TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA ĐIỆN TỬ VIỄN THÔNG

**BÁO CÁO**

**LẬP TRÌNH ĐA NỀN TẢNG**

**Text và Styling trong Flutter**

**Sinh viên thực hiện:**

**01. Văn Tiến Lộc Lớp: 22KTMT2 MSSV:  106220257**

**02. Nguyễn Trung Thành Lớp: 22KTMT2 MSSV:  106220270**

**Người hướng dẫn:**

**TS. Nguyễn Duy Nhật Viễn**

**Đà Nẵng, 2025.**

**THUYẾT MINH**

**BÁO CÁO**

**LẬP TRÌNH ĐA NỀN TẢNG**

**Text và Styling trong Flutter**

BẢNG PHÂN CÔNG CÔNG VIỆC TRONG NHÓM

|  |  |  |  |
| --- | --- | --- | --- |
| STT | HỌ VÀ TÊN | NHIỆM VỤ | KHỐI LƯỢNG |
| 01 | Văn Tiến Lộc | Tìm hiểu, soạn nội dung, viết báo cáo | 50% |
| 02 | Nguyễn Trung Thành | Tìm hiểu, soạn nội dung, làm slide | 50% |

 Link code github: [github.com/Cherikyatto](https://github.com/Cherikyatto/Text-and-Styling-trong-Flutter)

Mục lục

I. TextStyle properties..........................................................................................................4

II. Text overflow và text alignment.....................................................................................8

III. RichText và TextSpan..................................................................................................10

IV. Custom fonts và theme................................................................................................12

NỘI DUNG

**I. TextStyle properties**.

Trong **Flutter**, lớp **TextStyle** dùng để định dạng giao diện (style) của chữ trong **Text widget**. Bốn thuộc tính cơ bản và hay dùng nhất là:

**a. fontSize.**

Như cái tên đã nói thuộc tính **fontSize** trong lớp **TextStyle** của **Flutter** được dùng để xác định kích thước chữ của văn bản.

* **Khái niệm**: **fontSize** là 1 kiểu thuộc tích **double?(*Can be Null*)** trong lớp **TextStyle**.
* **Đơn vị**: Giá trị được chỉ định bằng **logical pixels** **(đơn vị độc lập với mật độ điểm ảnh).**
* **Mặc định:** Nếu không thiết lập, kích thước chữ sẽ mặc định là **14 logical pixels** (hoặc kích thước được định nghĩa bởi Theme).

|  |
| --- |
| Text(           'Kích thước chữ lớn',           style: TextStyle(                    fontSize: 24.0,  // Ví dụ: Đặt kích thước là 24 logical pixels           ),  ), |

Điểm đặc biệt và quan trọng nhất của **fontSize** là nó luôn bị ảnh hưởng bởi Tỷ lệ chữ của hệ thống **(System Text Scale Factor)** do người dùng thiết lập trong cài đặt trợ năng của thiết bị.

* **Tính toán kích thước hiển thị:** Kích thước chữ cuối cùng được hiển thị trên màn hình sẽ là:

**Kích thước hiển thị=fontSize×textScaleFactor**

* **Mục đích (Accessibility):** Tính năng này đảm bảo ứng dụng của Dev tuân thủ các tiêu chuẩn trợ năng, cho phép người dùng nhìn thấy văn bản với kích thước phù hợp với nhu cầu của họ.

1. **fontWeight.**

**FontWeight** dùng để xác định độ dày của chữ (đơn giản chính là độ đậm của ký tự).

* **Kiểu dữ liệu: FontWeight (là một enum/class).**
* **Cách sử dụng:** Bạn thường sử dụng các hằng số được định nghĩa sẵn trong **FontWeight class.**
* **FontWeight.normal:** Độ đậm chuẩn (thường là 400).
* **FontWeight.bold:** Chữ đậm.
* **FontWeight.w100** đến **FontWeight.w900:** Cho phép chọn độ đậm cụ thể theo thang số từ 100 (mỏng nhất) đến 900 (đậm nhất).

|  |
| --- |
| Text(          'Chữ đậm vừa',           style: TextStyle(                     fontWeight:                     FontWeight.w600,  // Đậm hơn normal, mỏng hơn bold           ),  ), |

1. **Color.**

Thuộc tính **color** dùng để mang vô vàn màu sắc tới cho các ký tự văn bản.

