# **La Instrucción Using en C#**

En C#, la instrucción **using** se utiliza para importar namespaces en un archivo de código, permitiendo acceder a los tipos (clases, interfaces, estructuras, etc.) definidos dentro de esos namespaces sin necesidad de usar su nombre completo. Esto simplifica el código y mejora su legibilidad, ya que evita la necesidad de escribir los nombres de espacios largos y repetitivos. El ejemplo más común de la utilización de esta instrucción es **using System**, que al crear un nuevo proyecto nos aparece automáticamente en nuestra aplicación, si la quitáramos no podríamos usar ningún método de la clase system por lo cual no podríamos utilizar por ejemplo la clase **console** puesto que esta clase se encuentra dentro del namespace System.

# **Excepciones**

Una **excepción** es la indicación de que se produjo un error en el programa mientras se estaba ejecutando. Las excepciones, como su nombre lo indica, *se producen cuando la ejecución de un método no termina correctamente, sino que termina de manera excepcional como consecuencia de una situación no esperada*. Cuando ocurre una excepción el programa deja de funcionar abruptamente, y el usuario que lo estaba utilizando muchas veces no sabe qué fue lo que paso, por este motivo se busca evitar o controlar las excepciones en nuestra aplicación.

Cuando se produce una situación anormal durante la ejecución de un programa (por ejemplo se accede a un objeto que no ha sido inicializado o tratamos de acceder a una posición inválida en un vector), si no manejamos de manera adecuada el error que se produce, el programa va a terminar abruptamente su ejecución. Decimos que el programa deja de funcionar y es muy probable que el usuario que lo estaba utilizando ni siquiera sepa qué fue lo que pasó.

# **Control de excepciones**

Las características de control de excepciones del lenguaje C# nos ayudan a afrontar cualquier situación inesperada o excepcional que se produce cuando se ejecuta un programa. El control de excepciones usa las palabras claves/reservadas **try**, **catch** y **finally.**

El bloque **try** nos sirven para intentar realizar acciones que pueden no completarse correctamente.

El bloque **catch** sirve para ejecutar el código "seguro" aquí van a ir las líneas de código que se ejecutaran en caso de que se produzca un error en el bloque "try".

El bloque **finally** contiene el código que deseamos ejecutar siempre al finalizar un bloque try, independientemente de si este termina correctamente o con un error. Su ejecución es garantizada y su uso es opcional.

También podemos crear excepciones propias a través del uso de la palabra clave **throw**.

## **Ejemplo utilización Try Catch**

En el siguiente ejemplo se muestra una prueba de cómo evitar que el programa arroje una excepción a causa de un error por el usuario al ingresar un número.

**try**  
{  
    Console.**WriteLine**("Ingrese un número:");  
    string input = Console.**ReadLine**();  
      
    // Intentamos convertir el texto a número  
    **int** numero = **int**.**Parse**(input);  
  
    Console.**WriteLine**("El número ingresado es: " + numero);  
}  
**catch** (Exception ex)  
{  
    Console.**WriteLine**("Error: el valor ingresado no es un número válido.");  
}  
**finally**  
{  
    Console.**WriteLine**("Ejecución finalizada.");  
}

En este caso, si el usuario escribe un texto que no sea un número, se muestra un mensaje advirtiendo al usuario del error. Al utilizar el control de excepciones evitamos que el programa se “cuelgue” y se cierre, logrando así una aplicación más estable y sin errores.

# **📝 Actividad**

1. ¿Qué es la instrucción using y para qué sirve?
2. ¿Qué es una excepción y como se pueden evitar?
3. ¿Cuáles son las palabras reservadas para usar en el control de excepciones? Detalla cada una.