**BÁO CÁO NỘI DUNG HỌC TẬP**

**NIÊN LUẬN NGÀNH**

**FLUTTER AND DART**

**NGUYỄN THỊ CẨM TIÊN**

**B2004757**

**NĂM 2024**

**MỤC LỤC**

[Section 1: Introduction 1](#_Toc22052)

[1. Flutter là gì? 1](#_Toc26656)

[2. Dart là gì? 1](#_Toc24194)

[3. Flutter setup 1](#_Toc651)

[4. Tạo dự án và setting up code editor 1](#_Toc7248)

[5. Run dự án flutter đầu tiên 1](#_Toc11025)

[Section 2: Flutter and Dart Basic I: Getting a solid foundation - roll dive app 1](#_Toc24972)

[1. Phân tích cấu trúc file của một dự án flutter 1](#_Toc28947)

[2. Từ Dart đến mã máy 2](#_Toc22532)

[3. Một số khái niệm cơ bản 2](#_Toc5970)

[4. Phân chia giao diện thành các file nhỏ 3](#_Toc7929)

[5. Variable 5](#_Toc4480)

[6. final and const 5](#_Toc20195)

[7. Hiển thị hình ảnh 5](#_Toc900)

[8. Multiple constructor function 6](#_Toc23603)

[9. Column and Row Widget 7](#_Toc30628)

[10. Button 7](#_Toc22854)

[11. StatelessWidget and StatefulWidget 7](#_Toc27440)

[12. State và setState() 7](#_Toc22768)

[Section 3: Flutter and Dart Basic II: Fundamental deep dive - quiz app 8](#_Toc3433)

[1. Adding icon to button 8](#_Toc32316)

[2. (stateful) widget lifecycle methods: 8](#_Toc10936)

[3. Using "if" Statements In Lists 8](#_Toc27481)

[4. Dự án Quiz app và các khái niệm mới 9](#_Toc20073)

[Section 4: Debugging Flutter Apps 11](#_Toc4304)

[1. Đọc và tìm hiểu lỗi 11](#_Toc14221)

[2. Chạy ứng dụng với debug 11](#_Toc10092)

[3. Devtool 13](#_Toc25016)

[4. Chạy ứng dụng trên thiết bị thật 14](#_Toc12938)

[Section 5: add interactivity , more Widget and theming 15](#_Toc10403)

[1. Tạo dự án mới - dự án quản lý chi tiêu 15](#_Toc5535)

[2. Sử dụng dịch vụ từ bên thứ 3 UUID để tạo ra ID duy nhất cho dữ liệu 16](#_Toc1540)

[3. Giới thiệu về Enum 17](#_Toc24376)

[4. Sử dụng ListViewt thay cho Column 17](#_Toc8994)

[5. Sử dụng gói intl để format ngày giờ 18](#_Toc15558)

[Section 6: Building responsive and adaptive user interface 19](#_Toc993)

[Section 7: Flutter and Dart Internal 19](#_Toc26093)

[Section 8: Building multi screen App and navigating between screen 19](#_Toc32607)

# **Section 1: Introduction**

## Flutter là gì?

* là khung giao diện người dùng, một tập các khối mã được viết sẵn hỗ trợ cho việc viết single codebase chạy trên đa nền tảng
* Cung cấp các bộ công cụ để chuyển đổi mã code chạy trên nhiều nền tảng khác nhau

## Dart là gì?

* Dart là một ngôn ngữ lập trình

## Flutter setup

* install Flutter SDK
* install Android Studio
* set up virtual device

## Tạo dự án và setting up code editor

* mở terminal : flutter create (Tên thư mục dự án) : flutter create first\_app
* install Visual Studio code
* mở thư mục dự án với vs code
* cài extension cho vs code: flutter

## Run dự án flutter đầu tiên

* khởi động thiết bị ảo: ctrl + shift + p → flutter: select device → chọn thiết bị
* chạy dự án : flutter run / run without debugging

# **Section 2: Flutter and Dart Basic I: Getting a solid foundation - roll dive app**

## Phân tích cấu trúc file của một dự án flutter

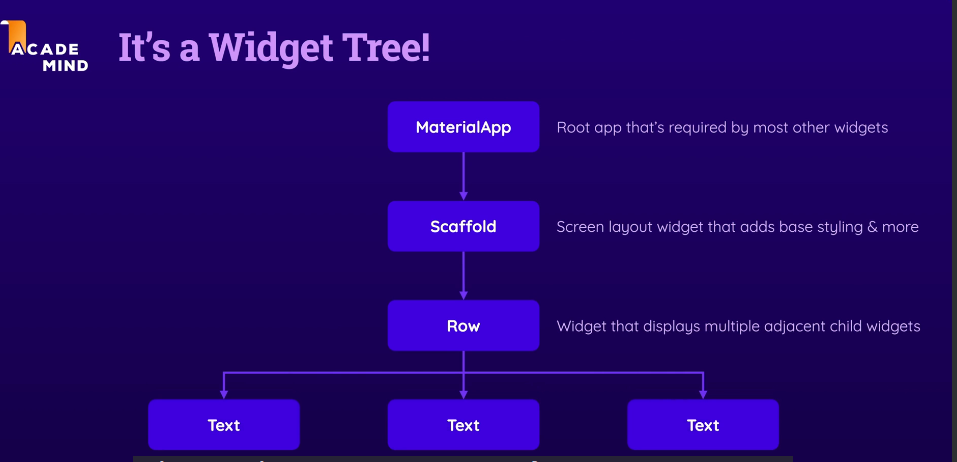
* thư mục lib chứa các file .dart là nơi chứa các mã code chính của ứng dụng
* các thư mục tên các nền tảng: Ios, android, web, macos,... chứa các tệp dành riêng cho các nền tảng khác nhau thông thường không cần thao tác đến
* Thư mục build chứa các tệp tạm thời và tệp đầu ra khi flutter xây dựng ứng dụng cho các nền tảng
* Thư mục test là nơi chứa các đoạn mã để kiểm tra ứng dụng
* các thư mục bắt đầu với dấu . chứa các cấu hình bổ sung
* tệp analyze\_option.yaml: cấu hình hiển thị cảnh báo và lỗi trong mã code kể cả trước khi run app
* tệp Pubspec.yaml: tệp khai báo các gói của bên thứ 3, thêm hình ảnh,..

