



ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN TỬ - VIỄN THÔNG

BÁO CÁO

LẬP TRÌNH ĐA NỀN TẢNG

CHỦ ĐỀ: TEXT VÀ STYLING TRONG FLUTTER

Giảng viên hướng dẫn: TS Nguyễn Duy Nhật Viễn
Sinh viên thực hiện: Nguyễn Thị Uyên Phương - 22KTMT1
Lê Thị Hải Yến - 22KTMT2

Đà Nẵng, tháng 10 năm 2025

Bảng phân công nhiệm vụ

STT	HỌ VÀ TÊN	NHIỆM VỤ	KHỐI LƯỢNG
01	NGUYỄN THỊ UYÊN PHƯƠNG	Tìm hiểu kiến thức Text và Styling, soạn nội dung Word	50%
02	LÊ THỊ HẢI YẾN	Tìm hiểu kiến thức Text và Styling, làm slide báo cáo	50%



Đề mục nội dung

- 1 Giới thiệu
- 2 TextStyle và các thuộc tính định dạng
- 3 Text Overflow và Text Alignment
- 4 RichText và TextSpan
- 5 Custom Fonts và Theme
- 6 Kết luận



I. Giới thiệu

- Trong Flutter, Text là widget cơ bản dùng để hiển thị chuỗi ký tự.
- Mỗi văn bản có thể được định dạng bằng TextStyle, giúp điều chỉnh màu, kích thước, độ đậm, kiểu chữ, v.v.
- Việc nắm vững cách hiển thị và định dạng chữ giúp ứng dụng trở nên nhất quán, dễ đọc và chuyên nghiệp.



II. TextStyle và các thuộc tính định dạng

- **fontSize**: kích thước chữ (px logic)
- **fontWeight**: độ đậm, ví dụ `FontWeight.bold`, `w500`
- **color**: màu chữ, ví dụ `Colors.blue`
- **fontStyle**: kiểu chữ *thường hoặc nghiêng*
- **letterSpacing**, **wordSpacing**: khoảng cách giữa ký tự/từ
- **decoration**: gạch chân, gạch giữa, gạch trên
- **fontFamily**: chọn kiểu font cụ thể



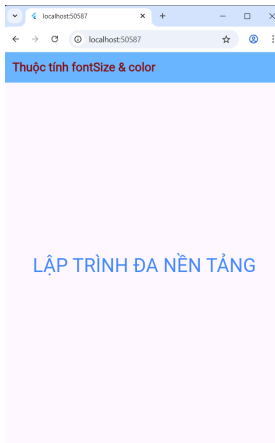
Ví dụ 1: Thuộc tính `fontSize` và `color`

```
21 |         body: Center(  
22 |           child: Text(  
23 |             'LẬP TRÌNH ĐA NỀN TẢNG',  
24 |             style: TextStyle(  
25 |               fontSize: 35, color: Colors.blueAccent,  
26 |             ), // TextStyle  
27 |           ), // Text  
28 |         ), // Center  
29 |       ), // Scaffold  
30 |     ); // MaterialApp  
31 |   }  
32 | }
```

- Thuộc tính `fontSize` điều chỉnh kích thước chữ theo pixel logic.
- `color` xác định màu văn bản, nhận giá trị từ thư viện `Colors`.



Kết quả hiển thị:



Kết quả hiển thị với `fontSize = 35` và `color = Colors.blueAccent`

Ví dụ 2: Thuộc tính `fontWeight` và `fontStyle`

```
21 |         body: Center(  
22 |           child: Column(  
23 |             mainAxisAlignment: MainAxisAlignment.center,  
24 |             children: const [  
25 |               Text(  
26 |                 'LẬP TRÌNH ĐA NỀN TẢNG',  
27 |                 style: TextStyle(  
28 |                   fontWeight: FontWeight.bold,  
29 |                   fontSize: 35,  
30 |                 ), // TextStyle  
31 |             ), // Text  
32 |             SizedBox(height: 20), //khoảng cách 2 dòng  
33 |             Text(  
34 |               '22KTMT',  
35 |               style: TextStyle(  
36 |                 fontStyle: FontStyle.italic,  
37 |                 fontSize: 36  
38 |               ), // TextStyle  
39 |             ), // Text  
40 |           ],  
41 |         ), // Column  
42 |       ), // Center  
43 |     ), // Scaffold  
44 |   ); // MaterialApp  
45 | }  
46 | }
```

