Sri SAI RAM ENGINEERING COLLEGE
INSTITUTE OF TECHNOLOGY
West Tambaram, Chennai - 44

SAIRAM DIGITAL RESOURCES

YEAR II SEM III

CS8392

OBJECT ORIENTED PROGRAMMING
(Common to EEE, CSE, EIE, ICE, IT)

UNIT NO 2

INHERITANCE AND INTERFACES

2.5 INTERFACES – DEFINITION, DECLARATION
CLASSES AND INTERFACES

COMPUTER SCIENCE & ENGINEERING

INTERFACES - DEFINITION

- Interfaces specify what a class must do and not how.
- It is the blueprint of the class.
- It is a collection of abstract methods. Along with abstract methods, an interface may also contain constants, default methods, static methods, and nested types.
- A class implements an interface, thereby inheriting the abstract methods of the interface.
- The interface in Java is a mechanism to achieve abstraction. There can be only abstract methods in the Java interface, not method body.
- It is used to achieve abstraction and multiple inheritance in Java.

DECLARING AN INTERFACE IN JAVA

- An interface is declared by using the "**interface**" keyword.
- All the methods in an interface are declared with an empty body and are public and all fields are public, static and final by default.

Syntax:

```
interface <interface_name>
{
        // declare constant fields
        // declare methods that are abstract
        // by default.
}
```

A SAMPLE INTERFACE PROGRAM

- The example taken is a “player” Interface indicating the capabilities.
- It specifies a set of methods that the class has to implement.

```
// A simple interface

interface player  {
    // public, static and finall
    final int a = 10;
    // public and abstract
    void display(); }
```

IMPLEMENTING AN INTERFACE IN JAVA

- A class that implements an interface must implement all the methods declared in the interface.
- To implement interface, the keyword "**implements**" is used.
- The class can implement more than one interface, so the **implements** keyword is followed by a comma-separated list of the interfaces implemented by the class.

Syntax:

```
class classname implements interface1, interface2,…interface n
    {  // Methods
    }
```

IMPLEMENTING AN INTERFACE IN JAVA

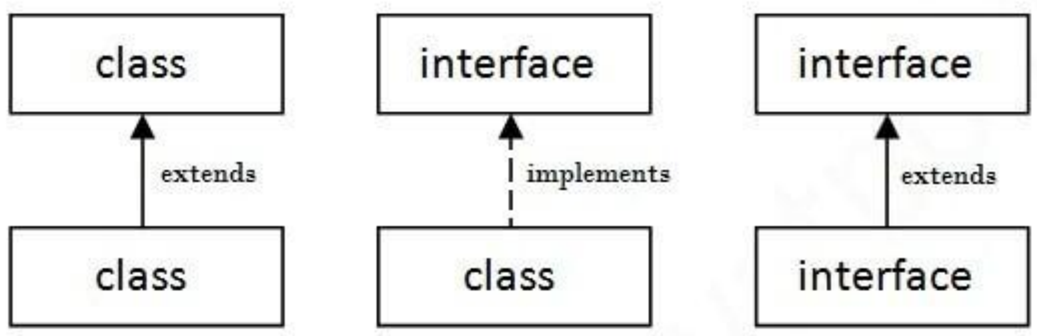
```
interface MyInterface
{
   /* compiler will treat them as:
    * public abstract void method1();
    * public abstract void method2();
    */
   public void method1();
   public void method2();
}
class Demo implements MyInterface
{
   /* This class must have to implement both the
abstract methods
    * else you will get compilation error
    */
 public void method1()
   {
         System.out.println("implementation of
method1");
   }
   public void method2()
   {
         System.out.println("implementation of
method2");
   }
   public static void main(String arg[])
   {
         MyInterface obj = new Demo();
         obj.method1();
   }
}
```

Output:

```
implementation of method1
```

RELATIONSHIP BETWEEN CLASSES AND INTERFACES

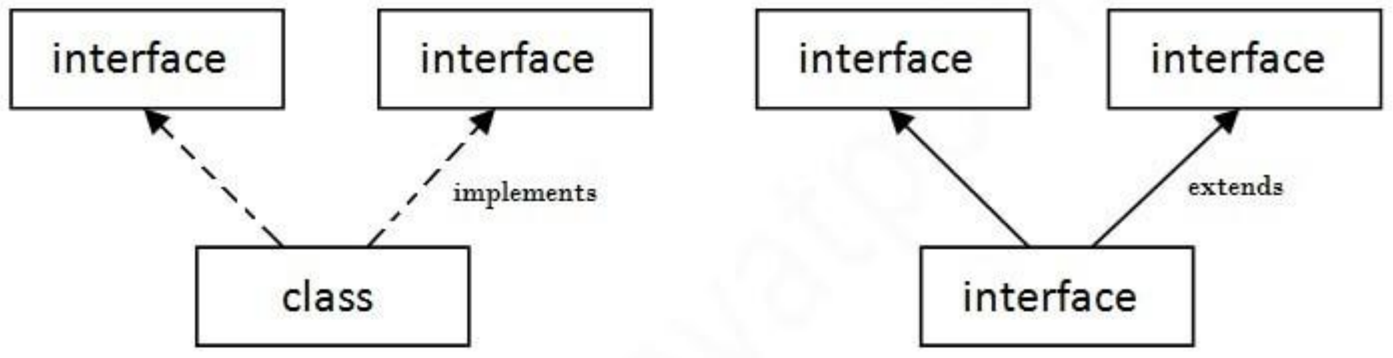
- A class extends another class, an interface extends another interface, but a **class implements an interface**.



DIFFERENCES BETWEEN CLASSES AND INTERFACES

Class	Interface
A class describes the attributes and behaviors of an object.	An interface contains behaviors that a class implements.
A class may contain abstract methods, concrete methods.	An interface contains only abstract methods.
Members of a class can be public, private, protected or default.	All the members of the interface are public by default.

MULTIPLE INHERITANCE USING INTERFACES



Multiple Inheritance in Java

VIDEO LINK

https://drive.google.com/file/d/1Cv5co_ZdPunbKEVmJ-2BzYNZ8i6jptc6/view?usp=sharing

QUIZ LINK

https://docs.google.co m/forms/d/1xoCDLfsUFIzUhy24v3TGaLj2xUoUfYF-hKWabeq4oNI/edit