* Kiểu dữ liệu: **Color?**
* **Giá trị:** Thường sử dụng các hằng số từ lớp **Colors** hoặc **tạo màu tùy chỉnh**.
* **Color** có thể được sử dụng thông qua 3 cách sau:

1. **Sử dụng các màu có sẵn:**

|  |
| --- |
| // Màu cơ bản  color: Colors.red,  // Sử dụng sắc thái (shade) màu  color: Colors.blue[600],  // Màu xanh dương với độ đậm 600 |

1. **Sử dụng màu Custome (Tùy chỉnh):**

Sử dụng mã màu **Hex** với **độ mờ (*opacity*)** 8 chữ số, bắt đầu bằng **0x.** Hai chữ số đầu tiên là kênh **Alpha (độ mờ)**, sau đó là **R,G,B.**

|  |
| --- |
| // Màu đỏ tươi (FF là Alpha (100% độ mờ))  color: const Color(0xFFFF0000),  // Màu xanh với độ mờ 50% (80 là Alpha)  color: const Color(0x8000FF00), |

Hoặc sử dụng phương thức **fromARGB(alpha, red, green, blue):**

|  |
| --- |
| color: Color.fromARGB(255, 0, 0, 255),  // Màu xanh dương (255 là độ mờ 100%) |

1. **Màu từ Theme (Chủ đề):**

Để đảm bảo tính nhất quán với thiết kế, nên lấy màu từ **Theme** của ứng dụng.

|  |
| --- |
| // Lấy màu chính (Primary color) từ Theme  color: Theme.of(context).colorScheme.primary,  // Lấy màu mặc định cho văn bản từ TextTheme  color: Theme.of(context).textTheme.bodyLarge?.color, |

Phân biệt với **backgroundColor:**

* **color:** Tô màu cho nội dung (các glyphs/ký tự).
* **backgroundColor:** Tô màu cho **phần nền (*background*)** ngay phía sau văn bản.

|  |
| --- |
| style: TextStyle(           color: Colors.white,  // Màu chữ là Trắng           backgroundColor: Colors.black, // Màu nền là Đen  ) |

1. **fontStyle.**

Thuộc tính **fontStyle** xác định kiểu chữ là **đứng (*normal*)** hay **nghiêng (*italic*).**

* **Kiểu dữ liệu: FontStyle?**
* **Giá trị:** Nó là một **enum** với hai lựa chọn:
  + **fontStyle.normal:** kiểu chữ mặc định đứng thẳng.
  + **dontStyle.italic:** chữ in nghiêng.

Để sử dụng gán trực tiếp giá trị enum cho thuộc tính.

|  |
| --- |
| // 1. Chữ nghiêng  Text(          'Văn bản nghiêng',              style: TextStyle(              fontStyle: FontStyle.italic,          ),  ),  // 2. Chữ đứng (thường không cần thiết vì đây là mặc định)   Text(          'Văn bản thường',              style: TextStyle(              fontStyle: FontStyle.normal,          ),  ), |

**II. Text overflow và text alignment.**

1. **Text overflow.**

Khi nội dung văn bản quá dài so với không gian chiều ngang hoặc chiều dọc đã được cung cấp thì lúc đó sẽ xảy ra tình trạng **Text Overflow.**

Thuộc tính được sử dụng để kiểm soát điều này là **overflow** trong **TextStyle** của **widget Text.** Nó nhận giá trị từ **enum TextOverflow.**

Các giá trị chính của **TextOverflow:**

* **TextOverflow.clip:** Cắt bỏ phần văn bản bị tràn, không hiển thị bất kỳ dấu hiệu nào.
* **TextOverflow.fade:** Phần văn bản bị tràn sẽ mờ dần thành trong suốt ở cuối để tạo hiệu ứng chuyển tiếp mềm mại.
* **TextOverflow.ellipsis:** Thay thế phần văn bản bị tràn bằng **(...)** để báo hiệu nội dung còn tiếp tục. Đây là lựa chọn phổ biến nhất.
* **TextOverflow.visible:** Văn bản sẽ tiếp tục hiển thị bên ngoài giới hạn của widget.
  + **Lưu ý:** Điều này có thể gây ra lỗi tràn pixel (render overflow) nếu không được xử lý cẩn thận.

**Ví dụ:** Thường phải kết hợp **overflow** với thuộc tính **maxLines** để áp dụng xử lý tràn sau khi giới hạn số dòng:

|  |
| --- |
| Text(       'Đây là một đoạn văn bản rất dài cần bị giới hạn và cắt bớt.',        maxLines: 2,  // Giới hạn chỉ hiển thị 2 dòng overflow:        TextOverflow.ellipsis,  // Nếu tràn, hiển thị "..."        style: TextStyle(fontSize: 16),  ), |

