Introducción al entorno: Visual Studio.NET

OBJETIVOS

- Primera toma de contacto con el entorno de desarrollo Visual Studio .NET: Opciones del entorno y tipología de proyectos.

1.- VISUAL STUDIO .NET: VISIÓN GENERAL

Este entorno de desarrollo o IDE (Integrated Development Environment) será el medio que utilizaremos para la implementación de las prácticas de la asignatura, para lo cual comenzaremos con esta primera sesión que nos guiará en la creación de uno de los diversos tipos de proyectos que podemos seleccionar mediante este IDE.

Los objetivos que se persiguen mediante esta primera auto-práctica son pocos pero muy concretos, y son:

Introducción al entorno de desarrollo de .NET
Creación de un primer proyecto utilizando el lenguaje de desarrollo C#

De esta forma, el primer paso a ejecutar en el desarrollo de esta primera auto-práctica es la inicialización del entorno de desarrollo o IDE de .NET, para lo cual procederemos según la figura siguiente.

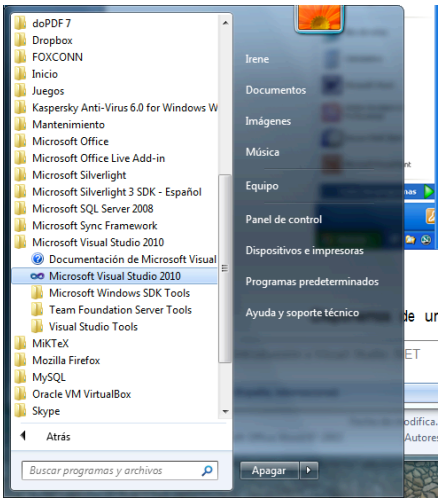


Figura 1.

Una vez tenemos arrancado el entorno de desarrollo, deberemos seleccionar el tipo de proyecto .NET que deseamos crear, aunque antes procederemos a realizar una identificación de los elementos existentes en éste con la ayuda de la siguiente figura.

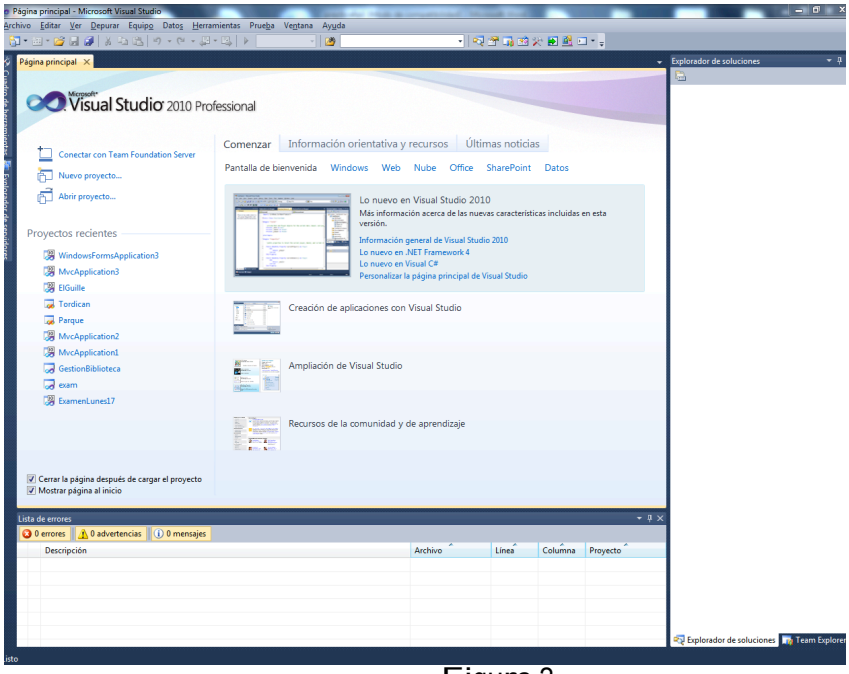


Figura 2.

Si nos fijamos en la página de inicio (Start Page) observamos que para poder comenzar con el uso de este entorno, podemos gestionar los proyectos/sitios web a través de sus dos acciones fundamentales que son la creación de nuevos proyectos/sitios web y la apertura de los ya creados previamente. Asimismo, cuando ya hemos trabajado en la creación de varios proyectos en nuestra máquina, tendremos una lista de proyectos/sitios web más recientes.[1]

2.- SOLUCIONES Y PROYECTOS:

En este punto nos centraremos en primer lugar en dar unas definiciones para que puedan quedar claros tres conceptos fundamentales que nos acompañarán a lo largo de las sesiones de prácticas de HADA.

De esta manera, y utilizando la figura siguiente que aparece al pinchar sobre el botón de Create: Project... de la figura anterior, tenemos en primera instancia varias opciones que se explican seguidamente.

- Tipos de proyecto: Mediante este entorno podemos crear una serie de proyectos de muy diferente índole, que varían desde los lenguajes de programación que podemos utilizar para su implementación (Visual Basic .NET, C#, Visual C++) hasta la naturaleza del propio proyecto, para lo cual disponemos de las
- Plantillas de proyecto y Plantillas predefinidas: según sea la naturaleza del problema que debamos resolver, podemos optar por la creación de un programa

[1] Ten en cuenta que una vez hayas utilizado el entorno durante varias sesiones de prácticas, serás capaz incluso de modificar este acceso al IDE de forma personalizada si lo deseas, pero las primeras veces que lo utilices hemos preparado esta pequeña guía para que puedas acceder a las opciones apropiadas sin demasiado esfuerzo.

