**INTERFACES**

Interface looks like class but it is not a class. An interface can have methods and variables just like the class but the methods declared in interface are by default abstract (only method signature, no body). Also, the variables declared in an interface are public, static & final by default. We will discuss these points in detail, later in this post.

**What is the use of interfaces?**

As mentioned above they are used for abstraction. Since methods in interfaces do not have body, they have to be implemented by the class before you can access them. The class that implements interface must implement all the methods of that interface. Also, java programming language does not support multiple inheritance, using interfaces we can achieve this as a class can implement more than one interfaces, however it cannot extend more than one classes.

**Declaration**

Interfaces are declared by specifying a keyword “interface”. E.g.:

**interface MyInterface**

**{**

 /\* All the methods are public abstract by default

 \* Note down that these methods are not having body

 \*/

 public void method1();

 public void method2();

}

**Interface Implementation**

This is how a class implements an interface. It has to provide the body of all the methods that are declared in interface.

Note: class implements interface but an interface extends another interface.

interface MyInterface

{

 public void method1();

 public void method2();

}

class XYZ implements MyInterface

{

 public void method1()

 {

 System.out.println("implementation of method1");

 }

 public void method2()

 {

 System.out.println("implementation of method2");

 }

 public static void main(String arg[])

 {

 MyInterface obj = new XYZ();

 obj. method1();

 }

}

**Output:**

implementation of method1