



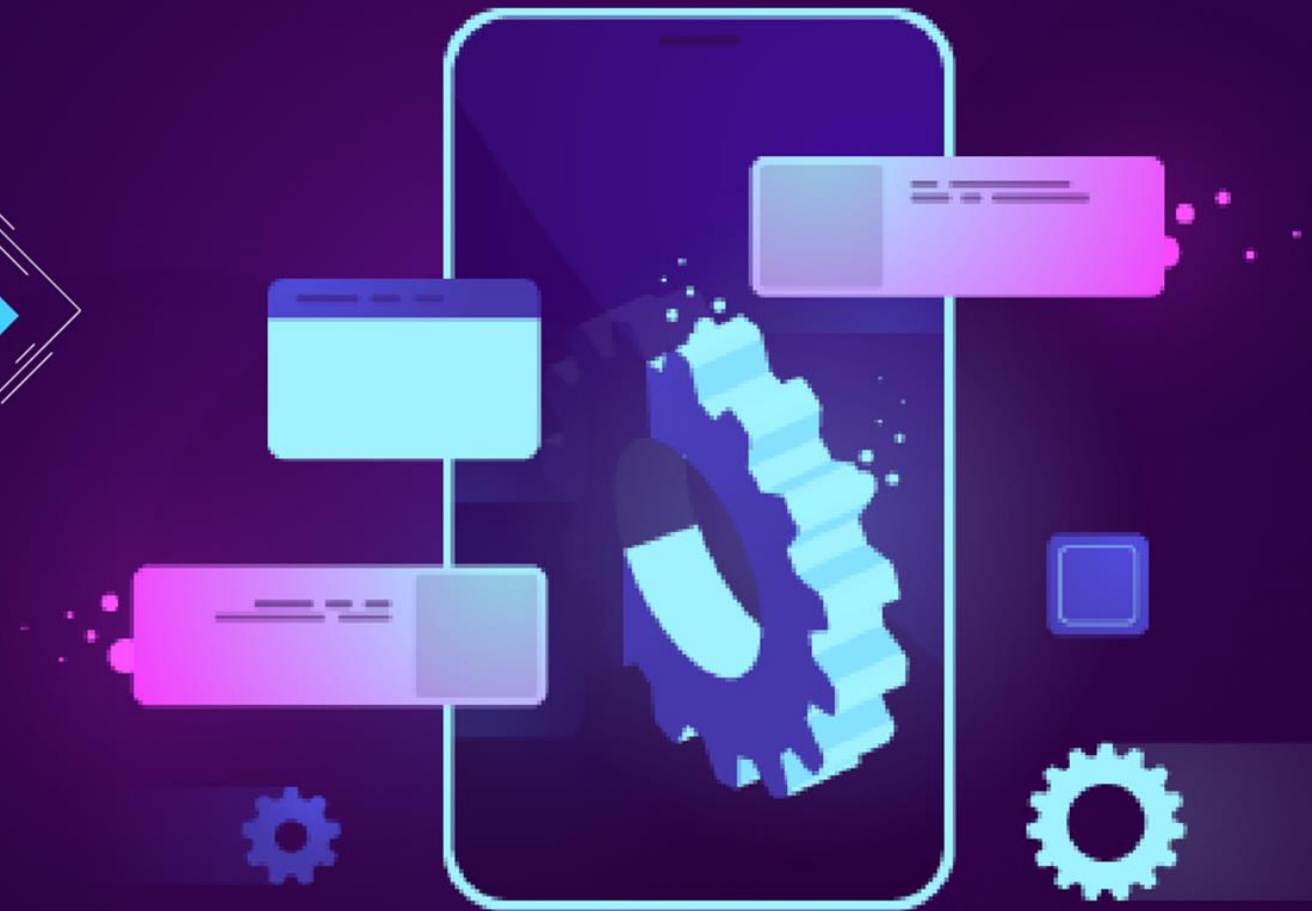
Sayed Abdul-Aziz
Flutter Developer & Instructor

