Interface in Java

Interface

Example of Interface

Multiple Inheritance by 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*. There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple Inheritance

In other words, you can say that interfaces can have abstract methods and variables. It cannot have a method body

Java Interface also **represents the IS-A relationship**.

An interface is declared by using the interface keyword. 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

    // by default.

}

#### **The relationship between classes and interfaces**

As shown in the figure given below, a class extends another class, an interface extends another interface, but a **class implements an interface**.



**interface** Bank{

**float** rateOfInterest();

}

**class** SBI **implements** Bank{

**public** **float** rateOfInterest(){**return** 9.15f;}

}

**class** PNB **implements** Bank{

**public** **float** rateOfInterest(){**return** 9.7f;}

}

**class** TestInterface2{

**public** **static** **void** main(String[] args){

Bank b=**new** SBI();

System.out.println("ROI: "+b.rateOfInterest());

}}

O/p

ROI: 9.15

## **Multiple inheritance in Java by interface**

If a class implements multiple interfaces, or an interface extends multiple interfaces, it is known as multiple inheritance.



interface A

{

int x=10;

}

interface B

{

int x=100;

}

class Hello implements A,B

{

public static void Main(String args[])

{

System.out.println(x);

System.out.println(A.x);

System.out.println(B.x);

}

}

|  |  |
| --- | --- |
| **Abstract class** | **Interface** |
| 1) Abstract class can **have abstract and non-abstract** methods. | Interface can have **only abstract** methods. it can have **default and static methods** also. |
| 2) Abstract class **doesn't support multiple inheritance**. | Interface **supports multiple inheritance**. |
| 3) Abstract class **can have final, non-final, static and non-static variables**. | Interface has **only static and final variables**. |
| 4) Abstract class **can provide the implementation of interface**. | Interface **can't provide the implementation of abstract class**. |
| 5) The **abstract keyword** is used to declare abstract class. | The **interface keyword** is used to declare interface. |
| 6) An **abstract class** can extend another Java class and implement multiple Java interfaces. | An **interface** can extend another Java interface only. |
| 7) An **abstract class** can be extended using keyword "extends". | An **interface** can be implemented using keyword "implements". |
| 8) A Java **abstract class** can have class members like private, protected, etc. | Members of a Java interface are public by default. |