Windows, o bien de una dll (Dynamic Link Library), que más tarde podemos necesitar migrar a un servicio de Windows NT o XP en función de las necesidades de nuestro proyecto. Es decir, que este IDE nos provee de diversas plantillas con una estructura de ficheros que resuelven problemas tipo que nos encontramos cuando trabajamos en el desarrollo de sw para entornos Windows como por ejemplo: Ficheros ejecutables, dll, ATL, Servicios, páginas asp, proyectos de instalación de sw, etc.

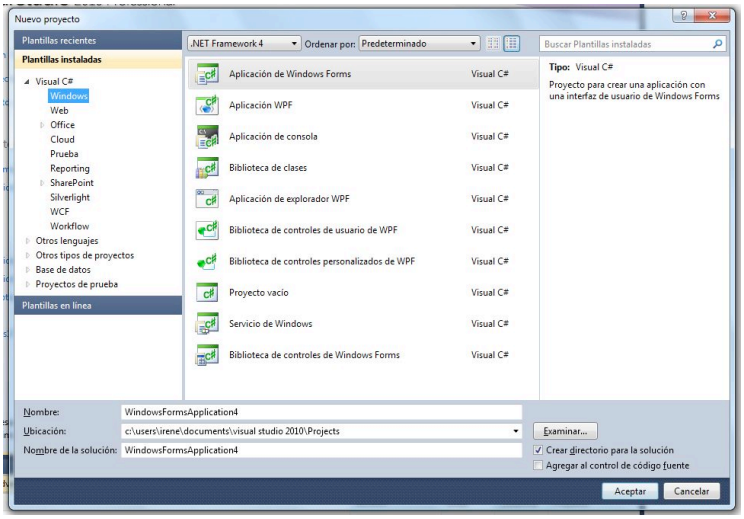


Figura 3.

Para finalizar este punto, creemos necesario escribir unas líneas más para aclarar la diferencia de concepto que existe entre las soluciones y los demás tipos de proyectos de .NET.

Como se puede apreciar en la figura 4, uno de los tipos de proyecto que se pueden desarrollar son las "Soluciones de Visual Studio", las cuales se diferencian del resto de tipos de proyecto básicamente en que:

UNA SOLUCIÓN ES UN CONTENEDOR DE PROYECTOS

Cuando el alumno tenga una experiencia más dilatada en el desarrollo de proyectos con VStudio .NET, será capaz de incluir en la misma solución diferentes tipos de proyectos que pueden estar desarrollados en diferentes lenguajes de programación.

3.- MI PRIMER PROGRAMA:

Para comenzar a desarrollar con el IDE de .NET, vamos a proceder a la creación de nuestro primer programa; y siguiendo la costumbre del gremio de la tecla, vamos a proceder a la escritura del programa "Hola mundo". Para esto, necesitamos pensar en una serie de pasos antes de lanzarnos a teclear[2].

Ingredientes del programa:

1. Una SOLUCIÓN que actuará a modo de contenedora de programa/s.
2. Un proyecto que desarrollaremos en lenguaje C# mediante la utilización de la plantilla de proyecto que se denomina "Aplicación de Consola".
3. Unos leves conocimientos de C# que nos permitan al menos insertar una línea de código en la plantilla que nos presentará .NET.

Primer Paso: Creación de la solución. Para ello, lo primero que haremos será abrir el entorno de desarrollo o IDE para, seguidamente, crear una Solución en blanco, que será la que contenga el/los proyecto/s que vayamos creando en esta fase de las prácticas. Para e lo, seleccionaremos el botón de Create: Project... tras lo cual se nos mostrará el diálogo de la siguiente figura, en el cual modificaremos el nombre de la solución, así como el path en el que se grabará ésta y que nos propone Visual Studio.

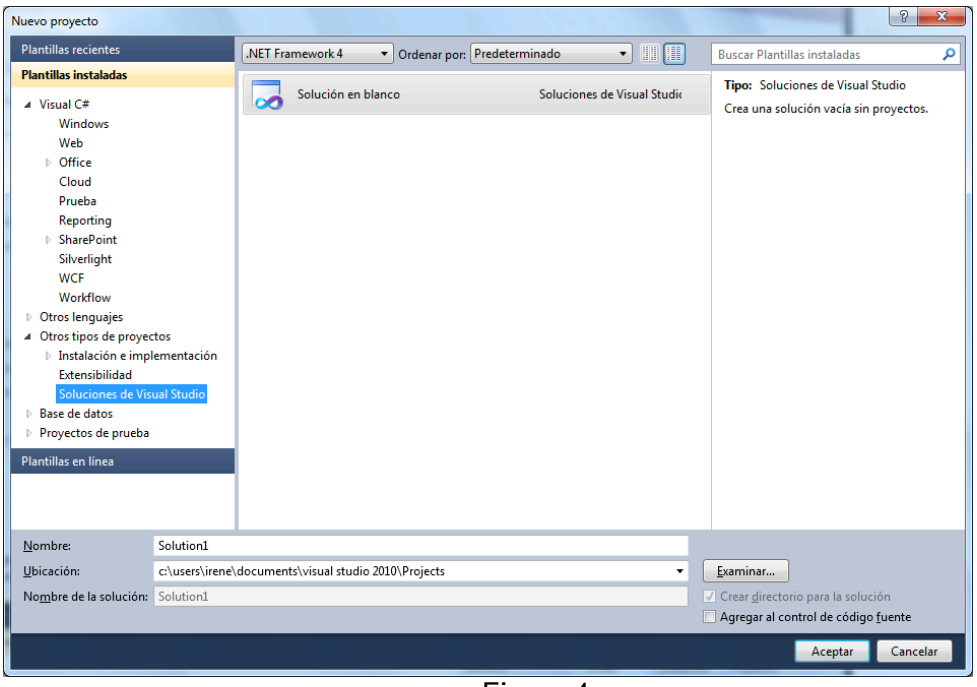


Figura 4.

