GUSTO UNIVERSITY

Object Oriented programming with Dart Programming Language

Daw A Mar Myo Thu

Lecturer

GUSTO University

INHERITANCE IN DART

- Inheritance is a sharing of behavior between two classes.
- It allows you to define a class that extends the functionality of another class.
- The extend keyword is used for inheriting from parent class.
- **Note**: Whenever you use inheritance, it always create a **is-a** relation between the parent and child class like **Student is a Person**, **Truck is a Vehicle**, **Cow is a Animal** etc.
- Dart supports single inheritance, which means that a class can only inherit from a single class.
- Dart does not support multiple inheritance which means that a class cannot inherit from multiple classes.

Syntax

```
class ParentClass {
  // Parent class code
}

class ChildClass extends ParentClass {
  // Child class code
}
```

Terminology

- **Parent Class**: The class whose properties and methods are inherited by another class is called parent class. It is also known as base class or super class.

- **Child Class**: The class that inherits the properties and methods of another class is called child class. It is also known as derived class or sub class.

Example 1: Inheritance In Dart

In this example, we will create a class Person and then create a class Student that inherits the properties and methods of the Person class.

```
class Person {
  // Properties
  String? name;
  int? age;

  // Method
  void display() {
    print("Name: $name");
    print("Age: $age");
  }
}
```

```
// Here In student class, we are extending the
// properties and methods of the Person class
class Student extends Person {
  // Fields
  String? schoolName;
  String? schoolAddress;

  // Method
  void displaySchoolInfo() {
    print("School Name: $schoolName");
    print("School Address: $schoolAddress");
  }
}
```

```
void main() {
  // Creating an object of the Student class
  var student = Student();
  student.name = "John";
  student.age = 20;
  student.schoolName = "ABC School";
  student.schoolAddress = "New York";
  student.display();
  student.displaySchoolInfo();
}
```

Advantages Of Inheritance In Dart

- It promotes reusability of the code and reduces redundant code.
- It helps to design a program in a better way.
- It makes code simpler, cleaner and saves time and money on maintenance.
- It facilitates the creation of class libraries.
- It can be used to enforce standard interface to all children classes.

Example 2: Inheritance In Dart

In this example, here is parent class Car and child class Toyota. The Toyota class inherits the properties and methods of the Car class.

```
class Car{
  String? color;
  int? year;

  void start(){
    print("Car started");
  }
}
```

```
class Toyota extends Car{
  String? model;
  int? price;


  void showDetails(){
    print("Model: $model");
    print("Price: $price");
  }
}
```

```
void main(){
  var toyota = Toyota();
  toyota.color = "Red";
  toyota.year = 2020;
  toyota.model = "Camry";
  toyota.price = 20000;
  toyota.start();
  toyota.showDetails();
}
```

Example 3: Single Inheritance In Dart

In this example below, there is super class named Car with two properties name and price. There is sub class named Tesla which inherits the properties of the super class. The sub class has a method display to display the values of the properties.

```
class Car {
  // Properties
  String? name;
  double? price;
}
```

```
class Tesla extends Car {
  // Method to display the values of the properties
  void display() {
    print("Name: ${name}");
    print("Price: ${price}");
  }
}
```

```
void main() {
  // Create an object of Tesla class
  Tesla t = new Tesla();
  // setting values to the object
  t.name = "Tesla Model 3";
  t.price = 50000.00;
  // Display the values of the object
  t.display();
}
```

Example 4: Multilevel Inheritance In Dart

- In this example below, there is super class named Car with two properties name and price.
- There is sub class named Tesla which inherits the properties of the super class.
- The sub class has a method display to display the values of the properties.
- There is another sub class named Model3 which inherits the properties of the sub class Tesla.
- The sub class has a property color and a method display to display the values of the properties.

Example 4: Multilevel Inheritance In Dart

```
class Car {
// Properties
String? name;
double? price;
}


class Tesla extends Car {
// Method to display the values of the properties
void display() {
  print("Name: ${name}");
  print("Price: ${price}");
}
}
```

```
class Model3 extends Tesla {
// Properties
String? color;


// Method to display the values of the properties
void display() {
  super.display();
  print("Color: ${color}");
}
}
```

```
void main() {
// Create an object of Model3 class
Model3 m = new Model3();
// setting values to the object
m.name = "Tesla Model 3";
m.price = 50000.00;
m.color = "Red";
// Display the values of the object
m.display();
}
```

Example 5: Multilevel Inheritance In Dart

- In this example below, there is class named Person with two properties name and age.
- There is sub class named Doctor with properties listofdegrees and hospitalname.
- There is another subclass named Specialist with property specialization.
- The sub class has a method display to display the values of the properties.

Example 5: Multilevel Inheritance In Dart

```
class Person {
  // Properties
  String? name;
  int? age;
}
```

```
class Doctor extends Person {
  // Properties
  List<String>? listofdegrees;
  String? hospitalname;

  // Method to display the values of the properties
  void display() {
    print("Name: ${name}");
    print("Age: ${age}");
    print("List of Degrees: ${listofdegrees}");
    print("Hospital Name: ${hospitalname}");
  }
}
```

```
class Specialist extends Doctor {
  // Properties
  String? specialization;

  // Method to display the values of the properties
  void display() {
    super.display();
    print("Specialization: ${specialization}");
  }
}
```

Example 5: Multilevel Inheritance In Dart

```
void main() {
  // Create an object of Specialist class
  Specialist s = new Specialist();
  // setting values to the object
  s.name = "John";
  s.age = 30;
  s.listofdegrees = ["MBBS", "MD"];
  s.hospitalname = "ABC Hospital";
  s.specialization = "Cardiologist";
  // Display the values of the object
  s.display();
}
```

What’s problem Of Copy Paste Instead Of Inheritance?

- If you copy the code from one class to another class, then you will have to maintain the code in both the classes.
- If you make any changes in one class, then you will have to make the same changes in the other class.
- This can lead to errors and bugs in the code.

Applied Programming and Design Principles