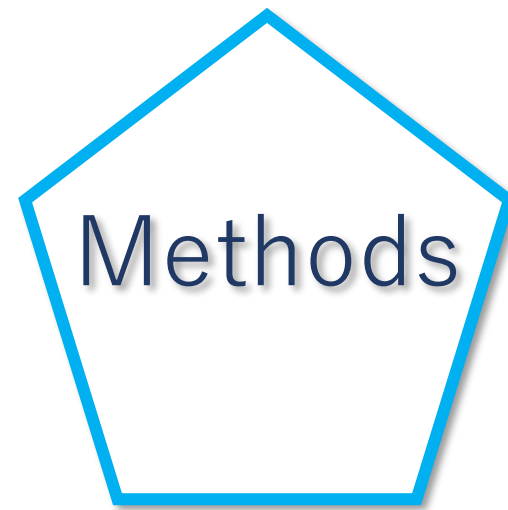


Flutter 3



Session – 4 : Methods and Intro to OOP





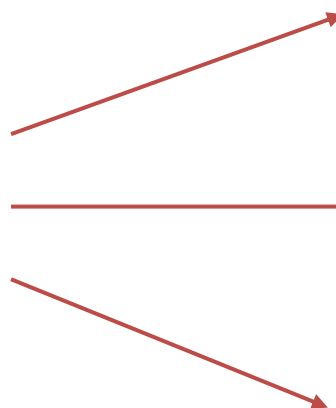
Sayed Abdul-Aziz

@sayedabdulaziz





Task 1
Task 2
Task 3



Sayed Abdul-Aziz

@sayedabdulaziz





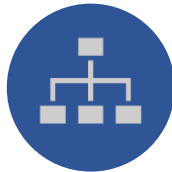
Reusability

Time savers and help us to reuse the code without retyping the code. Write one use many!



Maintenance

Make the maintenance process easier.



Manageable

It divides the code to tasks so that make it easier to control.



Sayed Abdul-Aziz

@sayedabdulaziz





return-type Parameters

Method-name

```
int    add   (int a, int b)
{
    //method body
}
```

- Return Type:
 - With returning a value → int, String, double, ...etc
 - Without returning a value → void
- If a method has return type with returning a value. Use **return** keyword.



With returning a value

```
class MyClass {  
    int display(int x) {  
        // code to be execute  
        return x;  
    }  
}
```

Without returning a value


```
class MyClass {  
    void display() {  
        // code to be executed  
    }  
}
```





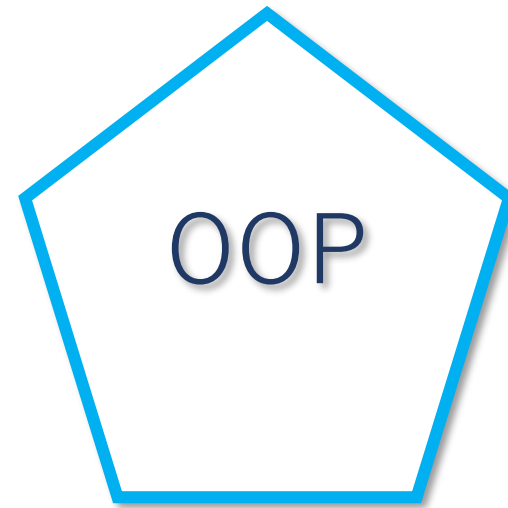
HOW TO CALL A METHOD?

```
void main() {  
    display();  
}  
  
void display() {  
    // code to be executed  
}
```



