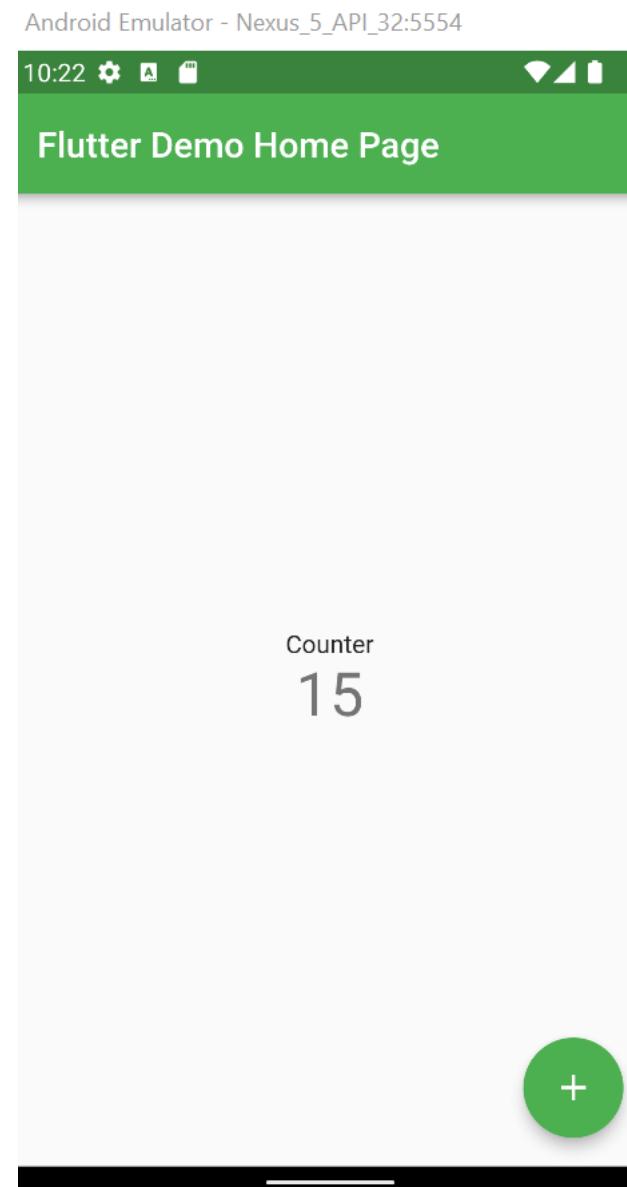
  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  int _count = 0;

  void _increaseCounter() {
    setState(() {
      _count++;
    });
  }

  @override
  Widget build(BuildContext context) {
```

Hello, Flutter!



Hello, Flutter!

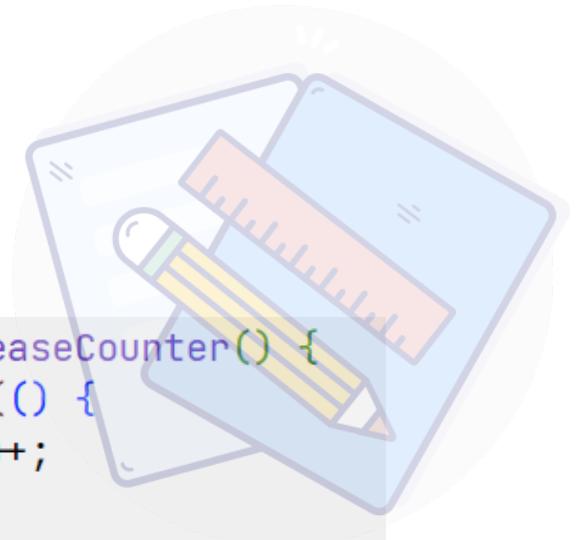
- Thêm các nút giảm, đặt lại giá trị của counter

```
75 |   const Text('Counter'),
76 |   Text(
77 |     '_count',
78 |     style: Theme.of(context).textTheme.headline4,
79 |   ), // Text
80 |   Row(
81 |     mainAxisSize: MainAxisSize.spaceAround,
82 |     children: <Widget>[
83 |       ElevatedButton(
84 |         onPressed: _increaseCounter,
85 |         child: const Icon(Icons.add),
86 |       ), // ElevatedButton
87 |       ElevatedButton(
88 |         onPressed: _decreaseCounter,
89 |         style: ElevatedButton.styleFrom(
90 |           primary: Colors.red,
91 |         ),
92 |         child: const Icon(Icons.remove),
93 |       ), // ElevatedButton
94 |     ], // <Widget>[]
95 |   ) // Row
```

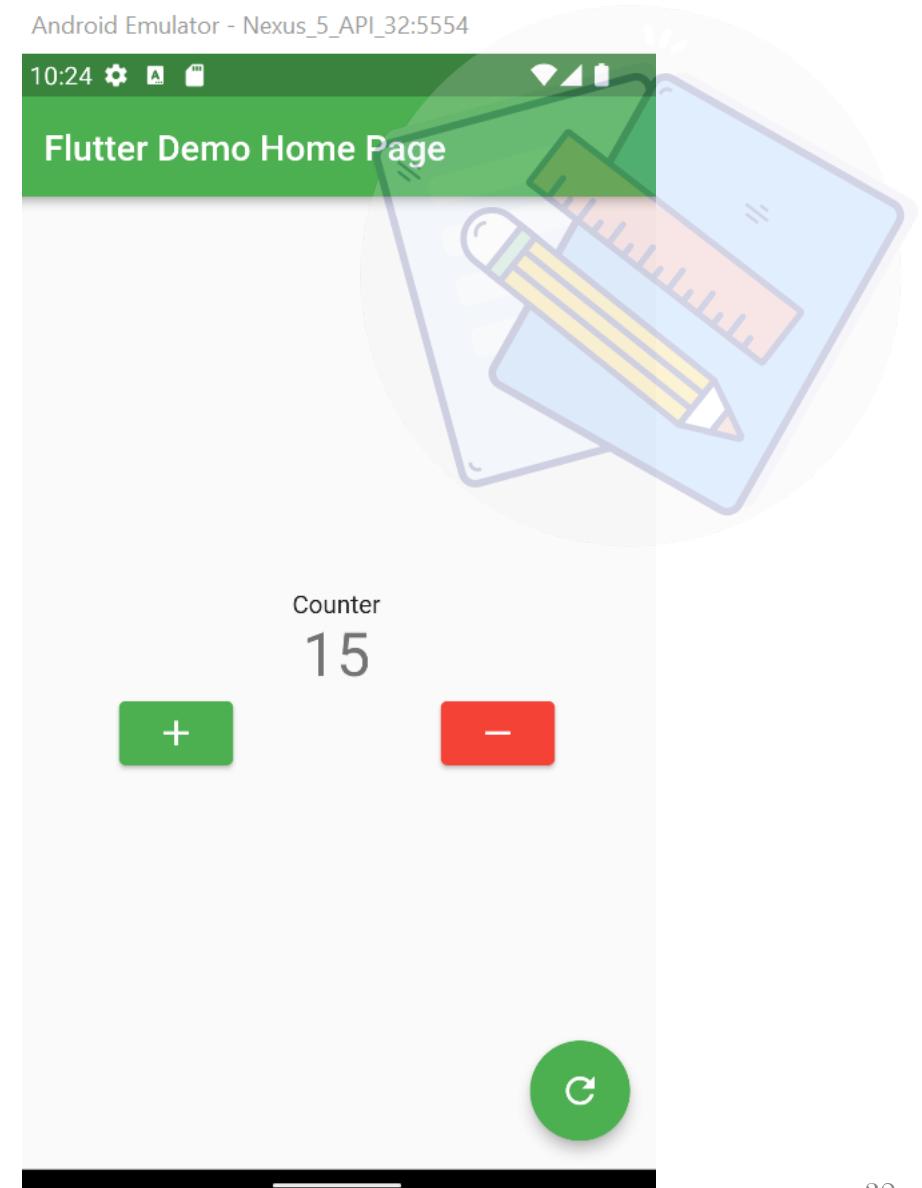
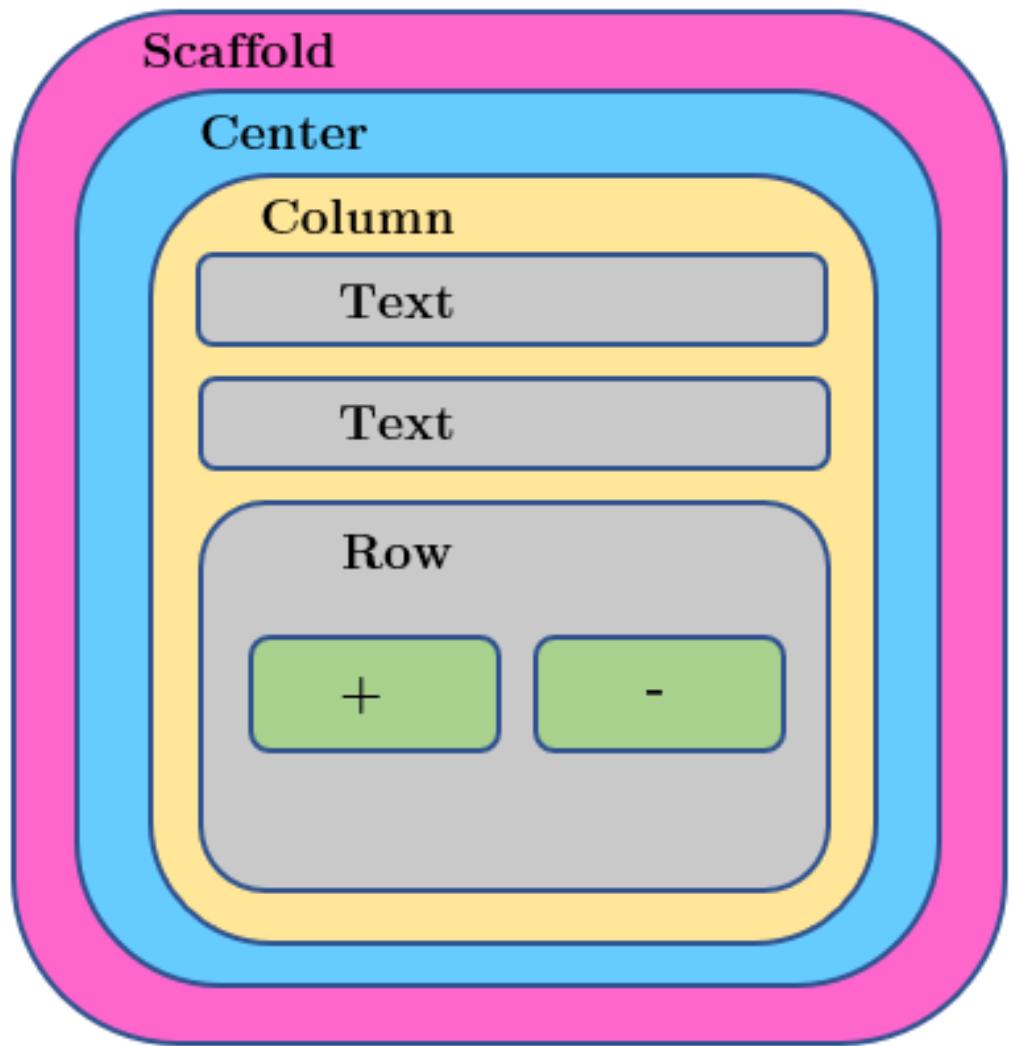
```
void _increaseCounter() {
  setState(() {
    _count++;
  });
}

