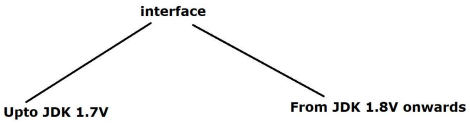
interface in Java :



interface Upto JDK 1.7V

* interface is a keyword in Java which is similar to a class.

Example :
Drawable.java

```
public interface Drawable
{

}
```

javac

Drawable.class

```
public interface Drawable
{
   //Compiler will add few
   methods of Object  class
}
```

* An interface describes the **"Working functionality of a class"**
* An interface contains only abstract methods and static and final (static blank final field) fields
* interface methods are by default public and abstract where as fields are by default **public, static & final.**
* In order to implement the abstract methods of an interface, We should use **implements** keyword.
* All the abstract methods defined inside an interface must be **overridden in the implementer classes otherwise implementer class will become as an abstract class.**
* As we know, interface methods are by default **public and abstract** so due to **public**, we cannot reduce the visibility at the time of overriding.
* An interface contains only abstract methods so, We can achive 100% abstraction.
* The main purpose of interface to provide **loose coupling facility.**
* We **cannot** write non static field, any kind of initializer (static block OR non static block), constructor.
* By using interface, we can achieve **multiple inheritance**
* An interface is implicitly an abstract class.
* We cannot create an object for interface.
* From JDK 1.8V (java 8 features) we are allowed to write default and static method inside an interface.

//Program :

```java
package com.ravi.interface_demo;

interface Drawable
{
    void draw(); //public + abstract
}

class Draw implements Drawable
{
    @Override
    public void draw()
    {
        IO.println("Drawing something!!!!");
    }
}

public class InterfaceDemo
{
    public static void main(String[] args)
    {
        Drawable d = new Draw();
        d.draw();

    }

}
```