call

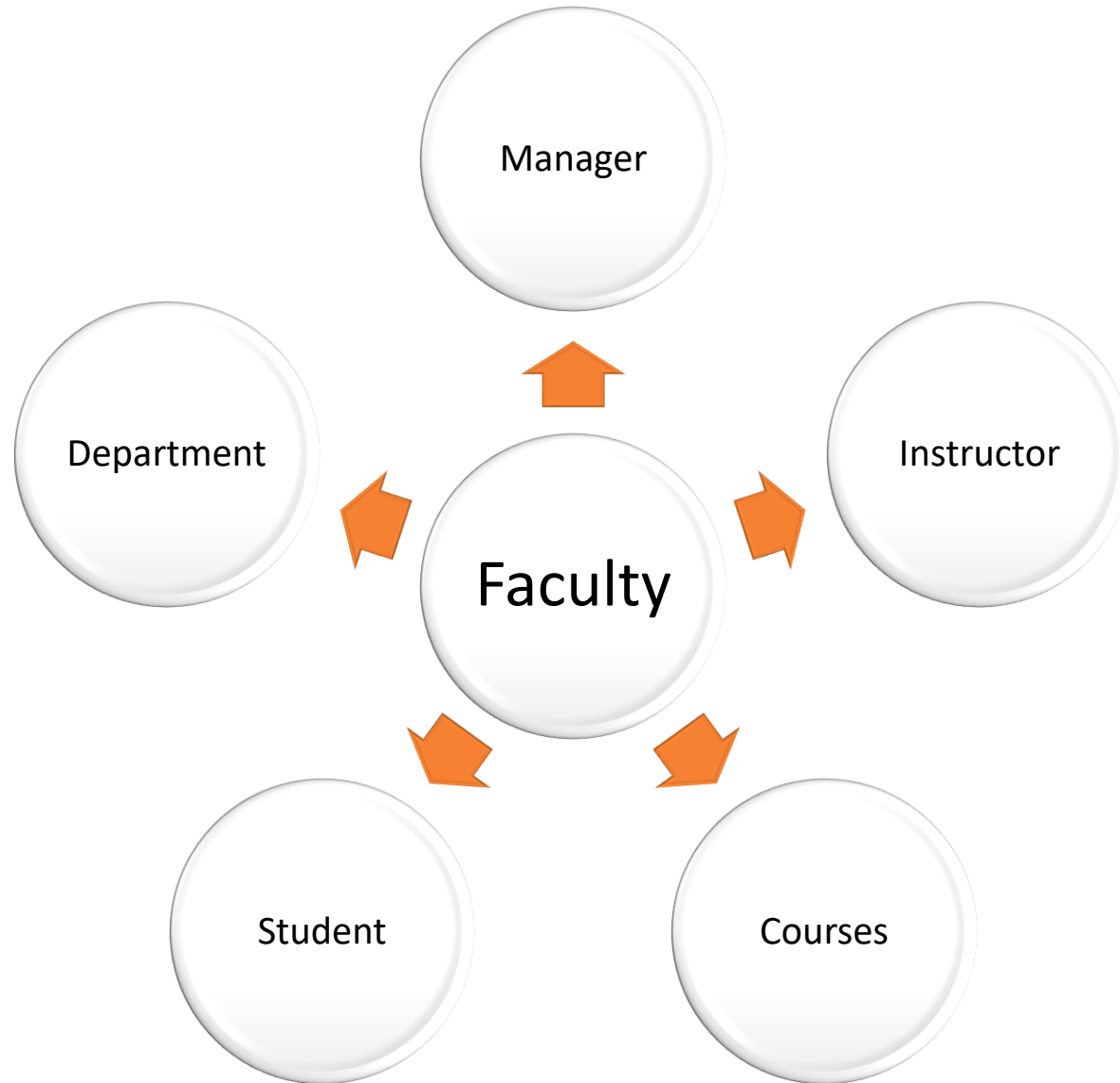




Sayed Abdul-Aziz

@sayedabdulaziz





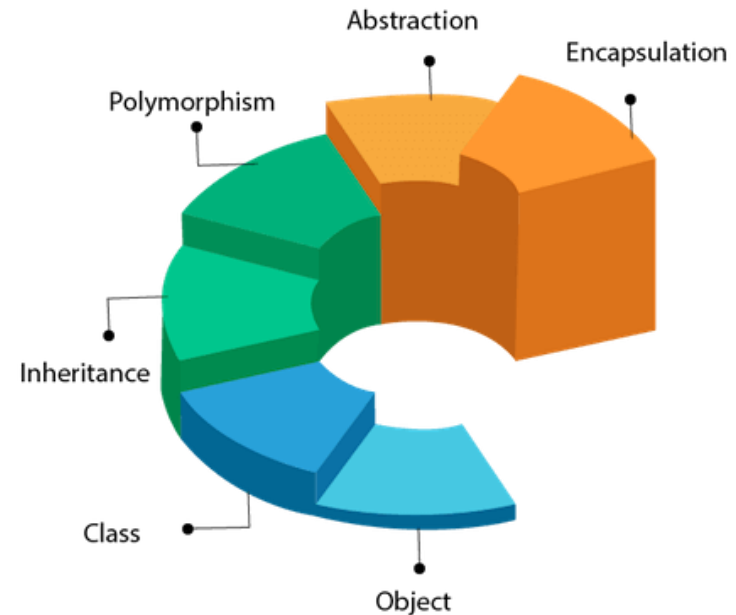
Sayed Abdul-Aziz

@sayedabdulaziz





Object-oriented programming (OOP) is a programming method that uses objects and their interactions to design and program applications. It is one of the most popular programming paradigms and is used in many programming languages, such as Dart, Java, C++, Python, etc.