1. **Text alignment.**

**Text Alignment** xác định cách văn bản được căn chỉnh trong không gian chiều ngang của **widget Text.**

* Thuộc tính được sử dụng là **textAlign,** nhận giá trị từ **enum TextAlign.**
* Các Giá trị chính của **TextAlign:**
  + **TextAlign.left:** Căn chỉnh văn bản về phía bên trái.
  + **TextAlign.right:** Căn chỉnh văn bản về phía bên phải.
  + **TextAlign.center:** Căn chỉnh văn bản ở giữa không gian.
  + **TextAlign.justify:** Dàn đều khoảng cách giữa các từ để làm cho lề trái và lề phải thẳng hàng (chủ yếu dùng cho các đoạn văn dài).
  + **TextAlign.start:**Căn chỉnh theo hướng bắt đầu của ngôn ngữ (trái cho hầu hết các ngôn ngữ LTR như tiếng Việt/Anh, phải cho các ngôn ngữ RTL).
  + **TextAlign.end:**Căn chỉnh theo hướng kết thúc của ngôn ngữ (phải cho LTR, trái cho RTL).

**Ví dụ:** căn giữa.

|  |
| --- |
| Text(         'Văn bản này được căn giữa.',         textAlign: TextAlign.center,  // Căn giữa văn bản style:        TextStyle(fontSize: 18),  ), |

**Lưu ý quan trọng về Canh lề:**

* **textAlign** chỉ căn chỉnh văn bản bên trong hộp văn bản mà **widget Text** chiếm giữ.
* Nếu muốn căn chỉnh chính **widget Text** trong **widget cha** của nó, cần sử dụng các thuộc tính của **widget cha** như **mainAxisAlignment (*trong Column hoặc Row*)** hoặc bọc **Text** trong một **widget Center.**

**III. RichText và TextSpan**

1. **Tổng quan về RichText**

* **RichText** là **widget** dùng để hiển thị văn bản có nhiều kiểu định dạng khác nhau trong cùng một đoạn. Nội dung hiển thị được mô tả dưới dạng cây các đối tượng **TextSpan,** trong đó mỗi **TextSpan** mang một **TextStyle** riêng và phong cách đó sẽ được áp dụng cho toàn bộ phần tử con của nó.
* Tùy theo giới hạn **bố cục (*layout constraints*),** đoạn văn bản có thể xuống dòng hoặc nằm trên một dòng duy nhất.
* Văn bản trong **RichText** phải được chỉ định **kiểu (*style*)** một cách rõ ràng. Thông thường, nên dùng **“DefaultTextStyle.of(context)”** để lấy kiểu chữ mặc định của giao diện hiện tại, sau đó tùy chỉnh lại khi cần.

**Ví dụ** dưới đây hiển thị câu **"Hello bold world",** trong đó: từ **"bold"** được in đậm bằng cách áp dụng **fontWeight.bold.**

|  |
| --- |
| RichText(    text: TextSpan(      text: 'Hello ',      style: DefaultTextStyle.of(context).style,      children: const <TextSpan>[        TextSpan(text: 'bold', style: TextStyle(fontWeight: FontWeight.bold)),        TextSpan(text: ' world!'),      ],    ),  ) |

* **Selection trong RichText**
* Để cho phép người dùng bôi đen và sao chép văn bản trong **RichText**, **widget** này cần nằm trong cây con của **SelectionArea** hoặc **SelectableRegion,** và phải gán **SelectionRegistrar** thông qua thuộc tính **selectionRegistrar.**
* Ngoài ra, phải cấu hình thêm **selectionColor** để hiển thị màu đánh dấu khi văn bản được chọn.

**Ví dụ:**

|  |
| --- |
| RichText(    text: const TextSpan(text: 'Hello'),    selectionRegistrar: SelectionContainer.maybeOf(context),    selectionColor: const Color(0xAF6694e8),  ) |

1. **TextSpan**

* **TextSpan** là một đối tượng bất biến đại diện cho một đoạn văn bản có định dạng. Một **TextSpan** có thể chứa:
  + Chỉ một chuỗi text duy nhất.
  + Danh sách children là các **TextSpan con**, mỗi phần lại có **style riêng**.
* Nếu một **TextSpan** vừa có **text** vừa có **children,** thì phần **text** sẽ được xem như một **TextSpan** không có **style** đặt ở đầu danh sách con. Khi **text** **bị để null**, **TextSpan** đóng vai trò như một **node rỗng** trong cây **văn bản (InlineSpan tree).**
* Để vẽ **TextSpan** lên Canvas, sử dụng **TextPainter.**
* Để hiển thị trong **widget,** sử dụng **RichText.**
* Nếu chỉ dùng một kiểu chữ duy nhất, hãy dùng **Text** thay vì **RichText.**

**Ví dụ đơn giản:**

|  |
| --- |
| const TextSpan(    text: 'Hello world!',    style: TextStyle(color: Colors.black),  ) |

