

ISW-011

Desenv. para Dispositivos Móveis 1 (DDM1)

DDM-1 – ISW-011
Prof. Davi dos Reis

Aula 05 – Flutter – Primeiros passos

Dart – entrada de dados

Dart – Entrada e saída de dados

- Entrada:

- Deve-se fazer a importação de uma biblioteca específica: “dart.io”.
- Por isso, somente é possível utilizá-la em uma IDE: não dá para utilizá-la no <https://dartpad.dev/>!!

- Ex.:

```
import 'dart:io';

void main(){
  print ("Qual seu nome?");
  String nome = stdin.readLineSync();
  print ("Olá, $nome, seja bem-vindo(a).");
}
```

Prof. Davi

ISW-011 (DDM-1)

VS Code

3

Dart – Entrada e saída de dados

- Conversão de tipos (*type casting*):

```
import 'dart:io';

void main(){
  print ("Qual seu nome?");
  String nome = stdin.readLineSync();
  print ("Olá, $nome, seja bem-vindo(a).");
  print ("Qual sua idade?");
  String? temp = stdin.readLineSync();
}
```

```
if (input != null){
  int idade = int.parse(input);
  print ("Então você nasceu em $(2023 –
idade).");
}
else {
  print ("Não foi possível determinar o ano de
nascimento.");
}
}
```

Prof. Davi

ISW-011 (DDM-1)

VS Code

4

Hello, World! (Flutter)

Prof. Davi

ISW-011 (DDM-1)

5

Flutter – Hello, World (via prompt)

- Abrir um “Prompt de comando”
- Validar se o “path” do Flutter está ok
- Fazer o caminho no qual se deseja criar o projeto
- Então, digitar: “flutter create <<nome_projeto>>”
- Entrar na pasta do projeto (com o comando “cd <<nome_projeto>>”)
- Executar o projeto (“flutter run”)

Sugestão de nome para o projeto: “aula05_01”

Prof. Davi

ISW-011 (DDM-1)

6

Flutter – Primeiros passos: Widgets

- O Flutter é baseado em Widgets
- Widgets são componentes que são criados a partir de uma classe
- Um widget pode chamar outro dentro dele, criando widgets aninhados
- Para saber mais, vamos começar um novo projeto:
- VS Code → CTRL+SHIFT+P → “Flutter: New Project” → “Application”

Nome: ~~“app_hello”~~
“aula05_01”

Prof. Davi

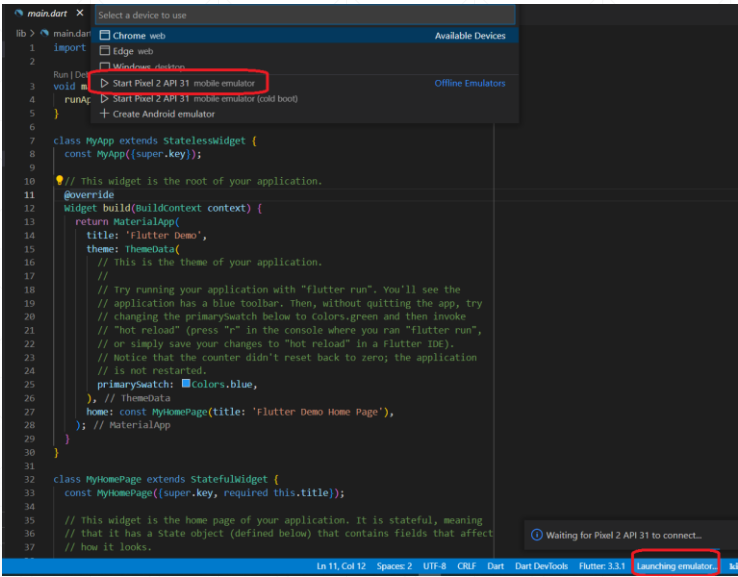
ISW-011 (DDM-1)

7

Flutter

Vamos agora executar o projeto.

