INTERFACE

PURPOSE OF INTERFACE:

1. **Provides communication** − Through the interface you can specify how you want the methods and fields of a particular type.
2. **Multiple inheritance** − Java doesn’t support multiple inheritance, using interfaces you can achieve multiple inheritance
3. **Abstraction** − All the methods of the interface are abstract and the user doesn’t know how a method is written except the method signature/prototype. Using interfaces, you can achieve (complete) abstraction.
4. **Loose coupling** − Coupling refers to the dependency of one object type on another, if two objects are completely independent of each other and the changes done in one doesn’t affect the other both are said to be loosely coupled.You can achieve loose coupling in Java using interface.

INTERFACE :

An interface in Java is a specification of method prototypes. Whenever you need to guide the programmer or make a contract specifying how the methods and fields of a type should be you can define an interface.

To create an object of this type you need to implement this interface, provide a body for all the abstract methods of the interface and obtain the object of the implementing class.

SYNTAX:

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

EXAMPLE 1:

```
interface printable
{
      void print();
}
class A6 implements printable
{
      public void print()
      {
           System.out.println("Hello");
       }
}
public class Main
{
      public static void main(String args[])
      {
          A6 obj = new A6();
         obj.print();
          }
}
```

Output :

```
      Hello
```

In this example,the printable interface has only one method,and its implementation is provided in the A6 class.

Example 2:

```
class A
{
     int a=10;
}
interface B
{
    int b=30;
}
class C extends A implements B
{
     int t;
     void sum()
     {
          t=a+b;
         System.out.println("hi"+t);
     }
}
public class Main
{
     public static void main (String[] args)
    {
       C c=new C();
       c.sum();
    }
}
```

Output:

```
  hi40
```

Through creating a object to class C ,class A could be inherited and properties of Class B could be inherited through the property of interface.

Note :

Any number of_interfaces could be created but not two classes could be inherited at the same time.