## Từ Dart đến mã máy

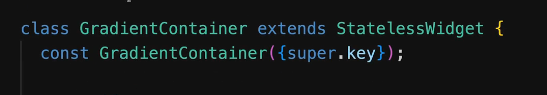
* Các đoạn code chương trình viết bằng ngôn ngữ Dart các mã này được phân tích cú pháp sau đó được biên dịch thành ngôn ngữ native trên nền tảng muốn triển khai → mã biên dịch này được thực thi trên máy mục tiêu

## Một số khái niệm cơ bản

* hàm void runApp(Widget) : được cung cấp bởi flutter , nhiệm vụ run code và hiển thị giao diện người dùng lên màn hình
* thêm các gói được xây dựng sẵn bằng từ khóa import
* void main() là một hàm đặc biệt trong dart được tự động khởi chạy khi run app
* Widget: là thành phần cơ bản của flutter, kết hợp nhiều Widget tạo thành cây Widget cũng chính là Flutter UI, có nhiều Widget được xây dựng sẵn để thực hiện một số công việc cơ bản như hiển thị : text, image, button,...

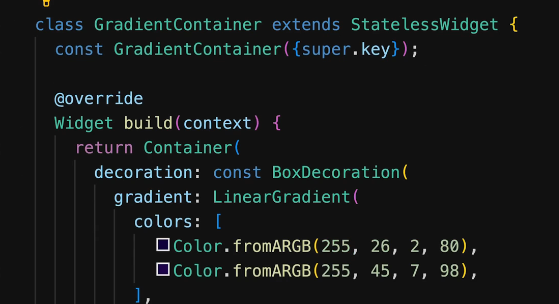


* Position Argument và name Argument:
* position int add(a,b) → add(1,2)
* name argument add({num1, num2}) → add(num1: 1, num2:2) or add(num2:2, num1:1)
* Const keyword: Giúp tối ưu hóa hiệu năng: khi được khai báo với từ khóa const các giá trị sẽ được lưu trong bộ nhớ của thiết bị chạy ứng dụng , khi sử dụng lại lần thứ 2 giá trị đó sẽ được lấy ra từ bộ nhớ thay vì phải tạo ra đối tượng mới → điều này giúp làm giảm sự trùng lặp trong bộ nhớ.
* Kiểu dữ liệu : int, string,widget..... Một giá trị có thể thuộc nhiều kiểu dữ liêu:
* 29 → int, num, object
* “Hello world” → String, object
* Object Color:
* Colors.màu : Colors.white,....
* Color.fromARGB
* List : một danh sách nhiều giá trị cùng loại , vd: 1 danh sách số nguyên, …
* List<kiểu>
* Tất cả các giá trị trong Dart đều có kiểu cuối cùng là kiểu Object
* Class:
* giống như blueprint của dữ liệu
* từ một lớp có thể sử dụng phương thức xây dựng để tạo ra đối tượng
* tạo ra 1 lớp: class ClassName{}
* lớp StatelessWidget lớp giao diện dành cho những thành phần giao diện cố định không thay đổi khi dữ liệu thay đổi→ có phương thức Widget build( context ) cần override
* Hàm xây dựng:



## Phân chia giao diện thành các file nhỏ

* Việc phân chia giao diện thành các file riêng biệt với mục đích tái sử dụng hoặc chỉ đơn giản là dễ quản lý
* Trong các file các phần giao diện được viết trong các class

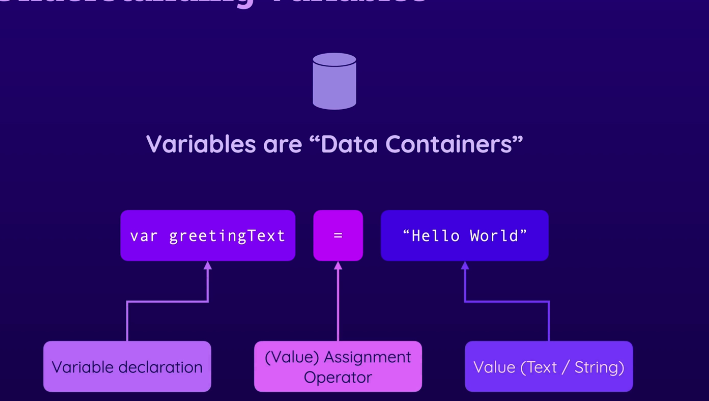


* để sử dụng trong các file khác sử dụng từ khóa import



## Variable

* Đặt tên cho một vùng nhớ để lưu trữ giá trị nào đó
* Biến khai báo bằng từ khóa var có thể thay đổi giá trị
* Cú pháp:



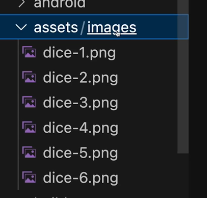
* có thể khai báo kiểu dữ liệu cho biến vd: String text = ‘hello’
* khi không khai báo kiểu dữ liệu và không gán trị kiểu mặc nhiên là dynamic
* để chỉ rõ biến có thể có giá trị null thêm vào ? : vd String? text

## final and const

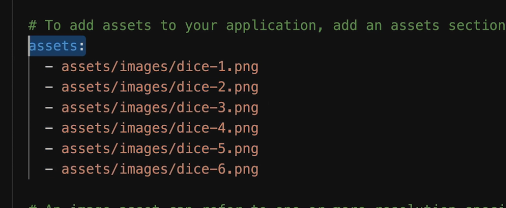
* final: biến được khai báo với từ khóa này không thể gán lại giá trị mới
* const : cũng gần giống như final chứa giá trị hằng không thể thay đổi
* const : là hằng số runtime, giá trị của biến được khóa ngay thời điểm biên dịch
* final: là hằng số thực thi, giá trị được khóa lúc thực thi thật sự

## Hiển thị hình ảnh

* tạo thư mục để chứa các hình ảnh



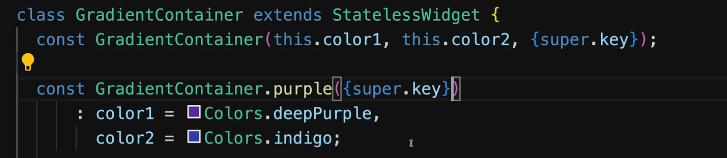
* mở file pubspec.yaml thêm vào đường dẫn của các hình ảnh



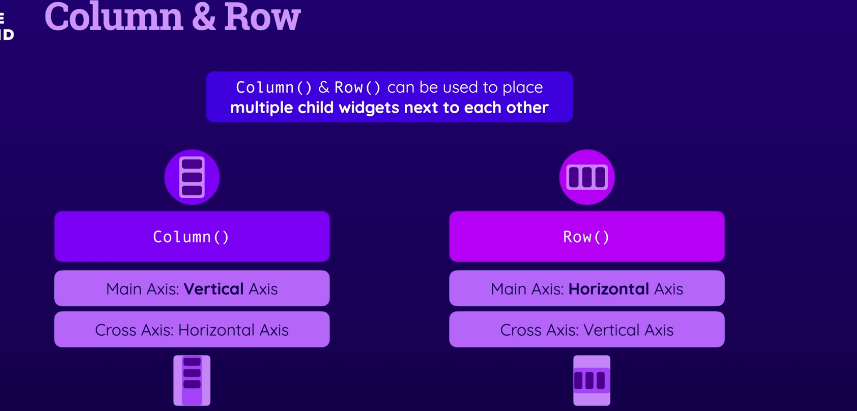
* sử dụng hình ảnh trong file dart



## Multiple constructor function



## Column and Row Widget



## Button

* ElevatedButton : nút có background và border
* OutlineButton: không có màu nền, có viền
* TextButton: văn bản có thể nhấn được
* TextButton(onPressed:function , child:widget)

## StatelessWidget and StatefulWidget

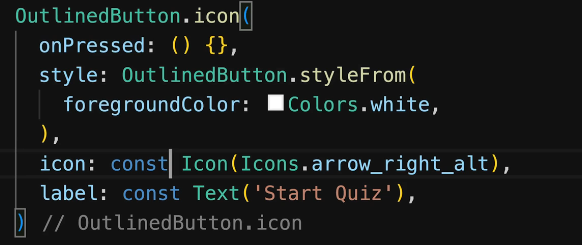
* StatelessWidget: chứa những phần UI cố định, dữ liệu không thay đổi
* StatefulWidget: một số dữ liệu bên trong có thể thay đổi làm thay đổi đến giao diện nên sử dụng widget này

## State và setState()

* StatefulWidget cần có 2 lớp tương tác với nhau
* Khi thực hiện thay đổi 1 giá trị cần đặt trong phuong thức setState để hàm build có thể được gọi là và UI mới được cập nhật

# **Section 3: Flutter and Dart Basic II: Fundamental deep dive - quiz app**

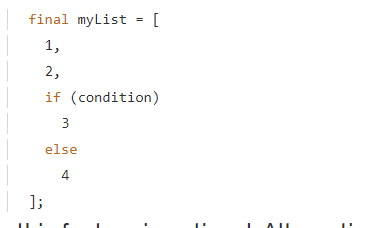
## Adding icon to button



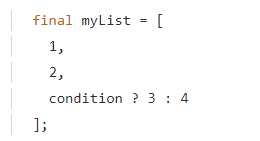
## (stateful) widget lifecycle methods:

* initState(): Được thực thi bởi Flutter khi đối tượng State của StatefulWidget được khởi tạo
* build(): Được thực thi bởi Flutter khi Widget được xây dựng lần đầu tiên VÀ sau khi setState() được gọi
* dispose(): Được thực thi bởi Flutter ngay trước khi Widget bị xóa (ví dụ: vì nó được hiển thị có điều kiện)

## Using "if" Statements In Lists

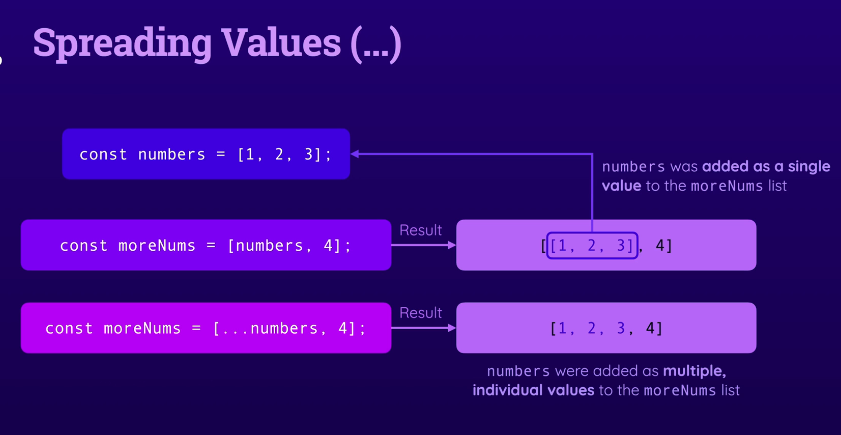


* số 3 sẽ chỉ được thêm vào myList nếu điều kiện được đáp ứng (điều kiện có thể đúng hoặc sai hoặc kiểm tra cho kết quả đúng hoặc sai - ví dụ: ngày == 'Chủ nhật').

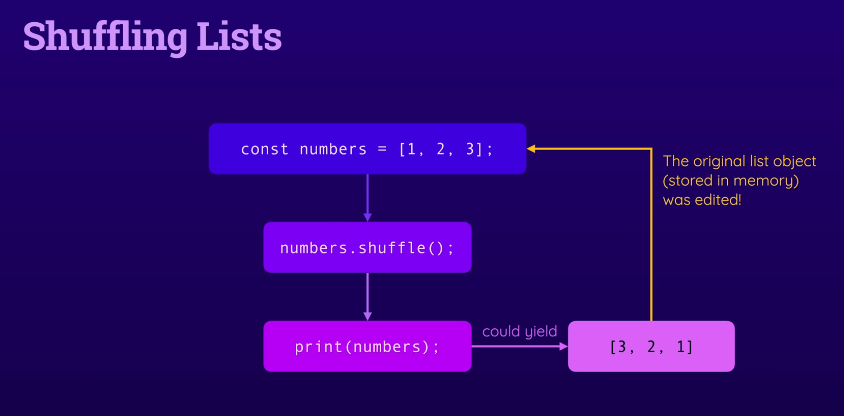


## Dự án Quiz app và các khái niệm mới

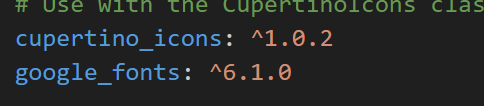
* map() : tạo ra danh sách mới trong bộ nhớ với nội dung được điều chỉnh toán tử spread



* shuffle(): xáo trộn danh sách

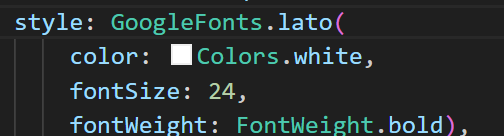


* List.of() : copy danh sách
* Use Third-party package : Google font
* install: $ flutter pub add google\_fonts



* sử dụng:





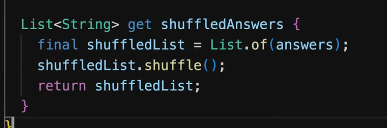
* kiểu dữ liệu map : lưu theo cặp key-value



* accessing map value



* truy cập để lấy dữ liệu từ map thông qua key
* Widget Expanded: chỉ chiếm phần không gian giới hạn theo Widget cha cung cấp
* Widget Column chiếm vùng không gian không giới hạn có thể vượt ra khỏi màn hình
* SingleChildScrollView: cho phép phần tử con có thể cuộn nếu nó vượt quá giới hạn cho phép
* getter : khai báo cho những phương thức chỉ dùng để lấy dữ liệu



* sử dụng như một biết/ thuộc tính thay vì phải sử dụng như cách gọi hàm



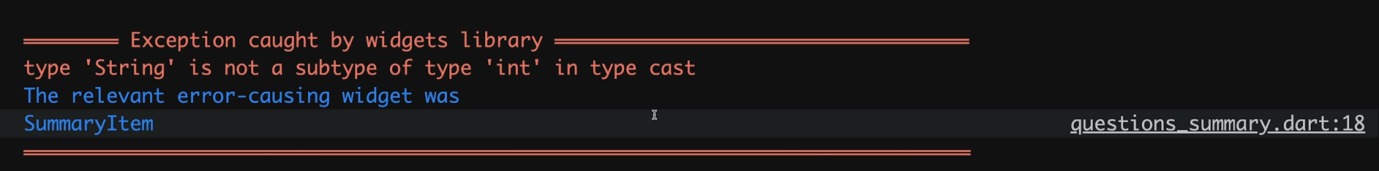
* ký tự underscore trong dart đồng nghĩa với private
* arrow function: có thể sử dụng thay thế cho anonymous function, sử dụng với cú pháp ngắn gọn với những thân hàm chỉ có return

