Object-Oriented Concepts:

Dart is an object-oriented programming language, and it supports all the concepts of object-oriented programming such as classes, object, inheritance, mixin, and abstract classes. As the name suggests, it focuses on the object and objects are the real-life entities. The Object-oriented programming approach is used to implement the concept like polymorphism, data-hiding, etc. The main goal of oops is to reduce programming complexity and do several tasks simultaneously. The oops concepts are given below.

- Class
- Object
- Inheritance
- Polymorphism
- Interfaces
- Abstract class

Class

Dart classes are defined as the blueprint of the associated objects. A Class is a user-defined data type that describes the characteristics and behavior of it. To get all properties of the class, we must create an object of that class. The syntax of the class is given below.

Syntax:

```
class ClassName {
   <fields>
   <getter/setter>
   <constructor>
   <functions>
}
```

Dart provides **class** keyword followed by a class name is used to define a class; all fields and functions are enclosed by the pair of curly braces ({}).

Here, the ClassName represents the actual name of the class, which is defined by the user. In curly braces, we provide a class definition. A class can consist of fields, constructors, getters setters, and methods.

Note - According to the naming convention rule of identifiers, the first letter of the class name must be capital and use no separators.

Let's understand the following example.

Example -

```
void main() {
  // Defining class
 class Student {
  var stdName;
  var stdAge;
  var stdRoll_nu;

  // Class Function
   showStdInfo() {
     print("Student Name is : ${stdName}");
     print("Student Age is : ${stdAge}");
     print("Student Roll Number is : ${stdRoll_nu}")
}
```

In the above example of class, we declared a class called **Student**. This class has three fields **stdName, stdAge**, and **stdRoll_nu**. The **showStdInfo()** is a class function which prints the fields of class. To access the properties of the class, we need to create its object.

Object

An object is a real-life entity such as a table, human, car, etc. The object has two characteristics - state and behavior. Let's take an example of a car which has a name, model name, price and behavior moving, stopping, etc. The object-oriented programming offers to identify the state and behavior of the object.

We can access the class properties by creating an object of that class. In Dart, The object can be created by using a new keyword followed by class name. The syntax is given below.

Syntax:

```
var objectName = new ClassName(<constructor_arguments>)
```

Working with Constructor:

A constructor is a different type of function which is created with same name as its class name. The constructor is used to initialize an object when it is created. When we create the object of class, then constructor is automatically called. It is quite similar to class function but it has no explicit return type. The generative constructor is the most general form of the constructor, which is used to create a new instance of a class.

It is option to declare within the class. All class have own constructor but if we don't declare or forget then Dart compiler will create default constructor automatically by passing the default value to the member variable. If we declare own constructor, then default constructor will be ignored.

Example -

Suppose we have a class name **Student** and we will create an object of it as follow.

```
Student std = new Student()
```