Una vez tenemos creada la solución, Visual Studio nos proporcionará una serie de archivos para comenzar a desarrollar los contenidos del proyecto, lo cual nos hace

[2] Es muy recomendable que antes de abordar cualquier tipo de proyecto, se proceda a pensar en la mejor forma de estructurarlo.

ahorrar un porcentaje considerable en el tiempo que debemos dedicar a la construcción de nuestra solución. Piensa que los entornos integrados o IDE's nos deben proporcionar facilidades palpables a la hora de desarrollar soluciones a problemas de negocio o de cualquier otra naturaleza que se nos presenten a lo largo de nuestra vida profesional.
Este es uno de los motivos principales por los que se nos proporcionan una serie de pasos predefinidos siempre que creamos un proyecto con Visual Studio .NET, de forma que podemos centrarnos en el desarrollo de la solución y dejar de lado ciertos detalles que asumirá el entorno.

Hagamos un pequeño paréntesis en la creación de nuestro programa para dar un pequeño paseo por la solución que acabamos de crear. Una vez tenemos creada la solución podemos recorrer los diferentes elementos del IDE para comenzar a tener algo de background con este sistema, de forma tal que nos vayan sonando las zonas de la pantalla, o los iconos de acceso directo a diferentes partes del entorno o del proyecto en el que nos encontremos trabajando.

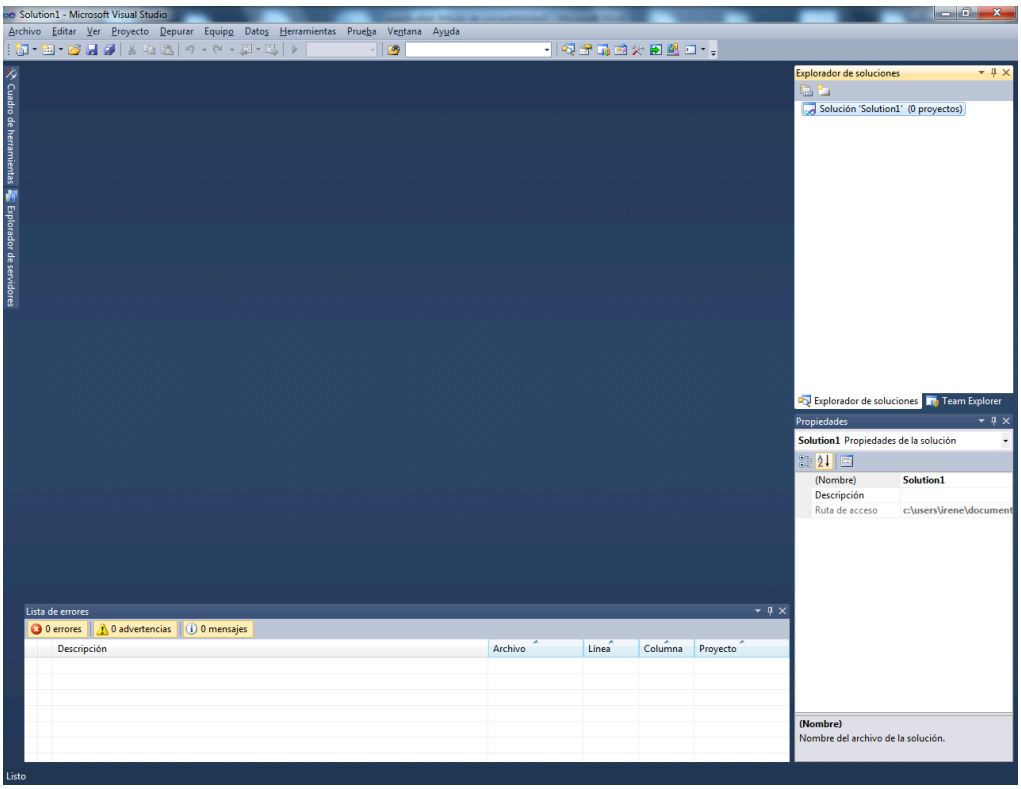


Figura 5.

Tenemos seleccionada a la derecha del entorno la solución "Solution1" que hemos creado antes y que vemos que contiene 0 proyectos. Así pues, y dado que un proyecto de tipo: solución en blanco, no sirve para mucho, debemos proceder a agregarle el segundo de los ingredientes de nuestra receta para esta autopráctica.

Para comenzar a sacar partido de este entorno, podemos proceder mediante la creación de un proyecto tipo MS-DOS, es decir, una aplicación de consola que nos va a permitir –romper el hielo" con VStudio y desarrollar algo visible en pocos momentos,

de forma que, si pinchamos con el botón derecho del ratón sobre el nombre de nuestra solución en la zona redondeada de azul de la figura anterior, podremos ver una imagen similar a la que se muestra figura siguiente.

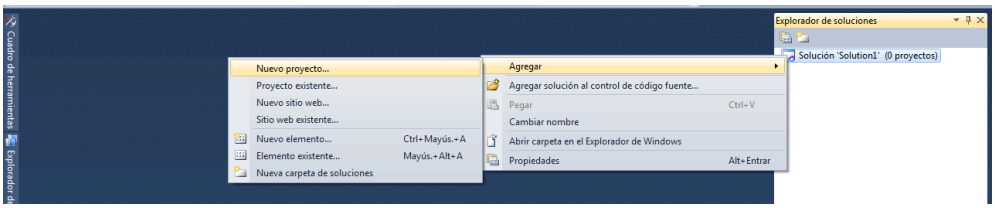


Figura 6.