void _decreaseCounter() {
  setState(() {
    _count--;
  });
}

void _resetCounter() {
  setState(() {
    _count = 0;
  });
}
```



Hello, Flutter!



Recipe calculator

Tạo mới dự án: `flutter create recipe_calculator`

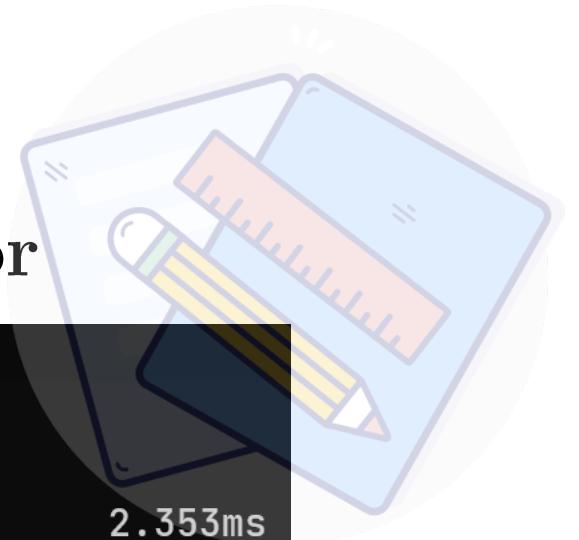
```
PowerShell 7
examples> flutter create recipe_calculator
Creating project recipe_calculator...
Running "flutter pub get" in recipe_calculator...
Wrote 127 files.

All done!
In order to run your application, type:

$ cd recipe_calculator
$ flutter run

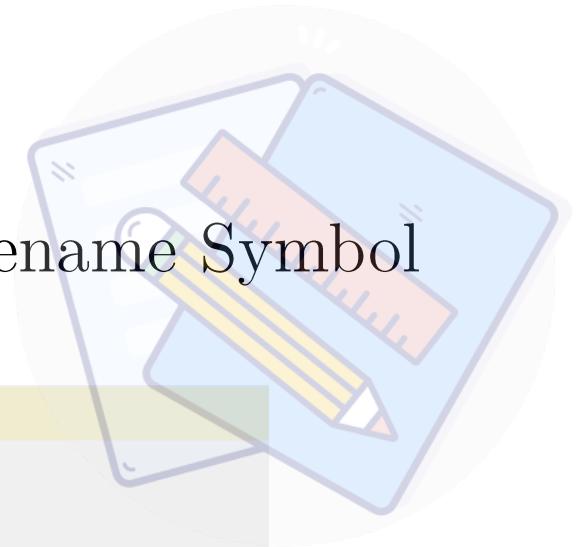
Your application code is in recipe_calculator\lib\main.dart.

examples>
```



Recipe calculator

Đổi tên lớp: nhấp chuột phải vào tên lớp MyApp > Rename Symbol

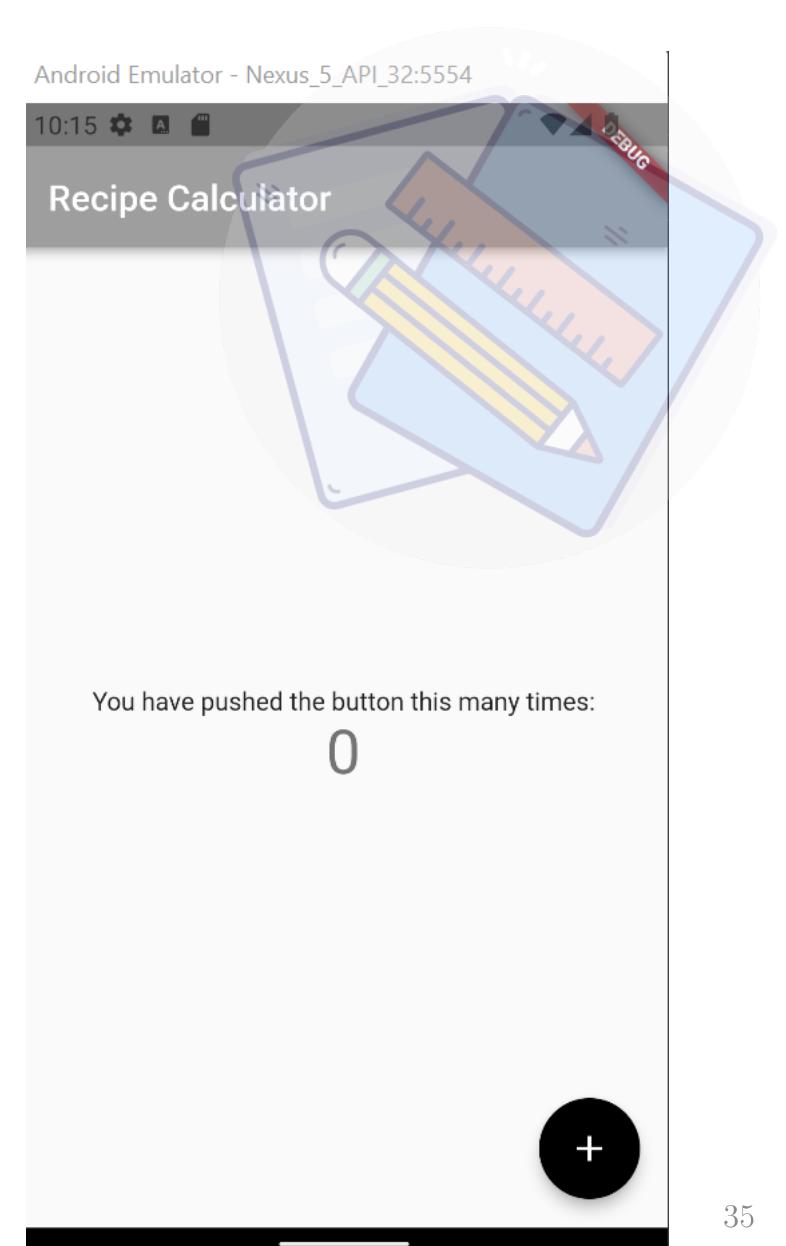


```
7 class MyApp extends StatelessWidget {  
8     const RecipeApp({  
9         Key key,  
10    }) : super(key: key);  
11  
12    @override  
13    Widget build(BuildContext context) {  
14        return MaterialApp(  
15            title: 'Flutter Demo',  
16            theme: ThemeData(  
17                primarySwatch: Colors.blue,  
18            ), // ThemeData  
19            home: const MyHomePage(title: 'Flutter Demo Home Page'),  
20        ); // MaterialApp
```

Recipe calculator

Thay đổi theme cho ứng dụng

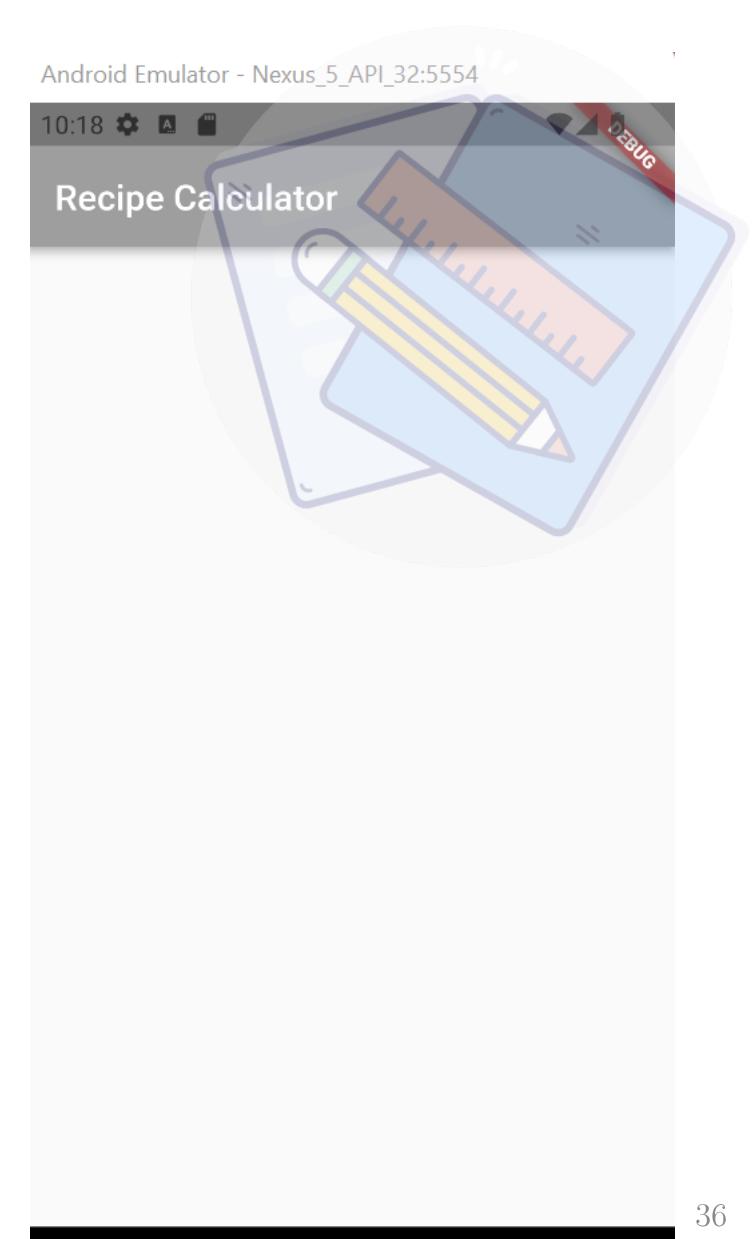
```
10  @override
11  Widget build(BuildContext context) {
12    final ThemeData theme = ThemeData();
13    return MaterialApp(
14      title: 'Recipe Calculator',
15      theme: theme.copyWith(
16        colorScheme: theme.colorScheme.copyWith(
17          primary: Colors.grey,
18          secondary: Colors.black,
19        ),
20      ),
21      home: const MyHomePage(title: 'Recipe Calculator'),
22    ); // MaterialApp
23  }
```



Recipe calculator

Chuẩn bị cho ứng dụng mới