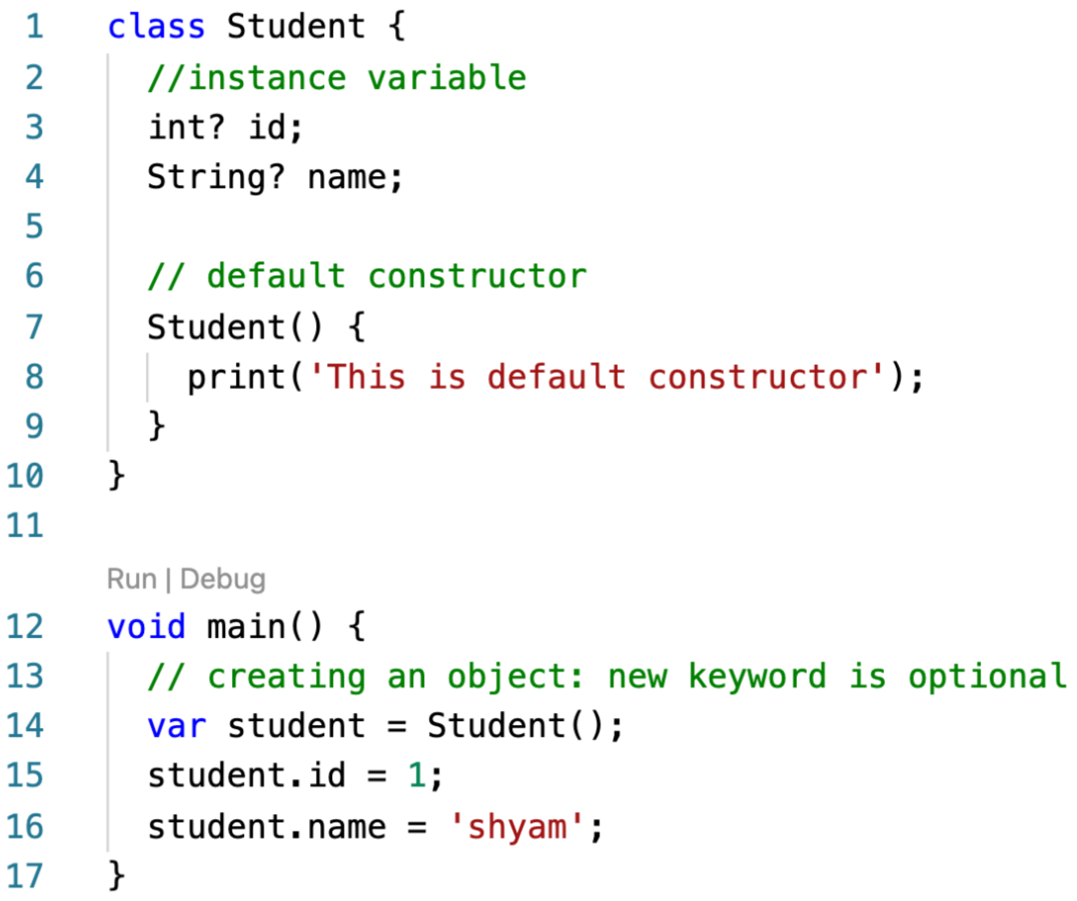
It invoked the default constructor of the **Student** class.

Types of Constructor:

- Default Constructor
- Parameterized Constructor
- Named Constructor
- Constant Constructor

Default Constructor:

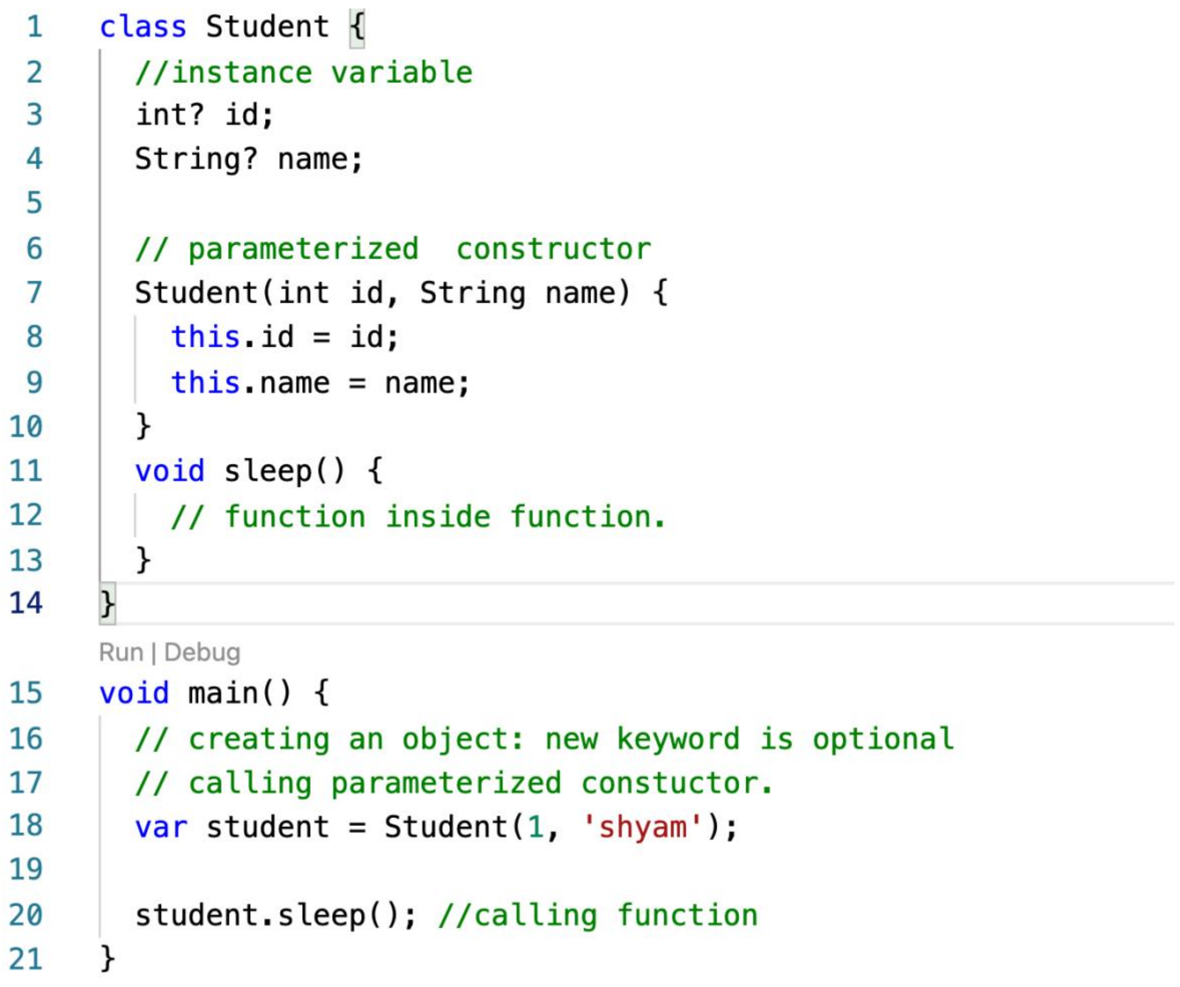
- Constructor having no parameter is known as default constructor.
- If we do not define any constructor, then dart constructor automatically add default constructor.



```dart
class Student {
  //instance variable
  int? id;
  String? name;

  // default constructor
  Student() {
    print('This is default constructor');
  }
}

void main() {
  // creating an object: new keyword is optional
  var student = Student();
  student.id = 1;
  student.name = 'shyam';
}
```

