

Interfícies

La interfície d'una classe defineix el **comportament d'aquesta classe**, ja que defineix que podem i que no podem fer amb objectes d'aquesta classe.

Totes les **classes tenen una interfície** que defineix què podem fer amb els objectes d'aquesta classe.

```
interface IContenedor
{
    int Quitar();
    void Meter(int i);
}
```

Una interfície defineix la llista de mètodes que pot portar un objecte, només declara el comportament que serà implementat per les classes que implementin la interfície.

No es poden crear objectes, instàncies d' interfície...

```
IContenedor c = new IContenedor();  
// Error: No se puede crear una interfaz!
```




Una interfície pot ser implementada per més d'una classe.

I una classe pot implementar més d'una interfície, seria lo més semblant a l'herència múltiple.

No tenim herència múltiple en C#

Qualsevol classe que implementi una interfície deu definir tots els mètodes d'aquesta interfície, sinó s'implementen tots els mètodes de la interfície aquesta haurà de ser declarada com abstracta.

```
class Contenedor : IContenedor
{
    public int Quitar() { ... }
    public void Meter(int i) { ... }
}
```

```
class Contenedor : IContenedor
{
    public void Meter(int i) { ... }
}
// Error: Y el método Quitar()???
```


Quan es fa servir interfícies:

- Sempre que veiem que més d'una classe poden fer lo mateix.
- No tota classe ha d'implementar una interfície obligatòriament.
- Fer servir interfícies permet després poder canviar una classe per una altra que implementi la mateixa interfície(només s'haurien d'adaptar les instàncies i la resta de codi quedaria igual).

```
ial.cs | intTutorial.cs | Program.cs
-----
ProyectoInter.intTutorial
1 using System.Linq;
2 using System.Text;
3 using System.Threading.Tasks;
4
5 namespace ProyectoInter
6 {
7     1 referencia
8     interface intTutorial
9     {
10         2 referencias
11         void SetTutorial(int pId, string pName);
12         2 referencias
13         string GetTutorial();
14     }
15 }
```

Exemple1

Debug Any CPU Iniciar

ClsEditorial.cs x intTutorial.cs Program.cs

C# ProyectoInter ProyectoInter.ClsEditorial GetTutorial()

```
7 namespace ProyectoInter
8 {
9     2 referencias
10    class ClsEditorial:intTutorial
11    {
12        protected int IdTutorial;
13        protected string NameTutorial;
14        2 referencias
15        public void SetTutorial(int pId, string pName) { IdTutorial=pId;
16            NameTutorial = pName;
17        }
18        2 referencias
19        public string GetTutorial() { return this.NameTutorial; }
20    }
21 }
```

105 %


```
{
    0 referencias
    static void Main(string[] args)
    {
        {
            ClsEditorial tutorial1 = new ClsEditorial();

            tutorial1.SetTutorial(1, "Js");

            Console.WriteLine(tutorial1.GetTutorial());

            Console.ReadKey();
        }
    }
}
```

Definir dos interfícies: **IntArchivo** y **IntImprimir**

La primera tiene dos métodos que se llaman:Leer y Escribir, no necesitan argumentos y no devuelven nada.

La segunda tiene un método Imprimir que tampoco necesita argumentos ni devuelve nada.

Definir dos clases:ClsDocumento y ClsRectangulo.

La primera tiene:

- sólo una propiedad que es contenido que es un texto.

- un constructor que inicializa la propiedad.

- hereda de las dos interfícies, sus métodos dan por salida un texto diciendo la acción que realizan.

La segunda tiene:

- dos propiedades ancho y alto que son números enteros.

- un constructor que inicializa las dos propiedades.

- hereda de la interfície IntImprimit y su método tiene el mismo comportamiento que en la clase anterior.

Definir instancias y llamar a los diferentes métodos.

```
IntImprimir.cs  ClsRectangulo.cs  IntArchivo.cs  Progi
C# ProyectoInterfacesImpri  ProyectoInterfacesImpri.IntArch
5  using System.Threading.Tasks;
6
7  namespace ProyectoInterfacesImpri
8  {
9      1 referencia
10     interface IntArchivo
11     {
12         2 referencias
13         void Leer();
14         2 referencias
15         void Escribir();
16     }
17 }
```

```
IntImprimir.cs  ClsRectangulo.cs  IntArchivo.cs
C# ProyectoInterfacesImpri  ProyectoInterfacesImpri
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ProyectoInterfacesImpri
8  {
9      2 referencias
10     interface IntImprimir
11     {
12         3 referencias
13         void Imprimir();
14     }
15 }
```

emple2


```
ir.cs  ClsRectangulo.cs  IntArchivo.cs  Program.cs
toInterfacesImpri
using System.Threading.Tasks;

namespace ProyectoInterfacesImpri
{
    3 referencias
    class ClsRectangulo: IntImprimir
    {
        int ancho;
        int alto;
        1 referencia
        public ClsRectangulo(int lado1,int lado2)
        {
            this.ancho = lado1;
            this.alto = lado2;
        }
        3 referencias
        public void Imprimir()
        {
            Console.WriteLine("ancho={0} y alto=
            {1}",this.ancho,this.alto);
        }
    }
}
```

```
s  ClsRectangulo.cs  ClsDocumento.cs*  IntArchiv
InterfacesImpri  ProyectoInterfacesImpri.ClsDocume
3 referencias
class ClsDocumento: IntArchivo, IntImprimir
{
    string contenido;
    1 referencia
    public ClsDocumento(string frase)
    {
        this.contenido = frase;
    }
    3 referencias
    public void Imprimir()
    {
        Console.WriteLine(contenido);
    }
    2 referencias
    public void Leer()
    {
        Console.WriteLine("Leyendo contenido");
    }
    2 referencias
    public void Escribir()
    {
        Console.WriteLine("Escribiendo contenido");
    }
}
```










```

namespace ProyectoInterfacesImpri
{
    0 referencias
    class Program
    {
        0 referencias
        static void Main(string[] args)
        {
            ClsDocumento unDoc = new ClsDocumento("Primer contenido");
            ClsRectangulo unRect = new ClsRectangulo(4,7);
            unDoc.Escribir();
            unDoc.Leer();
            unDoc.Imprimir();
            unRect.Escribir; //Da error
        }
    }
}

```

Buscar en Explorador de soluciones (Ctrl+Shift+B)

Solución "ProyectoInterfacesImpri"

-  **ProyectoInterfacesImpri**
 -  Properties
 -  Referencias
 -  App.config
 -  ClsDocumento.cs
 -  ClsRectangulo.cs
 -  IntArchivo.cs
 -  IntImprimir.cs
 -  Program.cs