

ISW-011

Desenv. para Dispositivos Móveis 1 (DDM1)

DDM-1 – ISW-011
Prof. Davi dos Reis

Aula 04 – DART – Primeiros passos, conceitos e OO

Dart – conceitos básicos

Dart – Introdução

- Características gerais:
 - Linguagem de programação criação pela empresa Google;
 - Fortemente utilizada para criar apps para dispositivos móveis;
 - É *cross-platform*, ou seja, permite criar apps para Android e iOS;
 - Também permite criar para desktop e web;
 - Semelhante ao Java em alguns aspectos;
 - Utiliza uma VM chamada Dart VM – permite *hot reload* ao desenvolver;

Dart – Introdução

- Plataformas:
 - Nativa
 - Para apps que serão executados em dispositivos móveis ou desktop;
 - Dart VM com compilação *just-in-time* (JIT) e compilador *ahead-of-time* (AOT);
 - Web:
 - Para apps que tem como alvo os navegadores web;
 - Transforma Dart em JavaScript;

Dart – Introdução

- Bibliotecas:
 - Possui um extenso conjunto de bibliotecas, tais como:
 - dart:core – tipos próprios, collections e demais funcionalidades padrões;
 - dart:collection – tipos de collections mais estruturadas, como filas, listas ligadas, hashmaps e árvores binárias;
 - dart:convert – codificadores e decodificadores entre diferentes representações de dados, incluindo JSON e UTF-8;
 - dart:math – funções, constantes matemáticas e geradores de números aleatórios;
 - dart:io – manipulação de arquivos, criação de socket, conexão HTTP e outros tipos de I/O;
 - dart:async – suporte para programação assíncrona, com classes como Future e Stream;
 - dart:typed_data – listas que manipulam eficientemente dados de tamanho fixo (p. ex.: inteiros de 8 bits) e tipos SIMD;
 - dart:async – suporte para programação assíncrona, com classes como Future e Stream;
 - dart:ffi – interfaces com funções estrangeiras para interop com outros códigos lembram uma interface em C;
 - dart:isolate – programação concorrente, usando *workers* independentes e semelhantes às threads, mas que não compartilham memória, trocando apenas mensagens;
 - dart:html - elementos HTML e outros recursos para apps que precisam interagir com o navegador e o DOM (*document object model*) da página.

Dart – Introdução

- Pontos importantes:
 - Pode-se utilizar aspas ou apóstrofo ao manipular Strings;
 - É possível concatenar String com variáveis sem usar caracteres especiais;
 - Não é obrigatório especificar o tipo de variáveis, métodos ou parâmetros (ou seja, não é fortemente “tipada”);
 - Os operadores lógicos no Dart são: && (AND) e || (OR);

Dart – Projeto inicial padrão

```
void main() {  
  for (int i = 0; i < 5; i++) {  
    print ('hello ${i + 1}');  
  }  
}
```

VS Code
<https://dartpad.dev/>
<https://dartpad.dartlang.org/>

**Hello, World!
(Dart)**

Primeiro Projeto – “Hello, world”

```
void main() {

    final String msg = "Hello, world!";

    // Comentário em linha
    print("Exibirá a primeira mensagem!!");
    print("Mensagem: $msg");
}
```

<https://dartpad.dev/>
<https://dartpad.dartlang.org/>

Prof. Davi

ISW-011 (DDM-1)

9

Segundo Projeto – sub-rotina e variáveis

```
void main() {

    final String nome = "Rubens Lara";

    int a = 1;
    a = 3;

    var b = 2;
    int c = somar(a, b);

    // Comentário em linha

    /*
    Comentário em bloco
    */

    if (c > 0){
        print("A soma foi positiva");
    }
    else {
        print("A soma foi negativa");
    }

    print("Soma: R\$$ $c");
    print("Nome: $nome");
}

somar(a, b) {
    return a + b;
}
```

<https://dartpad.dev/>
<https://dartpad.dartlang.org/>

Prof. Davi

ISW-011 (DDM-1)

