CONSTRUCTOR IN DART

Introduction

In this section, you will learn about constructor in Dart programming language and how to use constructors with the help of examples. Before learning about the constructor, you should have a basic understanding of the class and object in dart.

Constructor In Dart

A constructor is a special method used to initialize an object. It is called automatically when an object is created, and it can be used to set the initial values for the object's properties. For example, the following code creates a **Person** class object and sets the initial values for the **name** and **age** properties.

```
Person person = Person("John", 30);
```

Save this Snippet

Without Constructor

If you don't define a constructor for class, then you need to set the values of the properties manually. For example, the following code creates a **Person** class object and sets the values for the **name** and **age** properties.

```
Person person = Person();
person.name = "John";
person.age = 30;
```

Save this Snippet

Things To Remember

- The constructor's name should be the same as the class name.
- Constructor doesn't have any return type.

Syntax

```
class ClassName {
  // Constructor declaration: Same as class name
  ClassName() {
    // body of the constructor
  }
}
```

Save this Snippet

Info

Note: When you create a object of a class, the constructor is called automatically. It is used to initialize the values when an object is created.

Example 1: How To Declare Constructor In Dart

In this example below, there is a class **Student** with three properties: **name**, **age**, and **rollNumber**. The class has one constructor. The constructor is used to initialize the values of the three properties. We also created an object of the class **Student** called **student**.

```
class Student {
  String? name;
  int? age;
  int? rollNumber;

  // Constructor
  Student(String name, int age, int rollNumber) {
    print(
        "Constructor called"); // this is for checking the constructor is
called or not.
    this.name = name;
    this.age = age;
    this.rollNumber = rollNumber;
  }
}

void main() {
  // Here student is object of class Student.
```

```
  Student student = Student("John", 20, 1);
  print("Name: ${student.name}");
  print("Age: ${student.age}");
  print("Roll Number: ${student.rollNumber}");
}
```

Save this Snippet

Show Output

Run Online

Info

Note: The **this** keyword is used to refer to the current instance of the class. It is used to access the current class properties. In the example above, parameter names and class properties of constructor **Student** are the same. Hence to avoid confusion, we use the **this** keyword.

Example 2: Constructor In Dart

In this example below, there is a class **Teacher** with four properties: **name**, **age**, **subject**, and **salary**. Class has one constructor for initializing the values of the properties. Class also contain method **display()** which is used to display the values of the properties. We also created 2 objects of the class **Teacher** called **teacher1** and **teacher2**.

```
class Teacher {
  String? name;
  int? age;
  String? subject;
  double? salary;

  // Constructor
  Teacher(String name, int age, String subject, double salary) {
    this.name = name;
    this.age = age;
    this.subject = subject;
    this.salary = salary;
  }
  // Method
  void display() {
    print("Name: ${this.name}");
    print("Age: ${this.age}");
    print("Subject: ${this.subject}");
    print("Salary: ${this.salary}\n"); // \n is used for new line
```

```
  }
}

void main() {
  // Creating teacher1 object of class Teacher
  Teacher teacher1 = Teacher("John", 30, "Maths", 50000.0);
  teacher1.display();

  // Creating teacher2 object of class Teacher
  Teacher teacher2 = Teacher("Smith", 35, "Science", 60000.0);
  teacher2.display();
}
```

Save this Snippet

Show Output

Run Online

Info

Note: You can create many objects of a class. Each object will have its own copy of the properties.

Example 3: Constructor In Dart

In this example below, there is a class **Car** with two properties: **name** and **price**. The class has one constructor for initializing the values of the properties. The class also contains method **display()**, which is used to display the values of the properties. We also created an object of the class **Car** called **car**.

```
class Car {
  String? name;
  double? price;

  // Constructor
  Car(String name, double price) {
    this.name = name;
    this.price = price;
  }

  // Method
  void display() {
    print("Name: ${this.name}");
    print("Price: ${this.price}");
```

```
  }
}

void main() {
  // Here car is object of class Car.
  Car car = Car("BMW", 500000.0);
  car.display();
}
```