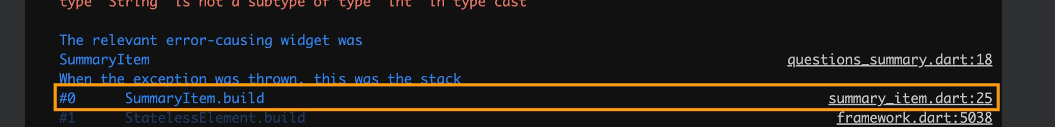


# **Section 4: Debugging Flutter Apps**

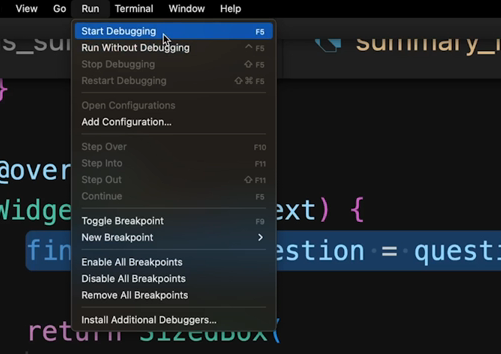
## Đọc và tìm hiểu lỗi

* Khi chạy ứng dụng đôi lúc sẽ bắt gặp những lỗi không mong muốn, trước hết nên đọc nội dung của những thông báo lỗi và thông tin về file có liên quan

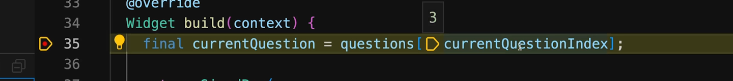




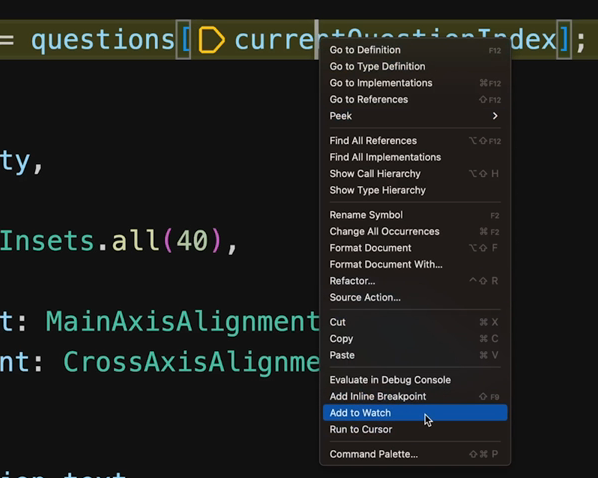
## Chạy ứng dụng với debug

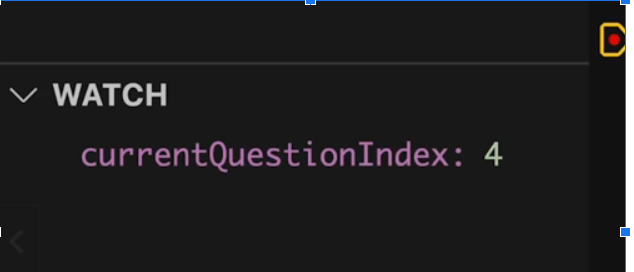


* trong chế độ debug có một số tính năng hỗ trợ cho việc tìm ra lỗi:
* breakpoint: tạo ra các điểm ngắt trong khi thực thi chương trình để kiểm tra giá trị được lưu thực tế khi ứng dụng chạy

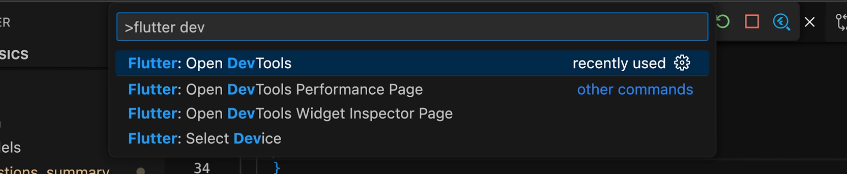


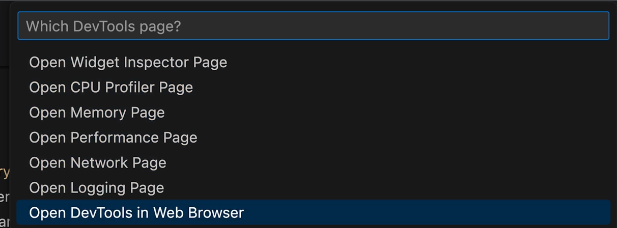
có thể thêm 1 biến vào watch để theo dõi sự thay đổi giá trị của biến trong mỗi lần chạy