Sayed Abdul-Aziz

@sayedabdulaziz





Sayed Abdul-Aziz

@sayedabdulaziz





WHAT IS AN OBJECT ?

- ❑ In real world everything around us is **object** .. Including us 😊 .
- ❑ Real-world Objects share two characteristics: They all have state and behavior.
 - **Object's state(attributes)** is the object description.
 - **Object's behavior(methods)** is the functionality of this.
- ❑ Think of any object around you ... What is the state and behavior of it?
- ❑ Humans have common **states(age ,length, weight, eye color, ...)** and common **related-behavior(walk, talk, eat, drink)**.





WHAT IS A CLASS?

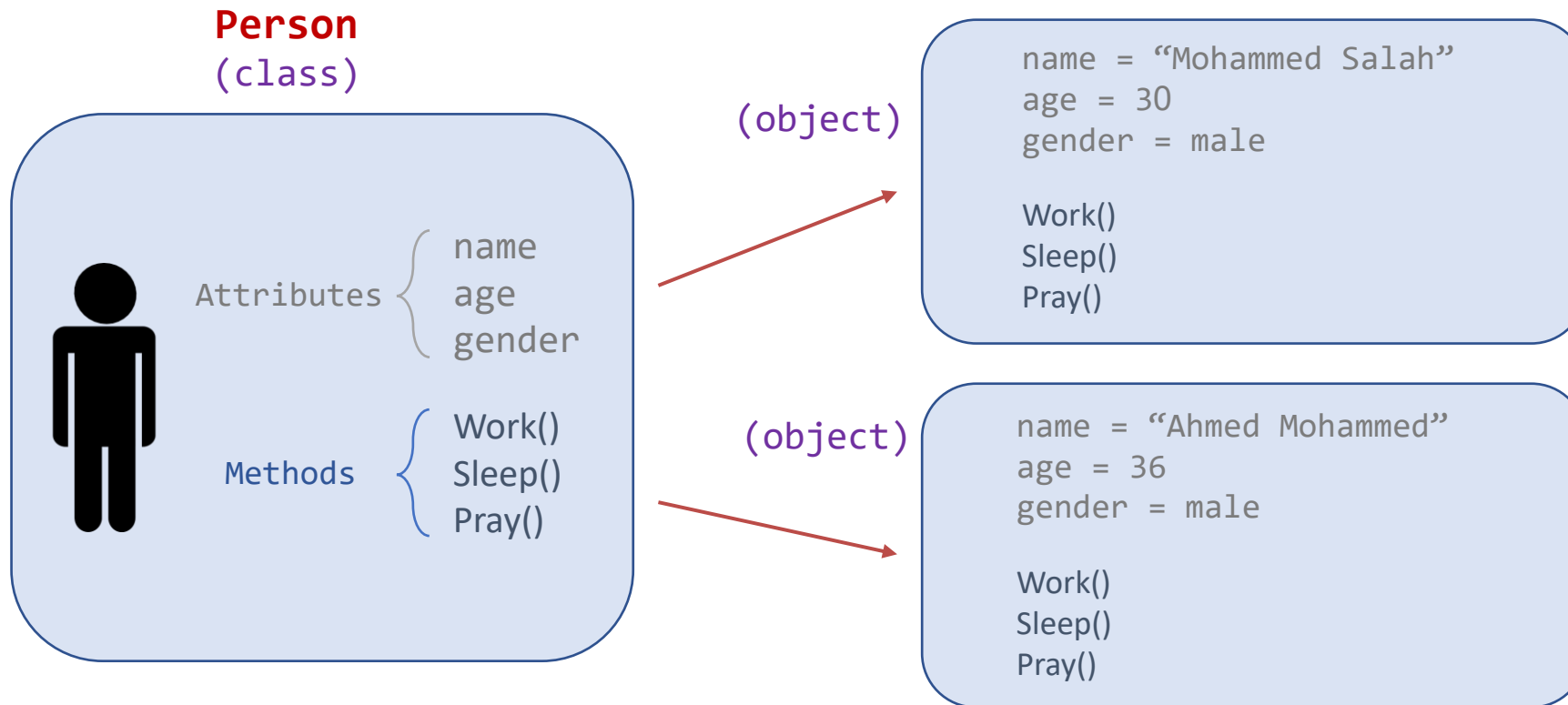
- ❑ It is a **template or blueprint** from which objects are created.
- ❑ A class is a **group of objects which have common properties**.
- ❑ A class in Dart can contain:
 - Fields
 - Methods
 - Constructors



