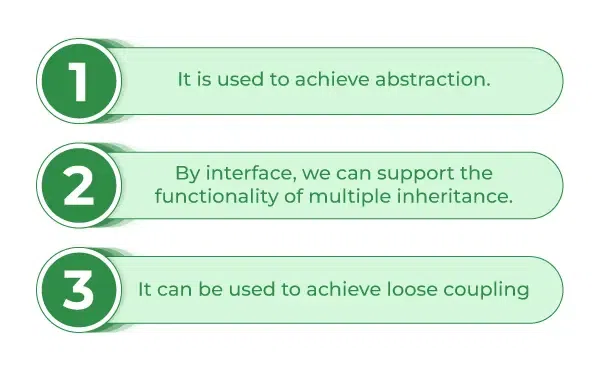
# **Interfaces in Java**

An **Interface in Java** programming language is defined as an abstract type used to specify the behavior of a class. An interface in Java is a blueprint of a behavior. A Java interface contains static constants and abstract methods.

## **What are Interfaces in Java?**

The interface in Java is *a* mechanism to achieve abstraction. There can be only abstract methods in the Java interface, not the method body. It is used to achieve abstraction and multiple inheritance in Java using Interface. variables in the interface are **final, public**, and **static**.

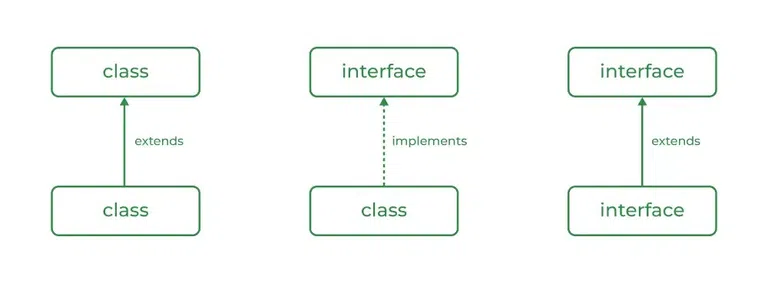


*Uses of Interfaces in Java:*

* *It is used to achieve total abstraction.*
* *Since java does not support multiple inheritances in the case of class, by using an interface it can achieve multiple inheritances.*
* *Any class can extend only 1 class, but can any class implement an infinite number of interfaces.*
* *It is also used to achieve loose coupling.*

## **Relationship Between Class and Interface**

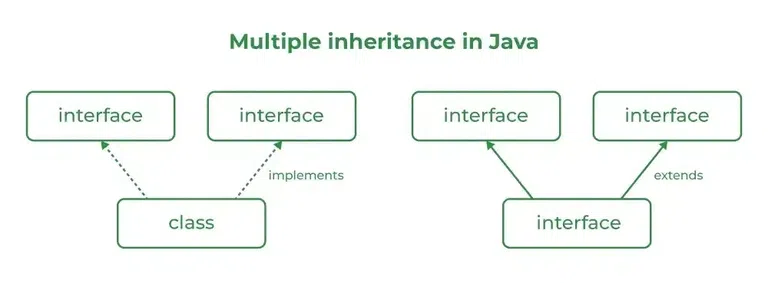
A class can extend another class similar to this and an interface can extend another interface. But only a class can extend to another interface, and vice-versa is not allowed.



| **Class** | **Interface** |
| --- | --- |
| Objects can be created for a concrete class | In an interface, you can’t instantiate and create an object |
| A class can contain concrete (with implementation) methods | The interface cannot contain concrete (with implementation) methods. |
| The access specifiers used with classes are default, private, protected, and public. | In Interface only one specifier is used- Public. All methods are **public** and variables are **public**, **static** and **final** by default |

## **Multiple Inheritance in Java Using Interface**

Multiple Inheritance is an OOPs concept that can’t be implemented in Java using classes. But we can use multiple inheritance in Java using Interface.



The interface contains multiple abstract methods, so provide the implementation in implementation classes. If the implementation class is unable to provide the implementation of all abstract methods, then declare the implementation class with an abstract modifier, and complete the remaining method implementation in the next created child classes. It is possible to declare multiple child classes but at final we should have completed the implementation of all abstract methods.

*In general, the development process is step by step:*

***Level 1*** *– interfaces: It contains the service/contract details - only abstract methods****Level 2*** *– abstract classes: It contains partial implementation - abstract and non-abstract methods****Level 3*** *– implementation classes: It contains all implementations - all concrete methods and no abstract methods****Level 4*** *– Final Code / Main Method: It has access to all interface data.*