Al seleccionar esta opción tendremos una ventana que se muestra seguidamente en la figura siguiente y que es la misma que cuando creamos el proyecto sobre el que estamos trabajando actualmente, es decir, la solución "Solution1". De esta forma, y dado que ya conocemos el aspecto de la ventana de creación de proyectos, podremos de forma rápida, seleccionar una de las "plantillas" para "proyectos" a desarrollar con el lenguaje "C#" que se nos ofrecen, el cual será, para esta primera sesión el denominado "Aplicación de consola". Debemos recordar cambiar el nombre y la ruta de grabación si fuera necesario a este nuevo proyecto que formará parte de nuestra primera solución en .NET y que ya consta de un proyecto.

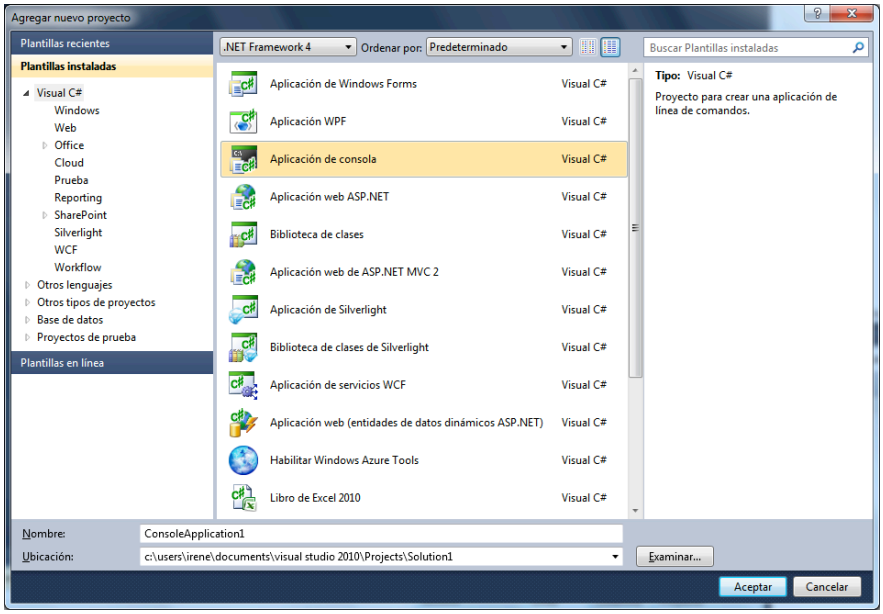


Figura 7.

Una vez creado el proyecto en lenguaje C#, de nuevo el entorno .NET nos crea una serie de ficheros en diferentes directorios de nuestro sistema, con el objetivo de adelantarnos trabajo y ahorrarnos tareas poco productiva.

Siguiendo con la creación de nuestro programa "Hola Mundo", comprobaremos que Visual Studio nos proporciona la imagen que podemos ver en la figura siguiente, en la que ya tenemos preparada toda la infraestructura para poder comenzar a desarrollar

una aplicación de Consola (o MS-DOS), utilizando el lenguaje de desarrollo C#, que será el que mayoritariamente utilizaremos a lo largo de la asignatura.

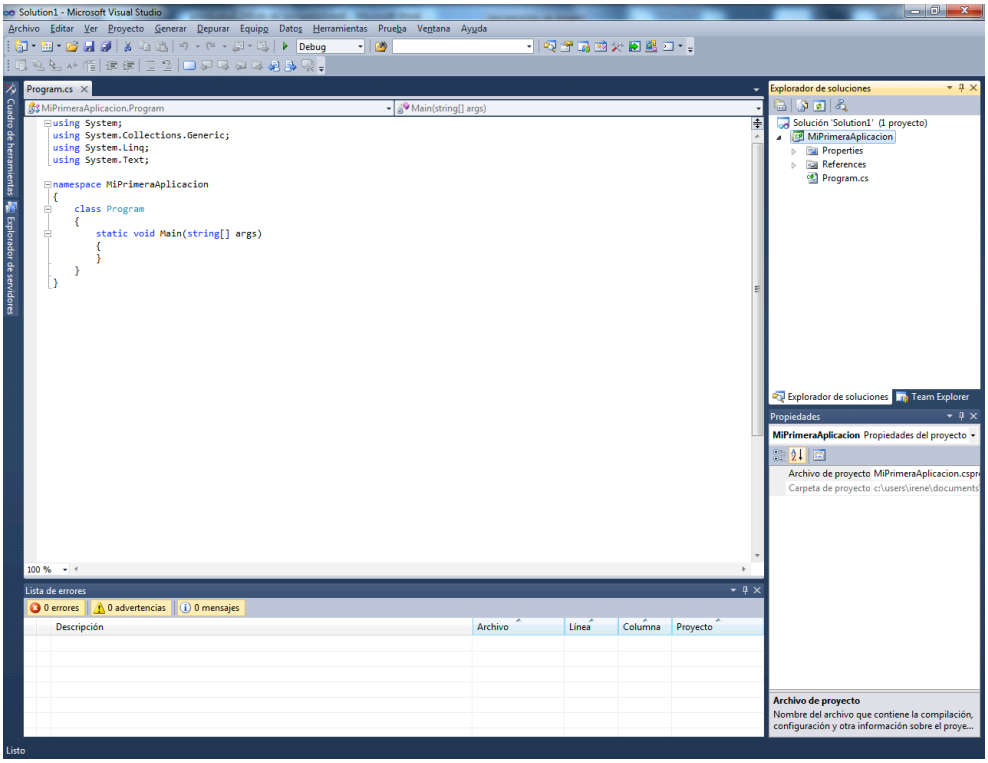


Figura 8.

El primer paso para poder comenzar a realizar el desarrollo de nuestro programa consiste en cambiar el nombre al fichero Program.cs, que es el nombre por defecto que el IDE proporciona a los objetos creados de manera automática, es decir:

[Tipo Fichero][número]*\.[extensión]

Ejemplos:

Solution1.sln → [Solution][1].[sln]
Class1.cs → [Class][1].[cs]

…y seguidamente ya podemos comenzar a escribir código fuente C# con el editor en la zona dedicada a ello y utilizando todos los conocimientos que hemos obtenido en nuestras clases de teoría así como en nuestras sesiones de autoaprendizaje mediante las bibliografías recomendadas en la asignatura.

