Object Oriented Programming using Java

Prepared By:
Suyel, PhD
Assistant Professor
Dept. of CSE, NIT Patna

Outline

- Inheritance
- Interface
- Examples

Inheritance

- Java allows us to derive new class from an existing class. It is known as inheritance.
- Inheritance is an important concept of Java.
- Sometimes, we need to derive a subclass from several classes. Thus, inheriting their data and methods.
- **What is the issue of inheritance?**
  - Only one parent class is allowed (multiple inheritance is not allowed).
  - **What is the solution?**
  - With interface, we can obtain multiple inheritance.

Interface

- A Java **interface** is a collection of constants and abstract methods. So, it is the responsibility of the class to implement an interface to give the code of methods.
- Here, all the fields are public, static and final by default.
- An interface in Java is a blueprint of a class.
- It is a mechanism to achieve total abstraction.
- Interface cannot be instantiated like abstract class. Only classes that implement interfaces can be instantiated.
- **interface** keyword allows us to utilize the “one interface, multiple methods” aspect of polymorphism.

Interface (Cont...)

- Methods in an interface have public visibility by default.
- We can have default and static methods in an interface from Java 8.
- Java also allows private method in an interface from Java 9.
- One class can implement many interfaces.
- One interface can be implemented by many classes.
- An interface does not contain any constructors.
- An interface is not extended by a class. But, it is **implemented** by a class.
- An interface can extend multiple interfaces.

Interface (Cont…)

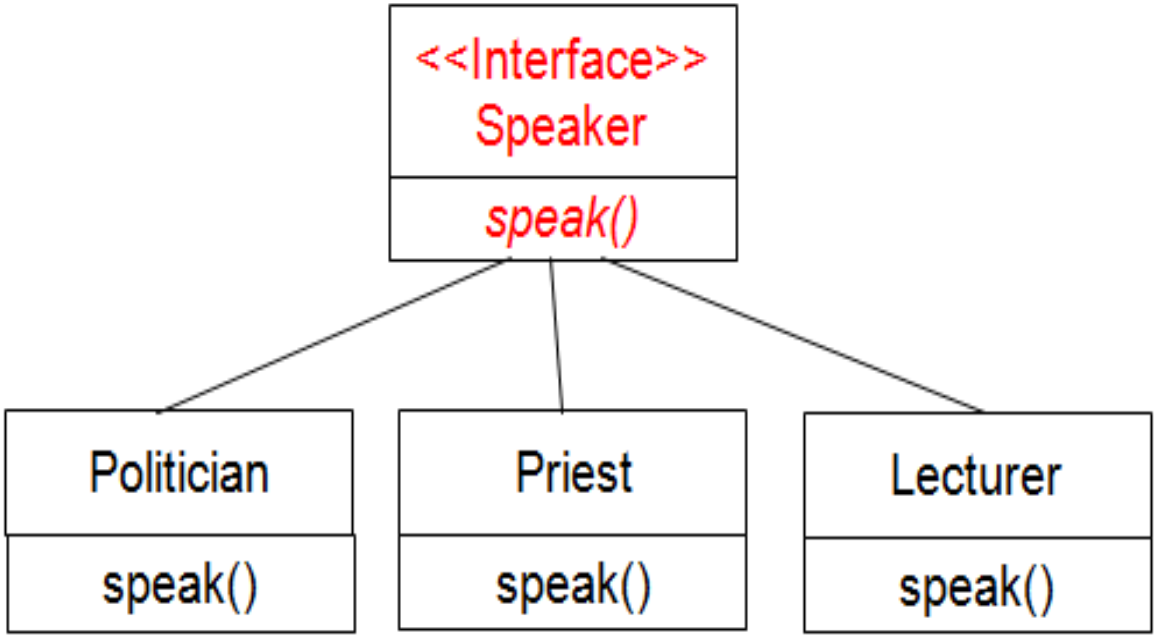
Application

- Similar to abstract class, interface supports us to achieve abstraction in Java.
- Interfaces are used to achieve multiple inheritance in Java.
- Interface can be used for achieving loose coupling.

Interface (Cont...)

Class	Interface
We can create an object in class.	We can’t create an object in interface.
Class contain concrete methods.	Interface cannot contain concrete methods.
Access specifiers: private, protected and public.	Access specifiers: public.

Interface (Cont...)



Interface (Cont...)

- **Syntax:**

```
interface Interface_Name
{
    Constant/Final Variable Declaration
    Methods Declaration
}
```

- **Example:**

```
interface Speaker
{
    public void speak( );
}
```

Interface (Cont...)

- **Syntax to implement Interface:**

```
class Class_Name implements Interface_Name [, Interface_Name2, …]
{
    Body of Class
 }
```

- **Example:**

```
class  Politician  implements Speaker
{
    public void speak()
    {
        System.out.println("Talk politics");
    }
}
```

Interface (Cont...)

- **Syntax to extend Interface:**

```
interface Interface_Name2 extends Interface_Name1
{
        Body of Interface_Name2
}
```

- **Syntax to implement Inheritance and Interface:**

```
class Class_Name extends Super_Class implements Interface_Name [,
 Interface_Name2, …]
{
        Body of Class
}
```

Example

```
interface Test
{
   void print();
}

class Interface implements Test
{
   public void print()
   {
      System.out.println("Hello!!! How are you?");
   }

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

Example (Cont...)

- **Output**

  Hello!!! How are you?

Example (Cont...)

```
interface Test
{
    final int x = 5;
    void display();
}

class Interface1 implements Test
{
    public void display()
    {
        System.out.println("We are at NIT Patna");
    }

    public static void main (String args[])
    {
        Test obj = new Interface1();
        obj.display();
        System.out.println(x);
    }
}
```

Example (Cont...)

- **Output**

  We are at NIT Patna

  5

Example (Cont...)

