

## Clase 2 - No Presencial. Por Guillermo Guastavino

### Ambiente de Desarrollo, repaso de C para C#, bases de VB .NET a partir de C, y primer proyecto

Saludos a todos!. Bienvenidos a la Clase 2

En la clase anterior, vimos que el Framework es un gran conjunto de clases, que se va abriendo en ramas, para "en la copa", tener a los lenguajes del VS .NET. Entonces, no hay lenguajes "más rápidos" que otros, ni lenguajes para "pavadas" y otros para "cosas importantes"; porque todos se construyen con las mismas clases primitivas y están al mismo nivel. El lenguaje es "un pretexto", sólo una forma de acceder al Framework y cuanto más se avanza, más se reduce, hasta casi ser una mínima parte, como dije "un pretexto" el lenguaje elegido contra todo lo que se hace en Framework

Como el framework son clases el código es casi completamente igual para llamarlas, estés en C# o en VB. Cuando veamos ADO (clases para acceder a bases de datos), casi todo lo que van a escribir es ADO NET, y el código es 99% igual si lo hacés en cualquiera de los lenguajes del Framework

La elección de un lenguaje para programar en .NET, es completamente a gusto del programador, o bien a exigencia del grupo o lugar en dónde se trabaja.

De hecho, como todos "salen de la misma bolsa", podés hacer muchas cosas en VB que es mucho más fácil y rápido para escribir, y otras en C# o J#, más estructurados, pero más "duros" e "inflexibles"; compilados en clases que se llaman entre si. De hecho es un entorno multi lenguaje. Cuando lleguemos al 2do cuatrimestre, ASP .NET, vamos a estar programando al mismo tiempo con HTML (lo que ve el cliente - usuario -), ASP .NET el lenguaje cliente servidor para la interface con el cliente, que luego genera el HTML, JAVA script, y el *code behind* que es el código que se ejecuta en el servidor, y que es Framework .NET principalmente, sobre el lenguaje que quieras.

En los avisos de pedidos de programadores para .NET, van a notar que piden VB y C#, dependiendo de la "historia" de la empresa. Si es una empresa nueva, tal vez pidan C# o indistinto VB. Si es una empresa que fue migrando sus programas, es muy probable que soliciten VB.

VB es "más flexible", permite muchas maneras de hacer la misma cosa y no es muy riguroso con la forma que elijas, lo cual agiliza la programación, aunque permite que cada programador tenga su propio estilo, que tal vez no sea el de las "buenas prácticas".

Las instrucciones son más coloquiales y menos abstractas que C#.

C# es muy estructurado y cero flexible. O sea, hay una sola forma de hacer las cosas, lo que obliga a programar correctamente y de la forma que corresponde.

Ustedes ya saben C, y C# es igual a lo que conocen. Los años anteriores agregué enseñar también VB que facilita mucho la programación, especialmente cuando lo que estamos usando es Framework y es lo que importa. Lo que vamos a ver es como es VB partiendo de lo que ya conocen de C. Entonces van a saber 2 lenguajes de uso comercial "por el precio de uno".

La idea ir viendo a los dos al mismo tiempo, que en clase se use VB, que agiliza las cosas y sirve para unificar las explicaciones, al utilizar un mismo lenguaje que le sea simple al que sabe un poquito y al que sabe mucho, sin tener que detenernos en cuestiones de lenguaje, y enfocarnos en las cosas nuevas. En paralelo, la idea es hacer lo mismo que hicimos en clase, en C#, así les van quedando ejemplos y trabajos en ambos lenguajes y pueden acceder a más ofertas laborales.

Repasemos algunas cosas básicas de C que ya conocen y veamos como son en VB. Vamos a ir viendo estas cosas durante las primeras clases, mientras que enfocamos hacia el Entorno de Desarrollo y el Framework, aprendiendo lo que necesitamos aplicar.