```
35 class _MyHomePageState extends State<MyHomePage> {
36   @override
37   Widget build(BuildContext context) {
38     return Scaffold(
39       appBar: AppBar(
40         title: Text(widget.title),
41       ), // AppBar
42       body: Container(),
43     ); // Scaffold
44   }
45 }
```

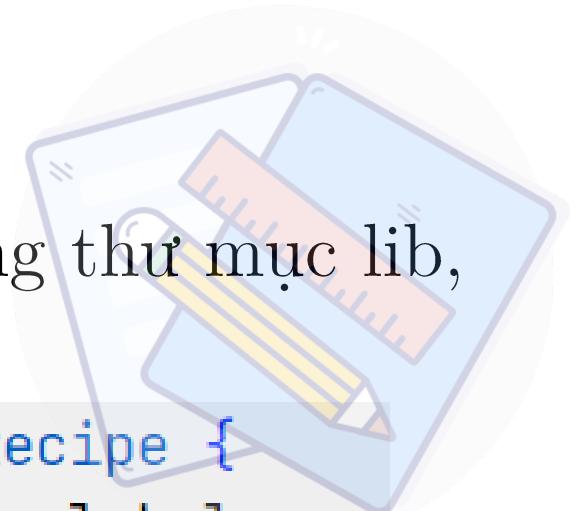


Recipe calculator

Định nghĩa lớp mô hình dữ liệu (data model): trong thư mục lib, tạo tập tin **recipe.dart**



```
1 class Recipe {  
2     String label;  
3     String imageUrl;  
4  
5     Recipe(  
6         this.label,  
7         this.imageUrl,  
8     );  
9 }
```



Recipe calculator

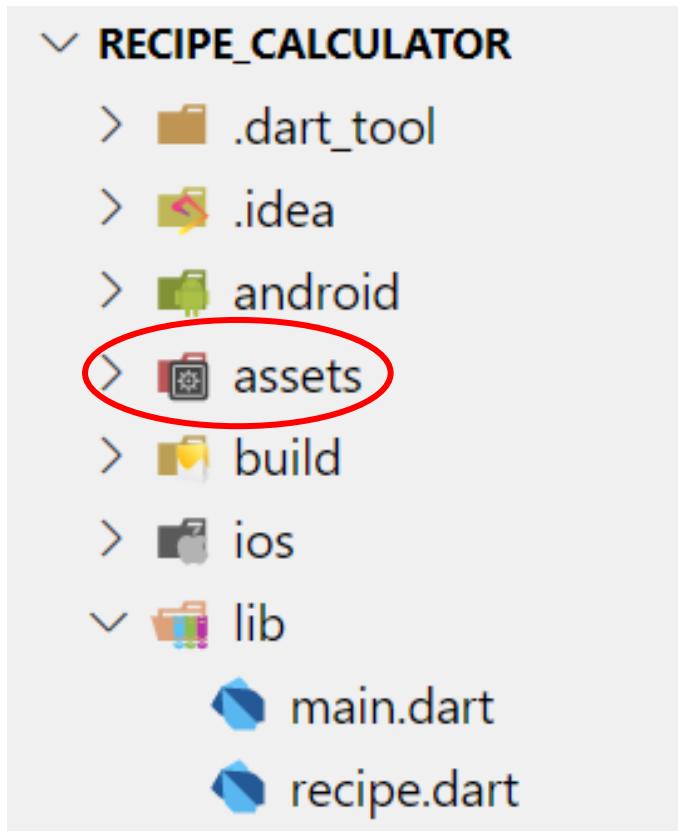
Thêm dữ liệu ví dụ vào **recipe.dart**

```
class Recipe {  
  ...  
  static List<Recipe> samples = [  
    Recipe(  
      'Spaghetti and Meatballs',  
      'assets/2126711929_ef763de2b3_w.jpg',  
    ),  
    Recipe(  
      'Tomato Soup',  
      'assets/27729023535_a57606c1be.jpg',  
    ),  
    Recipe(  
      'Grilled Cheese',  
      'assets/3187380632_5056654a19_b.jpg',  
    ),  
    Recipe(  
      'Chocolate Chip Cookies',  
      'assets/15992102771_b92f4cc00a_b.jpg',  
    ),  
    Recipe(  
      'Taco Salad',  
      'assets/8533381643_a31a99e8a6_c.jpg',  
    ),  
    Recipe(  
      'Hawaiian Pizza',  
      'assets/15452035777_294cefced5_c.jpg',  
    ),  
  ];  
}
```

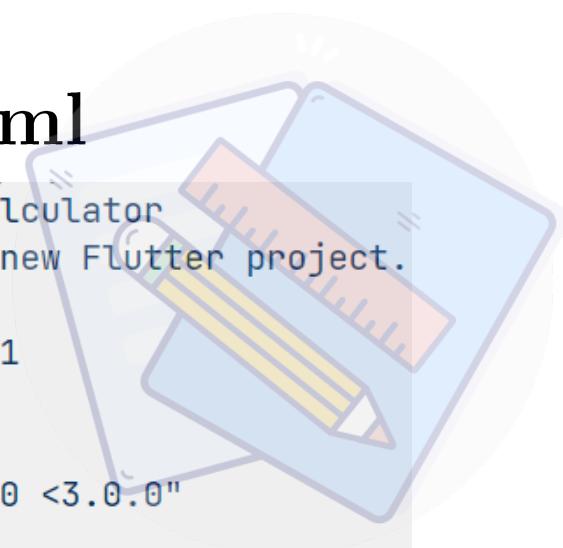


Recipe calculator

Thêm các tập tin asset vào dự án



pubspec.yaml



```
1 name: recipe_calculator
2 description: A new Flutter project.
3
4 version: 1.0.0+1
5
6 environment:
7   sdk: "≥2.17.0 <3.0.0"
8
9 dependencies:
10  flutter:
11    sdk: flutter
12    cupertino_icons: ^1.0.2
13
14 dev_dependencies:
15  flutter_test:
16    sdk: flutter
17    flutter_lints: ^2.0.0
18
19 flutter:
20   uses-material-design: true
21   assets:
22     - assets/
```

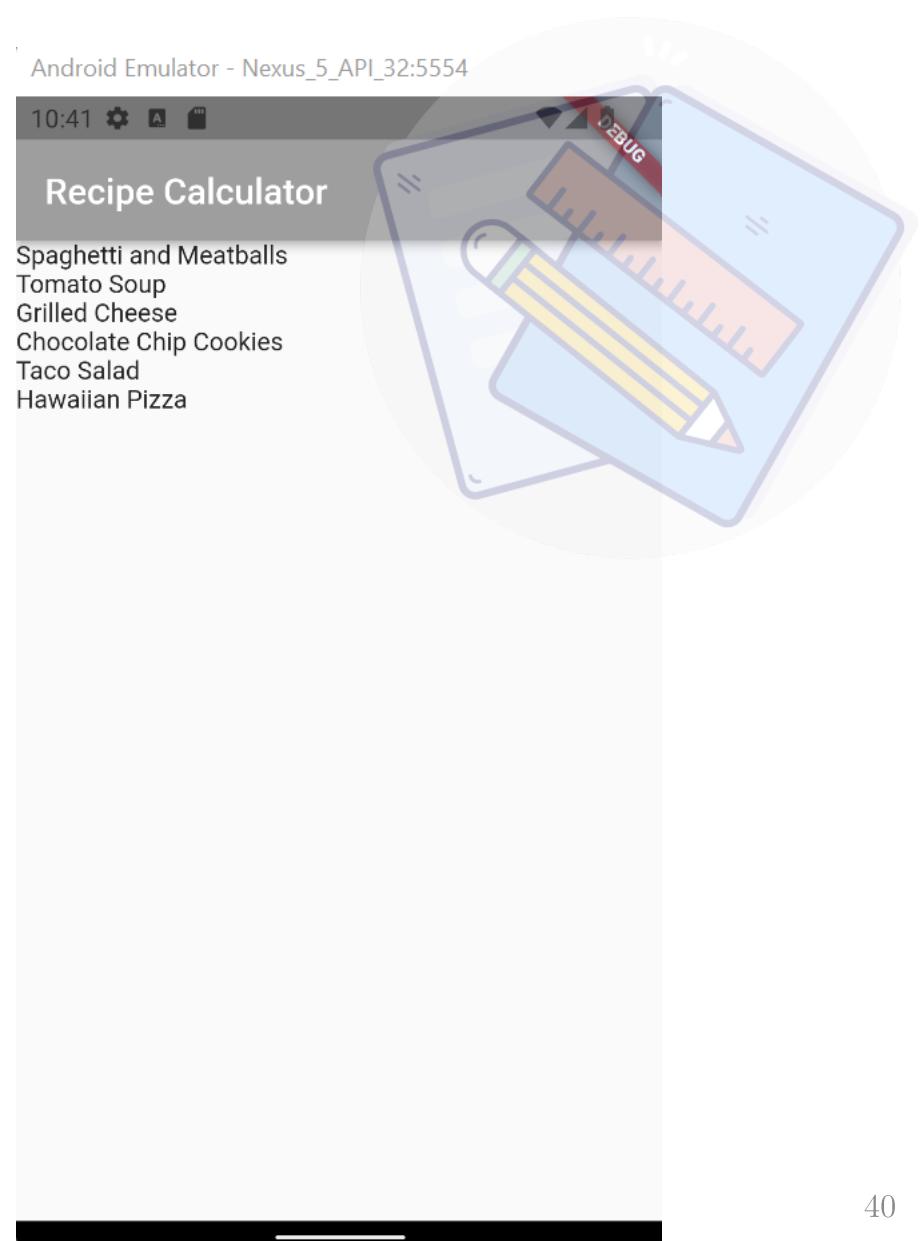
Recipe calculator

Hiển thị dạng danh sách

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

void main() {
  runApp(const RecipeApp());
}

...
class _MyHomePageState extends State<MyHomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      ...
      body: ListView.builder(
        itemCount: Recipe.samples.length,
        itemBuilder: (BuildContext context, int index) {
          return Text(Recipe.samples[index].label);
        },
      ),
    );
  }
}
```



