Tópicos Especiais em Desenvolvimento Móvel

M.Sc Nairon Viana

nairon.viana@ifpi.edu.br

Tópicos Especiais em Desenvolvimento Móvel

Roteiro

- Revisão dos Conceitos – Estruturas Dart+VS Code + Plugin
- Variáveis, Tipos, Comandos de Entrada e Saída, Estruturas de Decisão Simples e Composta, Estruturas de Repetição
- Listas em Dart
- Exercícios

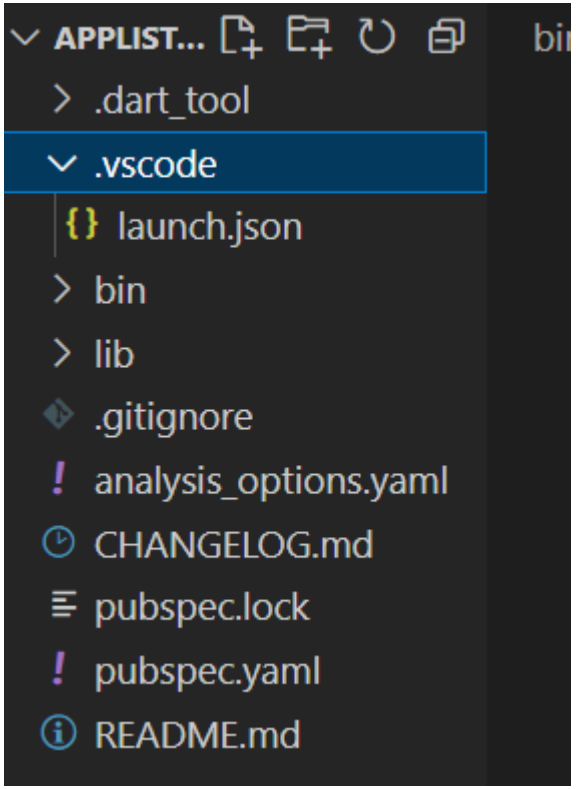
Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

Instalação da Ferramenta Linux (Dart+Flutter + Plugin VS Code)
Criação e Configuração da App Dart com Plugin
Estrutura de uma Aplicação Dart Console no VS Code
Configurando o app nos arquivos dart das pastas **bin/** e **lib/**
Configurando o arquivo launcher.json na pasta .vscode
Executando um App Simples



Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

1. Módulo principal **main()**

2. Módulos externos **void()** na pasta lib

3. Importação de módulos externos da pasta lib

bin > applistas.dart > ...

```dart
import 'package:applistas/lista1.dart' as l1;
import 'package:applistas/lista1.dart' as l2;

Run | Debug
void main(List<String> arguments) {
  l1.mostrarIdades();
  l2.mostrarNomes();
}
```

Módulo principal → main

lista1.dart

lib > lista1.dart > ...

```dart
void mostrarNomes() {
  print('Lista de Nomes: ');
  print('Nome 0: ${nomes[0]}');
  print('Nome 1: ${nomes[1]}');
  print('Nome 2: ${nomes[2]}');
  print('Nome 3: ${nomes[3]}');
}
```

Módulo e funções importadas (lib)

Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

1. Módulo principal **main()**
2. Módulos externos **void()** na pasta lib
3. Importação de módulos externos da pasta lib

applistas.dart > ...

```dart
import 'package:applistas/lista1.dart' as l1;
import 'package:applistas/lista1.dart' as l2;

Run | Debug
void main(List<String> arguments) {
  l1.mostrarIdades();
  l2.mostrarNomes();
}
```

Módulo principal → main

lista2.dart

lib > lista2.dart > ...

```dart
// De String
List<String> nomes = ['Maria', 'Pedro', 'Marcelo'];

void mostrarNomes() {
  print('Lista de Nomes: ');
  print('Existem ${nomes.length} nomes!');
  for (int i = 0; i < nomes.length; i++) {
    print('Nome $i: ${nomes[i]}');
  }
}
```

Módulo e funções importadas (lib)

Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

Declaração de Variáveis e Tipos em Dart/Flutter

Podem ser declaradas de forma simples, variáveis são tipadas. Tipos suportados, mais comuns: **int**, **double**, **String**, **bool**, **list**, etc...

