Dart Object-Oriented Programming

Key Points	Notes

1. Object

a. Dart is an object-oriented language with classes and mixin-based inheritance (inherit one superclass while sharing behaviour from different classes).

b. In Dart, a class is defined as a blueprint or template of associated objects. A class is a wrapper that encapsulates the data and methods together. Think a class as a user-defined data type.

c. We can create an instance or object of the class and access the class properties (local variables) and methods (functions).

d. **Class Declaration**:

```
class Student {

 //properties
 var name;

 //constructor
 Student(){
  name = “”;
}

 //setter
 void setName(String name){
  this.name = name;
 }

 //getter
 String getName(){
  return name;
 }

 //function
 void display(){
  print(getName());
 }
}
```

Summary

Key Points	Notes

e. **Object Creation**.

```
void main(){

 //instantiation with default constructor
 var myStudent = new Student(); //name = “”

 print(myStudent); //Instance of Student

 //accessing setter
 myStudent.setName(“Ali Bin Abu”); // name = “Ali Bin Abu”

 //accessing getter and function
 print(myStudent.getName());
 myStudent.display();
}
```

2. Constructor

a. A special method used to initialize an object when it is created. It is automatically called when object is instantiated.

b. A constructor must have the same name as the class name and does not have return type.

c. It is **NOT** a mandatory to include a constructor in a class because every class has its own default constructor created by the compiler.

d. There are **three (3)** types of constructors:
   i. Default constructor or no arguments.
      - It is created automatically by the compiler is there is **NO** constructor defined in the class.
   ii. Parameterized constructor.
      - A constructor that accept parameters or arguments to initialize object variables.
   iii. Named constructor.
      - It allows a class to define multiple constructors.

Summary

Key Points	Notes
	e. Examples.

```
//Default Constructor

class Student{
  var name;

  Student(){
   name = “”;
  }

}


void main(){
  Student a = new Student();
}
```

```
//Parameterized Constructor

class Student{
  var name;

  Student(String name){
  this.name = name;
  }

}


void main(){
  Student b = new Student(“Aminah Hassan”);
}
```

Summary

Key Points	Notes

```
//Named Constructor

class Student{
  var name;

  Student(){
    name = "";
  }

  Student.kict(String name){
   this.name = name;
  }

}


void main(){
  Student c = new Student();
  Student d = new Student.kict("Ali Ahmad");
}
```

3. `this` **Keyword**

a. **`this`** keyword pointing to the current class variables or properties. It refers to the current instant of the class in a method or constructor.

b. It eliminates the confusion between class variables or properties and parameters with the same name.

```
class Student {
 var name;

 //setter
 void setName(String name){
  this.name = name;
 }
}
```

Summary

Key Points	Notes

4. Class Inheritance

a. A class inheritance allows a dart program to create a new class from any existing class. The extended class to create a new class is called super class or parent class. The newly created class is called sub class or child class.

b. To inherit all **properties** and **methods** from existing class the **extends** keyword will be used. Please take note, constructor will **NOT** inherit **constructors** from the parent class.

```
class Kulliyyah extends Student {
  var major;

  Kulliyyah(){
    major = “”;
  }

  void setMajor(String major){
   this.major = major;
  }

  String getMajor(){
   return major;
  }

  void setName(String name){
   super.setName(name);
  }

  String getName(){
   return super.getName();
  }

  void display(){
   super.display();
   print(getMajor());
  }
}
```

Summary

Key Points	Notes

c. **super** keyword is used to access all variables or properties and methods of the parent class. Refer to `setName()`, `getName()` and `display()` methods from `Kulliyyah` class.

d. Method **overriding** allows sub class or child class to use the same method name to modify its original purposes from the super class or parent class. Refer to `setName()` and `getName()` methods from `Kulliyyah` class.

e. Dart support **ONLY** single and multi-level class inheritance from a parent class. You can include other classes in the sub or child class by using Mixin "**`with`**" keyword.

f. **Setters** and **getters** are used to initialize and retrieve local variables or properties. Refer to `setName()` and `getName()` methods from `Student` class.

5. `static` **Keyword**

a. A static keyword is used for memory management of global data members. It can be applied to local variables or properties and methods and allows developer to access local variables or properties and methods without having to instantiate a class.

```
class test{
 static var x;
 var y;

 void setY(int y)[
  this.y = y;
 }

}


void main(){
 test z = new test();
 z.setY(10);
 test.x = 20;
}
```

Summary

Key Points	Notes

6. Encapsulation

a. In dart data encapsulation happens at **library level** instead of class level, which is totally different from the other object-oriented programming languages.

b. To set class local variables or properties to private, use underscore (_) for each local variable or property name declared.

```
//student.dart
class Student {
  //properties
  var _name;

  //constructor
  student() {
    _name = "";
  }

  //setter
  void setName(String name) {
    this._name = name;
  }

  //getter
  String getName() {
    return this._name;
  }

  void display() {
    print(getName());
  }
}
```

Summary

Key Points	Notes

```
//main.dart
import 'student.dart';

void main() {
  //instantiation with default constructor
  Student myStudent = new Student(); //name = “”

  //Syntax error, _name is not defined.
  myStudent._name = “Ali”;

}
```

7. Cascade Operator

a. Cascade operator allows developer to call class methods in sequence without having to make an object reference.

```
void main(){
  new Student()
  ..setName(“ali”)
  ..display();
}
```

8. References

a. A tour of the Dart Language. Retrieved on October 2nd, 2020 from https://dart.dev/guides/language/language-tour
b. Howard, J. (2019). Dart Basics. Retrieved from https://www.raywenderlich.com/4482551-dart-basics#toc-anchor-014
c. Dart Programming Tutorial. Retrieved on October 2nd, 2020 from https://www.tutorialspoint.com/dart_programming/index.htm
d. Dart Tutorial. Retrieved on October 30th,-2020 from https://www.w3adda.com/dart-tutorial

Summary