Recipe calculator



ListView: danh sách cuộn (scrollable) các widget

- **ListView.builder**: phương thức xây dựng các phần tử danh sách theo yêu cầu (on demand)
 - **itemCount**: cho biết số lượng các phần tử trong danh sách
 - **itemBuilder**: hàm trả về một widget là phần tử con trong danh sách

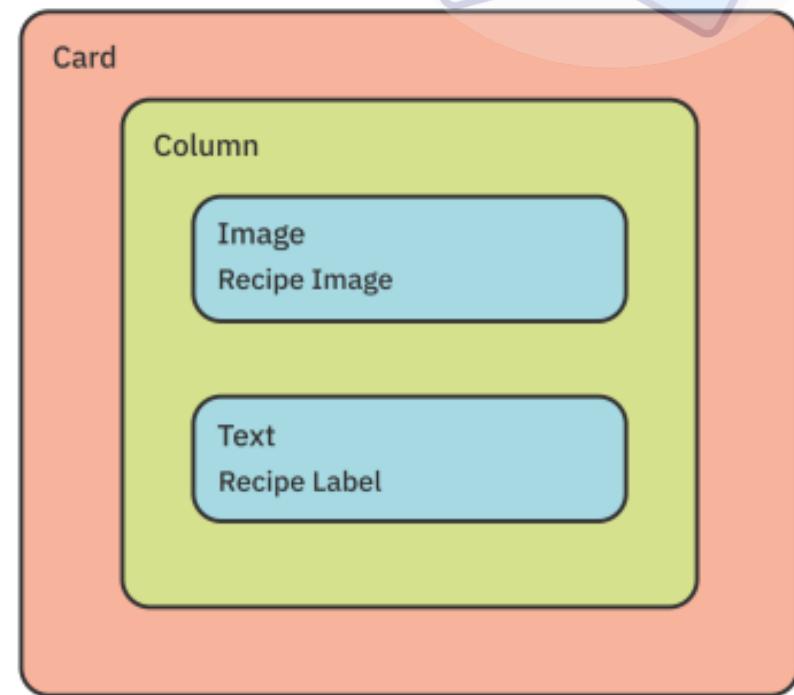
```
child: ListView.builder(  
    itemCount: Recipe.samples.length,  
    itemBuilder: (BuildContext context, int index) {  
        return Text(Recipe.samples[index].label);  
    },  
) // ListView.builder
```

Recipe calculator

Hiển thị phần tử danh sách trong Card

- **Card**: vùng UI hiển thị một nhóm các thông tin liên quan

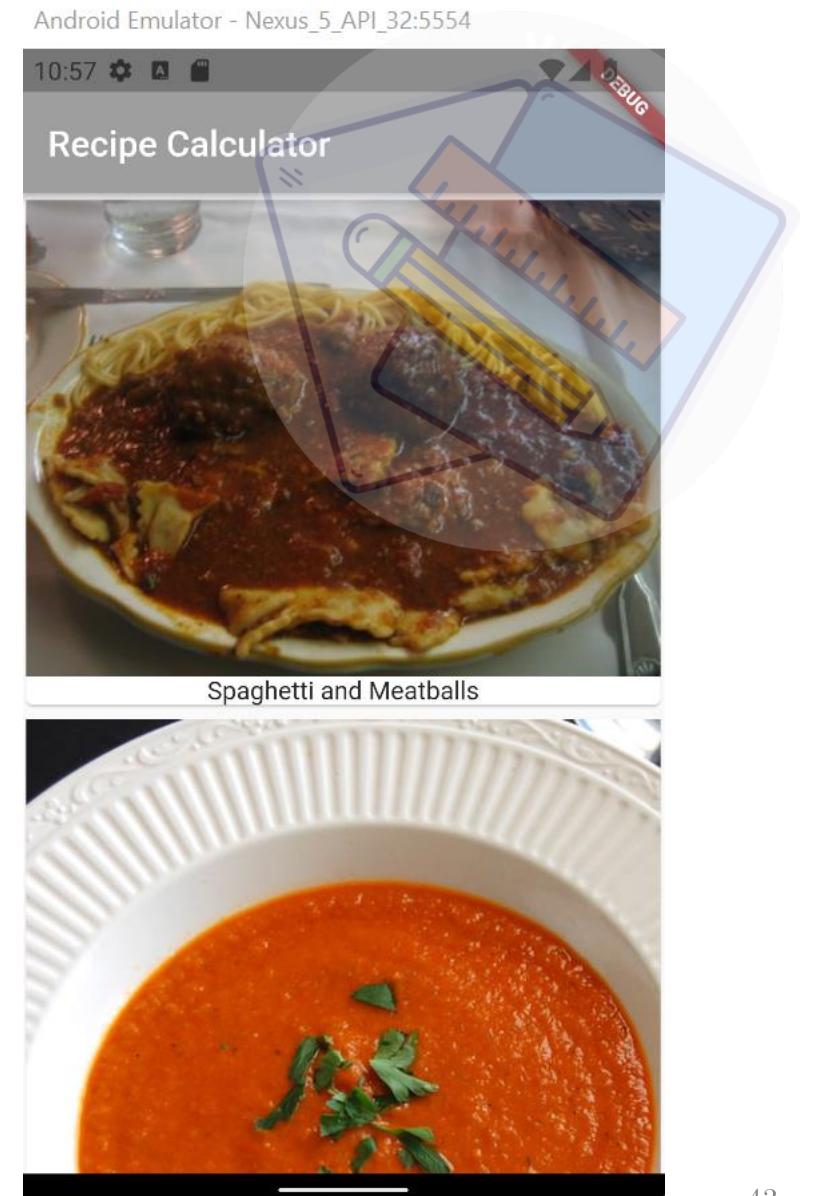
```
36 class _MyHomePageState extends State<MyHomePage> {  
37     Widget buildRecipeCard(Recipe recipe) {  
38         return Card(  
39             child: Column(  
40                 children: <Widget>[  
41                     Image(image: AssetImage(recipe.imageUrl)),  
42                     Text(recipe.label),  
43                 ], // <Widget>[]  
44             ), // Column  
45         ); // Card  
46     }  
47  
48     @override  
49     Widget build(BuildContext context) {
```