Para acessar as variáveis e mostrar seus valores usamos **comandos de saída** (print)

Para concatenar o valor de uma variável usamos o operador **$** dentro da string de saída

exemplo1.dart > ...

```
17    */
      Run | Debug
18    void main() {
19      // Declaração de Variáveis em Dart
20      int idade = 18;
21      double nota = 7.5;
22      String nome = "Maria";
23      bool tipo = false;
24      List<String> frutas = ["Maça", "Banana", "Melão"];
25      //Comando de Saída em Dart
26      print("Nome do usuário: $nome");
27      print("Idade do usuário: $idade");
28      print("Nota do usuário: $nota");
29      print("Valor lógico: $tipo");
30      print("1o Item da Lista: ${frutas[0]}");
31      print("2o Item da Lista: ${frutas[1]}");
32      print("3o Item da Lista: ${frutas[2]}");
33    }
```

Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

Declaração de Variáveis e Tipos em Dart/Flutter

```
String? sn1, sn2;
double? n1, n2;
```

Variáveis declaradas com a **?**: são aquelas em que o valor pode ser indefinido (pode assumir valores nulos)

Sempre que declarar com **?,** para acessar o valor da variável utilize **!** (primeiro acesso)

```
void calculos() {
  double soma = n1! + n2!;
  double subt = n1! - n2!;
  double prodt = n1! * n2!;
  double divi = n1! / n2!;
}
```

Variáveis podem ser declaradas:

- Dentro de Módulos (Locais)
- Fora de Módulos (Globais)

Programação em Dart/Flutter

Elementos da Linguagem

- **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

Entrada de Dados em Dart

Importação da biblioteca **dart::io**

Variável global **stdin** com o comando **readLineSync()** para ler uma entrada de dados do terminal

Para usar o valor da entrada com tipos diferentes de String deve-se fazer a conversão

- Inteiro: **int.parse**
- Real: **double.parse**

lib > questao1.dart > lerValores

```dart
import "dart:io";

String? sn1, sn2;
double? n1, n2;

void lerValores() {
  print('Digite o primeiro número: ');
  sn1 = stdin.readLineSync();
  n1 = double.parse(sn1!);

  print('Digite o segundo número: ');
  sn2 = stdin.readLineSync();
  n2 = double.parse(sn2!);
}
```

Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

Estruturas de Decisão

Simples e Composta

if → simples
if..else → simples

```dart
import 'dart:io';

void testes() {
  print("Qual a sua idade?");
  String? input = stdin.readLineSync();
  print("Sua idade é $input.");
  if (idade >= 18) {
    print('Maior de Idade!');
  }
}
```

Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

Estruturas de Decisão

Simples e **Composta**

if → simples
if..else → simples

```dart
double media = (nota1 + nota2) / 2;

// Estruturas de descisão
if (media >= 7) {
  print('aprovado com media: $media');
} else if (media >= 4.0) {
  print('exame final com media: $media');
} else {
  print('reprovado com media: $media');
}
```

Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

Estruturas de Decisão

Estrutura Múltipla

Switch..case
break..default

exemplo3.dart > ...

Run | Debug

```dart
void main() {
  // Estrutura seleção múltipla
  int semana = 7;
  switch (semana) {
    case 1:
      print('Segunda-Feira');
      break;
    case 2:
      print('Terça-Feira');
      break;
    default:
      print('Valor inválido!');
      break;
  }
}
```

Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

Estruturas de Repetição
Laço for
Início e final definido
Variável contadora inteira

```
Run | Debug
void main() {
  int maximo = 10;
  // Laço for crescente
  for (var x = 0; x < maximo; x++) {
    print('Valor de x: $x');
  }
}
```

Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

Estruturas de Repetição
Laço while
Condição Lógica analisada no **INÍCIO**
Não precisa de variável contadora
Trecho de código é repetido **enquanto** a condição for verdadeira

```
Run | Debug
void main() {
  int x = 0;

  while (x < 10) {
    print('Valor de x: $x');
    x++;
  }
  print('Valor final de x: $x');
```

Programação em Dart/Flutter

Elementos da Linguagem

– **Recursos já Estudados:**

Recursos já estudados do Ambiente DART/vs Code:

Estruturas de Repetição
Laço do..while
Condição Lógica analisada no **FINAL**
Não precisa de variável contadora
Trecho de código é repetido **enquanto** a condição for verdadeira

```
Run | Debug
void main() {
  int x = 0;
  do {
    print('Valor de x: $x');
    x++;
  } while (x < 10);
  print('Valor final de x: $x');

}
```

Programação em Dart/Flutter

Variáveis compostas → Listas em Dart

Tópicos Especiais em Desenvolvimento Móvel

Listas em Dart

- Definição de uma Lista Dart
- Declaração de Listas em Dart
- Operações na Lista em Dart
  - Acessando Elementos, Mostrando elementos
  - Inserção
  - Remoção
  - Busca por Elemento / Busca por índice
  - Copiar 2 Listas
- Exercícios

Programação em Dart/Flutter

Listas em Dart

– Definições

Geral

Listas em Dart são grupos de dados, estruturas compostas semelhantes aos **arrays/vetores** onde uma única variável pode fazer referência a cada item do conjunto