- `fontWeight` quy định độ dày nét chữ (w100-w900 hoặc bold).
- `fontStyle` quy định chữ thường hoặc nghiêng.



Kết quả hiển thị:



Kết quả hiển thị với *fontWeight.bold* và *fontStyle.italic*

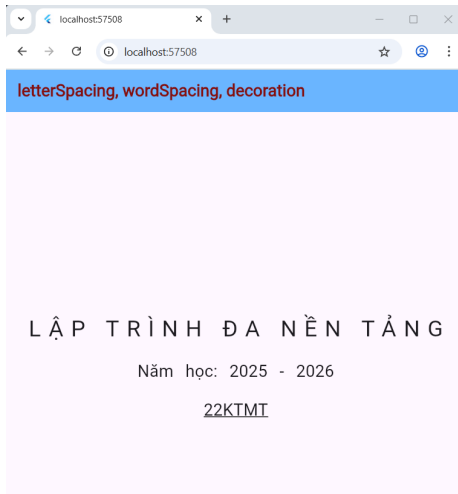


Ví dụ 3: Khoảng cách và gạch chân trong TextStyle

```
21 | | body: Center(child: Column(  
22 | |   mainAxisAlignment: MainAxisAlignment.center,  
23 | |   children: const [  
24 | |     Text( 'LẬP TRÌNH ĐA NỀN TẢNG',  
25 | |       style: TextStyle(  
26 | |         letterSpacing: 10.0, //Giãn khoảng cách giữa các ký tự  
27 | |         fontSize: 30, ), ), // TextStyle // Text  
28 | |     SizedBox(height: 20),  
29 | |     Text('Năm học: 2025 - 2026',  
30 | |       style: TextStyle(  
31 | |         wordSpacing: 10.0, //Giãn khoảng cách giữa các từ  
32 | |         fontSize: 22, ), ), // TextStyle // Text  
33 | |     SizedBox(height: 20),  
34 | |     Text('22KTMТ',  
35 | |       style: TextStyle(  
36 | |         decoration: TextDecoration.underline, //Gạch chân  
37 | |         decorationColor: Colors.black, //Màu gạch chân  
38 | |         decorationThickness: 1, //Độ dày gạch chân  
39 | |         fontSize: 22, ), ), ], ), ), // TextStyle // Text // Column // Center  
40 | |   ), // Scaffold  
41 | | ); // MaterialApp  
42 | }  
43 | }
```



Kết quả hiển thị:



Kết quả hiển thị với letterSpacing, wordSpacing và decoration



III. Overflow và Alignment

1. Text Overflow: Thuộc tính `overflow` dùng để xử lý khi văn bản vượt quá không gian hiển thị.

- `TextOverflow.clip` – Cắt phần văn bản dư (không hiển thị thêm).
- `TextOverflow.fade` – Làm mờ dần phần cuối văn bản.
- `TextOverflow.ellipsis` – Thêm dấu “...” ở cuối (phổ biến nhất).

2. Text Alignment: Thuộc tính `textAlign` xác định cách căn chỉnh văn bản trong widget.

- `TextAlign.left`, `TextAlign.center`, `TextAlign.right`
- `TextAlign.justify` – Căn đều hai bên (thường dùng cho đoạn văn dài).

→ Hai thuộc tính này thường được dùng cùng nhau để hiển thị văn bản đẹp và rõ ràng.