Recipe calculator

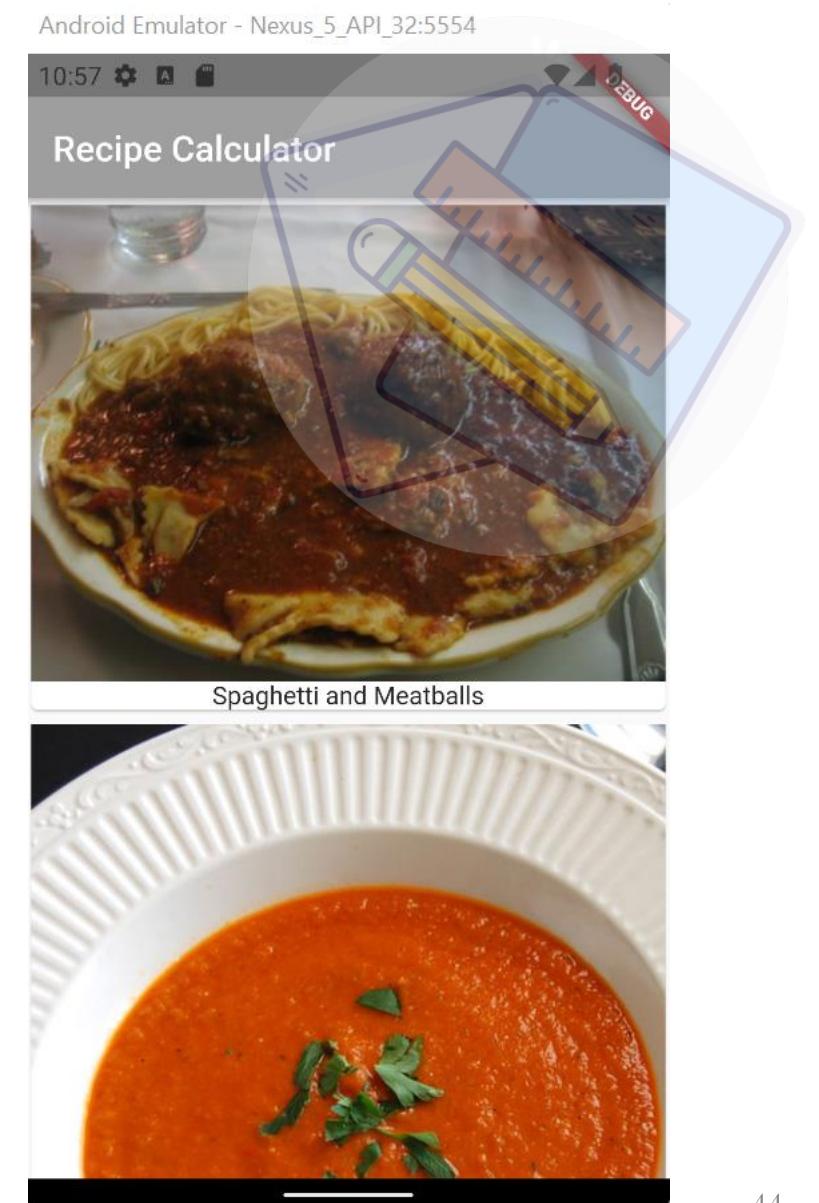
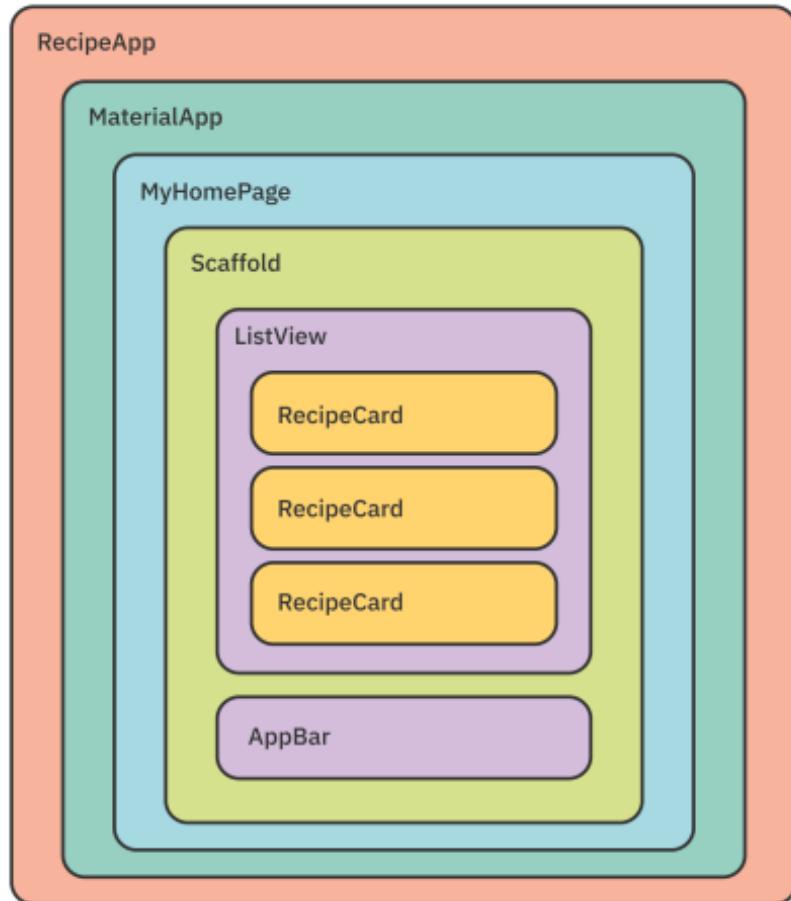
Hiển thị phần tử danh sách trong Card

```
child: ListView.builder(  
    itemCount: Recipe.samples.length,  
    itemBuilder: (BuildContext context, int index) {  
        return buildRecipeCard(Recipe.samples[index]);  
    },  
, // ListView.builder
```



Recipe calculator

Xem cây widget của ứng dụng



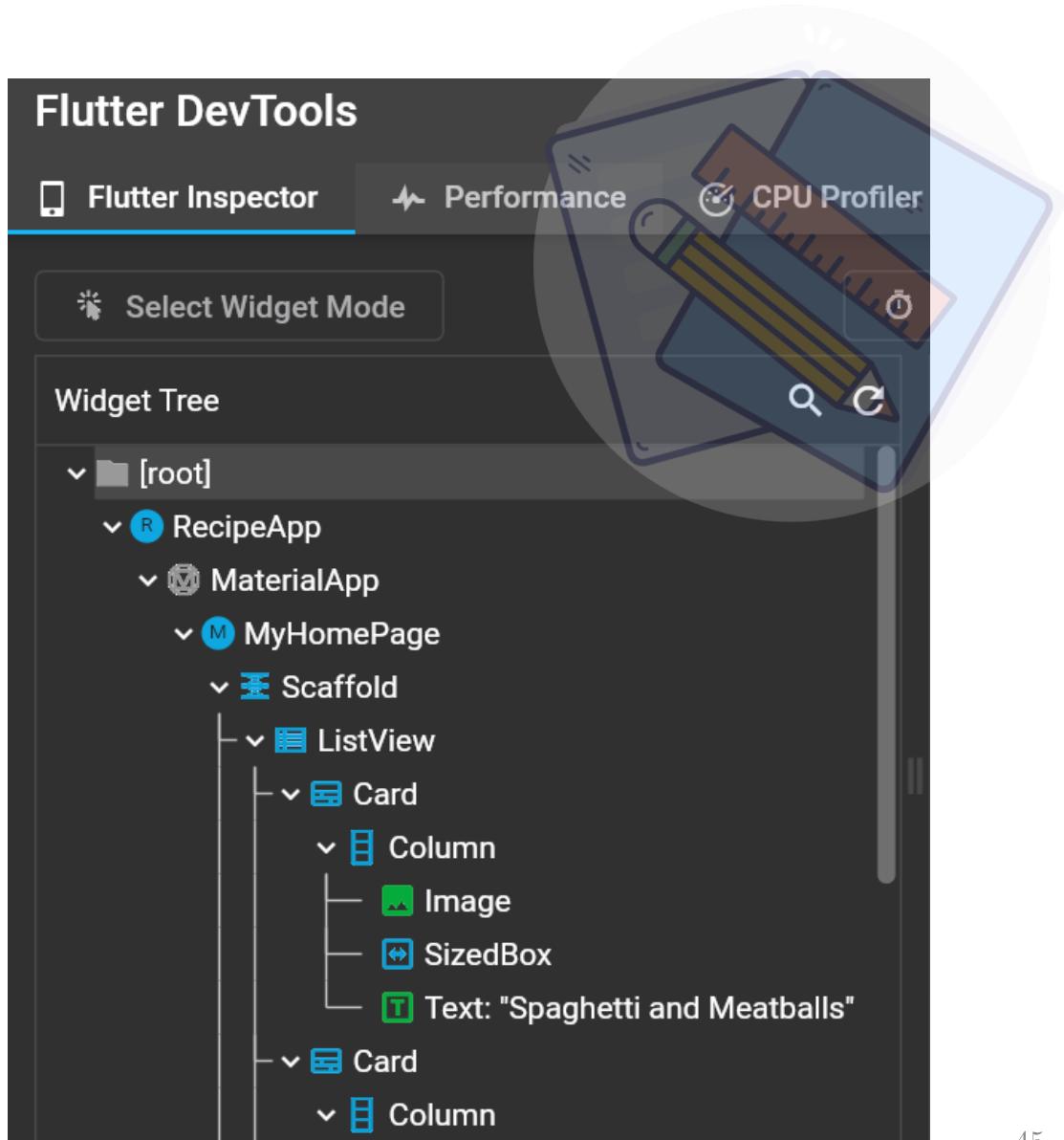
Recipe calculator

Mở DevTools:

View > Command Palette...