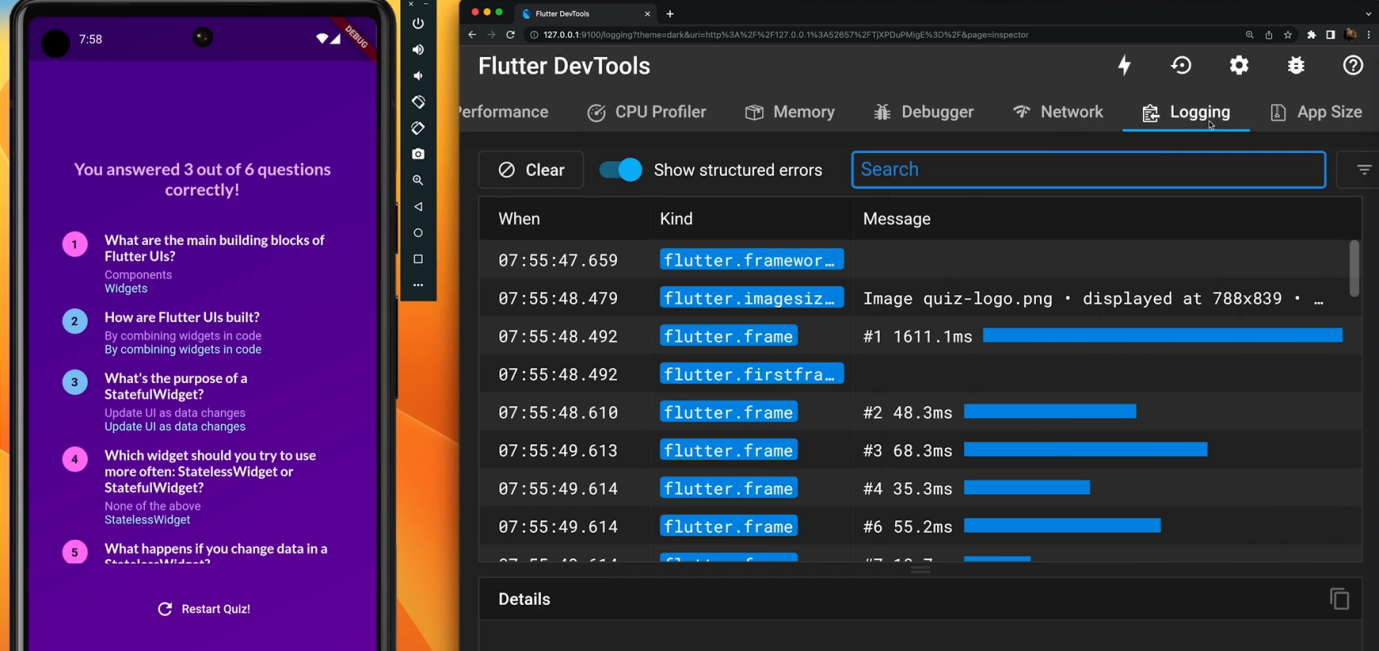




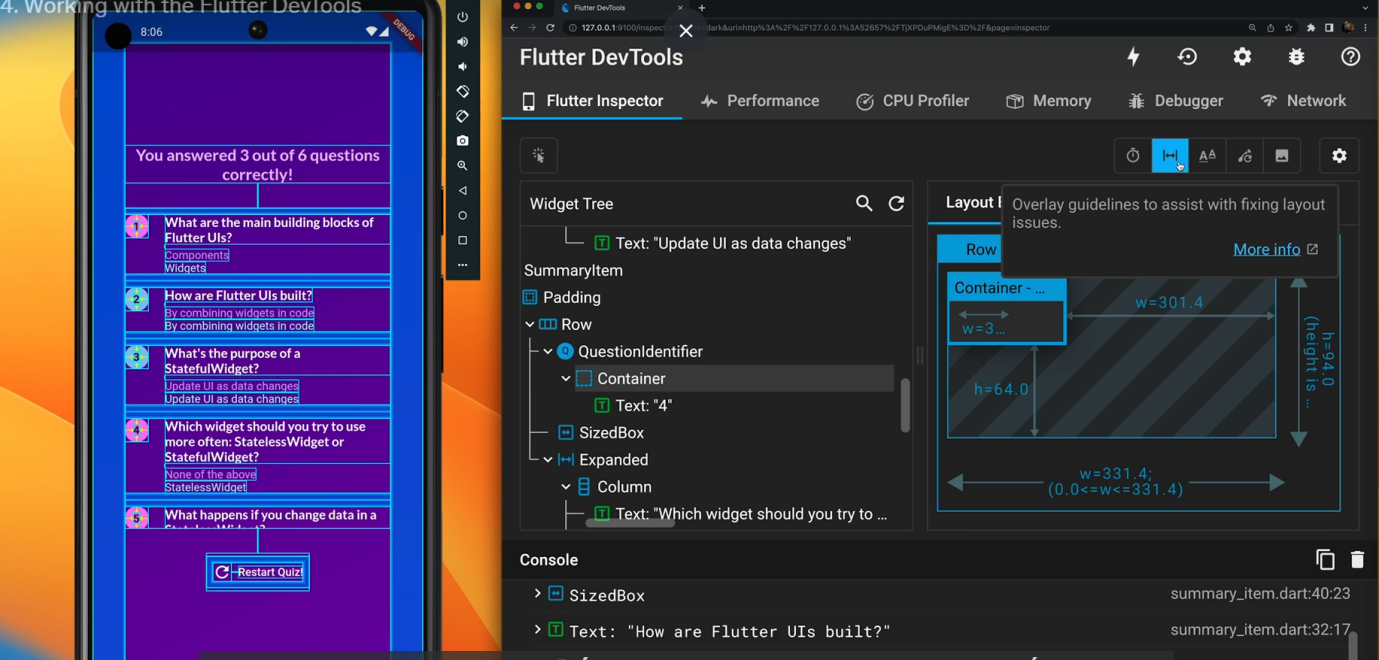
## Devtool





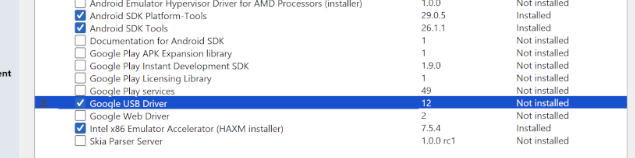


* Cung cấp nhiều phân tích trong khi ứng dụng đang chạy
* Tương tự như google devtool khi xây dựng web



## Chạy ứng dụng trên thiết bị thật

* Chỉ có thể xây dựng + chạy ứng dụng iOS khi sử dụng thiết bị macOS. Tất nhiên, mã sẽ giống nhau, bất kể bạn đang sử dụng nền tảng nào - nhưng bạn chỉ có thể xây dựng + chạy ứng dụng iOS từ máy Mac.