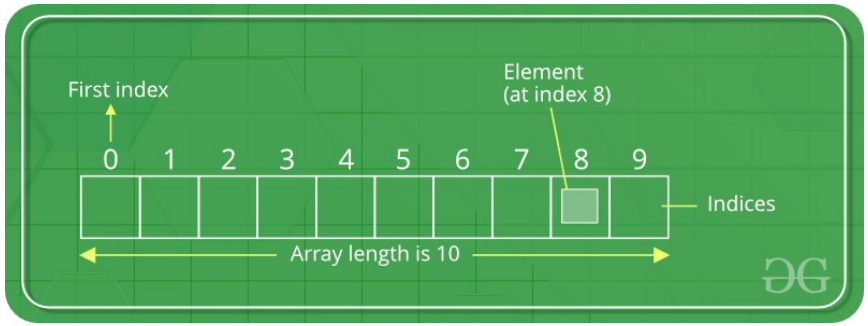
Listas em Dart

Cada item da lista assume uma posição lógica sequencial

Cada posição é identificada por um index inteiro

A posição inicial é o index **zero** (0)

Listas podem ser definidas de todos os tipos suportados em Dart



Lista em Dart com 10 elementos indexados de 0 a 9

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Listas podem ser declaradas:

1. Sem especificar seu tipo (usando a palavra reservada **var**)

2. Especificando seu tipo (usando a classe List e o tipo **entre <>**)

Na declaração inicial deve-se atribuir **[ ] (colchetes)** à variável definida pela lista

```dart
var nomes = [];
```

Listas sem tipos definidos

```dart
List<String> paises = ['Brasil', 'Japao', 'Russia'];
```

Listas com tipos definidos

Programação em Dart/Flutter

Listas em Dart

- **Definições**

Acessando o tamanho de uma lista:

Função → **length**

```dart
var nomes = [];
print('Tamanho da lista de nomes: ${nomes.length}'); // 0

List<String> paises = ['Brasil', 'Japao', 'Russia'];
print('Tamanho da lista de países: ${paises.length}'); // 3
```

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Verificando se a lista está vazia:	**Verificando se a lista não está vazia:**
Função → **isEmpty**	Função → **isNotEmpty**

```
Run | Debug
void main() {
  var nomes = [];
  if (nomes.isEmpty) {
    print('Nomes - VAZIA');
  }

  List<String> paises = ['Brasil', 'Japao'];
  if (paises.isNotEmpty) {
    print('Países - NÃO VAZIA');
  }
}
```

Programação em Dart/Flutter

Listas em Dart

- **Definições**

Acessando elementos de uma lista – elementos do início e final da lista

Exemplo: **var nomes = ['Maria', 'Pedro', 'José', 'Ana'];**

- Podem ser acessados usando os índices:
  - **Índice 0:** primeiro elemento – **nomes[0]**
  - **Índice length-1:** último elemento – **nomes[length-1]**
- Podem ser acessados usando funções (**first – primeiro** e **last – último**)
  - **nomes.first** – primeiro elemento
  - **nomes.last** – último elemento

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Acessando elementos de uma lista – elementos do início e final da lista

lista4.dart > ...

```
Run | Debug
1  void main() {
2    var nomes = ['Maria', 'Pedro', 'José', 'Ana'];
3    print('Primeiro elemento da lista: ${nomes.first}'); // Maria
4    print('Primeiro elemento da lista: ${nomes[0]}'); // Maria
5    print('Úlltimo elemento da lista: ${nomes[nomes.length - 1]}'); // Ana
6    print('Último elemento da lista: ${nomes.last}'); // Ana
7  }
8
```

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Acessando demais elementos de uma lista – Iterando com laço for

```
Run | Debug
1  void main() {
2    var nomes = ['Maria', 'Pedro', 'José', 'Ana'];
3
4    for (var i = 0; i < nomes.length; i++) {
5      print('Elemento da posição $i: ${nomes[i]}');
6    }
7  }
```

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Acessando demais elementos de uma lista – Iterando com laço for in
laço for exclusivo para conjuntos/listas

Acessando a posição de um elemento da lista – usando função **indexOf()**

```
Run | Debug
1  void main() {
2    var nomes = ['Maria', 'Pedro', 'José', 'Ana'];
3
4    for (var elemento in nomes) {
5      int pos = nomes.indexOf(elemento);
6      print('Elemento da posição $pos: $elemento');
7    }
8  }
```

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Acessando demais elementos de uma lista – Iterando com laço for in
laço for exclusivo para conjuntos/listas

Somando os elementos de uma lista de valores reais **(double)**

lista7.dart > main

Run | Debug

```dart
void main() {
  var precos = [4.25, 5.25, 1.25, 5.25];
  double soma = 0;

  for (var p in precos) {
    int pos = precos.indexOf(p);
    print('Preco da posição $pos: $p');
    soma = soma + p;
  }
  print('Soma total dos preços: $soma');
}
```