## RESUMEN DEL SISTEMA DE TIPOS C#

Tipo CTS	Alias C#	Descripción	Valores que acepta
System.Object	object	Clase base de todos los tipos del CTS	Cualquier objeto
System.String	string	Cadenas de caracteres	Cualquier cadena
System.SByte	sbyte	Byte con signo	Desde -128 hasta 127
System.Byte	byte	Byte sin signo	Desde 0 hasta 255
System.Int16	short	Enteros de 2 bytes con signo	Desde -32.768 hasta 32.767
System.UInt16	ushort	Enteros de 2 bytes sin signo	Desde 0 hasta 65.535
System.Int32	int	Enteros de 4 bytes con signo	Desde -2.147.483.648 hasta 2.147.483.647
System.UInt32	uint	Enteros de 4 bytes sin signo	Desde 0 hasta 4.294.967.295
System.Int64	long	Enteros de 8 bytes con signo	Desde -9.223.372.036.854.775.808 hasta 9.223.372.036.854.775.807
System.UInt64	ulong	Enteros de 8 bytes sin signo	Desde 0 hasta 18.446.744.073.709.551.615
System.Char	char	Caracteres Unicode de 2 bytes	Desde 0 hasta 65.535
System.Single	float	Valor de coma flotante de 4 bytes	Desde 1,5E-45 hasta 3,4E+38
System.Double	double	Valor de coma flotante de 8 bytes	Desde 5E-324 hasta 1,7E+308
System.Boolean	bool	Verdadero/falso	true ó false
System.Decimal	decimal	Valor de coma flotante de 16 bytes (tiene 28-29 dígitos de precisión)	Desde 1E-28 hasta 7,9E+28

Seguramente ya conocen y usaron estos tipos para variables en C. La primera columna es el namespace (Clase 1) de la clase sobre la cual se crea el tipo, por ejemplo el INT hereda de System.Int32, lo mismo pasa con INTEGER de VB, que hereda de la misma clase..., lo mismo que todos los otros tipos

Normalmente vamos a usar:

<b>C#:</b>	<b>VB</b>
object	object
string	string
int	integer
long	long
bool	boolean
decimal	decimal

### **Declaración de Variables C# (igual que C):**

int num                'declara la variable num que es de tipo int y no la inicializa, queda con el valor por defecto 0 para las variables numéricas

int num=10;           'declara la variable num que es de tipo int y al mismo tiempo le asigna el valor 10

pero como es lo mismo usar int, que la clase que lo genera, podríamos haber escrito:

```
System.Int32 num=10;
```

otras declaraciones:

```
double num=10.75;
```

```
decimal num=10.7508;
```

```
System.DateTime fecha;
```

### **Declaración de Variables en VB**

dim num as integer                'es lo mismo que int num  
dim num as integer=10            'es lo mismo que int num=10

dim a, b, c, d as integer        'a,b,c,d son variables integer

dim a, b, c, d as integer=2      'es error.... no puedo asignar valor a un grupo

dim a as integer=2, b as integer=2, c as integer=2, d as integer=2 'asi si...

dim a as integer=2, b as integer=a+1, c as integer=a\*b

Esto es una declaración de variables que al mismo tiempo asignan valores en cadena, se garantiza la ejecución de izquierda a derecha: a vale 2, b vale 3, c vale 6

## Concatenación de variables

Como saben de C, la concatenación, que es unir 2 cadenas, en C se hace con el caracter +, que en vez de sumar concatena

**en VB:**

```
dim cosa as string="casa"
```

```
cosa=cosa + "21"
```

como "21" es un literal, una cadena de caracteres, o sea el caracter 2 pegado al caracter 1, no el número 21..., no tengo problemas de concatenarlo con "21", ya que concatena cadena con cadena. Concatena "casa" y "21", quedando cosa="casa21"

pasaría lo mismo con C

El tema es si quiero ejecutar:

```
cosa=cosa+21
```

cosa es un string, pero 21 es un número. No puedo concatenar cosas de diferentes tipos. En C daría error, también en VB, pero en VB existe un operador específico para concatenar que es el & (no confundir con AND).

En VB: cosa=cosa & 21 'es correcto

En VB hace una "conversión implícita de tipo de dato", o sea yo no le dije que convierta 21 en string para poder concatenarlo, pero VB lo hace automáticamente para resolver la situación, gracias al operador &.

Podría haber hecho una "conversión explícita de tipo de dato" que es decirle que primero convierta 21 a string antes de concatenarlo. En éste caso funciona tanto con + como con &

```
cosa=cosa + cstr(21)
```

"c" es de "convertir", siendo lo que sigue "str" a qué lo voy a convertir. CDEC a decimal, CSTR a string como vimos, CINT a entero etc.