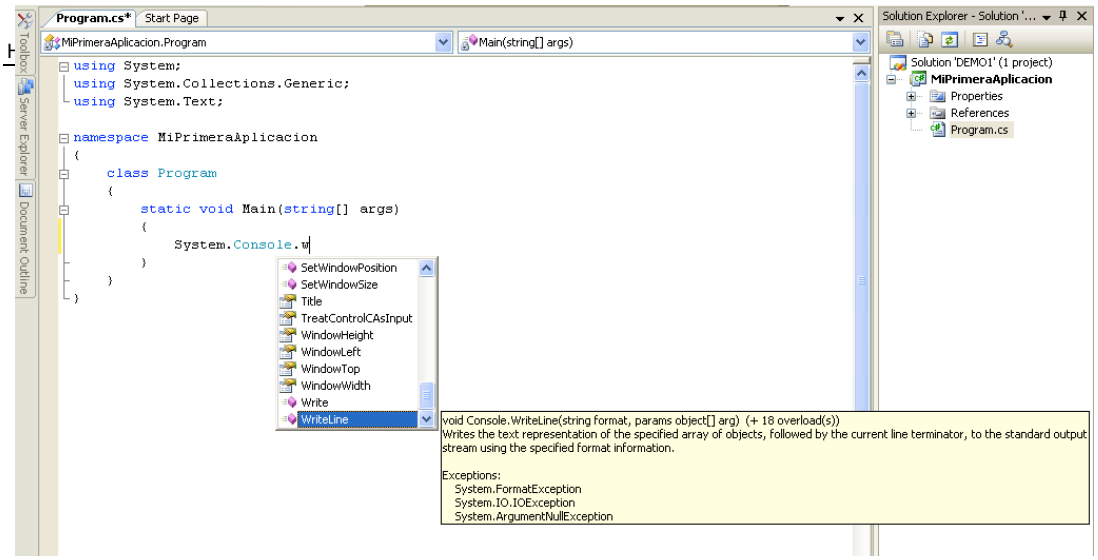


Figura 9.

Observando la imagen anterior, podemos ir comprobando cómo, ante la introducción de determinadas instrucciones de C#, el propio IDE detecta que System es una palabra especial y que además es un espacio de nombres, con lo cual los ofrece mediante un menú contextual todos los métodos y propiedades que tiene en forma de menú, a lo que se le añade una ayuda contextual mediante el típico bocadillo de ayuda Windows.

4.- MI PRIMER ERROR

4.1.- Mi primer error sintáctico

Supongamos que hemos tecleado la primera y única línea del programa para saludar al mundo mediante C#, pero en un alarde de mala coordinación entre la escritura y la pronunciación del inglés, tecleamos la sentencia de manera incorrecta como podemos observar en la figura siguiente.

Como podemos observar, nuestro error aparece convenientemente subrayado para que no se nos olvide que está ahí. Ese subrayado azul estará ahí hasta que escribamos correctamente el nombre del método que queremos emplear, y que en este caso es WriteLine.

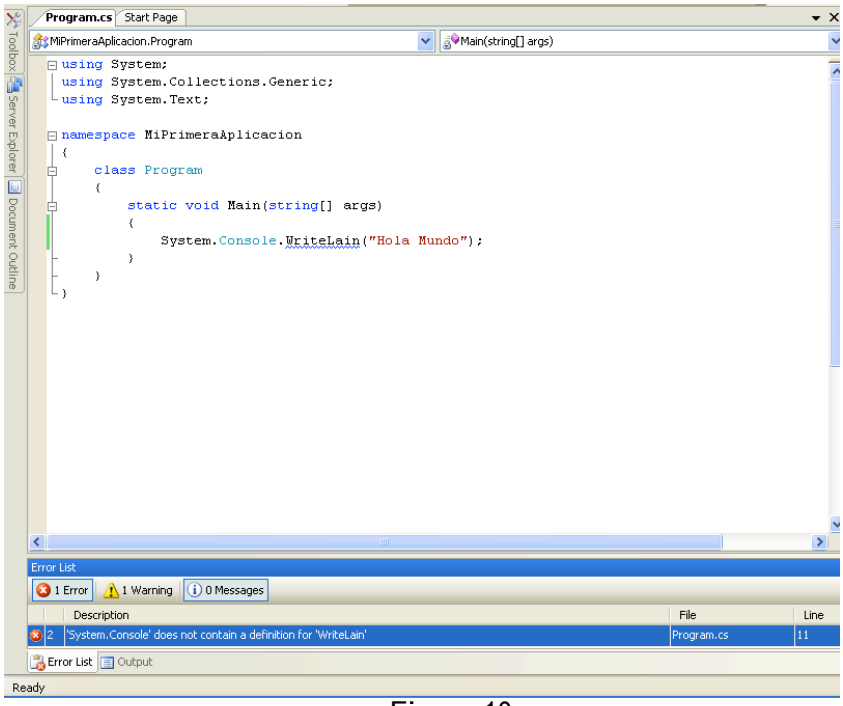


Figura 10.

4.2.- Mi primera corrección de errores:

Como podemos observar en la figura anterior, nos aparece el problema que hemos tenido en lo que se llama la "Ventana de Tareas" o "Ventana de errores". Si hacemos doble click en la línea en la que se nos muestra el error, automáticamente el cursor se situará sobre la zona del código fuente en la que está éste, quedando el IDE a la espera de que utilicemos alguna de nuestras neuronas para arreglarlo.