> Flutter: Open DevTools

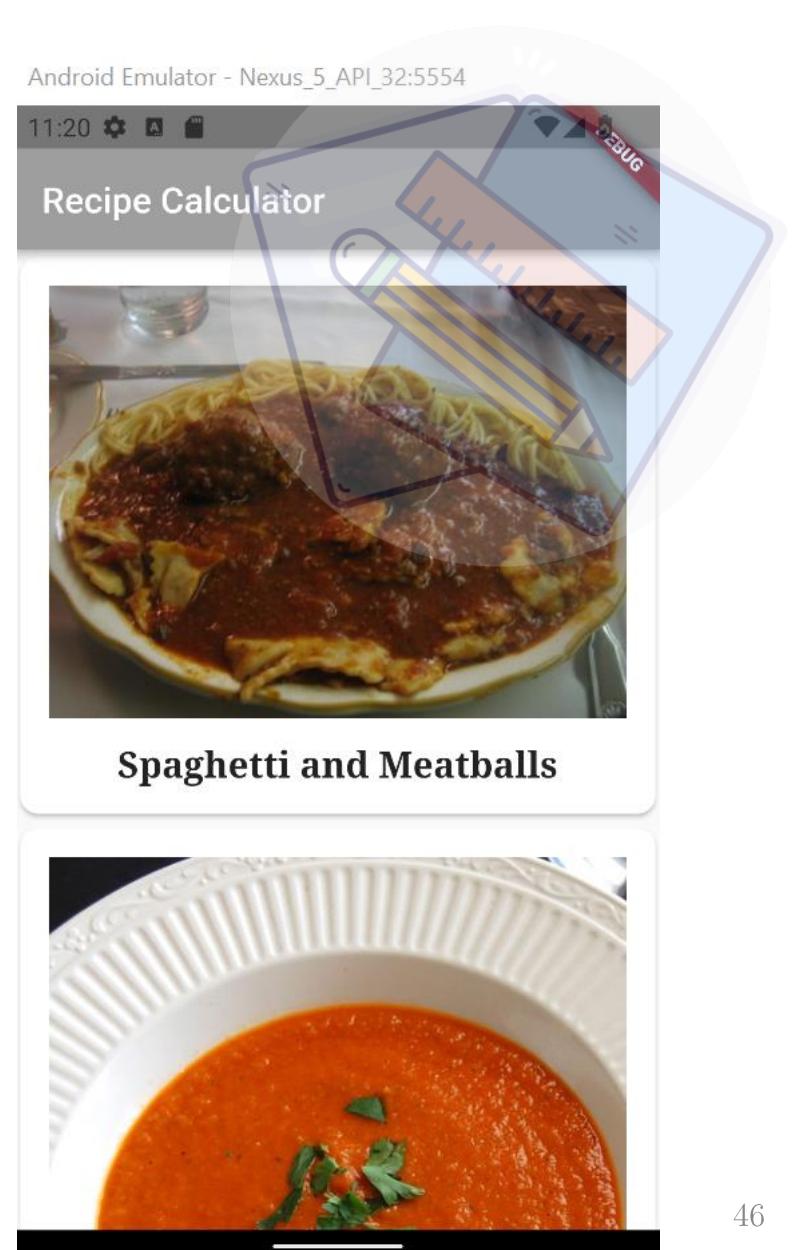
> Open DevTools in Web Browser



Recipe calculator

Tùy chỉnh định dạng của Card

```
Widget buildRecipeCard(Recipe recipe) {  
    return Card(  
        elevation: 2.0,  
        shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(10.0)),  
        child: Padding(  
            padding: const EdgeInsets.all(16.0),  
            child: Column(  
                children: <Widget>[  
                    Image(image: AssetImage(recipe.imageUrl)),  
                    const SizedBox(height: 14.0),  
                    Text(  
                        recipe.label,  
                        style: const TextStyle(  
                            fontSize: 20.0,  
                            fontWeight: FontWeight.w700,  
                            fontFamily: 'Palatino',  
                        ), // TextStyle  
                    ), // Text  
                ], // <Widget>[]  
            ), // Column  
        ), // Padding  
    ); // Card  
}
```



Recipe calculator



Tùy chỉnh định dạng của Card:

- **elevation**: độ cao, ảnh hưởng đến bóng đổ (shadow)
- **shape**: hình dạng của Card

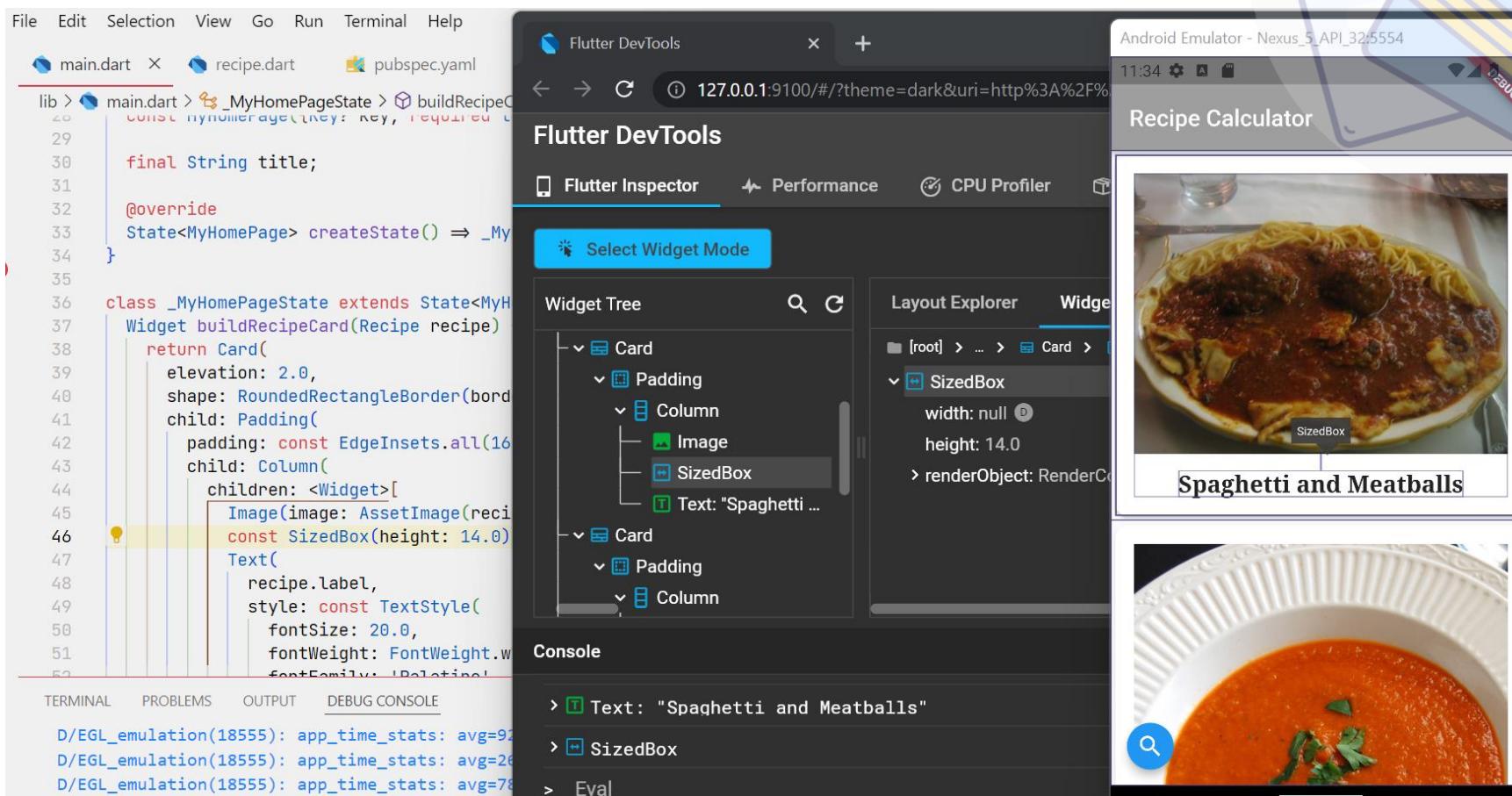
Padding: widget dùng để tạo padding cho widget con của nó

- **EdgeInsets.symmetric**: đặt giá trị offset đối xứng (trái/phải, trên/dưới)
- **EdgeInsets.all**: đặt một giá trị offset cho cả bốn cạnh
- **EdgeInsets.only**: đặt giá trị offset cho một hoặc một số cạnh

SizedBox: tạo khoảng không gian có kích thước cố định

Recipe calculator

Xem các thông số của các widget qua DevTools



Recipe calculator

Thêm trang chi tiết công thức nấu ăn

Nhấp chuột phải vào **buildRecipeCard** > Refactor...
> Wrap with widget... > GestureDetector



```
body: ListView.builder(  
    itemCount: Recipe.samples.length,  
    itemBuilder: (BuildContext context, int index) {  
        return buildRecipeCard(Recipe.samples[index]);  
    },  
, // ListView.bu      Wrap with widget...  
// Scaffold           Wrap with Builder
```

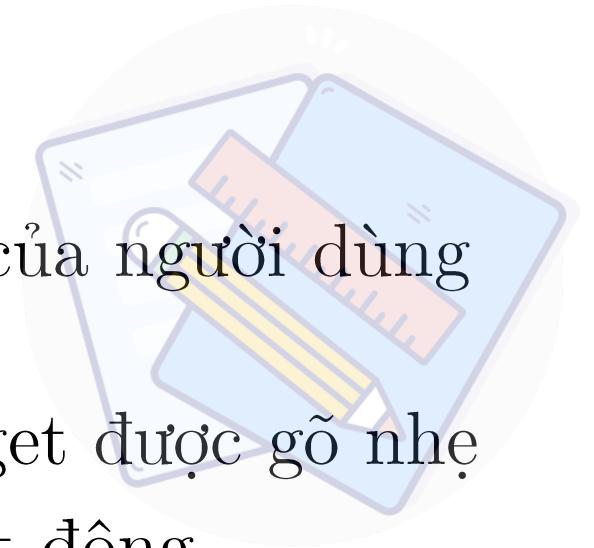
Recipe calculator

Thêm trang chi tiết công thức nấu ăn

```
itemBuilder: (BuildContext context, int index) {  
  return GestureDetector(  
    onTap: () {  
      Navigator.push(  
        context,  
        MaterialPageRoute(  
          builder: (context) {  
            return const Text('Detail page');  
          },  
        ), // MaterialPageRoute  
      );  
    },  
    child: buildRecipeCard(Recipe.samples[index]),  
  ); // GestureDetector  
},
```



Recipe calculator



GestureDetector: widget nhận dạng các cử chỉ của người dùng và gọi hàm callback tương ứng

- **onTap**: định nghĩa callback sẽ được gọi khi widget được gõ nhẹ
- **child**: vùng UI mà việc nhận dạng cử chỉ sẽ hoạt động

Navigator: widget quản lý một ngăn xếp các trang

- Gọi hàm **push** với một **MaterialPageRoute** sẽ đẩy một trang material vào ngăn xếp
- **builder** của MaterialPageRoute tạo widget trang đích đến

Recipe calculator

Thêm trang chi tiết công thức nấu ăn

Tạo tập tin **recipe_detail_screen.dart** trong thư mục lib/

```
1 import 'package:flutter/material.dart';
2 import 'recipe.dart';
3
4 class RecipeDetailScreen extends StatefulWidget {
5   final Recipe recipe;
6
7   const RecipeDetailScreen({Key? key, required this.recipe}) : super(key: key);
8
9   @override
10  State<RecipeDetailScreen> createState() => _RecipeDetailScreenState();
11 }
```

