Mobile App Dev

Table of Contents

Installation

Creating a project

Keywords

Tasks

- *Day 1*

Dart Programming Language

- ***Datatypes***
  - 1. Numbers
  - 2. Strings
  - 3. Booleans
  - 4. Lists
  - 5. Maps
  - 6. Sets
  - 7. Runes
- *Variables*

Installation

1. Go to https://docs.flutter.dev/get-started/install/
2. Choose your operating system.
3. Choose the platform and follow procedure.
4. move the zip file to a folder in c : drive.

5. Extract the zip file.
6. Update windows path variable.
7. Run **flutter doctor** on the cmd.

Creating a project

command: flutter create [project name]

project name cannot use upper case letters

running the project:

command: flutter run

Open the folder in vs code.

Keywords

1. Integer
2. Double
3. void
4. dynamic

Tasks

Day 1

- installation
- project creation
- datatypes - string operations

- run the app on chrome and emulator

Dart Programming Language

File extension for dart files is **.dart**

dart language is used to create applications using the flutter framework.

A dart file should have a **void main(){}** function that is the entry point of the program.

To print on the console, we use the **print()** function.

Every line of code or statement should end in a semicolon (;)

```
void main(){
    print("Hello World");
}
```

The above code prints “Hello World” on the console.

Datatypes

There are 7 datatypes in dart, they are:

1. Numbers

In dart, numbers are used to represent numeric literals.

2. Strings

It is used to represent a sequence of characters. It is a sequence of UTF-16 code units. The keyword string is used to represent string literals. String is a collection of characters enclosed in single, double or triple quotes.

String str1 = “ ABC|abc ”;

string in-built methods:

toUpperCase() → str1.toUpperCase() → “ ABC|ABC ” toLowerCase() → str1.toLowerCase() → “ abc|abc ”

trim() → str1.trim() → “ABC|abc”

trimLeft() → str1.trimLeft() → “ABC|abc ”

trimRight() → str1.trimRight() → “ ABC|abc”

split() → str1.split(”|”) → [” ABC”, “abc ”]

3. Booleans

True or False values

bool isTrue = true;

bool isFalse = false;

4. Lists

List is a collection of objects separated by commas and enclosed in square brackets.

[1, 23, 69, 7, 10] is a list of numbers or integers.

Every element is positioned with an index number. Indices start with 0 for the first element.

syntax:

List<DataType> ListName = [element1, element2, element3, element4];

if you give the datatype as **dynamic**, it can take any datatype and be a heterogenous list.

functions:

```
void main(){
    // declaration of studentNames list
    List<String> studentNames = [”safwan”];
    studentNames.add("mohammed");    // adds an element to the l
    studentNames.addAll(["jswanth", "chakradhar"]);   // adds mu
    studentNames.insert(2, "deva");    // adds "deva" to index 2
    studentNames.removeAt(2);    // removes element from index 2
    studentNames.remove("mohammed");    // removes "mohammed" el

    // printing using index
    print(studentNames[0]);
    print(studentNames[3]);
    print(studentNames.first);
    print(studentNames.last);

    studentNames.clear(); // removes all elements from the list
}
```

5. Maps

The map object is a key value pair. Keys and values on a map may be of any type. It is a dynamic collection.

syntax: Map<keyDataType, valueDataType> MapName = { };

Map <String, String> students = {};

```
void main(){
    Map Newmap = new Map();
    Newmap['First'] = 'Dart';
    Newmap['Second'] = 'For';
    Newmap['Third'] = 'Developing apps';

    print(Newmap);
    // output: {'First': 'Dart', 'Second': 'For', 'Third': 'Deve
}
```

6. Sets

7. Runes

Variables

variable declaration syntax: [**typeOfVariable] [nameOfVariable] = value;**

examples :

int num1 = 2;

double num2 = 1.5;

bool isRemember = true;

When a variable name starts with an underscore (_variableName), it is set to private access specifier. It cannot be accessed directly outside the local scope. It will need getter and setter methods to manipulate it.

String concatenation: to add a variable inside a string we use the dollar ($) symbol

to add more than a variable, the dollar is accompanied with curly braces { }

```
void main(){
    int x = 5;
    print("The value of x is $x");
    print("The upper case of hello is ${'hello'.toUpperCase()}"
}
```