Desenvolvimento para Dispositivos Móveis I

Prof. Jorge Luiz Chiara

Dart

Linguagem Dart

- A Linguagem DART
  - Link – dart.dev
  - Criada em 2011 pela Google com o objetivo principal para a utilização no desenvolvimento de aplicações multiplataforma.
    - Dispositivos móveis através do Flutter
    - WEB
    - Desktop

Linguagem Dart

- Dart é uma linguagem de Programação open-source.
- Orientada a Objetos com sintax estilo “C”

Linguagem Dart

- Instalação do Dart SDK
  - dart.dev
    - Get Dart
      - Seguir orientação
        - » Install SDK
          - Utilize “Install using setup wizard”

Get the Dart SDK

The Dart SDK has the libraries and command-line tools that you need to develop Dart web, command-line, and server apps. If you're developing only mobile apps, then you don't need the Dart SDK; just install Flutter.

To learn about other tools you can use for Dart development, see the Dart tools page. To learn about what's in the SDK, see Dart SDK overview.

Install the Dart SDK

As the following instructions show, you can use a package manager to easily install and update the Dart SDK. Alternatively, you can build the SDK from source or download the SDK as a zip file.

⚠ Dart tools may send usage metrics and crash reports to Google. By downloading the Dart SDK, you agree to the Google Terms of Service. Note: The Google Privacy Policy describes how data is handled in this service.

Windows | Linux | Mac

Choose one of these options:

- Install using Chocolatey
- Install using a setup wizard

Install using a setup wizard

Alternatively, use the community-supported Dart SDK installer for Windows. You can use the wizard to install **stable** or **dev** versions of the Dart SDK.

Utilizando

- Ambiente de Desenvolvimento Dart
  - Visual Studio Code
- Links
  - https://dart.dev/#try-dart
  - https://dart.dev/codelabs

Dart-Comentários

- // define um comentário de linha

- /*
  define um bloco de comentários
- */

Dart-Variáveis e tipos de dados

- Números:
  - int
  - double
- String
  - Representado por ‘ ’ ou “ ”
- Boolean
  - true ou false
- Listas e Maps

Dart-Variáveis e tipos de dados

- Declaração:
  - Utilizando a palavar *var <nome_variavel>;*
    - Ex: var x;
      - x terá o valor null.
  - Utilizando um tipo de dado <*tipo nome_variavel>;*
    - *Ex: var x="Dispositivos Móveis";*
      - *x, no caso, será uma String*
    - *Ex: x="Dispositivos Móveis"; ou*
    - *x='Dispositivos Móveis';*
      - *Declaração explícita de um tipo String*

Dart-Variáveis e tipos de dados

– Embora haja uma prática para declarar variáveis utilizando *var* quando ela for uma **variável local**. Quando for **variável global**, utilizar a notação de tipo, faça sua escolha.

Dart-Variáveis e tipos de dados

- Utilizando a palavra *dynamic*
  - Ex: *dynamic x;*
    - Desta forma, indicamos ao Dart que a variável x não tem um tipo predefinido. Assim, seu tipo poderá mudar no decorrer da aplicação conforme necessidade.

Dart-Variáveis e tipos de dados

- Importante:
  - Na linguagem Dart, tudo é um objeto, mesmo o mais simples número, funções e até *null.* Todos são objetos, instâncias de classes; e, todos são estendidos a partir de uma classe comum, ***Object.***
  - *Object y=10;*

- ***Exemplo:***

```
main() {
        var x=10;
        print('oi x=' + x.toString());
        Object y=10;
        print("oi y=" + y.toString());
}
```

Dart-Constantes

A declaração *const* ou *final* declara uma constante.

Ex: const x = 10;

const pi = 3.1415;

const String nome = “Dispositivos Móveis”;

Ou

final x= 10;

final nome = “Dispositivos Móveis”;

A diferença é que *const* são definidas em tempo de compilação e não funcionaria, por exemplo em *const data=DateTime(now);*

Mas, funcionaria em: final x=DateTime(now);

*Ou seja, **final** declara a constante uma vez, mas pode definir no tempo de execução; enquanto que **const** indica que ela só pode ser definida uma vez, com seu valor já conhecido em tempo de compilação.*

Dart-String

- Inicializada com *aspas simples* ou *duplas*
- Inclui expressões com o uso da sintaxe:
  - *${expressão}* ou
  - *$identificador*
- *Exemplo:*

```
main() {
  String s1="Alo Dart!!!";
  String s3="$s1 muito legal!";
  String s2="${s1} muito legal!";     //desnecessário!!!!!
  print(s2);
  print(s3);
}
```

- Concatenação (+):
  - String s4=s1+s2;

Dart-Valores numéricos

- Os tipos ***int*** *e* ***double*** são subclasses de ***num***
  - *Podemos utilizar:*
    - *num x=3; //número inteiro*
    - *num y=3.3; //número double*
    - *int z=3;*
    - *double t=3.3;*
  - *A transformação número para String e vice-versa:*
    - *String sz = x.toString();*
    - *double w = double.parse(sz);*

Dart – experimente!

```
main() {
        String s1="Alo Dart!!!";
        String s3="$s1 muito legal!";
        String s2="${s1} muito legal!";
        print(s2);
        print(s3);
        print(s1+" muito legal!!!");
        var x1=10;
        print("$x1");
        var s4=s1+s2;
        print(s4);
        num x=3;    //número inteiro
        num y=3.3; //número double
        int z=3;
        double t=3.3;
        String st = x.toString();
        double w = double.parse(st);
        print("\nx=$x \ny=$y \nz=$z \nt=$t \nst=$st \nw=$w");

}
```

Dart – valor boleano

- As variáveis booleanas são do tipo *bool* e admitem os valores *true* ou *false*

```
main() {
  bool v=true;
  print(v);
  print(!v);
  print(v && !v);
  print(v && v);
  print(!v && !v);
  print(v || !v);
  print(!v || v);
  print(!v || !v);
}
```

Dart - Listas

- A classe List instancia um lista de valores utilizando da seguinte sintaxe:
- List nomelista = [];
  - Exemplo:
    - List n = [1,2,3,4,5];
    - List n2 = [];
    - n.add(6);
    - n2.add(1);

Dart - Listas

```
main() {
  List n=[1,2,3,4,5];
  print(n);
  for (var e in n){
    print(e);
  }
  n.add(6);
  n.add(3.5);
  print(n);
  print(n.length);
}
```

```
main(){
  print("Operando listas");
  List mlista = [1,3,4,11,33];
  print(mlista);
  mlista.add(9);
  print(mlista);
  mlista.sort();
  print(mlista);
  var lista2 = List();
  lista2.add(4);
  print(lista2);
}
```

```
var lista2 = List();
lista2.add(4);
print(lista2);
print("tamanho:"+lista2.length.toString());
print("primeirio:"+mlista.first.toString());
print("ultimo:"+mlista.last.toString());
print(mlista.isEmpty);  //pode utilizar o not !
print(mlista.isNotEmpty);
mlista.insert(3,3);
print('elemento inserido: ${mlista[2]}');
print(mlista);
}
```