Práctica A: Definición y uso de variables de tipo referencia

Objetivos

Al final de esta práctica, usted será capaz de:

Crear variables referencia y pasarlas como parámetros de métodos.

Usar el marco de trabajo del sistema.

Requisitos previos

Antes de realizar la práctica debe estar familiarizado con los siguientes temas:

Creación y uso de clases

Llamadas a métodos y paso de parámetros

Uso de tablas

Ejercicio 1
Inclusión de un método de instancia con dos parámetros

En la Práctica 7 desarrollamos una clase **BankAccount**. En este ejercicio volverá a utilizar esa clase y añadirá un nuevo método de instancia, llamado **TransferFrom**, que transfiere dinero de una cuenta especificada a la actual. Si no terminó la Práctica 7, puede conseguir una copia de la clase **BankAccount** en la carpeta Starter\Bank dentro del fichero lab08.zip.

Cómo crear el método TransferFrom

1. Abra el proyecto Bank.sln en la carpeta Starter\Bank dentro del fichero lab08.zip.
2. Edite la clase **BankAccount** como se indica a continuación:
   a. Cree un método de instancia público llamado **TransferFrom** en la clase **BankAccount**.
   b. El primer parámetro es una referencia a otro objeto **BankAccount** llamado **accFrom**, desde donde se va a transferir el dinero.
   c. El segundo parámetro es un valor **decimal** llamado *amount*, que se pasa por valor e indica la cantidad que se va a transferir.
   d. El método no tiene valor de devolución.
3. Añada al cuerpo de **TransferFrom** dos instrucciones que realicen las siguientes tareas:
   a. Deducir *amount* del saldo de **accFrom** (usando **Withdraw**).
   b. Comprobar que el dinero se ha podido retirar sin problemas. Si es así, sumar *amount* al saldo de la cuenta actual (usando **Deposit**).

   El código para la clase **BankAccount** debería ser como éste:

```
class BankAccount
{

   ... demás código omitido para mayor claridad ...

   public void TransferFrom(BankAccount accFrom, decimal
➥amount)
   {
       if (accFrom.Withdraw(amount))
           this.Deposit(amount);
   }

}
```

Cómo probar el método TransferFrom

1. Añada al proyecto el archivo Test.cs.
2. Añada a este archivo la clase siguiente, que será el sistema de prueba.

```
using System;

public class Test
{
   public static void Main()
   {

   }
}
```

3. En el método **Main**, añada código para crear dos objetos **BankAccount**, ambos con un saldo inicial de 100 $ (use el método **Populate**).
4. Añada código para mostrar el tipo, número de cuenta y saldo actual de cada cuenta.
5. Añada código para llamar a **TransferFrom** y transferir 10 € de una cuenta a otra.
6. Añada código para mostrar los saldos después de la transferencia.

   El código para la clase **Test** debería ser como éste:

```
public static void Main( )
{
    BankAccount b1 = new BankAccount( );
    b1.Populate(100);

    BankAccount b2 = new BankAccount( );
    b2.Populate(100);

    Console.WriteLine("Before transfer");
    Console.WriteLine("{0} {1} {2}",
        ➥b1.Type( ), b1.Number( ), b1.Balance( ));
    Console.WriteLine("{0} {1} {2}",
        ➥b2.Type( ), b2.Number( ), b2.Balance( ));

    b1.TransferFrom(b2, 10);

    Console.WriteLine("After transfer");
    Console.WriteLine("{0} {1} {2}",
        ➥b1.Type( ), b1.Number( ), b1.Balance( ));
    Console.WriteLine("{0} {1} {2}",
        ➥b2.Type( ), b2.Number( ), b2.Balance( ));
}
```

7. Guarde el trabajo realizado.
8. Compile el proyecto y corrija los posibles errores. Ejecute y pruebe el programa.

Ejercicio 2
Inversión de una cadena de caracteres

En el Módulo 5 desarrollamos una clase **Utils** que contenía distintos métodos.

En este ejercicio añadirá un nuevo método estático llamado **Reverse** (Invertir) a la clase **Utils**. Este método recibe una cadena de caracteres y devuelve otra nueva con los caracteres en orden inverso.

Cómo crear el método Reverse

1. Abra el proyecto Utils.sln en la carpeta Starter\Utils dentro del fichero lab08.zip.
2. Añada un método public static llamado **Reverse** a la clase **Utils**:
   a. Tiene un solo parámetros, llamado *s*, que es una referencia a una **string**.
   b. El método tiene un tipo de devolución **void**.
3. En el método **Reverse**, cree una variable **string** llamada *sInv* que contendrá la cadena devuelta como resultado. Inicialice esta variable a "".

4. Para crear una cadena de caracteres invertida:

   a. Escriba un bucle que extraiga caracteres uno a uno desde *s*. Comience por el final (use la propiedad **Length**) y retroceda hasta llegar al principio de la cadena. Puede emplear notación de tablas ([ ]) para examinar un carácter concreto en una cadena.

   Consejo El último carácter en una cadena está en la posición **Length** – 1, y el primero en la posición 0.

   b. Añada este carácter al final de *sInv*.

   La clase **Utils** podría contener lo siguiente:

```
class Utils
{

... demás métodos omitidos para mayor claridad ...

    //
    // Inversión de una cadena
    //

    public static void Reverse(ref string s)
    {

        string sRev = "";

        for (int k = s.Length – 1; k >= 0 ; k--)
                sInv = sInv + s[k];

        // Devolver resultado a llamador
        s = sInv;
   }
}
```

Cómo probar el método Invertir

1. Añada al proyecto el archivo Test.cs.
2. Añada a este archivo la clase siguiente, que será el sistema de prueba.

```
namespace Utils
{
    using System;

    public class Test
    {
        public static void Main()
        {

        }
    }
}
```

3. En el método **Main**, cree una variable **string**.
4. Utilice **Console.ReadLine** para leer un valor en la variable **string**.

