**Day 20**

**Interface in Java**

* 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](https://www.javatpoint.com/abstract-class-in-java).
* There can be only abstract methods in the Java interface, not method body.
* Java Interface also represents the IS-A relationship.

Usage

* It is used to achieve abstraction and
* multiple [inheritance in Java](https://www.javatpoint.com/inheritance-in-java).

Declaration

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 abstract methods

}

Example

interface printable

{

int min=5;

void print();

}

#### The Java compiler adds public and abstract keywords before the interface method. Moreover, it adds public, static and final keywords before data members.

interface printable

{

public static final int min =5;

public abstract void print();

}

**Relationships between classes and interfaces**

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

extends

class class

implements

class interface

extends

interface interface

Example

interface printable

{

void print();

}

class A implements printable

{

public void print()

{

System.out.println(“Hello");

}

public static void main(String args[])

{

A ob = new A();

ob.print();

}

}

**Difference between abstract class and interface**

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

**Multiple Inheritance in Java**

// different type method

interface Printable

{

void print();

}

interface Showable()

{

void show();

}

class A implements Printable, Showable

{

public void print()

{

System.out.println(“Print”);

}

public void show()

{

System.out.println(“Print”);

}

public static void main(String args[])

{

A ob = new A();

ob.print();

ob.show();

}

// same type method

interface Printable

{

void print();

}

interface Showable()

{

void print();

}

class A implements Printable, Showable

{

public void print()

{

System.out.println(“Print”);

}

public static void main(String args[])

{

A ob = new A();

ob.print();

}