* Các bước thiết lập với điện thoại android
* b1: Bật tùy chọn Nhà phát triển và gỡ lỗi USB trên thiết bị
* b2: Cài đặt Trình điều khiển USB của Google.

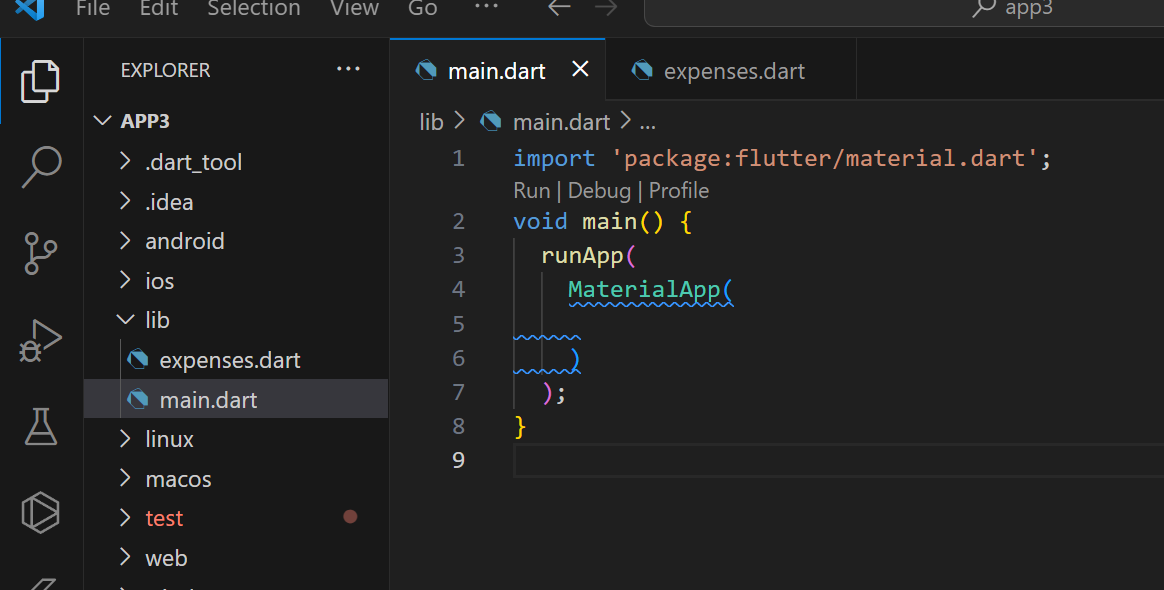


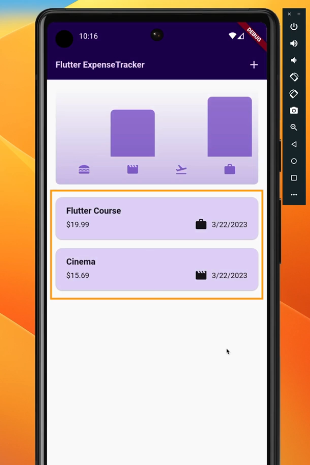
* b3: Cắm thiết bị của bạn vào máy tính Windows. Nếu thiết bị của bạn nhắc bạn, hãy cho phép máy tính truy cập vào thiết bị của bạn.
* b4: Xác minh rằng Flutter nhận ra thiết bị Android được kết nối của bạn