En C# es más rígido el tema:

```
int a = (int) cadena; // Error. Una cadena no se puede convertir a número
```

Para este caso necesitaríamos hacer uso de los métodos de conversión que proporcionan cada una de las clases del .NET Framework para los distintos tipos de datos:

```
int a = System.Int32.Parse(cadena); // Así sí
```

## Incrementar variables

En C, para incrementar en 1 a una variable uso ++  
si a es entero y vale 2

a++

es un post incremento, le suma 1 a la variable a, podría haber sido ++a que es un pre incremento, y en este caso es lo mismo, salvo que ahora agregue la variable b que es entero

b=a++

primero la variable a, le pasa el valor 2 a b, y luego se incrementa, quedando a con 3 y b con 2

b=++a

primero se incrementa la variable a al valor 3, luego le pasa el 3 a b, quedando a y b con 3

En VB no existe el ++ (nunca supe porque no lo agregaron... qué les costaba?...), en cambio existe algo bastante más loco:

a+=1 es lo mismo que decir a=a+1

a+=5 es lo mismo que decir a=a+5

también funciona para -, \*, / y &

a\*=2

a/=2

a-=2

y en el caso que teníamos de cosa=cosa + "21":

cosa &="21"

Con esto tendremos suficiente para realizar nuestro primer proyecto: Un Navegador de Internet con la Potencia de GOOGLE....(!!!!?????)

## Pero primero debemos entender el ambiente de desarrollo del VS.

Retomen el final de la clase 1, dónde quedamos al borde nomás de crear un proyecto. Esta vez continúen...

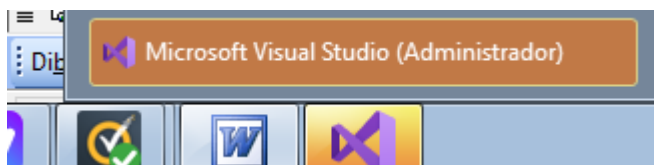
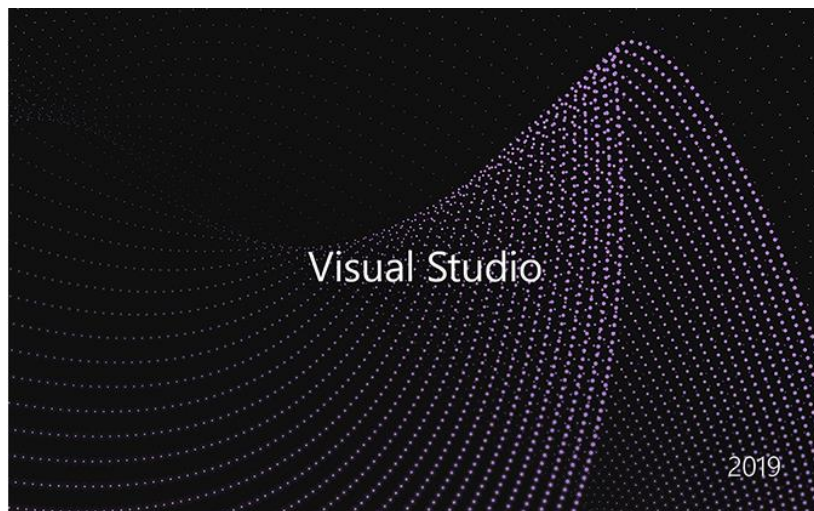
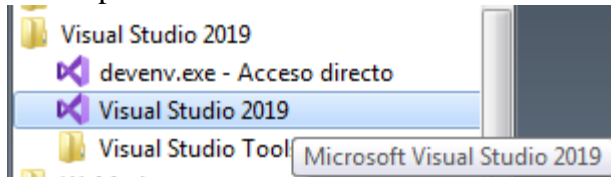
Vamos a trabajar C#, y **ustedes hacer el mismo proyecto con VB** (recuerden usar & para concatenar). Luego iremos pasándonos a VB a medida que profundicemos en Framework

Para evitar posibles diferencias por usar otras versiones de VS, verificar primero que la versión del Visual Studio que tengas, sea la misma que les pasé en la Clase 1 (versión original, gratis, bajada desde Microsoft). Esa versión posee todas las plantillas (aunque siempre trato de llevarlos a usar las menos plantillas posibles, para evitar que se agreguen un montón de cosas que no van a usar ahora, y que les complique “la visión” del ambiente de desarrollo; y aparte esas cosas también se compilan aunque no las usen. Entonces trato que siempre sea lo más “limpio” posible.

Por lo que leíste antes; como lo que vale es Framework, y vos ya viste C en otro curso, te enseño VB que es más fácil, y nos permite abocarnos de lleno en Framework. Entonces en las primeras clases vemos C# y vas aprendiendo VB, y luego seguimos con VB en las clases para que sea más simple y apuntar a lo importante. Por tu lado, sería recomendable que fueras haciendo lo mismo en C#, para practicar ambos, pero para las entregas, con que entregues en VB es suficiente.

