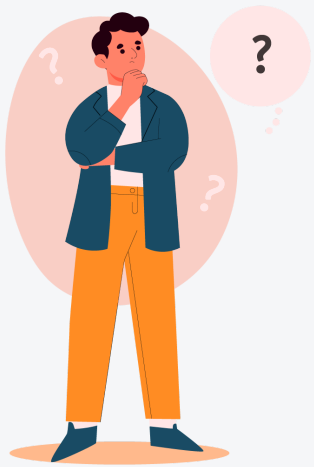
Atributos y métodos estáticos

Sitio: Agencia de Habilidades para el Futuro
Curso: Desarrollo de Sistemas Orientado a Objetos 1º D
Libro: Atributos y métodos estáticos

Imprimido por: Eduardo Moreno
Día: domingo, 1 de junio de 2025, 14:23

Tabla de contenidos

Preguntas orientadoras



Atributos Estáticos:

- ¿Qué es un atributo estático en C# y cuál es su propósito principal en la programación?
- ¿Cómo se declara un atributo estático en una clase? ¿Cuál es la diferencia entre un atributo estático y uno de instancia?
- ¿En qué tipo de escenarios sería útil utilizar un atributo estático en lugar de un atributo de instancia?
- ¿Cómo se accede a un atributo estático desde diferentes partes de tu código? ¿Cuál es la sintaxis adecuada para ello?
- ¿Pueden los atributos estáticos ser modificados después de su inicialización? ¿Por qué?

Métodos Estáticos:

- ¿Qué es un método estático y cómo se diferencia de un método de instancia en C#?
- ¿Cuál es la principal ventaja de utilizar métodos estáticos? ¿En qué situaciones son particularmente útiles?
- ¿Cómo se declara y se define un método estático en una clase? ¿Qué restricciones existen en cuanto a su acceso a miembros de instancia?
- ¿Puede un método estático llamar a otros métodos estáticos y no estáticos? ¿Hay alguna limitación en este sentido?
- ¿Cuál es la diferencia entre una llamada a un método estático y una llamada a un método de instancia?

Introducción

Hasta aquí, a lo largo de la cursada, hemos avanzando en distintos temas. En este libro, te proponemos dos recorridos: abordar los atributos y, luego, los métodos.

En primer lugar, descubriremos cómo los atributos estáticos pueden otorgar a nuestras clases y miembros propiedades compartidas en toda la aplicación, lo que resulta esencial para el mantenimiento y la gestión de datos comunes.

A continuación, exploraremos los métodos estáticos, que nos permiten ejecutar acciones sin necesidad de crear instancias de una clase, lo que puede llevar a una mayor eficiencia en términos de memoria y procesamiento.

A través de ejemplos prácticos y ejercicios, aprenderemos cómo declarar y utilizar atributos y métodos estáticos en diferentes situaciones. También analizaremos cómo estos conceptos se integran en proyectos más amplios y cómo pueden contribuir a la creación de aplicaciones sólidas y flexibles.

Prepárense para adentrarse en el mundo de la programación en C# y para dominar la utilización de atributos y métodos estáticos.

Sin más preámbulos, ¡comencemos!

Atributos Estáticos: Concepto y Utilidad

Acerca del qué y para qué

Los atributos estáticos son miembros de clase en C# que son compartidos entre todas las instancias de esa clase y que persisten a lo largo de la vida útil de la aplicación.

A diferencia de los atributos de instancia, que tienen una copia única para cada objeto creado, los atributos estáticos mantienen un solo valor para toda la clase.

Estos atributos pueden utilizarse para almacenar información común, valores compartidos o configuraciones globales que son relevantes para toda la aplicación.

A continuación, desglosemos cómo se declaran y acceden.

Declaración y Acceso

Para declarar un atributo estático, se utiliza la palabra clave static antes de la declaración del atributo en la clase.

Por ejemplo:

```
class MiClase
{
    static int contador = 0;
}
```

El acceso a los atributos estáticos se realiza mediante el nombre de la clase, seguido del nombre del atributo:

```
int valor = MiClase.contador;
```

¡Vayamos a repasar las ventajas, consideraciones y precauciones!

Ventajas y precauciones

Ventajas

- Compartir Información Global: Los atributos estáticos permiten almacenar información que es relevante para toda la clase y todas sus instancias, evitando la duplicación de datos
- Eficiencia de Memoria: Dado que solo hay una instancia del atributo para toda la clase, se ahorra memoria en comparación con múltiples copias de atributos de instancia.
- Configuración y Parámetros Globales: Son útiles para mantener configuraciones globales en la aplicación, como configuraciones de bases de datos, rutas de archivos o valores predeterminados.

Consideraciones y Precauciones

- Acceso a Miembros de Instancia: Los atributos estáticos no tienen acceso directo a miembros de instancia, ya que no están vinculados a objetos específicos. Solo pueden acceder a otros miembros estáticos.
- Multi-Hilo (Multithreading): Si se utiliza en un entorno multi-hilo, es importante implementar medidas de funcionamiento para evitar problemas de concurrencia.
- Iniciación y Carga: Los atributos estáticos se inicializan cuando la clase se carga en la memoria, lo que puede afectar el rendimiento.

A continuación, veamos un ejemplo práctico

Ejemplo práctico

Supongamos que estamos construyendo una aplicación de gestión de empleados.