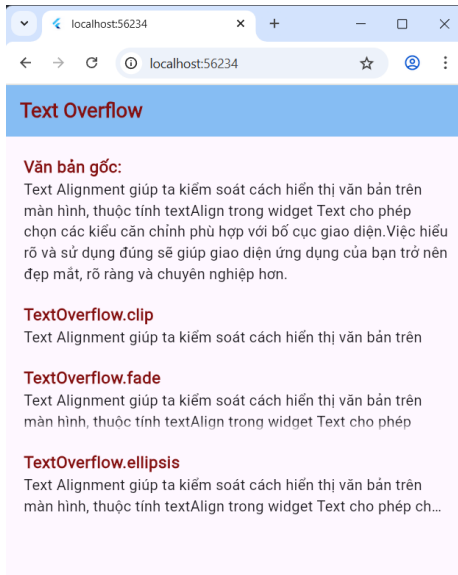


Ví dụ: Overflow trong Flutter

```
19 body: Padding(  
20   padding: const EdgeInsets.all(20.0),  
21   child: Column(  
22     crossAxisAlignment: CrossAxisAlignment.start,  
23     children: const [  
24       Text(  
25         'Văn bản gốc',  
26         style: TextStyle(  
27           fontSize: 18, fontWeight: FontWeight.bold, color: Color.fromARGB(255, 132, 17, 17)), // TextStyle  
28       ), // Text  
29       Text(  
30         'Text Alignment giúp ta kiểm soát cách hiển thị văn bản trên màn hình, thuộc tính textAlign trong widget Text cho phép chọn các kiểu căn chỉnh phù hợp với  
31         style: TextStyle(fontSize: 16)), // Text  
32       SizedBox(height: 20),  
33       Text(  
34         'TextOverflow.clip',  
35         style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold, color: Color.fromARGB(255, 132, 17, 17))), // Text  
36       Text(  
37         'Text Alignment giúp ta kiểm soát cách hiển thị văn bản trên màn hình, thuộc tính textAlign trong widget Text cho phép chọn các kiểu căn chỉnh phù hợp với  
38         style: TextStyle(fontSize: 16),  
39         overflow: TextOverflow.clip,  
40         maxLines: 1), // Text  
41       SizedBox(height: 20),  
42       Text(  
43         'TextOverflow.fade', style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold, color: Color.fromARGB(255, 132, 17, 17))), // Text  
44       Text(  
45         'Text Alignment giúp ta kiểm soát cách hiển thị văn bản trên màn hình, thuộc tính textAlign trong widget Text cho phép chọn các kiểu căn chỉnh phù hợp với  
46         style: TextStyle(fontSize: 16),  
47         overflow: TextOverflow.fade,  
48         maxLines: 2), // Text  
49       SizedBox(height: 20),  
50       Text(  
51         'TextOverflow.ellipsis',  
52         style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold, color: Color.fromARGB(255, 132, 17, 17))), // Text  
53       Text(  
54         'Text Alignment giúp ta kiểm soát cách hiển thị văn bản trên màn hình, thuộc tính textAlign trong widget Text cho phép chọn các kiểu căn chỉnh phù hợp với  
55         style: TextStyle(fontSize: 16),  
56         overflow: TextOverflow.ellipsis,  
57         maxLines: 2), // Text  
58     ],  
59   ), // Column  
60 ), // Padding  
61 ), // Scaffold  
62 ); // MaterialApp  
63 }  
64 }
```



Kết quả hiển thị:



Ví dụ: Text Alignment

```
21 child: Column(  
22   crossAxisAlignment: CrossAxisAlignment.stretch,  
23   children: const [  
24     Text(  
25       'TextAlign.left', style: TextStyle(fontSize: 18, color: Colors.red, fontWeight: FontWeight.bold)), // Text  
26     Text(  
27       'Text Alignment giúp ta kiểm soát cách hiển thị văn bản trên màn hình. Thuộc tính textAlign trong widget Text cho phép chọn các kiểu căn chỉnh phù hợp với  
28       textAlign: TextAlign.left,  
29       style: TextStyle(fontSize: 18, color: Colors.red)), // Text  
30     Text(  
31       'TextAlign.center', style: TextStyle(fontSize: 18, color: Colors.green, fontWeight: FontWeight.bold)), // Text  
32     Text(  
33       'Text Alignment giúp ta kiểm soát cách hiển thị văn bản trên màn hình. Thuộc tính textAlign trong widget Text cho phép chọn các kiểu căn chỉnh phù hợp với  
34       textAlign: TextAlign.center,  
35       style: TextStyle(fontSize: 18, color: Colors.green)), // Text  
36     Text(  
37       'TextAlign.right', style: TextStyle(fontSize: 18, color: Colors.orange, fontWeight: FontWeight.bold)), // Text  
38     Text(  
39       'Text Alignment giúp ta kiểm soát cách hiển thị văn bản trên màn hình. Thuộc tính textAlign trong widget Text cho phép chọn các kiểu căn chỉnh phù hợp với  
40       textAlign: TextAlign.right,  
41       style: TextStyle(fontSize: 18, color: Colors.orange)), // Text  
42     Text(  
43       'TextAlign.justify', style: TextStyle(fontSize: 18, color: Colors.blueGrey, fontWeight: FontWeight.bold)), // Text  
44     Text(  
45       'Text Alignment giúp ta kiểm soát cách hiển thị văn bản trên màn hình. Thuộc tính textAlign trong widget Text cho phép chọn các kiểu căn chỉnh phù hợp với  
46       textAlign: TextAlign.justify,  
47       style: TextStyle(fontSize: 18, color: Colors.blueGrey)), // Text  
48   ],  
49 ), // Column  
50 ), // Container  
51 ), // Scaffold  
52 ); // MaterialApp  
53 }
```



Kết quả hiển thị:



IV. RichText và TextSpan

- RichText cho phép hiển thị nhiều định dạng trong cùng một đoạn văn.
- TextSpan là các đoạn văn bản con có thể định dạng độc lập.



IV. RichText và TextSpan

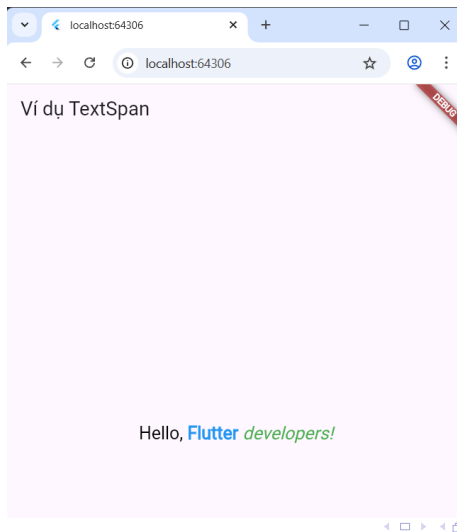
Ví dụ:

```
lib > main.dart > MyApp > build
6
7 class MyApp extends StatelessWidget {
8   @override
9   Widget build(BuildContext context) {
10     return MaterialApp(
11       home: Scaffold(
12         appBar: AppBar(title: Text('Ví dụ TextSpan')),
13         body: Center(
14           child: RichText(
15             text: TextSpan(
16               style: TextStyle(
17                 fontSize: 20.0,
18                 color: Colors.black,
19             ), // TextStyle
20             children: [
21               TextSpan(text: 'Hello, '),
22               TextSpan(
23                 text: 'Flutter ',
24                 style: TextStyle(
25                   fontWeight: FontWeight.bold,
26                   color: Colors.blue,
27                 ), // TextStyle
28               ), // TextSpan
29               TextSpan(
30                 text: 'developers!',
31                 style: TextStyle(
32                   fontStyle: FontStyle.italic,
33                   color: Colors.green,
34                 ), // TextStyle
35               ), // TextSpan
36             ],
37           ), // TextSpan
38         ), // RichText
39       ), // Center
40     ), // Scaffold
41   ); // MaterialApp
42 }
```