# **Section 5: add interactivity , more Widget and theming**

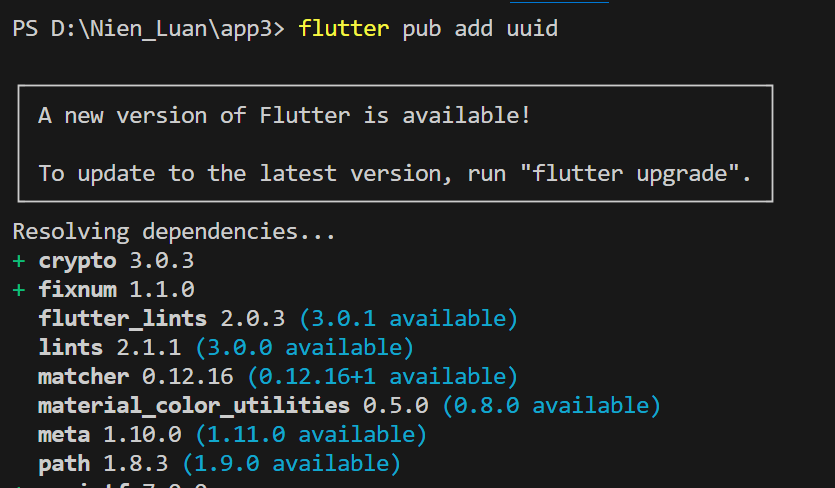
## Tạo dự án mới - dự án quản lý chi tiêu

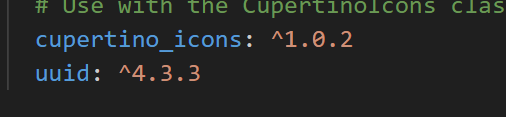




## Sử dụng dịch vụ từ bên thứ 3 UUID để tạo ra ID duy nhất cho dữ liệu

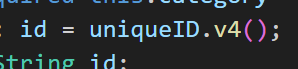
* cài đặt





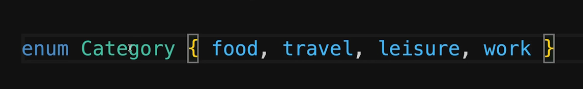
* sử dụng





## Giới thiệu về Enum

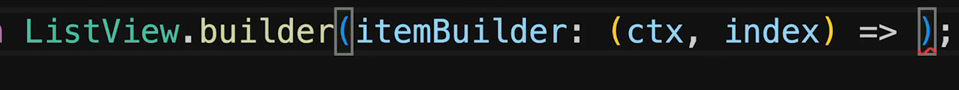
* Từ khóa cho phép tạo ra tập các giá trị xác định



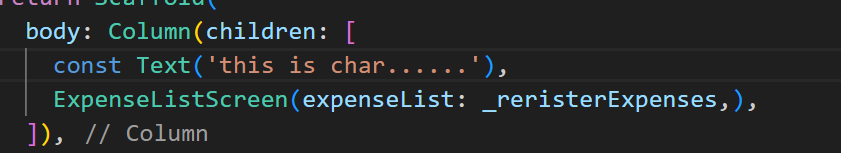


## Sử dụng ListViewt thay cho Column

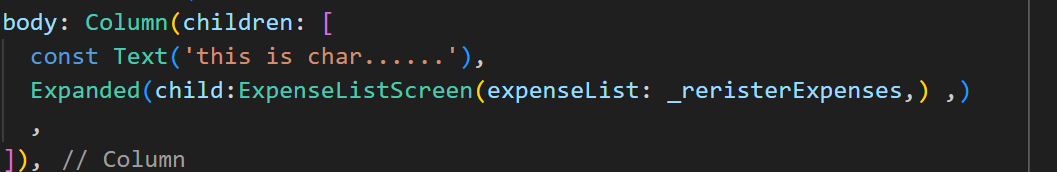
* Khi sử dụng ListViewt sẽ tạo ra 1 danh sách có thể cuộn vẫn sẽ tạo ra tất cả đối tượng khi được gọi hiển thị lên màn hình
* ListViewt có 1 constructor đặt biệt được xây dựng dành cho những danh sách với độ dài không xác định.Tạo ra 1 danh sách cuộn mà ở đó chỉ các item đang thấy hoặc sắp thấy trên màn hình mới được tạo



* Lưu ý sử dụng 1 danh sách trong 1 danh sách khác có thể tạo ra lỗi không hiển thị trong flutter vì không thể xác định được độ dài của phần tử bên trong
* lỗi



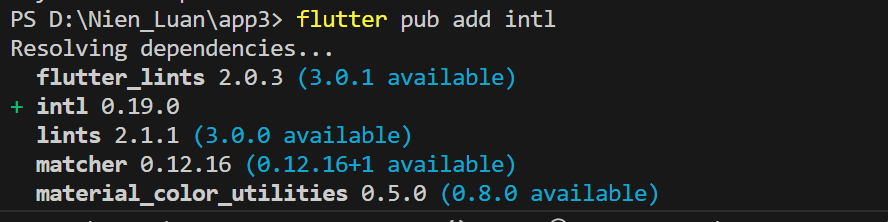
* fix

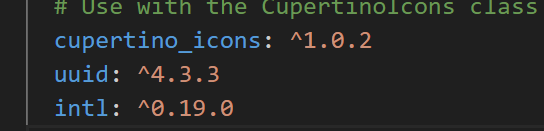


* Card widget: 1 thẻ được design nổi lên khỏi nền có đổ bóng ,...

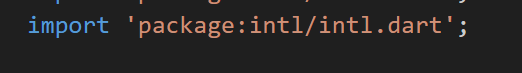
## Sử dụng gói intl để format ngày giờ

* Cài đặt





* Sử dụng







# **Section 6: Building responsive and adaptive user interface**

# **Section 7: Flutter and Dart Internal**

# **Section 8: Building multi screen App and navigating between screen**