10

Dart – tipos de dados

Prof. Davi

ISW-011 (DDM-1)

11

Dart – Tipos de dados

- Saída:
 - Como já vimos, para exibir uma saída podemos utilizar o comando “print”;
- Ex.:

```
int idade = 20;
print("Minha idade é " + idade.toString()); // conversão manual para concatenar
print("Minha idade é $idade"); // exibindo o valor sem concatenar (interpolação)
print("Minha idade ano que vem será: ${idade + 1}"); // uso de expressão
```

VS Code

<https://dartpad.dev/><https://dartpad.dartlang.org/>

Prof. Davi

ISW-011 (DDM-1)

12

Dart – Tipos de dados

- Saída:

- Imprimindo outros tipos de dados;

- Ex.:

```
String name = "Rubens Lara";  
//int idade = 20; // já declarado no slide anterior  
double altura = 1.75;  
bool isGeek = true;  
List<String> lista = ["Fatec", "SI", "5º. ciclo"];  
print(name);  
print(idade);  
print(altura);  
print(isGeek);  
print(lista);
```

VS Code

<https://dartpad.dev/>

Dart – Classes (POO)

Dart – Classes

```
void main() {
  Pessoa p = Pessoa("Rubens Lara");
  int c = Calculadora.somar (7, 8);

  print("Soma: $c");
  print("Nome: ${p.nome}");
}
```

```
class Pessoa {
  String nome;

  Pessoa (this.nome);
}

class Calculadora {
  static somar (a, b) {
    return a + b;
  }
}
```

VS Code
<https://dartpad.dev/>
<https://dartpad.dartlang.org/>

Prof. Davi

ISW-011 (DDM-1)

15

Dart – Classes e Lista

```
void main(){
  Carro carro1 = Carro("HB20");
  Carro carro2 = Carro("Gol");
  Carro carro3 = Carro("Argo");

  final carros = [carro1, carro2, carro3];
  // carros.add(carro1); carros.add(carro2); carros.add(carro3);

  print("Carros disponíveis: $carros, tamanho: ${carros.length}");

  // looping com For Each
  for (Carro carroTemp in carros){
    print("Carro disponível: ${carroTemp.modelo}");
  }

  // looping com convencional
  for (int i = 0; i < carros.length; i++) {
    print(" Carro: ${carros[i].modelo}");
  }
}
```

```
class Carro {
  String modelo;

  Carro (this.modelo);

  String toString(){
    return modelo;
  }
}
```

VS Code
<https://dartpad.dev/>
<https://dartpad.dartlang.org/>

Prof. Davi

ISW-011 (DDM-1)

16

Dart – Herança e Polimorfismo

```
void main() {
  Carro carro1 = Carro("HB20", 2019);
  Carro carro2 = Carro("Gol", 2008);
  Carro carro3 = Carro("Argo", 2021);

  final carros = [carro1, carro2, carro3];
  // carros.add(carro1);
  // carros.add(carro2);
  // carros.add(carro3);

  print("Lista: $carros, quantidade de carros: ${carros.length}");
  for (Carro carroTemp in carros) {
    //print("Carro: ${carroTemp.modelo} - Ano: ${carroTemp.ano}");
    print(carroTemp.dadosVeiculo());
  }
}

class Veiculo {
  String modelo;
  Veiculo(this.modelo);
```

```
String dadosVeiculo() {
  return modelo;
}

class Carro extends Veiculo {
  int ano;
  Carro(nomeModelo, this.ano) : super(nomeModelo);

  // usado para imprimir todos os objetos da lista
  @override
  String toString() {
    return "$modelo: $ano";
  }
  @override
  String dadosVeiculo() {
    // return "Modelo: ${super.dadosVeiculo()}; ano fabricação: $ano";
    return "Carro: $modelo e Ano: $ano";
  }
}
```