```
interface RBI
{
   float interest();
}

class SBI implements RBI
{
   public float interest()
   {
      return 1000.1565f;
   }

   void print()
   {
      System.out.println("It is one of the popular banks");
   }
}

class PNB implements RBI
{
   public float interest()
   {
      return 9.7f;
   }
}

class Interface2
{
   public static void main(String args[])
   {
       RBI obj = new SBI();
       System.out.println("Return is: "+ obj.interest());
       obj.print();
   }
}
```

Example (Cont...)

- **Output**

```
Interface2.java:34: error: cannot find symbol
        obj.print();
            ^
  symbol:   method print()
  location: variable obj of type RBI
1 error
```

Example (Cont...)

```
interface Multiple_Inheritance1
{
   void print();
}

interface Multiple_Inheritance2
{
   void show();
}

class Interface3 implements Multiple_Inheritance1, Multiple_Inheritance2
{
   public void print()
   {
      System.out.println("This is the First Program of Multiple Inheritance");
   }

   public void show()
   {
      System.out.println("This is one of the Important Topics of OOP");
   }

   public static void main(String args[])
   {
      Interface3 obj = new Interface3();
      obj.print();
      obj.show();
   }
}
```

Example (Cont...)

- **Output**

  This is the First Program of Multiple Inheritance

  This is one of the Important Topics of OOP

Example (Cont...)

```
interface Bihar
{
   void print();
}

interface Patna extends Bihar
{
   void show();
}

class Interface4 implements Patna
{
   public void print()
   {
       System.out.println("Hello All");
   }

   public void show()
   {
       System.out.println("You can always contact me for anything");
   }

   public static void main(String args[])
   {
       Interface4 obj = new Interface4();
       obj.print();
       obj.show();
   }
}
```

Example (Cont...)

- **Output**

  Hello All

  You can always contact me for anything

Example (Cont...)

```
interface Test
{
   void print();

   default void show()
   {
      System.out.println("This is Default method");
   }
}

class Interface5 implements Test
{
    public void print()
    {
        System.out.println("Are you enjoying here?");
    }

    public static void main(String args[])
    {
        Interface5 obj = new Interface5();
        obj.print();
        obj.show();
    }
}
```

Example (Cont...)

- **Output**

  Are you enjoying here?

  This is Default method

Example (Cont...)

```
interface NITP
{
   void print();

   static int square(int x)
   {
      System.out.println("Mathematics?");
      return x*x;
   }
}

class Patna implements NITP
{
    public void print()
    {
       System.out.println("Draw a Square");
    }
}

class Interface6
{
    public static void main(String args[])
    {
        NITP obj = new Patna();
        obj.print();
        System.out.println(NITP.square(10));
    }
}
```

Example (Cont...)

- **Output**

  Draw a Square

  Mathematics

  100

Example (Cont...)

```
interface NITP
{
   void print();

   static int square(int x)
   {
      int y = x*x;
      System.out.println(y);
      return -1;
   }
}

class Interface7 implements NITP
{
    public void print()
    {
       System.out.println("Draw a Square");
    }

    public static void main(String args[])
    {
       Interface7 obj = new Interface7();
       obj.print();
       NITP.square(10);
       System.out.println(NITP.square(10));
    }
}
```

Example (Cont...)

- **Output**

  Draw a Square

  100

  100

  -1

Example (Cont...)

```
interface NITP
{
   void print();
}

interface Bihar
{
   void print();
}

class Interface8 implements NITP, Bihar
{
   public void print()
   {
      System.out.println("How are you?");
   }

   public static void main(String args[])
   {
      Interface8 obj = new Interface8();
      obj.print();
   }
}
```

Example (Cont...)

- **Output**

  How are you?

Example (Cont...)

```
interface NITP
{
   void print();
}

interface Bihar
{
   int print();
}

class Interface9 implements NITP, Bihar
{
   public void print()
   {
      System.out.println("How are you?");
   }

   public int print()
   {
      System.out.println("Enjoy each moment of life");
      return 0;
   }

   public static void main(String args[])
   {
      Interface9 obj = new Interface9();
      obj.print();
   }
}
```

Example (Cont...)

- **Output**

```
Interface9.java:19: error: method print() is already defined in class Interface9
   public int print()
              ^
Interface9.java:12: error: Interface9 is not abstract and does not override abst
ract method print() in Bihar
class Interface9 implements NITP, Bihar
^
Interface9.java:14: error: print() in Interface9 cannot implement print() in Bih
ar
   public void print()
               ^
  return type void is not compatible with int
3 errors
```

Example (Cont...)

```
interface Polygon
{
    void getArea();

    default void getSide()
    {
        System.out.println("Default method of polygon");
    }
}

class Rectangle implements Polygon
{
    public void getArea()
    {
        int ln = 6, br = 5;
        int ar = ln * br;
        System.out.println("Area of the rectangle: " + ar);
    }

    public void getSide()
    {
        System.out.println("There are 4 sides");
    }
}

class Square implements Polygon
{
    public void getArea()
    {
        int ln = 5;
        int ar = ln * ln;
        System.out.println("Area of the square: " + ar);
    }
}

class Interface10
{
    public static void main(String args[])
    {
        Rectangle obj1 = new Rectangle();
        obj1.getArea();
        obj1.getSide();

        Square obj2 = new Square();
        obj2.getArea();
        obj2.getSide();
    }
}
```

Example (Cont...)

- **Output**

```
Area of the rectangle: 30
There are 4 sides
Area of the square: 25
Default method of polygon
```


thank
you . . .

Slides are prepared from various sources, such as Book, Internet Links and many more.