5. Pase la cadena de caracteres a **Reverse**. No olvide la palabra reservada **ref**.
6. Muestre el valor devuelto por **Reverse**.

   La clase **Test** podría contener lo siguiente:

```
static void Main( )
{
    string message;

    // Obtener una cadena de entrada
    Console.WriteLine("Escriba una cadena para invertirla:");
    mensaje = Console.ReadLine( );

    // Reverse
    Utils. Reverse (ref message);

    // Mostrar el resultado
    Console.WriteLine(message);
}
```

7. Guarde el trabajo realizado.
8. Compile el proyecto y corrija los posibles errores. Ejecute y pruebe el programa.

Ejercicio 3
Copia en mayúsculas de un archivo

En este ejercicio escribirá un programa que pida al usuario el nombre de un archivo de texto. El programa comprobará que el archivo existe, y en caso contrario mostrará un mensaje y se cerrará. Abrirá el archivo y lo copiará a otro (cuyo nombre preguntará al usuario), pero con todos los caracteres en mayúsculas.

Antes de empezar, tal vez sea conveniente que revise brevemente la documentación para **System.IO** en los documentos de ayuda del SDK de .NET Framework. En particular, consulte la documentación para las clases **StreamReader** y **StreamWriter**.

Cómo crear la aplicación de copia de archivos

1. Abra el proyecto CopyFileUpper.sln en la carpeta Starter\CopyFileUpper dentro del fichero lab08.zip.
2. Edite la clase **CopyFileUpper** y añada una instrucción **using** para el espacio de nombres **System.IO**.
3. En el método **Main**, declare dos variables **string** llamadas *sFrom* y *sTo* que contendrán los nombres de los archivos de entrada y salida.
4. Declare una variable de tipo **StreamReader** llamada *srFrom*. Esta variable contendrá la referencia al archivo de entrada.
5. Declare una variable de tipo **StreamWriter** llamada *swTo*. Esta variable contendrá la referencia a la secuencia de salida.
6. Pregunte el nombre del archivo de entrada, lea el nombre y almacénelo en la variable **string** *sFrom*.
7. Pregunte el nombre del archivo de salida, lea el nombre y almacénelo en la variable **string** *sTo*.
8. Las operaciones de entrada/salida que va a utilizar pueden producir excepciones. Comience un bloque **try-catch** que pueda capturar **FileNotFoundException** (para archivos que no existen) y **Exception** (para cualquier otra excepción). Imprima un mensaje adecuado para cada excepción.
9. En el bloque **try**, cree un nuevo objeto **StreamReader** usando el nombre del archivo de entrada en *sFrom*, y almacénelo en la variable referencia **StreamReader** *srFrom*.
10. Del mismo modo, cree un nuevo objeto **StreamWriter** usando el nombre del archivo de salida en *sTo*, y almacénelo en la variable referencia **StreamWriter** *swTo*.
11. Añada un bucle **while** que se ejecute si el método **Peek** de la secuencia de entrada no devuelve -1. Dentro del bucle:
    a. Use el método **ReadLine** sobre la secuencia de entrada para leer la línea siguiente en una variable **string** llamada *sBuffer*.
    b. Aplique el método **ToUpper** a *sBuffer*.
    c. Use el método **WriteLine** para enviar *sBuffer* a la secuencia de salida.
12. Una vez finalizado el bucle, cierre las secuencias de entrada y de salida.

13. El archivo CopyFileUpper.cs debería ser como éste:

```
using System;
using System.IO;

class CopyFileUpper
{
    static void Main( )
    {
        string       sFrom, sTo;
        StreamReader srFrom;
        StreamWriter swTo;

        // Pedir el nombre del archivo de entrada
        Console.Write("Copiar de:");
        sFrom = Console.ReadLine( );

        // Pedir el nombre del archivo de salida
        Console.Write("Copiar a:");
        sTo = Console.ReadLine( );

        Console.WriteLine("Copiar de {0} a {1}", sFrom,
        ➥sTo);

        try
        {
            srFrom = new StreamReader(sFrom);
            swTo   = new StreamWriter(sTo);

            while (srFrom.Peek( )!=-1)
            {
                string sBuffer = srFrom.ReadLine( );
                sBuffer = sBuffer.ToUpper( );
                swTo.WriteLine(sBuffer);
            }
            swTo.Close( );
            srFrom.Close( );

        }
        catch (FileNotFoundException)
        {
            Console.WriteLine("Archivo de entrada no
encontrado");
        }
        catch (Exception e)
        {
            Console.WriteLine("Excepción inesperada");
            Console.WriteLine(e.ToString( ));
        }
    }
}
```

14. Guarde el trabajo realizado. Compile el proyecto y corrija los posibles errores.

Cómo probar el programa

1. Abra una ventana Command y vaya a la carpeta bin\debug dentro del proyecto CopyFileUpper.
2. Ejecute CopyFileUpper.
3. Cuando lo pida el programa, indique como nombre de archivo de origen ***drive:\path*\CopyFileUpper.cs**

   (es el archivo que acaba de crear).
4. Indique **Test.cs** como archivo de destino.
5. Cuando el programa finalice, utilice un editor de texto para examinar el archivo Test.cs, que debería contener una copia del código fuente en letras mayúsculas.