**Creating and using Interface:**

An **interface** is like the **purely abstract** class in which we can have only abstract method to **define the prototype** and not the body of the function or method.

Using the interface we can also have the feature of **multiple inheritances because a class can inherits more than one interface at a time**.

Note:

1. All the functions in the interface are by **default abstract** so don't use the abstract keyword

2. All the function in the interface are **public** by default so don't even use the public modifier

3. An interface can inherit another interface

4. A class that extends the interface must provide the body / implementation of all the abstract functions of the interface

**Difference Between Interface and Abstract Class**

1. Main difference is methods of a Java interface are implicitly abstract and cannot have implementations. A Java abstract class can have instance methods that implements a default behavior.
2. Variables declared in a Java interface is by default final. An  abstract class may contain non-final variables.
3. Members of a Java interface are public by default. A Java abstract class can have the usual flavors of class members like private, protected, etc..
4. Java interface should be implemented using keyword “implements”; A Java abstract class should be extended using keyword “extends”.
5. An interface can extend another Java interface only, an abstract class can extend another Java class and implement multiple Java interfaces.
6. A Java class can implement multiple interfaces but it can extend only one abstract class.
7. Interface is absolutely abstract and cannot be instantiated; A Java abstract class also cannot be instantiated, but can be invoked if a main() exists.
8. In comparison with java abstract classes, java interfaces are slow as it requires extra indirection.