Interfaces

Las interfaces, como las clases, definen un conjunto de propiedades, métodos y eventos. Pero de forma contraria a las clases, las interfaces no proporcionan implementación. Se implementan como clases y se definen como entidades separadas de las clases. Una interfaz representa un contrato, en el cual una clase que implementa una interfaz debe implementar cualquier aspecto de dicha interfaz exactamente como esté definido.

Para definir una interfaz:

```
interface ISampleInterface
{
    void doSomething();
}
```

Para implementar una interfaz en una clase:

```
class SampleClass : ISampleInterface
{
    void ISampleInterface.doSomething()
    {
        // Method implementation.
    }
}
```

Una interfaz contiene las definiciones de un grupo de funcionalidades relacionadas que una clase o una estructura pueden implementar.

Mediante las interfaces puede incluir, por ejemplo, un comportamiento de varios orígenes en una clase. Esta capacidad es importante en C# porque el lenguaje no admite la herencia múltiple de clases. Además, debe usar una interfaz si desea simular la herencia de estructuras, porque no pueden heredar de otra estructura o clase.

Defina una interfaz mediante la palabra clave interface, como se muestra en el ejemplo siguiente.

```
interface IEquatable<T>
{
    bool Equals(T obj);
}
```

El nombre de la estructura debe ser un nombre de identificador de C# válido. Por convención, los nombres de interfaz comienzan con una letra `I` mayúscula.

Cualquier clase o estructura que implementa la interfaz IEquatable<T> debe contener una definición para un método Equals que coincida con la firma que la interfaz especifica. Como resultado, puede contar con una clase que implementa `IEquatable<T>` para contener un método `Equals` con el que una instancia de la clase puede determinar si es igual a otra instancia de la misma clase.

La definición de `IEquatable<T>` no proporciona una implementación para `Equals`. La interfaz define solo la firma.De esa manera, una interfaz en C# es similar a una clase abstracta en la que todos los métodos son abstractos.Sin embargo, una clase o estructura puede implementar varias interfaces, pero una clase solo puede heredar una clase única, ya sea abstracta o no.

Para obtener más información sobre las clases abstractas, vea Abstract and Sealed Classes and Class Members(Clases y miembros de clase abstractos y sellados [Guía de programación de C#]).

Las interfaces pueden contener métodos, propiedades, eventos, indizadores o cualquier combinación de estos cuatro tipos de miembros. Para obtener vínculos a ejemplos, vea Secciones relacionadas. Una interfaz no puede contener constantes, campos, operadores, constructores de instancias, finalizadores ni tipos. Los miembros de interfaz son públicos automáticamente y no pueden incluir modificadores de acceso. Los miembros tampoco pueden ser estáticos.

Para implementar un miembro de interfaz, el miembro correspondiente de la clase de implementación debe ser público, no estático y tener el mismo nombre y firma que el miembro de interfaz.

Cuando una clase o estructura implementa una interfaz, la clase o estructura debe proporcionar una implementación para todos los miembros que define la interfaz. La propia interfaz no proporciona ninguna funcionalidad que una clase o estructura puedan heredar de la misma la forma en que pueden heredar la funcionalidad de la clase base. Sin embargo, si una clase base implementa una interfaz, cualquier clase que se derive de la clase base hereda esta implementación.

En el siguiente ejemplo se muestra una implementación de la interfaz IEquatable<T>. La clase de implementación `Car` debe proporcionar una implementación del método Equals.

```
public class Car : IEquatable<Car>
{
    public string Make {get; set;}
    public string Model { get; set; }
    public string Year { get; set; }

    // Implementation of IEquatable<T> interface
    public bool Equals(Car car)
    {
        return this.Make == car.Make &&
               this.Model == car.Model &&
               this.Year == car.Year;
    }
}
```

Las propiedades y los indizadores de una clase pueden definir descriptores de acceso adicionales para una propiedad o indizador que estén definidos en una interfaz. Por ejemplo, una interfaz puede declarar una propiedad que tenga un descriptor de acceso get. La clase que implementa la interfaz puede declarar la misma propiedad con un descriptor de acceso `get` y set. Sin embargo, si la propiedad o el indizador usan una implementación explícita, los descriptores de acceso deben coincidir. Para obtener más información sobre la implementación explícita, vea Implementación de interfaz explícita y Propiedades de interfaces.

Las interfaces pueden heredar de otras interfaces. Una clase puede incluir una interfaz varias veces mediante las clases base que hereda o mediante las interfaces que otras interfaces heredan. Sin embargo, la clase puede proporcionar una implementación de una interfaz solo una vez y solo si la clase declara la interfaz como parte de la definición de la clase (`class ClassName : InterfaceName`). Si la interfaz se hereda porque se heredó una clase base que implementa la interfaz, la clase base proporciona la implementación de los miembros de la interfaz. Sin embargo, la clase derivada puede volver a implementar cualquier miembro de la interfaz virtual, en lugar de usar la implementación heredada.

Una clase base también puede implementar miembros de interfaz mediante el uso de los miembros virtuales. En ese caso, una clase derivada puede cambiar el comportamiento de la interfaz reemplazando los miembros virtuales. Para obtener más información sobre los miembros virtuales, vea la sección de Polimorfismo.

Resumen de interfaces

Una interfaz tiene las propiedades siguientes:

- Una interfaz es como una clase base abstracta con solo miembros abstractos. Cualquier clase o estructura que implementa la interfaz debe implementar todos sus miembros.
- No se puede crear una instancia de una interfaz directamente. Sus miembros se implementan por medio de cualquier clase o estructura que implementa la interfaz.
- Las interfaces pueden contener eventos, indizadores, métodos y propiedades.
- Las interfaces no contienen ninguna implementación de métodos.
- Una clase o estructura puede implementar varias interfaces. Una clase puede heredar una clase base y también implementar una o varias interfaces.