Bien, pues una vez corregido el problema, podremos proceder a generar la solución y disfrutar de la visualización posterior de nuestro mensaje "Hola Mundo" en la consola MS-DOS de nuestro pc. Disponemos de varias opciones para la generación/compilación de nuestros proyectos, aunque principalmente son tres las que emplearemos con mayor frecuencia, y son las que podemos observar en la figura siguiente.

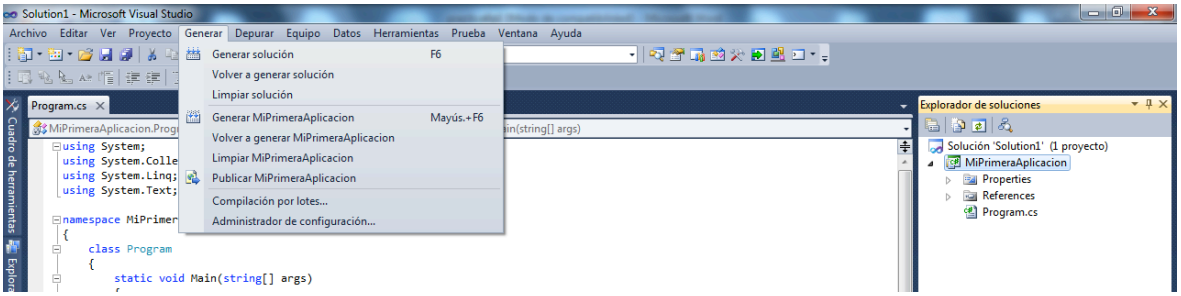


Figura 11.

Como siempre con Studio, observamos que hay varios caminos y formas de hacer lo mismo, es decir, de crear el deseado ejecutable mediante las opciones que el entorno nos proporciona para tal fin.

No obstante, y teniendo en las figuras anteriores los caminos para hacer esto, veamos algunas de las diferentes maneras de generar nuestros proyectos / soluciones con el IDE:

- Generar solución

  Esta opción realizará las tareas de compilación y linkado de los diferentes proyectos que compongan nuestra solución. Si nuestra solución únicamente está formado por una aplicación de consola, esta opción compilará el proyecto y generará un ejecutable sólo en caso de que el código fuente se encuentre libre de errores

- Compilación por lotes

  Si nuestra solución o proyecto son tan grandes que pueden llegar a comprometer los recursos de nuestra máquina (supongamos que desarrollamos un procesador de textos con C# y queremos compilarlo), tendremos la posibilidad de generarlos en batch o por lotes, es decir, cómo y cuando la máquina pueda, y a trozos

Aunque siempre nos quedará la combinación de teclas Ctrl + F5 que ejecuta el proyecto sin compilarlo si quiera.

5- EL DEPURADOR

Supongamos que, por alguna razón que desconocemos, no funciona el programa "Hola Mundo" que acabamos de desarrollar y necesitamos recurrir a algún tipo de herramienta de depuración, al estilo de los sufridos y conocidos printf que nos ayudan o ayudaban en las prácticas de otras asignaturas.

Afortunadamente, los entornos de que disponemos desde hace algunos años hasta la actualidad, no requieren que se rellenen los programas fuente de enormes cantidades de sentencias de control al estilo de:

```
cout<<"Voy por aquí y acabo de salir de allí"<<endl;
```

Como decimos, lejos de tener que emplearnos a fondo en estos menesteres, podemos establecer puntos de ruptura o "break points" en todas aquellas líneas de nuestro programa que consideremos que son sospechosas como podemos ver en la figura siguiente. Podemos establecer tantos puntos de interrupción como estimemos necesarios y el entorno detendrá la ejecución del programa en todas y cada una de las líneas que estén marcadas con un punto de interrupción.

Una vez detenida la ejecución, podremos comprobar los contenidos de las variables que nos interesen para proceder a la corrección de los errores que se estén produciendo.

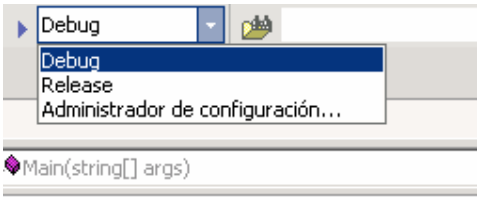
En la figura siguiente podemos comprobar un ejemplo de varios puntos de ruptura en nuestro programa.

Figura 12

- Administrador de configuración:

Esta opción es importante desde el punto de vista de la depuración de proyectos independientemente de su tamaño. Por defecto, disponemos de dos configuraciones estándar y predefinidas por el IDE, las cuales son Debug y Release. Por defecto, cuando nos encontramos desarrollando un proyecto nos encontramos con la configuración de Debug seleccionada por defecto y solemos decir que –estamos en debug o en modo depuración“. Esta configuración nos permite establecer puntos de ruptura (tan deseados cuando trabajamos con otros lenguajes o entornos) y seguir la ejecución de nuestro proyecto paso a paso, instrucción por instrucción comprobando dónde tenemos el error para poder solventarlo rápidamente. Esta configuración de depuración posee la característica de que utiliza librerías de Windows que no son las de producción, sino las de depuración, por lo cual, una vez finalizada la fase de desarrollo y depuración de errores en nuestro proyecto, deberemos tomarnos la molestia de cambiar de configuración y pasar a modo “Release“, para generar la versión definitiva de nuestro proyecto con las librerías de producción. Es decir, una vez que nuestro proyecto esté libre de errores y nos encontremos en situación de generar un ejecutable o una dll o cualquiera que sea la forma del “deliverable“ que estemos construyendo, deberíamos cambiar la configuración de nuestro proyecto y cambiar de modo Debug a modo Release, tal como podemos observar en la figura siguiente.

Cuando pasemos al modo Release con la intención de generar el ejecutable definitivo de nuestro programa no debemos dejar ningún punto de ruptura en la aplicación.