IV. RichText và TextSpan

Kết quả:

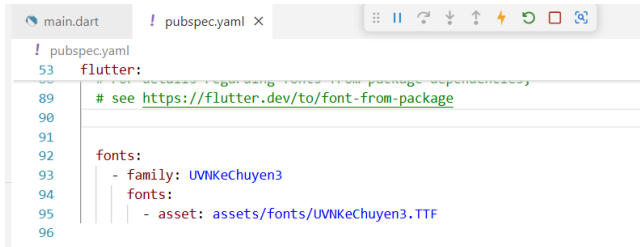


V. Custom Fonts và Theme

- Có thể thêm font tùy chỉnh vào ứng dụng Flutter để tạo sự đồng bộ về phong cách thiết kế.

- Các bước thực hiện:

Bước 1: Tạo thư mục `assets/fonts/` và thêm file `.ttf` vào, khai báo trong file `pubspec.yaml`.



```
main.dart  ! pubspec.yaml x
! pubspec.yaml
53 flutter:
54   # For details regarding fonts from package dependencies,
89   # see https://flutter.dev/to/font-from-package
90
91
92   fonts:
93     - family: UVNKeChuyen3
94       fonts:
95         - asset: assets/fonts/UVNKeChuyen3.TTF
96
```



V. Custom Fonts và Theme

Bước 2: Sử dụng trong widget với TextStyle(fontFamily: '...').

```
body: const Center(  
  child: Column(  
    mainAxisAlignment: MainAxisAlignment.center,  
    children: [  
      Text(  
        'Lập trình Đa Nền Tảng',  
        style: TextStyle(  
          fontSize: 36,  
          color: Color.fromARGB(255, 143, 3, 3),  
        ), // TextStyle  
      ), // Text  
      SizedBox(height: 20),  
      Text(  
        'Lập trình Đa Nền Tảng',  
        style: TextStyle(  
          fontSize: 46,  
          color: Color.fromARGB(255, 11, 2, 99),  
          fontFamily: 'UVNKeChuyen3', // font tùy chỉnh  
          fontWeight: FontWeight.bold  
        ), // TextStyle  
      ), // Text  
    ],  
  ), // Column  
) // Center
```



V. Custom Fonts và Theme

Kết quả sau khi áp dụng font tùy chỉnh:



V. Custom Fonts và Theme

- **Theme** giúp định nghĩa phong cách giao diện tổng thể cho ứng dụng (màu sắc, font, kích thước, nút,...).
- Giúp giao diện đồng nhất, dễ bảo trì, thay đổi nhanh toàn bộ ứng dụng.
- Cấu hình Theme trong MaterialApp thông qua thuộc tính theme: `ThemeData(...)`.
 - `primaryColor`: Màu chính của ứng dụng.
 - `scaffoldBackgroundColor`: Màu nền của các màn hình.
 - `appBarTheme`: Cấu hình cho AppBar.
 - `textTheme`: Định nghĩa kiểu chữ chung.
 - `buttonTheme` / `elevatedButtonTheme`: Thiết kế nút bấm.



V. Custom Fonts và Theme

- Sau khi khai báo ThemeData trong MaterialApp, ta có thể truy cập và áp dụng Theme trong từng widget.
- Sử dụng Theme.of(context) để lấy các thuộc tính giao diện hiện tại (màu sắc, font chữ, style,...).
- Cơ chế hoạt động: Theme.of(context) là hàm *context-aware*, tìm ThemeData được định nghĩa trong cây widget cha gần nhất.
- Khi thay đổi Theme, mọi widget dùng Theme.of(context) sẽ tự cập nhật lại giao diện.

