Course Name - Object Oriented Programming using Java

Lecture 23 – Interface: Definition, Use of Interface.

Presented By
Dr. Sudipta Sahana
Associate Professor
Dept. of CSE
UEM - Kolkata

Topic of Interest

- Definition of Interface
- Uses of Interface
- Interface inheritance
- Declaration of Interface
- Example
- Similarities in Interface & Class
- Differences in Interface & Class

Definition of Interface

- An **interface in Java** is a blueprint of a class.
- It has static constants and abstract methods.
- The interface in Java is a mechanism to achieve abstraction.
- Interfaces can have abstract methods and variables. It cannot have a method body.
- Java Interface also represents the IS-A relationship.
- It cannot be instantiated just like the abstract class.

Uses of Interface

- It is used to achieve abstraction.
- Interface support the functionality of multiple inheritance.
- It can be used to achieve loose coupling.

Declaration of Interface

- An interface is declared by using the interface keyword.
- It provides total abstraction; means all the methods in an interface are declared with the empty body, and all the fields are public, static and final by default.
- A class that implements an interface must implement all the methods declared in the interface.

Syntax:

```
interface<interface_name> {
        // declare constant fields
        // declare methods that abstract
 }
```

Interface inheritance

- A class implements interface but one interface extends another interface .
- Example:

```
interface Printdocument {
        void print();
    }
interface Showdocument extends Printdocument {
        void show();
    }
class Test implements Showdocument {
        public void print(){System.out.println("Hello JISCE");}
        public void show(){System.out.println("Welcome");}
        public static void main(String args[]) {
                Test obj = new Test();
                obj.print();
                obj.show();
            }
    }
```

Output:

```
Hello JISCE
Welcome
```

Example

```
interface Drawable {
        void draw();
    }
class Rectangle implements Drawable {
        public void draw() {
            System.out.println("drawing rectangle");
        }
    }
class Circle implements Drawable {
        public void draw() {
            System.out.println("drawing circle");
        }
    }
class Test {
        public static void main(String args[])
            {
                Drawable d;
                Circle c=new Circle();
                d=c;
                d.draw();
                Rectangle r=new Rectangle();
                d=r;
                d.draw();
            }
    }
```

Output -
drawing circle
drawing rectangle

Similarities in Interface & Class

- An interface can contain any number of methods.
- An interface is written in a file with a **.java** extension, with the name of the interface matching the name of the file.
- The byte code of an interface appears in a **.class** file.
- Interfaces appear in packages, and their corresponding bytecode file must be in a directory structure that matches the package name.

Differences in Interface & Class

- Interface cannot be instantiated.
- An interface does not contain any constructors.
- All of the methods in an interface are abstract.
- An interface cannot contain instance fields. The only fields that can appear in an interface must be declared both static and final.
- An interface is not extended by a class; it is implemented by a class.
- An interface can extend multiple interfaces.

UEM
UNIVERSITY OF ENGINEERING & MANAGEMENT
Good Education, Good Jobs

Thank You