# INSTITUTO TECNOLOGICO DEL SUR DE NAYARIT


## PROGRAMACION ORIENTADA A OBJETOS

**Docente: Ing. Mata Bravo Cinthia Anahí**

**Ingeniería en Tecnologías de la Información y comunicaciones**

**Semestre 2**

**Alumno: Diego Flores Nava**

**No. Control: 191140052**

# Índice

# Archivos generados en un proyecto

- Aplicación de consola
- Aplicación Windows Form

## C# AplicacionConsola

- **App.config**: Si no se utiliza el programa utilizara las configuraciones ya establecidas, al utilizar app.config el usuario puede modificar configuraciones como el servidor o los temas.
- **C# Program.cs:** Es una opción que muestra una ventana de código, donde nos permitirá agregar instrucciones al trabajo, las cuales deben ir antes de otras instrucciones en el archivo. Es lo principal de la aplicación y donde se derivan más opciones.
- **Program**

## C# WindowsForms

**App.config**

- **Form1.cs:** Es un archivo del formulario de Windows, donde se ingresan métodos para eventos próximos
- **Form1.designer.cs:** Este archivo será utilizado por el diseñador donde inicializará partes del formulario, al momento de que un elemento sea arrastrado y soltado en la ventana del formulario, se inicializará por default en esta clase.
- **Form1:**
- **C# Program.cs**
  **Program**

**Main (): void**: El main es el punto de entrada de los programas, la palabra void hace referencia específica a que el método utilizado no devuelve un valor, también se utiliza para declarar un contexto desconocido.

-

# Aplicación de consola

Este tutorial le enseña varias características de .NET Core y el lenguaje C#. Aprenderá lo siguiente:

Los aspectos básicos de la interfaz de línea de comandos (CLI) de .NET Core

La estructura de una aplicación de consola en C#

E/S de consola

Aspectos básicos de las API de E/S de archivo en .NET

Aspectos básicos de la programación asincrónica basada en tareas en .NET

Creará una aplicación que lea un archivo de texto y refleje el contenido de ese archivo de texto en la consola. El ritmo de la salida a la consola se ajusta para que coincida con la lectura en voz alta. Para aumentar o reducir el ritmo, presione las teclas "<" (menor que) o ">" (mayor que).

Hay muchas características en este tutorial. Vamos a compilarlas una a una.

## Requisitos previos

Configure la máquina para ejecutar .NET Core. Puede encontrar las instrucciones de instalación en la página Descargas de .NET Core. Puede ejecutar esta aplicación en Windows, Linux, macOS o en un contenedor de Docker.

Instale su editor de código favorito.

## Creación de la aplicación

El primer paso es crear una nueva aplicación. Abra un símbolo del sistema y cree un nuevo directorio para la aplicación. Conviértalo en el directorio actual. Escriba el comando dotnet new console en el símbolo del sistema. Esta acción crea los archivos de inicio para una aplicación básica "Hola mundo".

Antes de comenzar a realizar modificaciones, vamos a recorrer los pasos para ejecutar la aplicación Hola mundo sencillo. Después de crear la aplicación, escriba dotnet restore en el símbolo del sistema. Este comando ejecuta el proceso de restauración de paquetes de NuGet. NuGet es un administrador de paquetes .NET. Este comando permite descargar cualquiera de las dependencias que faltan para el proyecto. Como se trata de un nuevo proyecto, ninguna de las dependencias está en su lugar, así que con la primera ejecución se descargará .NET Core Framework. Después de este paso inicial, solo deberá ejecutar dotnet restore al agregar nuevos paquetes dependientes, o actualizar las versiones de cualquiera de sus dependencias.

Después de restaurar los paquetes, ejecutará dotnet build. Esta acción ejecuta el motor de compilación y crea el ejecutable de aplicación. Por último, ejecute dotnet run para ejecutar la aplicación.

El código de la aplicación sencilla Hola a todos está todo en Program.cs. Abra ese archivo con el editor de texto de su elección. Nos disponemos a realizar nuestros primeros cambios.

La primera característica que se va a agregar es la capacidad de leer un archivo de texto y visualizar todo ese texto en la consola. En primer lugar, vamos a agregar un archivo de texto. Copie el archivo sampleQuotes.txt del repositorio de GitHub de este ejemplo en su directorio del proyecto. Servirá como script para la aplicación.

# El método WriteLine ()

Este método es el que se usa para mostrar texto en la consola, el método escribe en la pantalla el valor que le pasemos como parámetro.

El parámetro que recibe el método puede ser de varios tipos, ya sea una cadena de caracteres, un número entero, una línea en blanco, etc...

VB.NET

```
Module Module1

    Sub Main ()

      'Escribimos una cadena de caracteres.

      Console. WriteLine ("Escribiendo una línea en la consola")

      'Escribimos un numero entero

      Console. WriteLine (23)

      'Escribimos una comparación lógica

      Console. WriteLine (3 > 1)

      Console.ReadLine ()

    End Sub

End Module
```

Es importante destacar que este método añade automáticamente el salto de carro al final de la línea, esto significa que la siguiente llamada a Console. WriteLine () escribe en la siguiente línea.