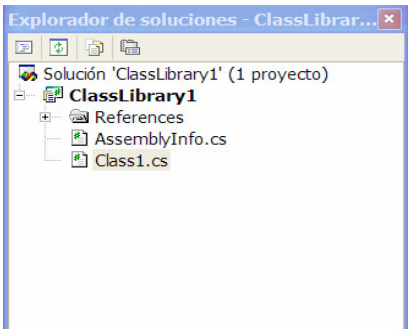
6.- MI PRIMERA APLICACIÓN

Procedemos seguidamente a crear una aplicación de manera más detallada mediante ejercicios paso a paso.

6.1.- Ejercicio de Desarrollo de una aplicación con VisualStudio .NET

6.1.1.- Generación de una clase con C#

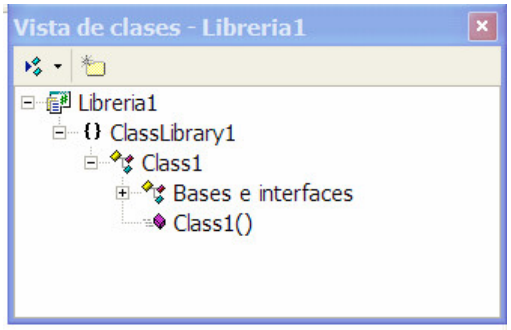
Paso 1) Desde el explorador de soluciones, selecciona con el botón derecho la solución y elige Agregar/Agregar Proyecto. (o desde Archivo Agregar N uevo Proyecto) Desde esta ventana selecciona Biblioteca de Clases. Lenguaje C#. Observarás como se añade un nuevo proyecto a la solución ya definida.



La idea de proyecto corresponde con la de conjunto de programas escritos con un único lenguaje de programación. La idea de solución corresponde con el concepto de ejecutable. Una solución puede estar formada por uno o más proyectos, del cuál solo uno de ellos es el proyecto inicial. La gestión de las soluciones se puede realizar a través del explorador de soluciones.

Paso 2) Accede con el botón derecho del ratón a la librería de clases (ClassLibrary1) dentro del explorador de soluciones, selecciona Cambiar Nombre e introduce Libreria1. De la misma forma cambia el nombre de la clase Class1 por Minombre.

Paso 3) Activa la ventana de la vista de clases, accediendo a la barra de menú Ver/Vista de Clases.



La vista de clases permite visualizar y modificar las propiedades de las clases definidas en la aplicación. Selecciona la clases y desde la ventana de propiedades cambia su nombre por el de Minombre. Observa como cambia automáticamente el código fuente asociado a la clase.

Paso 4) Añade el siguiente método a la clase:

```
// Este método devuelve mi nombre
public string MostrarMiNombre(){
    return Juan;
}
```

Paso 5) Para compilar el proyecto selecciona la opción Generar/Generar Solución del menú principal, con e lo se muestra la ventana de resultados en la que se indican los posibles errores detectados en el código.

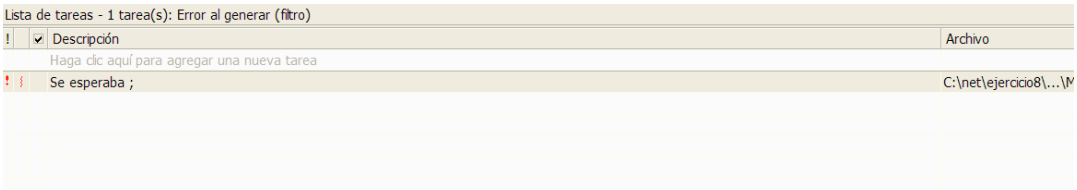
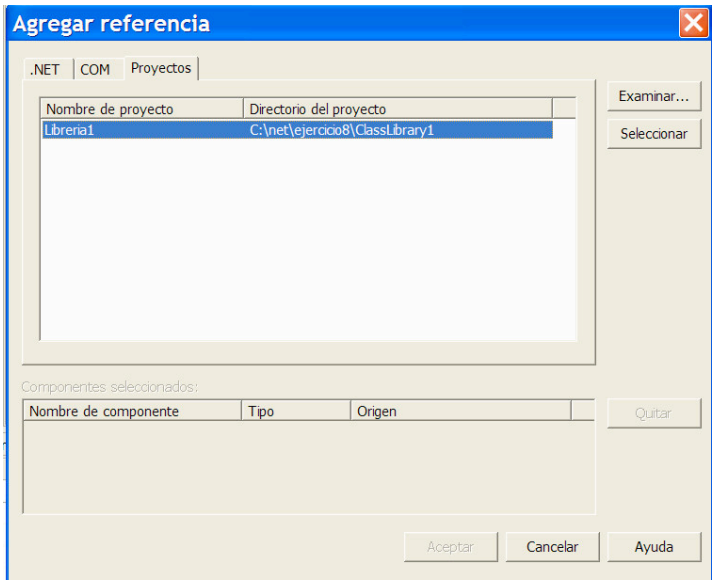


Figura 18.

En este caso se muestra un error, ya que hemos olvidado deliberadamente las comillas que cierran la cadena utilizada. Si pinchas con el ratón dos veces el cursor se posiciona en la parte del código donde se presume que se ha la el error. Añada las comillas – al inicio y final de Juan y vuelve a generar. En este caso la compilación no dará errores.

6.1.2.- Generación de una aplicación de consola

Paso 1) Igual que en el paso anterior, agrega un nuevo proyecto. Desde esta ventana selecciona Proyecto de C# con interfaz de consola. Observarás como se añade un nuevo proyecto a la solución ya definida.



Paso 2) Para poder utilizar la librería de clases definida en la parte anterior, desde el explorador de soluciones selecciona el nuevo proyecto creado con el botón derecho y elige Agregar Referencia. Desde la opción de Proyectos selecciona Librería1.