Les agrego un paso a paso:


Paso a paso desde abrir el VS 2019



# Visual Studio 2019


## Abrir recientes

Anterior

 Sistema.sln


16/12/2020 02:12 p.m.

D:\\_UNLZ\Clases UNLZ\PROG IV 2020\Sistema

 TrabajoFinalWeb.sln


13/12/2020 09:19 a.m.

D:\...\La-Lucha-Carrito-de-Compras-master\LaLucha\_CodigoFuente

 Sistema.sln


28/11/2020 09:28 a.m.

D:\Descargas\Ezequiel Javier Huguetti\Sistema\Sistema

 Sistema.sln


27/11/2020 06:03 p.m.

D:\Descargas\Lisandro OREZZOLI - CABA\TP Final Sistema\Sistema

 Sistema.sln


27/11/2020 09:31 a.m.

D:\...\Trabajos Alumnos 2020\AbalosDiego - campus\Nueva carpeta (9)

 Sistema.sln


26/11/2020 01:02 p.m.

D:\Descargas\Gabriel Navia - campus\Prog IV actualizar cada clase\Sistema


 Sistema.sln

26/11/2020 12:20 p.m.


## Tareas iniciales

 Clonar o extraer código del repositorio


Obtiene código desde un repositorio en línea, como GitHub o Azure DevOps.

 Abrir un proyecto o una solución

Abre un archivo .sln o proyecto de Visual Studio local.

 Abrir una carpeta local

Navegar y editar el código en cualquier carpeta

 Crear un proyecto

Elige una plantilla de proyecto con la técnica scaffolding de código para comenzar.

[Continuar sin código →](#)




## Crear un proyecto


Elige una plantilla de proyecto con la técnica scaffolding de código para comenzar.

# Crear un proyecto


## Plantillas de proyecto recientes

 Aplicación web ASP.NET (.NET Framework)


Visual Basic

 Aplicación de Windows Forms (.NET Framework)

Visual Basic

 Aplicación de Windows Forms (.NET Framework)

C#

 Aplicación de consola (.NET Core)

C#

Visual Basic

Todos los lenguajes

C#

C++

F#

Java

JavaScript

Lenguaje de consulta

Python

TypeScript

Visual Basic

Windows

Aplicación web ASP.NET (.NET Core)

Una aplicación de línea de comandos para Windows, Linux y MacOS.

Windows Linux macOS Con

Aplicación web ASP.NET (.NET Framework)

para crear aplicaciones ASP.NET, Pu

MVC o Web API y agregar muchas o

Esa lista desplegable, te permite elegir el lenguaje. Si elegís C# te muestra las plantillas de C#:

# Crear un proyecto

Plantillas de proyecto recientes

Aplicación web ASP.NET (.NET Framework)

Visual Basic

Aplicación de Windows Forms (.NET Framework)

Visual Basic

Aplicación de Windows Forms (.NET Framework)

C#

Aplicación de consola (.NET Core)

C#

Buscar plantillas (Alt+S)

Borrar todo

C#

Windows

Todos los tipos de proy...

Aplicación de consola (.NET Core)

Proyecto para crear una aplicación de línea de comandos que se puede ejecutar en .NET Core en Windows, Linux y MacOS.

C#LinuxmacOSWindowsConsola

Aplicación web ASP.NET Core

Plantillas de proyecto para crear aplicaciones web de ASP.NET Core y API web para Windows, Linux y macOS con .NET Core o .NET Framework. Cree aplicaciones web con Razor Pages, MVC o aplicaciones de una sola página (SPA) con Angular, React o React + Redux.

C#LinuxmacOSWindowsNubeServicioWeb

Aplicación Blazor

Plantillas de proyecto para crear aplicaciones Blazor que se ejecutan en el servidor en una aplicación ASP.NET Core o en el explorador en WebAssembly. Estas plantillas se pueden usar para crear aplicaciones web con interfaces de usuario (UI) dinámicas completas.

C#LinuxmacOSWindowsNubeWeb

Biblioteca de clases (.NET Standard)

Proyecto para crear una biblioteca de clases para .NET Standard.

