



Abstracción

Definición Formal:


- Efecto de separar conceptualmente algo de algo.

 El modificador abstract se utiliza para indicar que una clase está incompleta y que sólo se va a utilizar como una clase base.

Abstracción de Clases

 Una clase abstracta se diferencia de una clase no abstracta en lo siguiente:

- No se puede crear una instancia de una clase abstracta directamente, y es un error en tiempo de compilación utilizar el operador new en una clase abstracta. Aunque es posible tener variables y valores cuyos tipos en tiempo de compilación sean abstractos, tales variables y valores serán null o contendrán referencias a instancias de clases no abstractas derivadas de los tipos abstractos.
- Se permite que una clase abstracta contenga miembros abstractos, aunque no es necesario.
- No se puede sellar una clase abstracta.

 Cuando una clase no abstracta se deriva de una clase abstracta, la clase no abstracta debe incluir implementaciones reales de todos los miembros abstractos heredados; por lo tanto, reemplaza a estos miembros abstractos.

abstract class A

```
{  
    public abstract void F();  
}
```

abstract class B : A

```
{  
    public void G() {}  
}
```

class C : B

```
{  
    public override void F()  
    {  
        // actual implementation of F  
    }  
}
```

Métodos Virtuales

- 🔗 El modificador virtual para modificar un método, propiedad, indexador o evento declarado y permitir que sea anulado en una clase derivada.
- 🔗 Cuando una declaración de método de instancia incluye un modificador virtual, se dice que el método es un método virtual. Si no existe un modificador virtual, se dice que el método es un método no virtual.
- 🔗 La implementación de un método no virtual es invariable, o sea es la misma tanto si se invoca un método en una instancia de la clase en la que se declaró o en una instancia de una clase derivada.
- 🔗 En cambio, en un método virtual se puede sustituir por clases derivadas. El proceso de sustitución de la implementación de un método virtual heredado es conocido como reemplazamiento del método.
- 🔗 En la invocación de un método virtual, el tipo en tiempo de ejecución de la instancia para la que tiene lugar la invocación determina la implementación del método real a invocar. Cuando se invoca un método no virtual, el factor determinante es el tipo en tiempo de compilación de la instancia.
- 🔗 Concretamente, cuando se invoca un método denominado N con una lista de argumentos A en una instancia con un tipo C en tiempo de compilación y un tipo R en tiempo de ejecución (donde R es C o una clase derivada de C).
 - 🔗 la invocación se procesa de la forma siguiente:
 - 🔗 En primer lugar, la resolución de sobrecarga se aplica a C, N y A para seleccionar un método específico M del conjunto de métodos declarados en y heredados por C. Esto se describe con más detalle en la Sección 7.5.5.1.
 - 🔗 A continuación, si M es un método no virtual, se invoca M.
 - 🔗 Si no, M es un método virtual, y se invoca la implementación más derivada de M con respecto a R.

Virtuales vs No Virtuales

```
class A {  
    public void F() { Console.WriteLine("A.F"); }  
    public virtual void G() { Console.WriteLine("A.G"); }  
}  
class B : A {  
    new public void F() { Console.WriteLine("B.F"); }  
    public override void G() { Console.WriteLine("B.G"); }  
}  
class Test {  
    static void Main()  
    {  
        B b = new B();  
        A a = b;  
        a.F();  
        b.F();  
        a.G();  
        b.G();  
    }  
}
```

Métodos Abstractos - Abstracción de Métodos

- ✚ Cuando una declaración de método de instancia incluye un modificador abstract, se dice que el método es un método abstracto. Aunque un método abstracto es también implícitamente un método virtual, no puede tener el modificador virtual.
- ✚ Una declaración de método abstracto introduce un nuevo método virtual pero no proporciona una implementación del método. En cambio, es necesario que las clases derivadas no abstractas proporcionen su propia implementación mediante el reemplazo del método. Debido a que un método abstracto no proporciona una implementación real, el cuerpo-del-método de un método abstracto consiste simplemente en un punto y coma.
- ✚ Las declaraciones de métodos abstractos sólo se permiten en clases abstractas.

```
public abstract class Shape
{
    public abstract void Paint(Graphics g, Rectangle r);
}
public class Ellipse : Shape
{
    public override void Paint(Graphics g, Rectangle r)
    {
        g.DrawEllipse(r);
    }
}
public class Box : Shape
{
    public override void Paint(Graphics g, Rectangle r)
    {
        g.DrawRect(r);
    }
}
```

Redefinición de Métodos

- ✚ El modificador override es necesario para ampliar o modificar la implementación abstracta o virtual de un método, propiedad, indizador o evento heredado.
- ✚ Un método override proporciona una nueva implementación de un miembro que se hereda de una clase base. El método invalidado por una declaración override se conoce como método base invalidado. El método base reemplazado debe tener la misma firma que el método override.
- ✚ No se puede reemplazar un método estático o no virtual. El método base reemplazado debe ser virtual, abstract u override.
- ✚ Una declaración override no puede cambiar la accesibilidad del método virtual. El método override y el método virtual deben tener el mismo modificador de nivel de acceso.
- ✚ No se pueden usar los modificadores new, static o virtual para modificar un método override.
- ✚ Una declaración de propiedad de invalidación debe especificar exactamente el mismo modificador de acceso, tipo y nombre que la propiedad heredada, y la propiedad invalidada debe ser virtual, abstract u override.

```
public override string ToString()
{
    return "Mi propio ToString()";
}
```