Welcome you all

JAVA PROGRAMMING

DAY : 5

D.Sakthivel
Assistant Professor & Trainer,
KGiSL Micro College
KGiSL Campus, Coimbatore – 641 035.

Day 6

- Interface
- Multiple Inheritance
- **Interface Inheritance**

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*.
- There can be only abstract methods in the Java interface, not method body.
- It is used to achieve abstraction and multiple inheritance in Java.
- In other words, you can say that interfaces can have abstract methods and variables. It hod body.

Why use Java interface?

There are mainly three reasons to use interface. They are given below.

- It is used to achieve abstraction.
- By interface, we can support the functionality of multiple inheritance.
- It can be used to achieve loose coupling.

It is used to achieve abstraction.
1
2
By interface, we can support the functionality of multiple inheritance.
It can be used to achieve loose coupling.
3

How to declare an 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.*

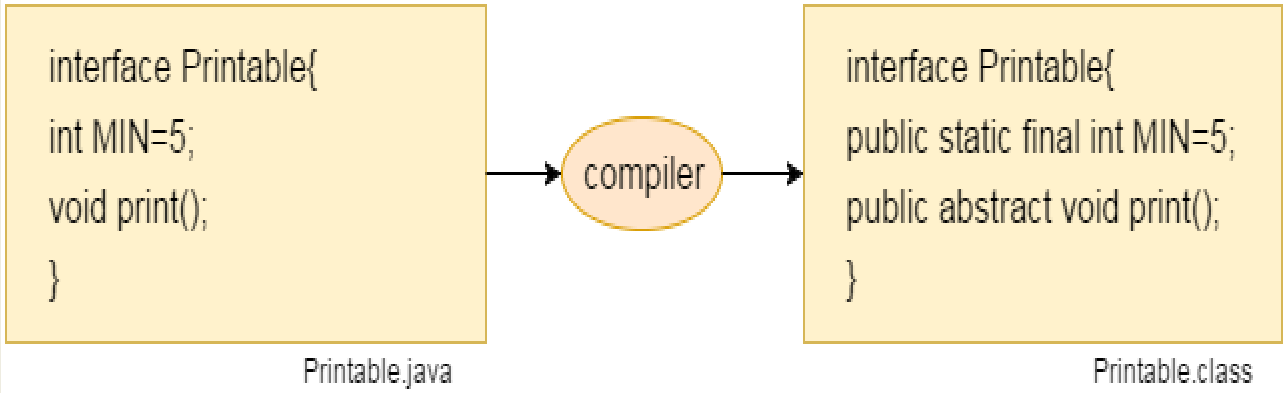
Syntax:

```
interface <interface_name>
{

    // declare constant fields
    // declare methods that abstrac
t
    // by default.
}
```

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

In other words, Interface fields are public, static and final by default, and the methods are public and abstract.



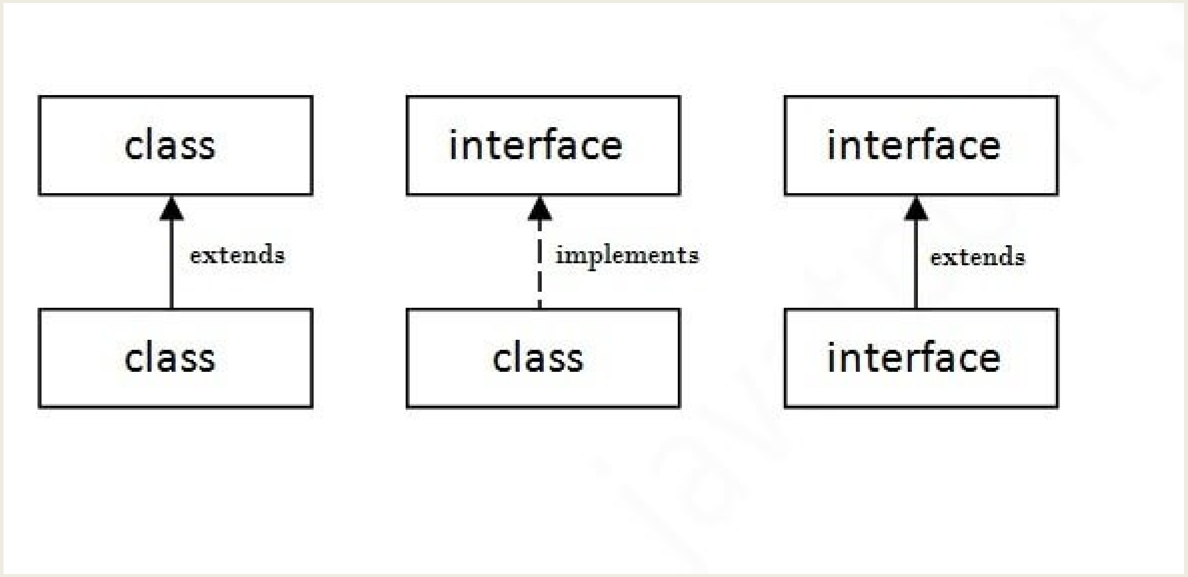
```
interface Printable{
int MIN=5;
void print();
}
```

Printable.java

```
interface Printable{
public static final int MIN=5;
public abstract void print();
}
```

Printable.class

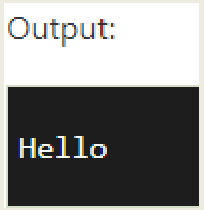
• The relationship between classes and interfaces



Interface Examples

```
interface printable{
void print();
}
class A6 implements printable{
public void print()
{System.out.println("Hello");}

public static void main(String args[]){
A6 obj = new A6();
obj.print();
 }
}
```