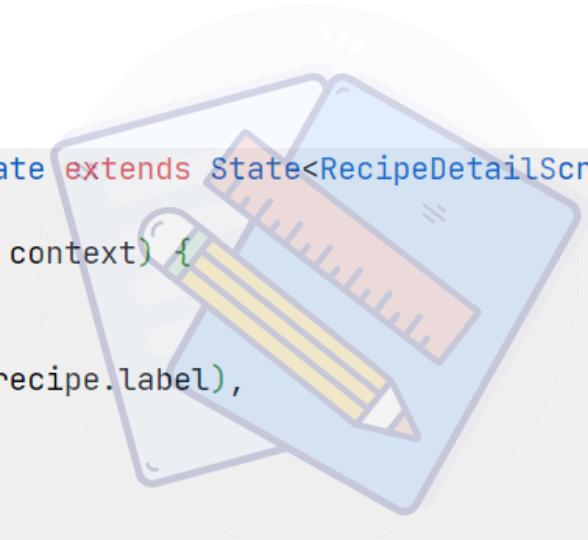


Recipe calculator

Thêm trang chi tiết công thức nấu ăn

Tạo tập tin
recipe_detail_screen.dart
trong thư mục lib/

```
13 class _RecipeDetailScreenState extends State<RecipeDetailScreen> {  
14   @override  
15   Widget build(BuildContext context) {  
16     return Scaffold(  
17       appBar: AppBar(  
18         title: Text(widget.recipe.label),  
19       ), // AppBar  
20       body: Column(  
21         children: <Widget>[  
22           SizedBox(  
23             height: 300,  
24             width: double.infinity,  
25             child: Image(image: AssetImage(widget.recipe.imageUrl)),  
26           ), // SizedBox  
27           const SizedBox(height: 4),  
28           Text(  
29             widget.recipe.label,  
30             style: const TextStyle(fontSize: 18),  
31           ), // Text  
32           ], // <Widget>[]  
33         ), // Column  
34       ); // Scaffold  
35     }  
36 }
```

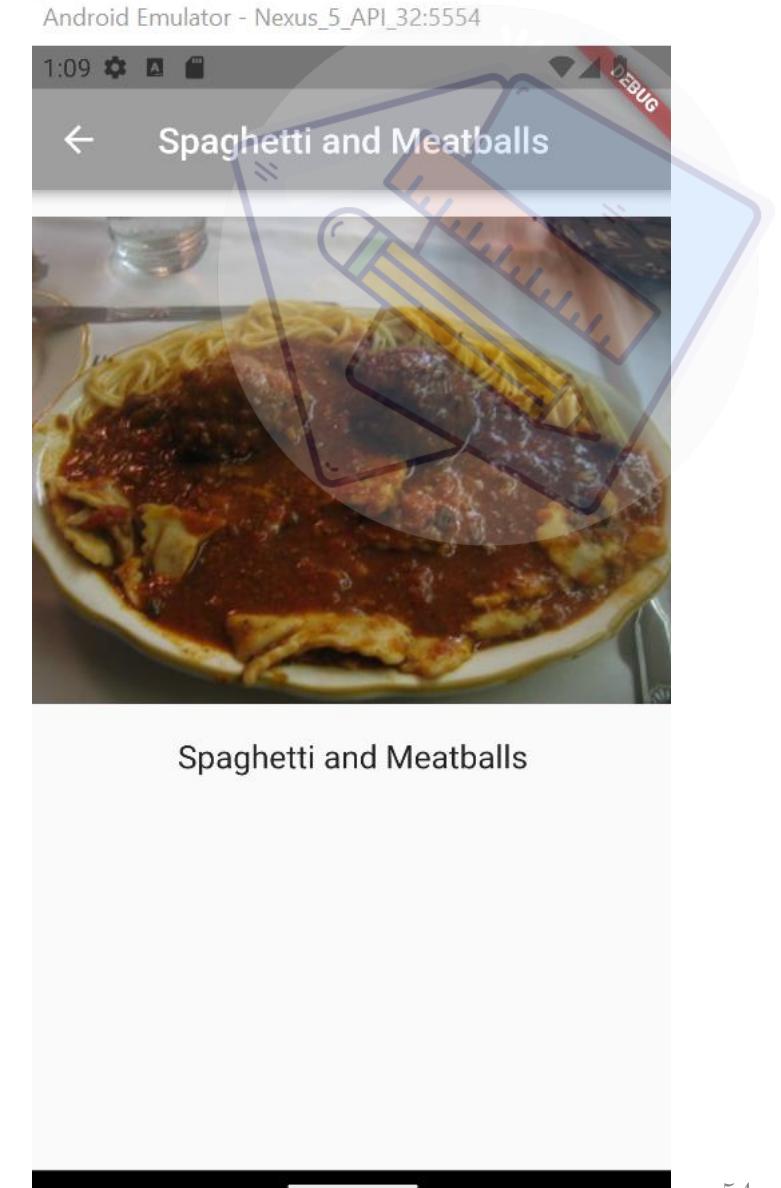


Recipe calculator

Thêm trang chi tiết công thức nấu ăn

Import **recipe_detail_screen.dart** vào **main.dart**

```
return GestureDetector(
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return RecipeDetailScreen(recipe: Recipe.samples[index]);
        },
      ), // MaterialPageRoute
    );
  },
  child: buildRecipeCard(Recipe.samples[index]),
); // GestureDetector
```

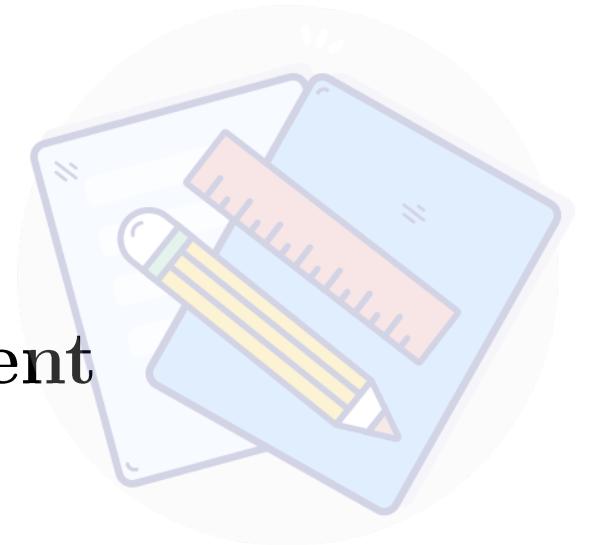


Recipe calculator

Thêm thông tin nguyên liệu

Mở tập tin **recipe.dart** và thêm vào lớp **Ingredient**

```
class Ingredient {  
    double quantity;  
    String measure;  
    String name;  
  
    Ingredient(  
        this.quantity,  
        this.measure,  
        this.name,  
    );  
}
```



Recipe calculator

Thêm thông tin nguyên liệu
Hiệu chỉnh lớp **Recipe**

```
class Recipe {  
    String label;  
    String imageUrl;  
    int servings;  
    List<Ingredient> ingredients;  
  
    Recipe(  
        this.label,  
        this.imageUrl,  
        this.servings,  
        this.ingredients,  
    );
```