Para isso abra o emulador no canto inferior direito da tela



Prof. Davi

ISW-011 (DDM-1)

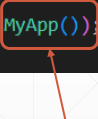
8

Flutter – Primeiros passos: Widgets

- Pode apagar todo o código, deixando apenas algo semelhante a:

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

Run | Debug | Profile
void main() {
  runApp(const MyApp());
}
```



- Agora há apenas o Widget **MyApp**:

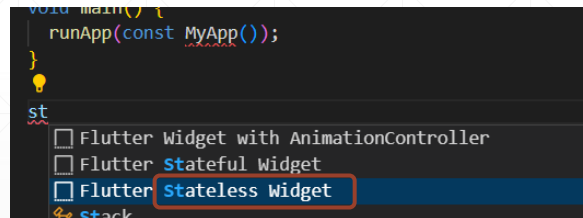
Flutter – Widgets

Stateless Widget x Stateful Widget

- **Stateless:**
 - É um componente que não guarda estado, ou seja, não armazena nada;
 - É utilizado apenas para exibição de um layout;
- **Stateful:**
 - É um componente que guarda estado, ou seja, armazena algo para ser repassado para outro componente ou para exibir na tela;

Flutter – Widgets

- Vamos criar o código novamente, agora entendendo o que ocorre
- Logo abaixo do “}”, basta começar a digitar “st”



```
void main() {  
  runApp(const MyApp());  
}  
st  
  ☐ Flutter Widget with AnimationController  
  ☐ Flutter StatefulWidget  
  ☒ Flutter StatelessWidget  
  ☐ Stack
```

Flutter – Primeiros passos: MaterialApp

- O MaterialApp é o widget (componente) principal para início de um app Flutter.
- A partir dele é possível chamar outros widgets para montar o app.
- O MaterialApp fornece um layout orientado ao **Material Design**.
- **Material Design**, em resumo, pode ser considerada a “linguagem de design” (ou “design system”) para apps desenvolvida pelo Google em 2014.

Flutter – Primeiros passos: MaterialApp

- Vamos criar o exemplo ao lado:
- Pontos importantes:
 - Dentro do widget MaterialApp há o atributo “theme”, que está chamando outro widget, o ThemeData utilizando um construtor
 - Dentro do ThemeData é definida a cor primária do app
 - O MaterialApp possui outro atributo chamado “home”, onde é definido o conteúdo que vai aparecer na primeira tela do app ao ser aberto
 - Nesse caso, o “home” está definido com outro widget (Container)
 - O widget Container funciona como uma grande caixa, dentro da qual é possível criar novos widgets ou apenas preenchê-la com uma cor (como feito no exemplo)

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

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

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        primaryColor: Colors.blue,
      ),
      home: Container(
        color: Colors.orange,
      ));
  }
}
```

Prof. Davi

ISW-011 (DDM-1)

13

Flutter – Primeiros passos: Scaffold

- O Scaffold é um widget muito utilizado, pois ele permite criar outros widgets do Material Design dentro dele
- Exemplo:
 - Pontos importantes:
 - appBar: Widget de título da tela
 - body: Widget de conteúdo da tela

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

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

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        primaryColor: Colors.red,
      ),
      home: Scaffold(
        appBar: AppBar(
          title: Text("App Hello"),
        ),
        body: Container(
          color: Colors.orange,
        ),
      ));
  }
}
```

Prof. Davi

Flutter – Primeiros passos: Scaffold

- Ainda com o Scaffold, podemos facilmente adicionar um menu (drawer) e um botão (floatingActionButton) na tela
- Exemplo:

```
@override
Widget build(BuildContext context) {
  return MaterialApp(
    debugShowCheckedModeBanner: false,
    theme: ThemeData(
      primaryColor: Colors.red,
    ),
    home: Scaffold(
      appBar: AppBar(
        title: Text("App Hello"),
      ),
      body: Container(
        color: Colors.orange,
      ),
      drawer: Container(
        color: Colors.amber,
      ),
      floatingActionButton: FloatingActionButton (
        onPressed: () {},
      ),
    ),
  );
}
```

Prof. Davi

ISW-011 (DDM-1)

15

Flutter – MaterialApp x Scaffold

- Basicamente, temos:
 - **MaterialApp**: define o ponto inicial do app (o primeiro Widget que será informado no "runApp"), indicando que ele vai usar componentes e seguir padrões do Material Design. Caso fosse seguir algo voltado aos padrões Apple, ele poderia ser substituído pelo CupertinoApp, por exemplo.
 - **Scaffold**: é responsável pela implementação básica de layout do Material Design, como os locais e estilos básico para a AppBar, conteúdo, etc. (se fosse no CupertinoApp, o equivalente seria usar um CupertinoPageScaffold).
- Em suma, o conjunto dos dois (o ponto de entrada MaterialApp + suas telas com Scaffold) é o que possibilita essa aplicação do Material Design.

