C# Interface

Interface in C# is a blueprint of a class. It is like abstract class because all the methods which are declared inside the interface are abstract methods. It cannot have method body and cannot be instantiated.

It is used *to achieve multiple inheritance* which can't be achieved by class. It is used *to achieve fully abstraction* because it cannot have method body.

Its implementation must be provided by class or struct. The class or struct which implements the interface, must provide the implementation of all the methods declared inside the interface.

C# interface example

Let's see the example of interface in C# which has draw() method. Its implementation is provided by two classes: Rectangle and Circle.

```
using System;
public interface Drawable
{
    void draw();
}
public class Rectangle : Drawable
{
    public void draw()
    {
        Console.WriteLine("drawing rectangle...");
    }
}
public class Circle : Drawable
{
    public void draw()
    {
        Console.WriteLine("drawing circle...");
    }
}
public class TestInterface
{
    public static void Main()
    {
        Drawable d;
        d = new Rectangle();
        d.draw();
        d = new Circle();
        d.draw();
    }
}
```

Output:

```
drawing ractangle...
drawing circle...
```

Note: Interface methods are public and abstract by default. You cannot explicitly use public and abstract keywords for an interface method.

```
using System;
public interface Drawable
{
    public abstract void draw();//Compile Time Error
}
```