Recipe calculator

Thêm thông tin nguyên liệu
Hiệu chỉnh dữ liệu ví dụ

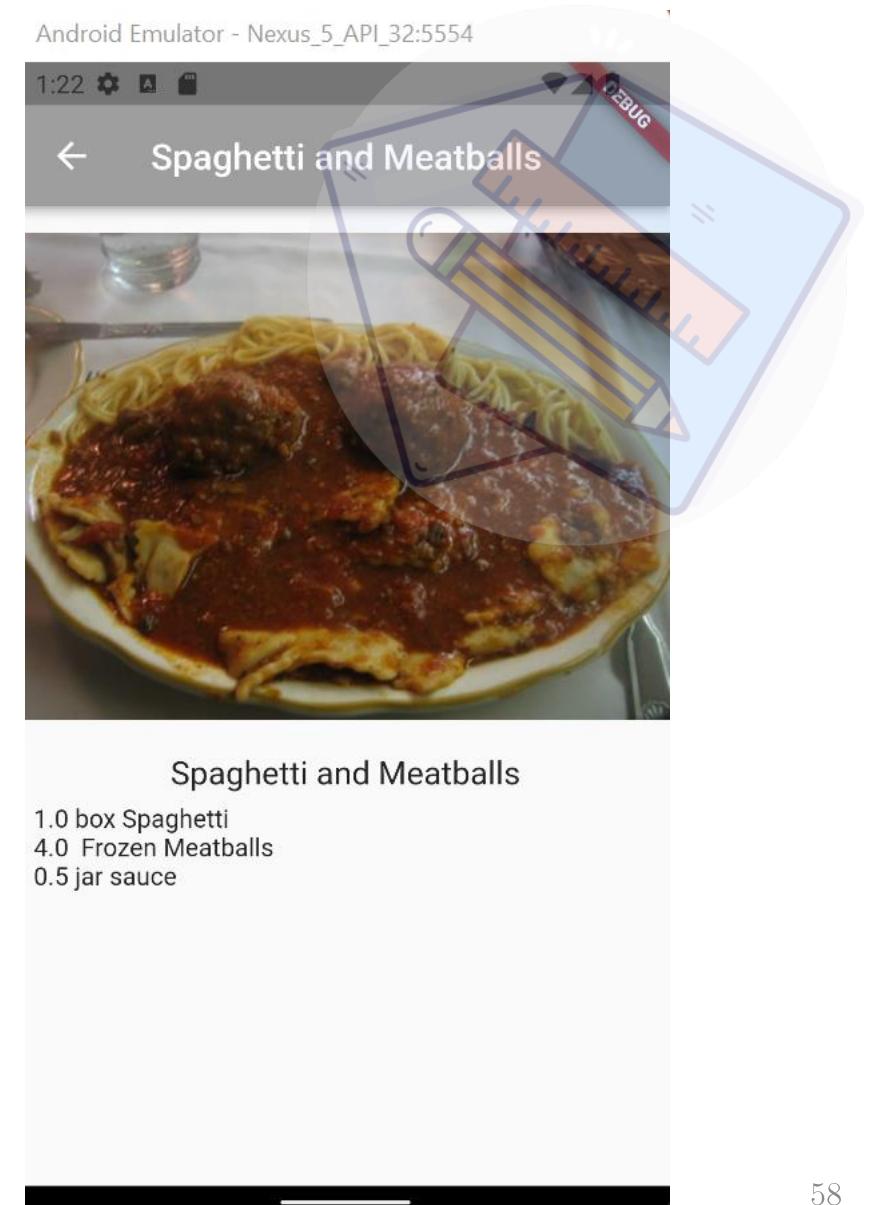
```
static List<Recipe> samples = [
    Recipe('Spaghetti and Meatballs', 'assets/2126711929_ef763de2b3_w.jpg', 4, [
        Ingredient(1, 'box', 'Spaghetti'),
        Ingredient(4, '', 'Frozen Meatballs'),
        Ingredient(0.5, 'jar', 'sauce'),
    ]), // Recipe
    Recipe('Tomato Soup', 'assets/27729023535_a57606c1be.jpg', 2, [
        Ingredient(1, 'can', 'Tomato Soup'),
    ]), // Recipe
    Recipe('Grilled Cheese', 'assets/3187380632_5056654a19_b.jpg', 1, [
        Ingredient(2, 'slices', 'Cheese'),
        Ingredient(2, 'slices', 'Bread'),
    ]), // Recipe
    Recipe(
        'Chocolate Chip Cookies', 'assets/15992102771_b92f4cc00a_b.jpg', 24, [
            Ingredient(4, 'cups', 'flour'),
            Ingredient(2, 'cups', 'sugar'),
            Ingredient(0.5, 'cups', 'chocolate chips'),
        ]), // Recipe
    Recipe('Taco Salad', 'assets/8533381643_a31a99e8a6_c.jpg', 1, [
        Ingredient(4, 'oz', 'nachos'),
        Ingredient(3, 'oz', 'taco meat'),
        Ingredient(0.5, 'cup', 'cheese'),
        Ingredient(0.25, 'cup', 'chopped tomatoes'),
    ]), // Recipe
    Recipe('Hawaiian Pizza', 'assets/15452035777_294cefced5_c.jpg', 4, [
        Ingredient(1, 'item', 'pizza'),
        Ingredient(1, 'cup', 'pineapple'),
        Ingredient(8, 'oz', 'ham'),
    ]), // Recipe
];
```



Recipe calculator

Hiển thị thông tin nguyên liệu

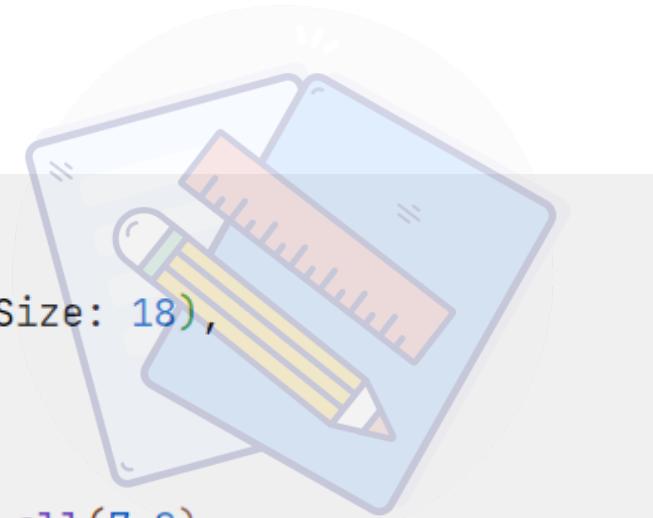
```
Text(  
  widget.recipe.label,  
  style: const TextStyle(fontSize: 18),  
, // Text  
Expanded(  
  child: ListView.builder(  
    padding: const EdgeInsets.all(7.0),  
    itemCount: widget.recipe.ingredients.length,  
    itemBuilder: (BuildContext context, int index) {  
      final ingredient = widget.recipe.ingredients[index];  
      return Text(  
        '${ingredient.quantity} '  
        '${ingredient.measure} '  
        '${ingredient.name}',  
      ); // Text  
    },  
, // ListView.builder  
, // Expanded
```



Recipe calculator

Expanded: widget bắt buộc widget con lắp đầy khoảng trống trong Column, Row hoặc Flex

```
Text(  
  widget.recipe.label,  
  style: const TextStyle(fontSize: 18),  
) // Text  
Expanded(  
  child: ListView.builder(  
    padding: const EdgeInsets.all(7.0),  
    itemCount: widget.recipe.ingredients.length,  
    itemBuilder: (BuildContext context, int index) {  
      final ingredient = widget.recipe.ingredients[index];  
      return Text(  
        '${ingredient.quantity} '  
        '${ingredient.measure} '  
        '${ingredient.name}',  
      ); // Text  
    },  
) // ListView.builder  
) // Expanded
```

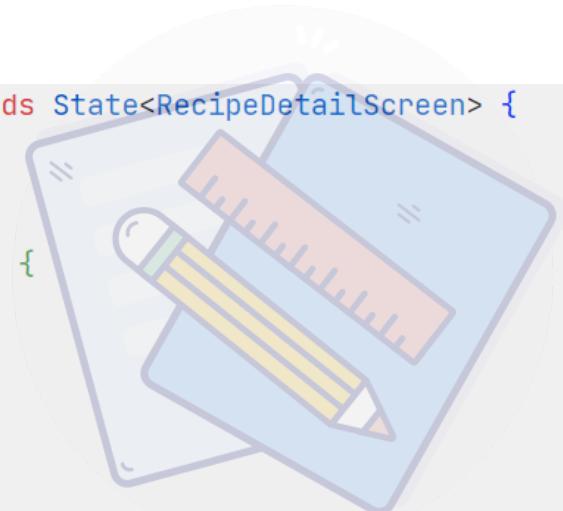


Recipe calculator

Thêm thanh trượt Slider
thay đổi số người phục vụ

```
class _RecipeDetailScreenState extends State<RecipeDetailScreen> {
  int _sliderVal = 1;

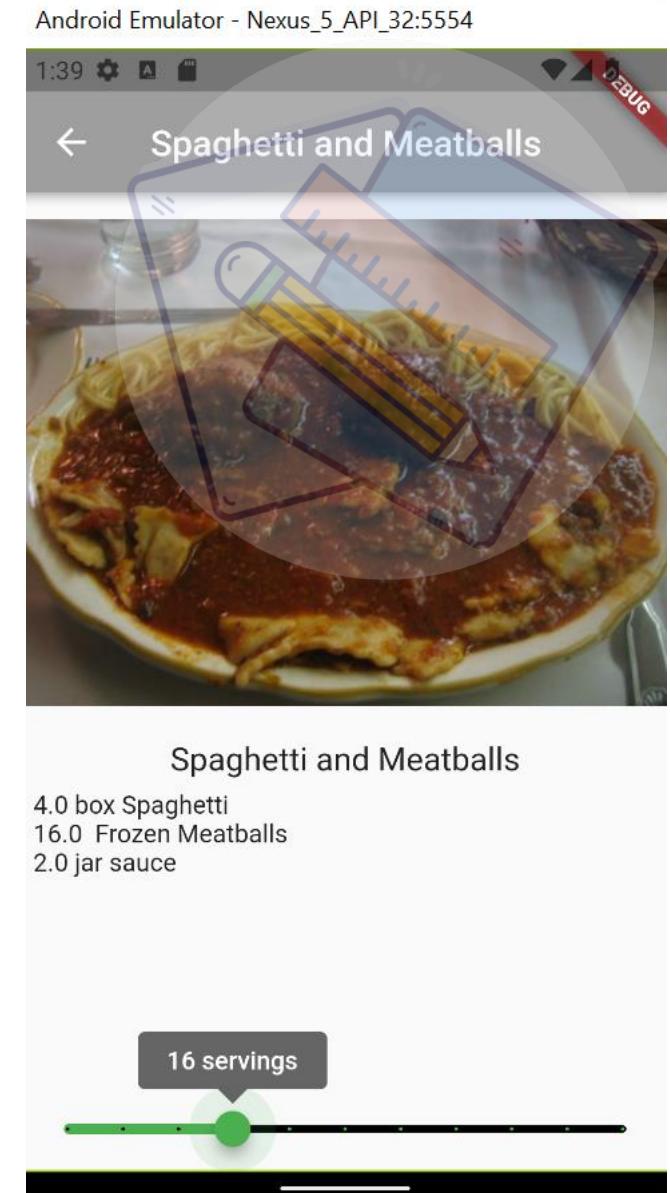
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      ...
      body: Column(
        children: <Widget>[
          ...
          Slider(
            min: 1, max: 10, divisions: 10,
            label: '${_sliderVal * widget.recipe.servings} servings',
            value: _sliderVal.toDouble(),
            onChanged: (newValue) {
              setState(() {
                _sliderVal = newValue.round();
              });
            },
            activeColor: Colors.green,
            inactiveColor: Colors.black,
          ),
        ],
      );
    }
}
```



Recipe calculator

Thêm thanh trượt thay đổi số người phục vụ

```
itemBuilder: (BuildContext context, int index) {  
  final ingredient = widget.recipe.ingredients[index];  
  return Text(  
    '${ingredient.quantity * _sliderVal} '  
    '${ingredient.measure} '  
    '${ingredient.name}',  
  ); // Text  
},
```



Câu hỏi?