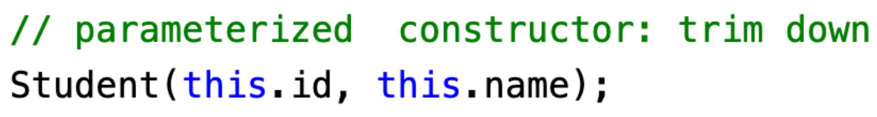
<u>Parametrized Constructor:</u>

- Constructor having parameter is known as Parameterized Constructor.
- You cannot have both default and parameterized constructor at the same time in same class.



```
class Student {
  //instance variable
  int? id;
  String? name;

  // parameterized  constructor
  Student(int id, String name) {
    this.id = id;
    this.name = name;
  }
  void sleep() {
    // function inside function.
  }
}
void main() {
  // creating an object: new keyword is optional
  // calling parameterized constuctor.
  var student = Student(1, 'shyam');

  student.sleep(); //calling function
}
```

Note: We can trim down the parameterized constructor.

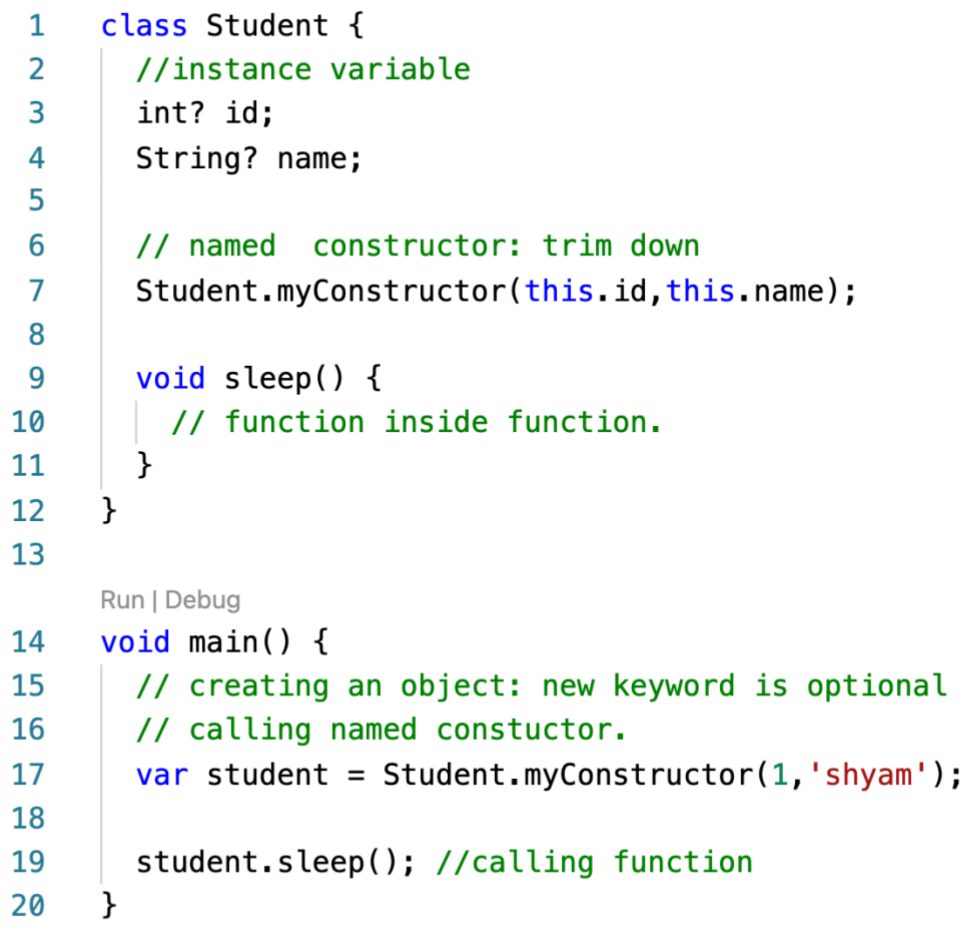


```
// parameterized  constructor: trim down
Student(this.id, this.name);
```

We can directly initialize the variable by such techniques.

Named Constructor:

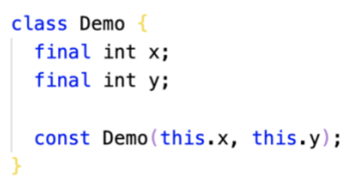
- You can define your own name of constructor.
- It is used to create multiple constructors in dart.



```dart
class Student {
  //instance variable
  int? id;
  String? name;

  // named  constructor: trim down
  Student.myConstructor(this.id,this.name);

  void sleep() {
    // function inside function.
  }
}

void main() {
  // creating an object: new keyword is optional
  // calling named constuctor.
  var student = Student.myConstructor(1,'shyam');

  student.sleep(); //calling function
}
```

Constant Constructor:

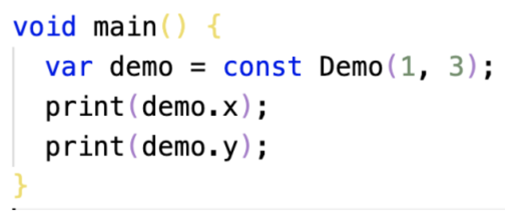
- To make a constant constructor make sure that all variables are final.



```dart
class Demo {
  final int x;
  final int y;

  const Demo(this.x, this.y);
}
```

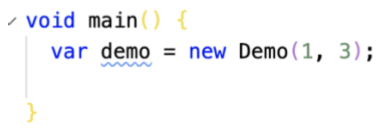
- Constant constructor does not have body.

- Creating objects of Constant constructor:



```
void main() {
  var demo = const Demo(1, 3);
  print(demo.x);
  print(demo.y);
}
```

- We can also create object by using new keyword:



```
void main() {
  var demo = new Demo(1, 3);

}
```

Brief:

- Class is a description of an Object.
- Class is a blueprint of an object.
- Object is the instantiation of a class.
- Objects have state (properties) and behavior (functions).
- A single class can have many Objects.
- Constructor is used to initialize an instance variable within constructors.
- It is used to create an Objects.
- You cannot have both default and parameterized constructor at the same time.
- You can have as many as Named Constructor you want.