La última línea en la que realizamos una llamada al método ReadLine () se utiliza para evitar que la pantalla se cierre automáticamente.

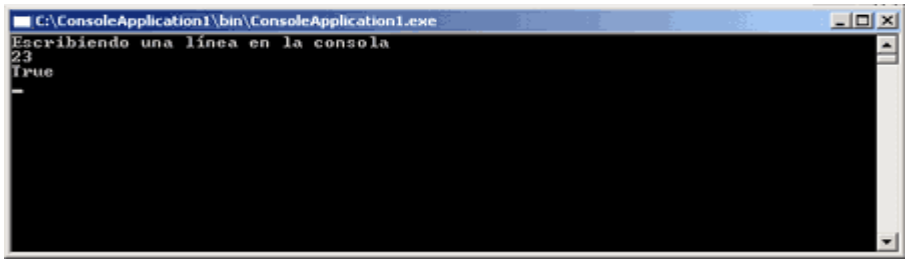



Fig1: Ejemplo del método WriteLine ()

## El método ReadLine ()

Este método se usa para recoger la información que el usuario introduce cuando la aplicación así lo requiera. Cuando invocamos al método Console.ReadLine () el sistema queda en espera hasta que el usuario pulsa la tecla Intro.

Si se asigna la llamada a Console.ReadLine () a una variable se consigue capturar el dato introducido por el usuario, para después poder operar con él.

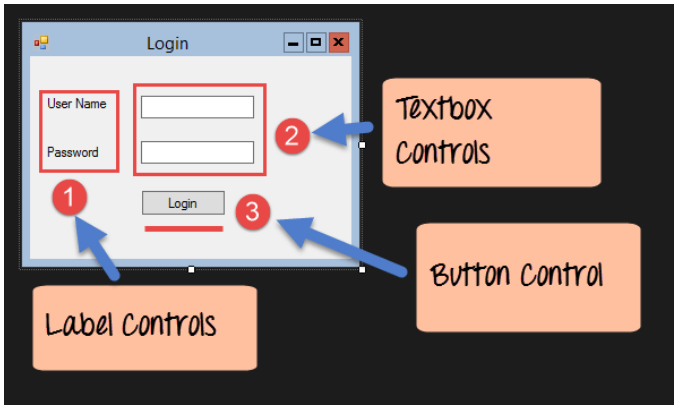
**VB.NET**

```
'Declaramos una variable de tipo cadena de caracteres
Dim cadena As String
'Mostramos un mensaje al usuario
Console.WriteLine("Por favor, introduzca su nombre:")
'Capturamos el dato introducido por el usuario
cadena = Console.ReadLine()
'Operamos con el dato
cadena = "El nombre introducido es: " & cadena
'Mostramos la cadena
Console.WriteLine(cadena)
Console.ReadLine()
```

# Windows Forms

Una aplicación de formularios de Windows es aquella que se ejecuta en la computadora de escritorio. Una aplicación de formularios de Windows normalmente tendrá una colección de controles como etiquetas, cuadros de texto, cuadros de lista, etc.

Muestra una pantalla de inicio de sesión simple, a la que puede acceder el usuario. El usuario ingresará las credenciales requeridas y luego hará clic en el botón Iniciar sesión para continuar.




1. Esta es una colección de controles de etiquetas que normalmente se utilizan para describir controles adyacentes. Entonces, en nuestro caso, tenemos 2 cuadros de texto, y las etiquetas se usan para decirle al usuario que un cuadro de texto es para ingresar el nombre de usuario y el otro para la contraseña.
2. Los 2 cuadros de texto se utilizan para guardar el nombre de usuario y la contraseña que ingresará el usuario.
3. Finalmente, tenemos el control de botón. El control de botón normalmente tendrá algún código adjunto para realizar un determinado conjunto de acciones. Entonces, por ejemplo, en el caso anterior, podríamos hacer que el botón realice una acción de validación del nombre de usuario y contraseña que ingresa el usuario.

# CONCLUCION

En fin la información recaudada nos muestra cosas básicas pero que debemos de conocer para trabajar en aplicaciones de consola y windows forma, teniendo la información necesaria podremos orientarnos y realizaremos de una manera correcta lo indicado como es el caso de crear Forms de una suma o login en alguna página, esto se crea mediante una serie de códigos y programas que eran de tras de lo que normalmente vemos, que es una pantalla, la aplicación de consola y la windows forms tienen sus diferencias y una es secuencia de la otra para complementar un trabajo, en la consola llevará todo el proceso de comandos e instrucciones para que el programa funcione, mientras que en Windows Form se verá el cómo quieres mostrar tu trabajo al exterior, es un diseño para darle más vista al trabajo y que este ordenado a la vez que bien hecho y funcional, entonces este trabajo muestra en parte algunos de los caminos a tomar para llegar a nuestro objetivo final.