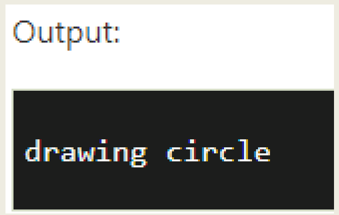


```
Output:

Hello
```

In this example, the Drawable interface has only one method. Its implementation is provided by Rectangle and Circle classes. In a real scenario, an interface is defined by someone else, but its implementation is provided by different implementation providers. Moreover, it is used by someone else. The implementation part is hidden by the user who uses the interface.

```
/Interface declaration: by first user
interface Drawable{
void draw();
}
//Implementation: by second user
class Rectangle implements Drawable{
public void draw(){System.out.println("drawing rectangle");}
}
class Circle implements Drawable{
public void draw(){System.out.println("drawing circle");}
}
//Using interface: by third user
class TestInterface1{
public static void main(String args[]){
Drawable d=new Circle();//
In real scenario, object is provided by method e.g. getD
d.draw();
}}
```

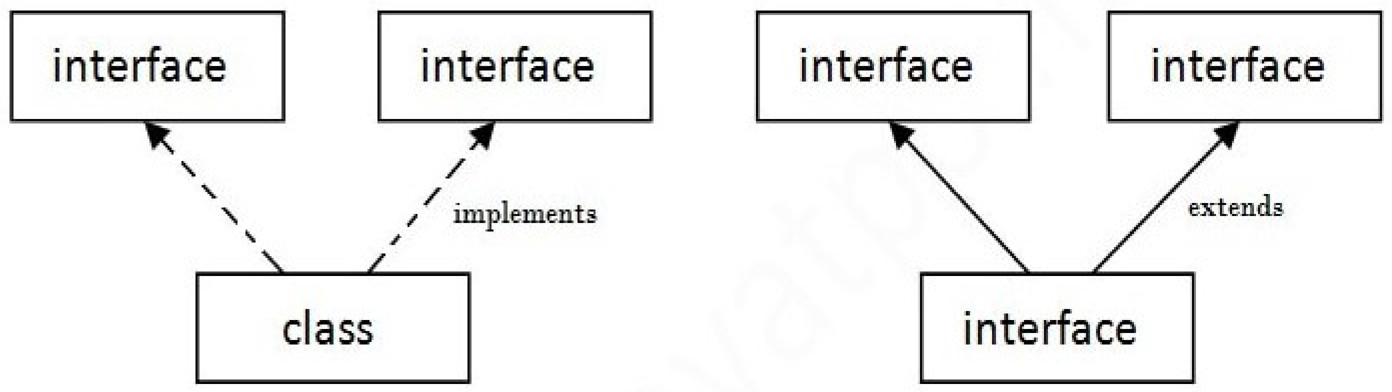


Output:

```
drawing circle
```

Multiple inheritance in Java by interface

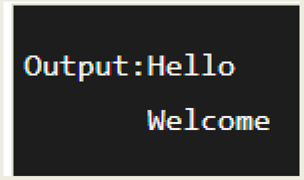
If a class implements multiple interfaces, or an interface extends multiple interfaces, it is known as multiple inheritance.



Multiple Inheritance in Java

```
interface Printable{
void print();
}
interface Showable{
void show();
}
class A7 implements Printable,Showable{
public void print(){System.out.println("Hello");}
public void show()
{System.out.println("Welcome");}

public static void main(String args[]){
A7 obj = new A7();
obj.print();
obj.show();
 }
}
```



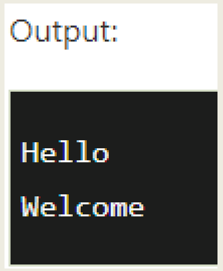
```
Output:Hello
       Welcome
```

interface inheritance

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

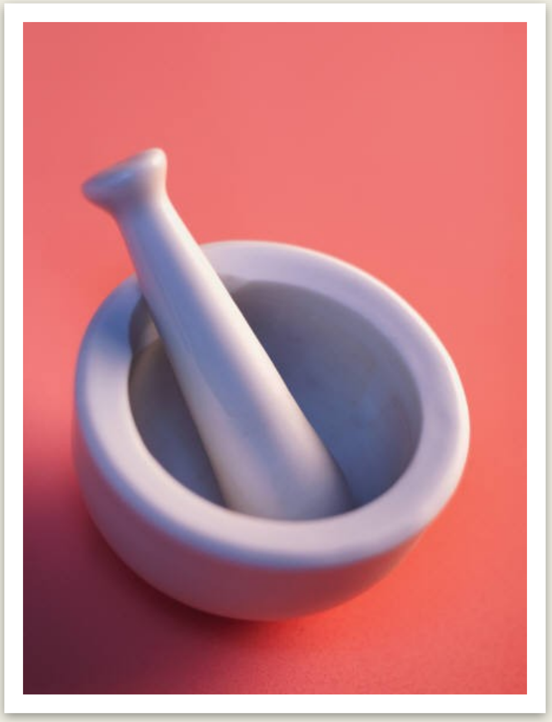
```
interface Printable{
void print();
}
interface Showable extends Printable{
void show();
}
class TestInterface4 implements Showa
ble{
public void print()
{System.out.println("Hello");}
public void show()
{System.out.println("Welcome");}

public static void main(String args[]){
TestInterface4 obj = new TestInterface4()
;
obj.print();
obj.show();
 }
}
```



Output:

```
Hello
Welcome
```



Thank You