Sayed Abdul-Aziz

@sayedabdulaziz







CLASS AND OBJECT

```
class Person {  
  
    String? name;  
    int? age;  
    String? gender;  
  
    void work( ) {  
        //body  
    }  
    void sleep( ) {  
        //body  
    }  
    void pray( ) {  
        //body  
    }  
}
```

```
main(){  
    Person p1 = Person( );  
    p1.name = "Mohammed Salah" ;  
    p1.age = 30 ;  
    p1.gender = "male" ;  
  
    Person p2 = Person( );  
    p2.name = "Ahmed Ali" ;  
    p2.age = 35 ;  
    p2.gender = "male" ;  
}
```





CLASS AND OBJECT

No.	Object	Class
1)	Object is an instance of a class.	Class is a blueprint or template from which objects are created.
2)	Object is a real world entity such as pen, laptop, mobile, bed, keyboard, mouse, chair etc.	Class is a group of similar objects.
3)	Object is a physical entity.	Class is a logical entity.
4)	Object is created through new keyword mainly e.g. <code>Student s1= Student();</code>	Class is declared using class keyword e.g. <code>class Student{}</code>
5)	Object is created many times as per requirement.	Class is declared once.
6)	Object allocates memory when it is created.	Class doesn't allocated memory when it is created.





EXAMPLE

Write a program that would print the information (name, year, salary) of three employees by creating a class named 'Employee'. The output should be as follows:

Name	Year	salary
Sayed	2000	2400
Ahmed	2002	5000
Alaa	2003	3560



Sayed Abdul-Aziz

@sayedabdulaziz





CONSTRUCTOR

- ❑ A **special method** that is used to initialize objects.
- ❑ The constructor is called when an **object of a class is created**. It can be used to set initial values for object attributes at the time of object creation.
- ❑ Constructor name must be **the same as its class name**.
- ❑ A Constructor must have **no explicit return type**.
- ❑ If a class doesn't have a constructor, the compiler **automatically creates a default constructor** during run-time. The default constructor initializes instance variables with **default values**.





CONSTRUCTOR CALLED

```
class Class1 {  
  
    //attributes  
    //Methods  
  
}
```

```
class Class2 {  
  
    //attributes  
    Class2( ){  
        //body of constructor  
    }  
    //Methods  
  
}
```

```
main(){  
    Class1 obj = Class1 ( );  
    Class2 obj = Class2 ( );  
}
```



Sayed Abdul-Aziz

@sayedabdulaziz





TYPES OF CONSTRUCTOR

- ❑ In dart, constructors can be divided into 3 types:

1) Default Constructor

- dart compiler will automatically create a no-argument constructor during run-time.
- The default constructor initializes any uninitialized instance variables with default values.

```
class Person{
  int? id ;
  String? name ;
}

main(){
  Person p = Person();
  print(p.id);      //0
  print(p.name);    //null
}
```



Sayed Abdul-Aziz

@sayedabdulaziz





TYPES OF CONSTRUCTOR

- ❑ In dart, constructors can be divided into 3 types:

2) No-arg Constructor

- The signature is same as default constructor, however body can have any code unlike default constructor where the body of the constructor is empty.

```
class Person{
  int? id ;
  String? name ;

  Person(){
    id = 5;
    name = 'Sayed';
  }
}

main(){
  Person p = Person();
  print(p.id);      //5
  print(p.name);    //Sayed
}
```



Sayed Abdul-Aziz

@sayedabdulaziz





TYPES OF CONSTRUCTOR

- ❑ In dart, constructors can be divided into 3 types:

3) Parameterized Constructor

- A constructor that has parameters is known as parameterized constructor. If we want to initialize fields of the class with your own values, then use a parameterized constructor for example:

```
class Person{
  int? id ;
  String? name ;

  Person(int id ,String name){
    this.id = id;
    this.name = name ;
  }
}
```

```
main(){

  Person p = Person(5, 'Sayed');
  print(p.id);      //5
  print(p.name);    //Sayed
}
```



Sayed Abdul-Aziz

@sayedabdulaziz



WRITE CONSTRUCTOR SINGLE LINE

```
class Person{
    String? name;
    int? id;

    // Constructor in short form
    Person(this.name, this.id);

    // display method
    void display(){
        print("Name: ${this.name}");
        print("ID: ${this.id}");
    }
}
```

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



Sayed Abdul-Aziz

@sayedabdulaziz