Figura 20.

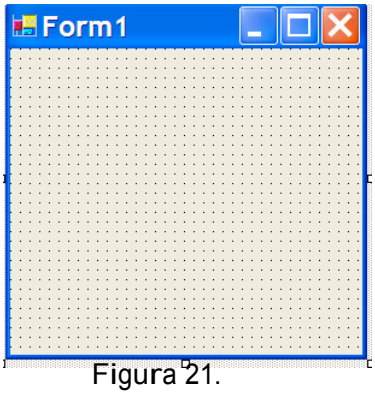
Ahora en el código escribe lo siguiente:

```
static void Main(string[] args){
      ClassLibrary1.Minombre cadena = new ClassLibrary1.Minombre();
      Console.WriteLine(cadena.MostrarMiNombre());
}
```

Para ejecutar la aplicaciónpulsa F5. Dará un error ya que por defecto la solución se asocia al primer proyecto que se ha creado, que en este caso era una biblioteca de clases y estas no son ejecutables. Para definir la interfaz de consola como proyecto inicial selecciona dentro del explorador de soluciones con el botón derecho del ratón y elige Establecer como proyecto de inicio. Al ejecutar ahora la solución se muestra por consola: Juan

6.1.3.- Generación de una aplicación de Windows con Visual Basic .NET

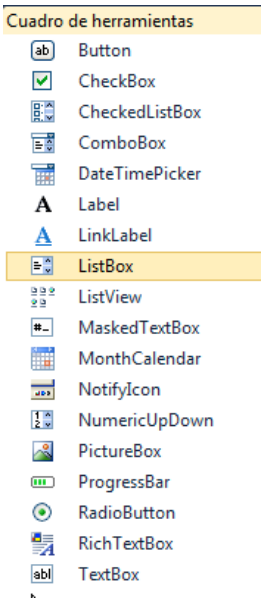
Paso 1) De la misma forma que en la parte anterior añade un nuevo proyecto (en este caso de Visual Basic y de tipo Aplicación para Windows. Aparecen nuevos elementos de interés.



El formulario es la parte central de las aplicaciones Windows. Permite contener controles como botones, cajas de texto, menús etc.

Figura 21.

El cuadro de herramientas contiene los controles que podemos asociar a un formulario. La forma de colocarlos es simplemente seleccionando uno y arrastrándolo dentro del formulario.

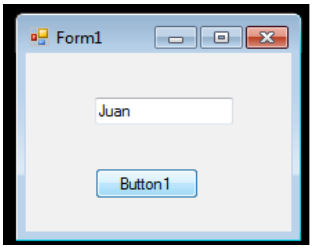


Paso 2) Selecciona y arrastra un botón (button) y una caja de texto (textBox1) dentro del formulario.

Paso 3) Agrega una referencia de la librería1 dentro de este proyecto, tal como hiciste en la parte anterior.

Paso 3) Selecciona el botón con el botón derecho y elige Propiedades. Dentro de propiedades pincha sobre el icono del rayo  y selecciona el evento Click y escribe el siguiente código.

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim nombre As New ClassLibrary1.Minombre
        TextBox1.Text = nombre.MostrarMiNombre()
    End Sub
```



Define este proyecto como proyecto de inicio de la solución. Tras ello ejecuta la solución y pulsa el botón del formulario.

6.1.4.- Generación de una aplicación Web con ASP .NET

1) Agregamos un nuevo proyecto del tipo C# y Aplicación Web ASP .NET VACIA
2) Establecemos el nuevo proyecto creado como proyecto de inicio de la aplicación.
3) Añadimos referencia a la librería de clases.
4) En el explorador de soluciones, en el proyecto web con el botón derecho seleccionamos Agregar nuevo elemento --> Web Forms, con lo que añade una nueva página web al proyecto (WebForm1.aspx)
5) En esa página, añadimos un botón (desde el cuadro de herramientas) y le cambiamos su nombre (utilizando la ventana propiedades) y le asignamos ¿Quieres saber mi nombre?.
6) Añadimos el siguiente código al evento por defecto del botón (Pulsar dos veces sobre el mismo).

```
private void Button1_Click(object sender, System.EventArgs e){
   ClassLibrary1.Minombre nombre = new ClassLibrary1.Minombre();
   Response.Write(nombre.MostrarMiNombre());
}
```

7) Ejecutamos (pulsando F5)

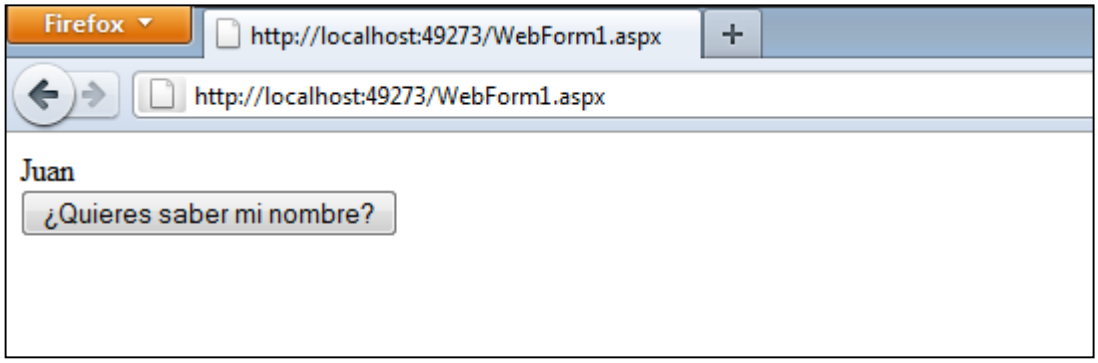


Figura 23.

Como podemos observar en la figura anterior, la ejecución se realiza sobre el navegador web de nuestra máquina.