VS Code
<https://dartpad.dev/>

Dart – Map

```
void main() {
  Carro carro1 = Carro("HB20");
  Carro carro2 = Carro("Gol");
  Carro carro3 = Carro("Argo");

  // map é um par de chave e valor, igual o json
  final carrosMap = {"1": carro1, "2": carro2};
  carrosMap["3"] = carro3;

  print("Imprimindo a lista toda de uma vez...");
  print("Lista: $carrosMap, length: ${carrosMap.length}");

  // looping pela "key" do map
  print("\nImprimindo a lista com foreach a partir da 'key'");
  for (String id in carrosMap.keys) {
    final carroTemp = carrosMap[id];
    print(" >> ${carroTemp?.modelo}");
    // o "?" acima serve porque "modelo" pode ser nulo
  }
}
```

```
// looping pelo "values" do map
print("\nImprimindo a lista com foreach a partir do 'value'");
for (Carro carroTemp in carrosMap.values) {
  print(" >> ${carroTemp.modelo}");
}

class Carro {
  String modelo;

  Carro(this.modelo);

  @override
  String toString() {
    return modelo;
  }
}
```

VS Code
<https://dartpad.dev/>

Dart – Mixin

Prof. Davi

ISW-011 (DDM-1)

19

Dart – Mixin

```
void main() {
  Carro carro1 = Carro("HB20");
  print("Modelo do carro: $carro1");
  carro1.acelerar(100);
  carro1.abastecer(40, "gasolina");
}
```

```
class Carro extends Veiculo with Combustivel {
  String modelo;
  Carro(this.modelo);
```

```
  @override
  void acelerar(int velocidade) {
    print("Acelerando a $velocidade km/h");
  }
}
```

```
  @override
  String toString(){
    return modelo;
  }
}
```

```
abstract class Veiculo {
  void acelerar(int velocidade);
}
```

```
mixin Combustivel {
  abastecer(int qtde, String tipo) {
    print("Abastecendo $qtde litros de $tipo");
  }
}
```

[VS Code](https://dartpad.dev/)
<https://dartpad.dev/>

Prof. Davi

ISW-011 (DDM-1)

20

Dart – Mixin

- Observações:
 - Mixin permite adicionar métodos de outras classes na classe em questão **sem precisar utilizar** herança;
 - para isso, basta adicionar a palavra “**with**” na declaração da classe;
 - é possível utilizar o “mixin” para incluir mais do que uma classe na classe em questão: para isso, basta separar as demais com vírgula;
 - é possível também declarar uma classe como **abstract**: com isso, todos os métodos dessa classe só terão assinatura (serão abstratos) e sua implementação deverá ser feita na classe que implementá-la.

Prof. Davi

ISW-011 (DDM-1)

21

Dart – Arrow Function

```
void main() {
  Carro carro1 = Carro("HB20");
  print("Modelo do carro: $carro1");
  carro1.acelerar(100);
  carro1.abastecer(40, "gasolina");
}

class Carro extends Veiculo with Combustivel {
  String modelo;
  Carro(this.modelo);

  @override
  void acelerar(int velocidade) {
    print("Acelerando a $velocidade km/h");
  }
}
```

```
@override
String toString() => modelo;
}

abstract class Veiculo {
  void acelerar(int velocidade);
}

mixin Combustivel {
  abastecer(int qtde, String tipo) {
    print("Abastecendo $qtde litros de $tipo");
  }
}
```

VS Code
<https://dartpad.dev/>

Prof. Davi

ISW-011 (DDM-1)

22

Dart – Arrow Function

- Observações:
 - é possível criar funções de forma mais resumida, utilizando apenas uma seta;
 - com isso, não é preciso ter as chaves e nem a palavra “return”;
 - é possível apenas para retornos e expressões, não para declarações.

Dart – Exercícios

Dart – entrada de dados

Prof. Davi

ISW-011 (DDM-1)

26

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).");
}
```

VS Code

Prof. Davi

ISW-011 (DDM-1)

27

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.");  
    }  
  }
```

[VS Code](#)

Prof. Davi

ISW-011 (DDM-1)

28

Obrigado!

Prof. Davi

ISW-011 (DDM-1)

29