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Acessando demais elementos de uma lista – Iterando com laço <u>for in</u> laço for exclusivo para conjuntos/listas

Uso da função **stdout.write**: para impressão em mesma linha – diferente de print

```dart
import 'dart:io';
Run | Debug
void main() {
  // invertendo uma lista de inteiros
  List<int> numeros = [4, 3, 2, 1];
  // Lista: 4, 3, 2, 1
  int x = numeros[3];
  int y = numeros[2];
  numeros[3] = numeros[0];
  numeros[2] = numeros[1];
  numeros[0] = x;
  numeros[1] = y;
  for (var n in numeros) { //Lista: 1, 2, 3, 4
    stdout.write('$n'); // Imprime na mesma linha
  }
}
```

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Inserindo elementos em uma Lista Dart – função add

```
lista9.dart > main
    Run | Debug
1   void main() {
2     List<String> capitais = [];
3     capitais.add('Teresina');
4     capitais.add('Fortaleza');
5     capitais.add('São Luís');
6     capitais.add('Salvador');
7     for (var i = 0; i < capitais.length; i++) {
8       print('Capital da posição $i: ${capitais[i]}');
9     }
10  }
```

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Inserindo elementos em uma Lista Dart – função add – com entrada do usuário

lista10.dart > main

```dart
import 'dart:io';

Run | Debug
void main() {
  List<String> capitais = [];
  // Adicionando 3 elementos na lista
  int num = 3;
  int i = 0;
  String? cap = '';
  while (i < num) {
    print('Digite um nome de capital: ');
    cap = stdin.readLineSync();
    capitais.add(cap!);
    i++;
  }
  for (var c in capitais) {
    int pos = capitais.indexOf(c);
    print('Capital da posição $pos: $c');
  }
}
```

Programação em Dart/Flutter

Listas em Dart

- **Definições**

Removendo elementos em uma Lista Dart – função remove

```
lista11.dart > main
Run | Debug
1   void main() {
2     List<String> capitais = [];
3     capitais.add('Teresina');
4     capitais.add('Fortaleza');
5     capitais.add('São Luís');
6     capitais.add('Maceio');
7     capitais.add('Salvador');
8     // removendo 'Teresina'
9     capitais.remove('Teresina');
10    // Removendo 'Salvador'
11    capitais.remove(capitais.last);
12    // Removendo 'Fortaleza'
13    capitais.remove(capitais.first);
14    for (var c in capitais) {
15      print('Capital: $c');
16    }
17  }
18
```

Programação em Dart/Flutter

Listas em Dart

- **Definições**

Removendo elementos em uma Lista Dart – função remove – com índice

lista12.dart > main

Run | Debug

```dart
void main() {
  List<String> capitais = [];
  capitais.add('Teresina');
  capitais.add('Fortaleza');
  capitais.add('São Luís');
  capitais.add('Maceio');
  capitais.add('Salvador');
  // removendo 'Teresina'
  capitais.removeAt(0);
  // Removendo 'Salvador'
  capitais.removeAt(capitais.length - 1);
  // Recuperando 'Fortaleza'
  String fort = capitais.elementAt(0);
  capitais.remove(fort);
  for (var c in capitais) {
    print('Capital: $c');
  }
}
```

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Buscar elemento em uma lista – função elementAt(index)

Retorna um elemento (se disponível) em uma lista de itens

lista13.dart > main

Run | Debug

```dart
void main() {
  List<String> nomes = ['Maria', 'Pedro', 'Ana'];
  // Mostrando nomes:
  for (var n in nomes) {
    print('Nome: $n');
  }
  // Buscando pelo nome 'Maria'
  // Passo 1: buscar pelo o índice
  String ped = 'Pedro';
  int index = nomes.indexOf(ped);
  // Se índice == -1, existe o elemento
  // Senão, não existe o elemento
  if (index != -1) {
    print('O elemento $ped existe na posição: $index');
    print('Remover o elemento $ped');
  } else {
    print('O elemento $ped não existe na lista');
  }
}
```

Programação em Dart/Flutter

Listas em Dart

– **Definições**

Copiar 2 listas com a função addAll(): adiciona uma lista ao final da outra

lista14.dart > main

```
Run | Debug
1   void main() {
2     List<String> frutas_ = ['Maça', 'Banana', 'Pera'];
3     List<String> frutas = ['Uva', 'Goiaba'];
4
5     frutas_.addAll(frutas);
6
7     for (var f in frutas_) {
8       print('Fruta: $f');
9     }
10  }
```

Programação em Dart/Flutter

Lista de Exercícios – Prática em Laboratório – Lista em Dart

Tópicos Especiais em Desenvolvimento Móvel

M.Sc Nairon Viana

nairon.viana@ifpi.edu.br