**IV. Custom fonts và theme**

1. **Custom Font.**

* Cho phép sử dụng **font tùy chỉnh** (do thiết kế riêng hoặc từ nguồn bên ngoài) trong ứng dụng **Flutter.**
* Áp dụng **font tùy chỉnh** cho toàn ứng dụng hoặc cho riêng từng **widget.**

**Các bước thực hiện:**

**1. Chọn font**

* Chọn **font** muốn dùng thường là **file .ttf, .otf hoặc .ttc.**
* Cần ưu ý kiểm tra xem **font** có hỗ trợ nhiều **weight, kiểu (italic, bold)** hay không, để dùng linh hoạt trong **app.**

**2. Nhập file font vào dự án**

* Tạo thư mục **fonts/ (hoặc assets/fonts/)** trong thư mục gốc dự án.
* Copy các **file .ttf, .otf** vào thư mục đó.

**3. Khai báo font trong pubspec.yaml**

**Mở pubspec.yaml, trong phần flutter: thêm mục fonts: như sau:**

|  |
| --- |
| flutter:  fonts:   - family: Raleway        fonts:          - asset: fonts/Raleway-Regular.ttf          - asset: fonts/Raleway-Italic.ttf            style: italic      - family: RobotoMono        fonts:          - asset: fonts/RobotoMono-Regular.ttf          - asset: fonts/RobotoMono-Bold.ttf            weight: 700 |

* **family là tên font được dùng trong TextStyle(fontFamily: …)**
* **Gán file cụ thể cho kiểu italic hoặc độ đậm khác nhau.**

**4. Sử dụng font trong widget**

|  |
| --- |
| Text(    'Văn bản tùy chỉnh',    style: TextStyle(fontFamily: 'Raleway', fontSize: 20),  ) |

Có thể kết hợp với **ThemeData** nếu muốn dùng **font** này làm mặc định cho toàn **app.**

1. **Theme và TextTheme**

* Dùng **ThemeData** và **TextTheme** để định nghĩa kiểu chữ, màu sắc chung, đảm bảo tính nhất quán trong toàn bộ ứng dụng.
* Dễ thay đổi **style toàn cục** hoặc **override** riêng cho vùng nhỏ nếu cần.

**Cách thực hiện:**

* Trong **MaterialApp,** bạn có thể đặt thuộc tính theme là một đối tượng **ThemeData.**
* **ThemeData** có các thuộc tính như:
* **colorScheme:** xác định bảng màu chính của ứng dụng.
* **textTheme:** xác định các kiểu chữ tiêu chuẩn **(headline, body, title, v.v).**

**Ví dụ:**

|  |
| --- |
| MaterialApp(    theme: ThemeData(      colorScheme: ColorScheme.fromSeed(seedColor: Colors.purple, brightness: Brightness.light),      textTheme: TextTheme(        displayLarge: TextStyle(fontSize: 72, fontWeight: FontWeight.bold),        titleLarge: TextStyle(fontSize: 30, fontStyle: FontStyle.italic),        bodyMedium: TextStyle(fontSize: 16),      ),    ),    home: MyHomePage(),  ); |

**Để sử dụng style từ theme:**

|  |
| --- |
| Text(    'Tiêu đề',    style: Theme.of(context).textTheme.titleLarge,  ) |

Nếu muốn **override theme** trong phần nhỏ của ứng dụng, thì có thể làm theo 2 cách sau:

* 1. Tạo một **Theme widget** mới với **ThemeData** riêng.
  2. Mở rộng **theme** hiện tại bằng **copyWith(...)** để thay đổi một phần nhỏ mà không thay đổi hết.

TÀI LIỆU THAM KHẢO

1. api.Flutter.dev: [api.flutter.dev/flutter/widgets/RichText](https://api.flutter.dev/flutter/widgets/RichText-class.html)
2. api.Flutter.dev: [api.flutter.dev/flutter/painting/TextStyle-class](https://api.flutter.dev/flutter/painting/TextStyle-class.html)
3. api.Flutter.dev: [api.flutter.dev/flutter/painting/TextSpan-class](https://api.flutter.dev/flutter/painting/TextSpan-class.html)
4. docs.Flutter.dev: [docs.flutter.dev/cookbook/design/themes](https://docs.flutter.dev/cookbook/design/themes)
5. docs.Flutter.dev: [docs.flutter.dev/ui/widgets/text](https://docs.flutter.dev/ui/widgets/text)
6. docs.Flutter.dev: [docs.flutter.dev/cookbook/design/fonts](https://docs.flutter.dev/cookbook/design/fonts)
7. FlutterDev Youtube: <https://www.youtube.com/@flutterdev>