Save this Snippet

Show Output

Run Online

Example 4: Constructor In Dart

In this example below, there is a class **Staff** with four properties: **name**, **phone1**, **phone2**, and **subject** and one method **display()**. Class has one constructor for initializing the values of only **name**, **phone1** and **subject**. We also created an object of the class **Staff** called **staff**.

```
class Staff {
  String? name;
  int? phone1;
  int? phone2;
  String? subject;

  // Constructor
  Staff(String name, int phone1, String subject) {
    this.name = name;
    this.phone1 = phone1;
    this.subject = subject;
  }

  // Method
  void display() {
    print("Name: ${this.name}");
    print("Phone1: ${this.phone1}");
    print("Phone2: ${this.phone2}");
    print("Subject: ${this.subject}");
  }
}

void main() {
  // Here staff is object of class Staff.
  Staff staff = Staff("John", 1234567890, "Maths");
  staff.display();
}
```

Save this Snippet

Show Output

Run Online

Example 5: Write Constructor Single Line

In the avobe section, you have written the constructor in long form. You can also write the constructor in short form. You can directly assign the values to the properties. For example, the following code is the short form of the constructor in one line.

```
class Person{
  String? name;
  int? age;
  String? subject;
  double? salary;

  // Constructor in short form
  Person(this.name, this.age, this.subject, this.salary);

  // display method
  void display(){
    print("Name: ${this.name}");
    print("Age: ${this.age}");
    print("Subject: ${this.subject}");
    print("Salary: ${this.salary}");
  }
}

void main(){
  Person person = Person("John", 30, "Maths", 50000.0);
  person.display();
}
```

Save this Snippet

Show Output

Run Online

Example 6: Constructor With Optional Parameters

In the example below, we have created a class **Employee** with four properties: **name**, **age**, **subject**, and **salary**. Class has one constructor for initializing the all properties values. For **subject** and **salary**, we have used optional parameters. It means we can pass or not pass the values of **subject** and **salary**. The Class also contain method **display()** which is used to display the values of the properties. We also created an object of the class **Employee** called **employee**.

```
class Employee {
  String? name;
  int? age;
  String? subject;
  double? salary;

  // Constructor
  Employee(this.name, this.age, [this.subject = "N/A", this.salary=0]);

  // Method
  void display() {
    print("Name: ${this.name}");
    print("Age: ${this.age}");
    print("Subject: ${this.subject}");
    print("Salary: ${this.salary}");
  }
}

void main(){
  Employee employee = Employee("John", 30);
  employee.display();
}
```

Save this Snippet

Show Output

Run Online

Example 7: Constructor With Named Parameters

In the example below, we have created a class **Chair** with two properties: **name** and **color**. Class has one constructor for initializing the all properties values with named parameters. The Class also contain method **display()** which is used to display the values of the properties. We also created an object of the class **Chair** called **chair**.

```
class Chair {
```

```
String? name;
String? color;

// Constructor
Chair({this.name, this.color});

// Method
void display() {
  print("Name: ${this.name}");
  print("Color: ${this.color}");
}
}

void main(){
Chair chair = Chair(name: "Chair1", color: "Red");
chair.display();
}
```

Save this Snippet

Show Output

Run Online

Example 8: Constructor With Default Values

In the example below, we have created a class **Table** with two properties: **name** and **color**. Class has one constructor for initializing the all properties values with default values. The Class also contain method **display()** which is used to display the values of the properties. We also created an object of the class **Table** called **table**.

```
class Table {
  String? name;
  String? color;

  // Constructor
  Table({this.name = "Table1", this.color = "White"});

  // Method
  void display() {
    print("Name: ${this.name}");
    print("Color: ${this.color}");
  }
}

void main(){
  Table table = Table();
  table.display();
}
```

Save this Snippet

Show Output

Run Online

Key Points

- The constructor's name should be the same as the class name.
- Constructor doesn't have any return type.
- Constructor is only called once at the time of the object creation.
- Constructor is called automatically when an object is created.
- Constructor is used to initialize the values of the properties of the class.

Challenge

Create a class **Patient** with three properties **name**, **age**, and **disease**. The class has one constructor. The constructor is used to initialize the values of the three properties. Also, create an object of the class **Patient** called **patient**. Print the values of the three properties using the object.

Video

Watch our video on class and object in Dart.