Cú pháp thường dùng:

- Theme.of(context).textTheme.bodyMedium
- Theme.of(context).colorScheme.primary
- Theme.of(context).appBarTheme.backgroundColor



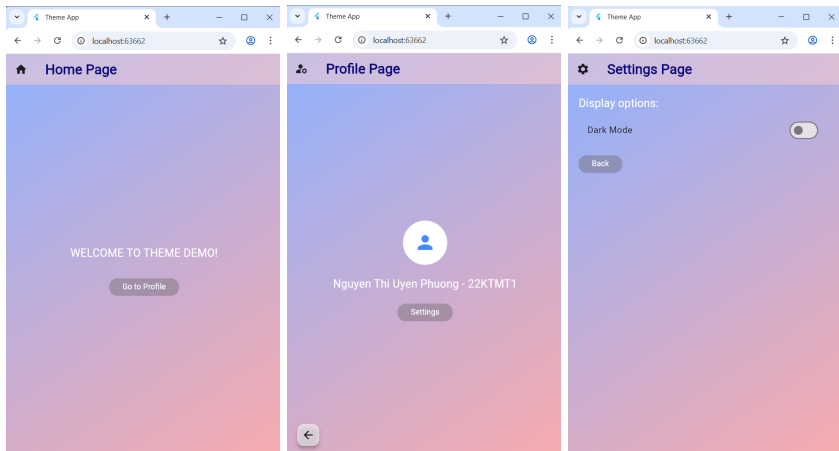
V. Custom Fonts và Theme

```
43
44 class MyApp extends StatelessWidget {
45   @override
46   Widget build(BuildContext context) {
47     return MaterialApp(
48       debugShowCheckedModeBanner: false,
49       title: 'Theme App',
50       theme: ThemeData(
51         primaryColor: hexToColor("#a6c0fe"),
52         appBarTheme: const AppBarTheme(
53           titleTextStyle: TextStyle(
54             color: Color.fromARGB(255, 10, 8, 120),
55             fontWeight: FontWeight.bold,
56             fontSize: 25,
57           ), // TextStyle
58           backgroundColor: Color.fromARGB(137, 248, 190, 190),
59           elevation: 0,
60         ), // AppBarTheme
61         textTheme: const TextTheme(
62           bodyMedium: TextStyle(
63             fontSize: 20,
64             color: Colors.white,
65           ), // TextStyle
66         ), // TextTheme
67         elevatedButtonTheme: ElevatedButtonThemeData(
68           style: ElevatedButton.styleFrom(
69             backgroundColor: Color.fromARGB(66, 255, 252, 252),
70             foregroundColor: Colors.white,
71             shape: RoundedRectangleBorder(
72               borderRadius: BorderRadius.all(Radius.circular(15)),
73             ), // RoundedRectangleBorder
74             padding: EdgeInsets.symmetric(horizontal: 24, vertical: 12),
75           ),
76         ), // ElevatedButtonThemeData
```



V. Custom Fonts và Theme

Demo 3 Page với Theme



Home Page

Profile Page

Settings Page

VI. Kết luận

- Text và Styling là nền tảng trong thiết kế giao diện Flutter.
- Sử dụng đúng cách giúp giao diện rõ ràng, mang tính thương hiệu và chuyên nghiệp.
- Kết hợp linh hoạt giữa Text, TextStyle, RichText và ThemeData sẽ giúp ứng dụng trở nên nhất quán và đẹp mắt.



Tài liệu tham khảo

- 1 Flutter Documentation – Text and Styling
[*https://docs.flutter.dev/ui/widgets/text*](https://docs.flutter.dev/ui/widgets/text)
- 2 Flutter Widget Catalog – Text Widgets
[*https://api.flutter.dev/flutter/widgets/Text-class.html*](https://api.flutter.dev/flutter/widgets/Text-class.html)
- 3 Material Design Guidelines – Typography
[*https://m3.material.io/styles/typography/overview*](https://m3.material.io/styles/typography/overview)

