

Java Interface

[< Previous](#)[Next >](#)

Java Interface

An **interface** is an abstract **class** that is used to group related methods with "empty" bodies:

Example

```
// interface
interface Animal {
    public void animalSound(); // interface method (does not have a body)
    public void run(); // interface method (does not have a body)
}
```

To access the interface methods, the interface must be "implemented" (kinda like inherited) by another class with the **implements** keyword (instead of **extends**):

Example

```
// interface
interface Animal {
    public void animalSound(); // interface method (does not have a body)
    public void sleep(); // interface method (does not have a body)
}

// Pig "implements" the Animal interface
class Pig implements Animal {
    public void animalSound() {
        System.out.println("The pig says: wee wee");
    }
}
```

```
public void sleep() {  
    System.out.println("Zzz");  
}  
}  
  
class MyMainClass {  
    public static void main(String[] args) {  
        Pig myPig = new Pig(); // Create a Pig object  
        myPig.animalSound();  
        myPig.sleep();  
    }  
}
```

[Run example »](#)

Notes on Interfaces:

- It **cannot** be used to create objects (in the example above, it is not possible to create an "Animal" object in the MyMainClass)
- Interface methods does not have a body - the body is provided by the "implement" class
- On implementation of an interface, you must override all of its methods
- Interface methods are by default **abstract** and **public**
- Interface attributes are by default **public**, **static** and **final**
- An interface cannot contain a constructor (as it cannot be used to create objects)

Why use Interfaces?

To achieve security - hide certain details and only show the important details of an object (interface).

Java does not support "multiple inheritance" (a class can only inherit from one superclass). However, it can be achieved with interfaces, because the class can **implement** multiple interfaces. **Note:** To implement multiple interfaces, separate them with a comma (see example below).

Multiple Interfaces

To implement multiple interfaces, separate them with a comma:

Example

```
interface FirstInterface {
    public void myMethod(); // interface method
}

interface SecondInterface {
    public void myOtherMethod(); // interface method
}

// DemoClass "implements" FirstInterface and SecondInterface
class DemoClass implements FirstInterface, SecondInterface {
    public void myMethod() {
        System.out.println("Some text..");
    }
    public void myOtherMethod() {
        System.out.println("Some other text...");
    }
}

class MyMainClass {
    public static void main(String[] args) {
        DemoClass myObj = new DemoClass();
        myObj.myMethod();
        myObj.myOtherMethod();
    }
}
```

[Run example »](#)

[< Previous](#)

[Next >](#)