```
@override
Widget build(BuildContext context) {
  return MaterialApp(
    debugShowCheckedModeBanner: false,
    theme: ThemeData(
      primaryColor: Colors.red,
    ),
    home: Scaffold(
      appBar: AppBar(
        title: Text("App Hello"),
      ),
      body: Container(
        color: Colors.orange,
      ),
      drawer: Container(
        color: Colors.amber,
      ),
      floatingActionButton: FloatingActionButton (
        onPressed: () {},
      ),
    ),
  );
}
```

Prof. Davi

ISW-011 (DDM-1)

16

Organizando o código em blocos

Prof. Davi

ISW-011 (DDM-1)

17

Flutter – organizando o código

- Utilizando métodos
- Neste caso, o conteúdo do Scaffold é encapsulado em um método (“metodoHome”), que é chamado no widget MaterialApp

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

void main() => runApp(MyApp());

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        primaryColor: Colors.blue
      ),
      home: metodoHome(),
    );
  }
}
```

```
metodoHome() {
  return Scaffold(
    appBar: AppBar(
      title: Text("App Hello"),
    ),
    body: Container(
      color: Colors.yellow,
    ),
  );
}
```

Prof. Davi

ISW-011 (DDM-1)

18

Flutter – organizando o código

- Utilizando **classes (Widgets)**
- Neste caso, o conteúdo do Scaffold é encapsulado em uma nova classe (“WidgetHome”), que é chamado no widget MaterialApp

main.dart

```
import 'corpo.dart';
import 'package:flutter/material.dart';

void main() => runApp(MyApp());

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      theme: ThemeData(primaryColor: Colors.blue),
      home: WidgetHome(),
    );
  }
}
```

corpo.dart

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("App Hello"),
        centerTitle: true,
      ),
      body: Container(
        color: Colors.orange,
      ),
    );
  }
}
```

Componentes básicos

Flutter – Widget Text

- O widget Text é responsável por exibir um texto na tela.
- Por ter de ficar dentro de outros widgets, vamos utilizar o atributo child dos widgets.

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

void main() => runApp(MyApp());
class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
```

```
@override
Widget build(BuildContext context) {
  return MaterialApp(
    debugShowCheckedModeBanner: false,
    theme: ThemeData(primaryColor: Colors.blue),
    home: Scaffold(
      appBar: AppBar(
        title: Text("Hello world, SI!"),
        backgroundColor: Colors.redAccent,
        centerTitle: true,
      ),
      body: Container(
        color: Colors.white,
        child: Center(
          child: Text("Hello, turma do 5o. SI!"),
        ),
      ),
    ),
  );
}
```

Prof. Davi

Flutter – Widget Text

- Também é possível utilizar outros widgets para formatar o Text

```
body: Container(
  color: Colors.white,
  child: Center(
    child: Text("Hello, turma do 5o. SI!"),
  ),
),
```

```
child: Text(
  "Hello Turma!",
  style: TextStyle(
    fontSize: 30,
    color: Colors.blue,
  ),
),
```

Prof. Davi

ISW-011 (DDM-1)

22

Flutter – Widget Text

- ...e é possível ir além:

```
color: Colors.blue,  
fontWeight: FontWeight.bold,  
fontStyle: FontStyle.italic,  
decoration: TextDecoration.underline,  
//decoration: TextDecoration.overline,  
decorationColor: Colors.red,  
decorationStyle: TextDecorationStyle.dashed,  
) ,
```

Obrigado!