Podríamos utilizar un atributo estático para mantener un contador de empleados en la empresa, que se incremente cada vez que se agregue un nuevo empleado:

```
class Empleado
{
    static int contador = 0;
    string nombre;
    public Empleado(string nombre)
    {
        this.nombre = nombre;
        contador++;
    }
```

```
    public static int ObtenerCantidadEmpleados()
    {
        return contador;
    }
}
```

Conclusión

Los atributos estáticos son una herramienta poderosa en C# para compartir información común y mantener valores globales en toda la aplicación.

Al comprender cómo se declaran, acceden y utilizan, puede optimizar el diseño y la estructura de su código de manera efectiva.

Sin embargo, es esencial ser consciente de las consideraciones y posibles desafíos al trabajar con atributos estáticos, especialmente en aplicaciones multi-hilo.

Métodos Estáticos: definición y características

Ahora, vamos a detenernos a analizar qué son los métodos estáticos.

En principio, podemos definir que los métodos estáticos son miembros de clase en C# que pertenecen a la clase en sí mismo en lugar de una instancia específica de la clase.

A diferencia de los métodos de instancia, los métodos estáticos no requieren la creación de objetos para ser invocados y se pueden llamar directamente utilizando el nombre de la clase.

Son utilizados para implementar funcionalidades que son independientes de las propiedades y estados de objetos individuales.

A continuación, desglosemos cómo se declaran y cuál es su uso.

Declaración y uso

Para declarar un método estático, se utiliza la palabra clave static antes de la firma del método en la clase.

Por ejemplo:

```
class Calculadora
{
    public static int Sumar(int a, int b)
    {
        return a + b;
    }
}
```

El método estático Sumarse puede llamar directamente usando el nombre de la clase:

```
int resultado = Calculadora.Sumar(5, 3);
```

Repasemos sus ventajas y limitaciones.

Ventajas y limitaciones

Ventajas de métodos estáticos

- Acceso Directo: Los métodos estáticos no requieren la creación de una instancia de la clase, lo que los hace convenientes para tareas que no dependen del estado del objeto
- Organización de Utilidades: Son ideales para funciones de utilidad y operaciones que no necesitan acceder a miembros de instancia.
- Eficiencia: Dado que no se requiere la creación de objetos, los métodos estáticos pueden ser más eficientes en términos de memoria y rendimiento.

Consideraciones y limitaciones

- Acceso a Miembros: Los métodos estáticos solo pueden acceder a otros miembros estáticos de la misma clase. No pueden acceder directamente a miembros de instancia.
- Herencia: Aunque los métodos estáticos se heredan, no pueden ser posteriores en una clase derivada utilizando la palabra clave override.
- Pruebas Unitarias : Pueden dificultar la realización de pruebas unitarias, ya que a menudo están acopladas directamente con clases y no pueden ser simuladas o fácilmente mejoradas.

Pero, ¿Cuáles son los escenarios de uso práctico? ¡Continuemos!

Escenarios de uso práctico

Repasemos algunos escenarios...

Funciones utilitarias:

Para implementar métodos que realizan cálculos matemáticos, conversiones de unidades, validaciones de entrada, etc.

Métodos de fábrica:

Al crear objetos complejos sin exponer la lógica interna de la construcción.

Métodos de ayuda:

Para funciones que brindan información adicional o ejecutan acciones secundarias en respuesta a eventos.

Ejemplo práctico

Consideremos un escenario en el que se necesita una función que convierta dólares a euros. Esta función no necesita acceder a ningún estado de objeto, por lo que es apropiado pensarlo como un método estático:

```
class ConversorDeMoneda
{
    static double ConvertirDolaresAEuros(double dolares)
    {
        return dolares * 0.85;
    }

}
```

Conclusión

Los métodos estáticos son una herramienta valiosa para implementar funcionalidades que son independientes del estado de los objetos.

Comprender cómo declararlos y utilizarlos adecuadamente te permitirá escribir un código más eficiente y modular.

Sin embargo, es importante tener en cuenta sus limitaciones, como el acceso sólo a miembros estáticos y las consideraciones de prueba unitaria.

En resumen

En C#, los atributos y métodos estáticos son elementos clave en la programación orientada a objetos que permiten definir comportamientos y características compartidas por todas las instancias de una clase, sin necesidad de crear objetos individuales.

Te dejamos un resumen de ambos conceptos:

Atributos estáticos:	Métodos estáticos:
• Los atributos estáticos son miembros de una clase que se asocian con la propia clase en lugar de con instancias individuales de la clase. • Se definen utilizando la palabra clave static en la declaración de campo, propiedad, método o evento. • Los atributos estáticos son compartidos por todas las instancias de la clase y se almacenan en un único lugar en la memoria. • Suelen utilizarse para almacenar valores o información que deben ser consistentes para todas las instancias de la clase. • Se accede a los atributos estáticos utilizando el nombre de la clase, seguido del nombre del atributo, por ejemplo: ClaseAtributos.NombreAtributo	• Los métodos estáticos son funciones que pertenecen a la clase en lugar de a una instancia específica de la misma. • Se definen utilizando la palabra clave static en la declaración del método. • Los métodos estáticos no pueden acceder a miembros de instancia (no pueden acceder a campos o propiedades no estáticos), pero pueden acceder a otros miembros estáticos. • Son útiles para realizar operaciones que no dependen del estado de un objeto específico y se pueden invocar directamente a través del nombre de la clase, por ejemplo: ClaseMetodos.MetodoEstatico().

Para recordar: Los atributos y métodos estáticos en C# son elementos que pertenecen a la clase en lugar de a instancias individuales, lo que les permite compartir información y comportamientos comunes entre todas las instancias de esa clase.