Dart Primer

Prashanth B S[1]

[1]Department of Information Science & Engineering, Nitte Meenakshi Institute of Technology,
Yelahanka - 560064, Bengaluru

1 Introduction to Dart

Dart is a programming language designed for client side development such as for the web and mobile Apps. It is developed by the Google and can be used to develop server and desktop Applications. Some of the features of the dart are as follows[1],

1. Dart is an *object oriented* Language
2. It is *statically typed* language with less learning curve
3. Supports *Classes* and has *inbuilt garbage collector* with *C style Syntax*
4. Dart can compile the code to *native code* or *JavaScript*
5. To develop apps using dart one should install Dart SDK(Installed along with Flutter)
6. Dart's performance is good with *Single Page Design*
7. It supports *Interfaces, Single level Inheritance, Collections and Generics*
8. Dart files will have an extension *dart.* and it can be run with the command *dartfilename.dart* from the command line.

A sample code of the dart looks as follows,

```
void main(){ // Entry point of the program
 print('This is a sample message'); // prints the message
 }
```

Dart programming support 2 styles of comments, single line comment(//), multi-line comment (/*... */).

1.1 Variables

In dart, everything is treated as *object*. Even the variables are declared as object. The default value of a variable which is not initialized is initialized to "NULL". Some of the variable definition styles are,

1. *Datattype Style* : Explicitly specify the datatype
2. *Implicit Style*: Also called as Deductive type, here the datatype is deduced based on the value that it holds

[1] https://en.wikipedia.org/wiki/Dart_(programming_language)

3. *Dynamic Type*: The datatype which can be assigned and changed dynamically

4. *Object Type*: Use the default object that can be assigned later.

The code shows the three styles of the variable definition,

```
void main(){ // Entry point of the program
int x = 20 ; // Type(1)
var y = 30.0 ; // Data type is deduced based on the value: Double
dynamic a = 20 ; // Integer type
a = 'xyz' ; // Permitted
Object m = "ABC"; // Valid Statement
}
```

To access the type of the variable at run-time, use *variable.runtimeType* as follows,

```
// filename: test.dart ( to run use "dart test.dart")
void main(){ // Entry point of the program
int x = 20 ; // Type(1)
print(x.runtimeType) ;
print('x value: $x');
var y = 30.0 ; // Data type is deduced based on the value: Double
print(y.runtimeType) ;
print('y value: $y');
dynamic a = 20 ; // Integer type
print(a.runtimeType) ;
a = 'xyz' ; // Permitted
print('a value: $a');
print(a.runtimeType) ;
Object m = "ABC"; // Valid Statement
print(m.runtimeType) ;
print('m value: $m');
}
```

1.2 const vs final

const and final define an immutable value for a variable, simply constant. But the difference lies in the way the value is assigned. const assign the value and treats it as constant at compile time, whereas final allows the computation to be done at run time for a constant and assigns it, it supports compile time assignment as well. Here is a simple program demonstrating the same,

```
// test2.dart
void main() {
  const a = 100; // defines a constant at compile time
  final m = 100; // defines a constant at compile time
  print('a and m are: $a and $m');
  const x = 10; // defines a constant at compile time
  print(x);
  final myx = DateTime.now(); // value for myx is computed at runtime
  print(myx);
}
```

1.3 Strings

The String data type represents a sequence of characters. String values in Dart can be represented using either single or double or triple quotes. Single line strings are represented using single or double quotes. Triple quotes are used to represent multi-line strings.[2]. Some of the operations that we can perform on the string is demonstrated below,

```
// strings.dart
void main() {
  String s1 = "Welcome"; // can be written as 'welcome' '''welcome''' or ''welcome''
  String s2 = " World";
  print(s1 + s2); // string concatenation
  String s3 = "${s1 + s2}, you are in for a game"; // String can be indexed using $
  print(s3);
  print(s1.toUpperCase()); // Uppercase
  print(s2.toLowerCase()); // Lowercase
  print(s1.length); // Length
  print(s2.isEmpty); // boolean
  print(s1.contains('We')); // matching substring
}
```

1.4 Numbers

The data can be represented using int, double, boolean or String, and the dart gives flexibility to type conversion as well.

```
void main() {
  int i = 20;
  var x = 20.0;
  bool b = true;
  String s1 = i.toString(); // integer->String
  String s2 = x.toString(); // double->String
  print(b);
  print(s1);
  print(s2);
  double m = double.parse(s1); // String->double
  print(m);
  double n = double.parse(s2); // String->double
  print(n);
  int o = int.parse(s1); // String->int
  print(o);
}
```

1.5 Other Data-types

Some of the other data types that are used are lists, set, dictionary and map. Here is a code snippet for the list and its operations.

```
// list.dart
void main() {
  List list = [1, 2, 3, 4]; // defining list
  print(list); // listindex ->[0,1,2,3]
```

[2] https://www.tutorialspoint.com/dart_programming/index.htm

```
  list.add(5); // adding a new element at the end
  print(list);

  print(list.length); // compute the length
  print(list.reversed); // print reversed
  print(list.first); // print first element
  print(list.last); // print last element
  print(list.indexOf(3)); // print the third element
  list.removeLast(); // remove from end
  print(list);

  list.remove(1); // remove the element 1
  print(list);

  list.removeAt(0); // removes 2
  print(list);
}
```

Similar to lists, we have set, which can also store a bunch of elements. The difference between set and lists is that set consists of only unique elements, i.e, set can not have multiple occurrences of the same element.

```
void main() {
  Set mySet = Set(); // Creating an empty Set
  mySet.addAll(['A', 'B', 'C']); // addAll takes List as an argument
  print(mySet);
  mySet.add('D'); // adding individual item
  print(mySet);

  mySet.remove('A'); // removing individual item
  print(mySet);
  print(mySet.length); // length of the set
  print(mySet.first); // first element
  print(mySet.last); // last element
}
```

Similar to Dictionary, dart Has Maps which stores the values in the form of a key-value pair. The following code snippets demonstrates the same.

```
// maps.dart
void main() {
  var student = {"Ashok": 1, "Ravi": 2}; // creates a default Map<String, int>
  print(student.runtimeType);
  print(student);

  Map teacher = Map(); // creates an Empty Map()
  teacher['first'] = "Rajesh"; // teacher[key] = value
  teacher['second'] = "Ankur";
  print(teacher);

  teacher.remove('second');
  print(teacher);

  Map coaches = Map<int, String>(); // Specifying the type of key-value pair
```

```
  coaches[1] = "Kapil";
  coaches[2] = "Ankit";
  print(coaches);

  print(coaches.keys); // print the keys of coaches map
  print(coaches.values); // print the values
}
```

Dart also supports defining Enumerated types. These types are used to define named constant values.

```
// enumdemo.dart
enum myText { This, was, boring } // named constants
void main() {
  for (myText t in myText.values) {
    // looping through constants
    print(t);
  }
}
```