C#AndroidiOSLinuxmacOSWindowsBiblioteca

Si elegís VB, que también vamos a usar:

# Crear un proyecto

Plantillas de proyecto recientes

Aplicación web ASP.NET (.NET Framework)

Visual Basic

Aplicación de Windows Forms (.NET Framework)

Visual Basic

Aplicación de Windows Forms (.NET Framework)

C#

Aplicación de consola (.NET Core)

C#

Borrar todo

Visual Basic

Windows

Todos los tipos de proy...

Aplicación de consola (.NET Core)

Proyecto para crear una aplicación de línea de comandos que se puede ejecutar en .NET Core en Windows, Linux y MacOS.

Visual BasicWindowsLinuxmacOSConsola

Aplicación web ASP.NET (.NET Framework)

Plantillas de proyecto para crear aplicaciones ASP.NET. Puede crear aplicaciones ASP.NET Web Forms, MVC o Web API y agregar muchas otras características en ASP.NET.

Visual BasicWindowsNubeWeb

Biblioteca de clases (.NET Standard)

Proyecto para crear una biblioteca de clases para .NET Standard.

Visual BasicAndroidiOSLinuxmacOSWindowsBiblioteca

Proyecto de prueba de MSTest (.NET Core)

Proyecto que contiene pruebas unitarias de MSTest que se pueden ejecutar en .NET Core en Windows, Linux y MacOS.

Visual BasicWindowsLinuxmacOSPrueba

Proyecto de prueba NUnit (.NET Core)

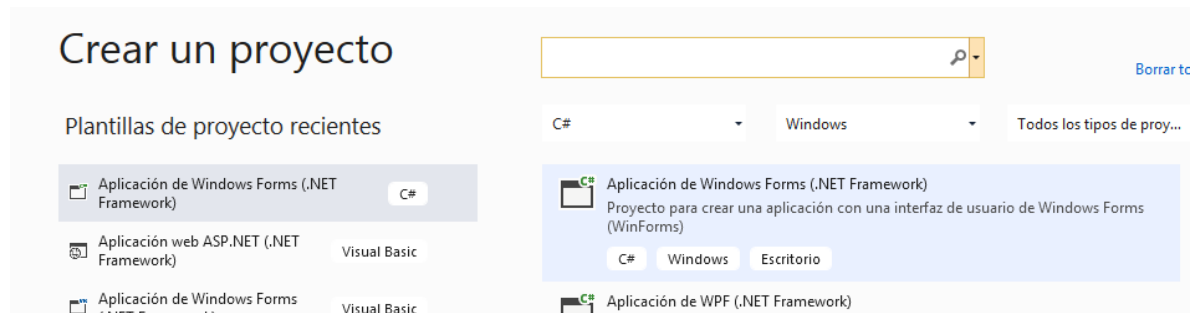
Atrás

Siguiente



Volvemos a C# para empezar.  
Desplazar la lista de la derecha para ir viendo las opciones...

Bajá en la lista hasta tener:



hacé click en Aplicación de Windows Forms (.NET Framework) y apretá

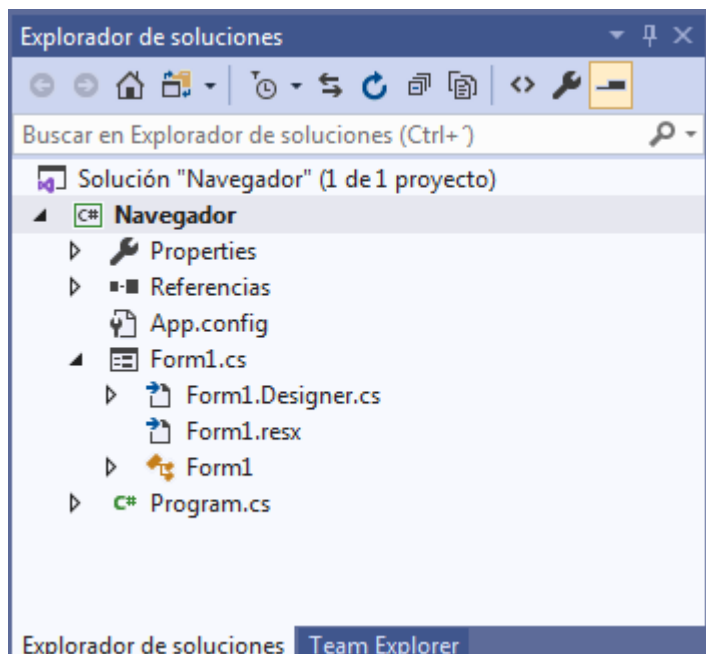
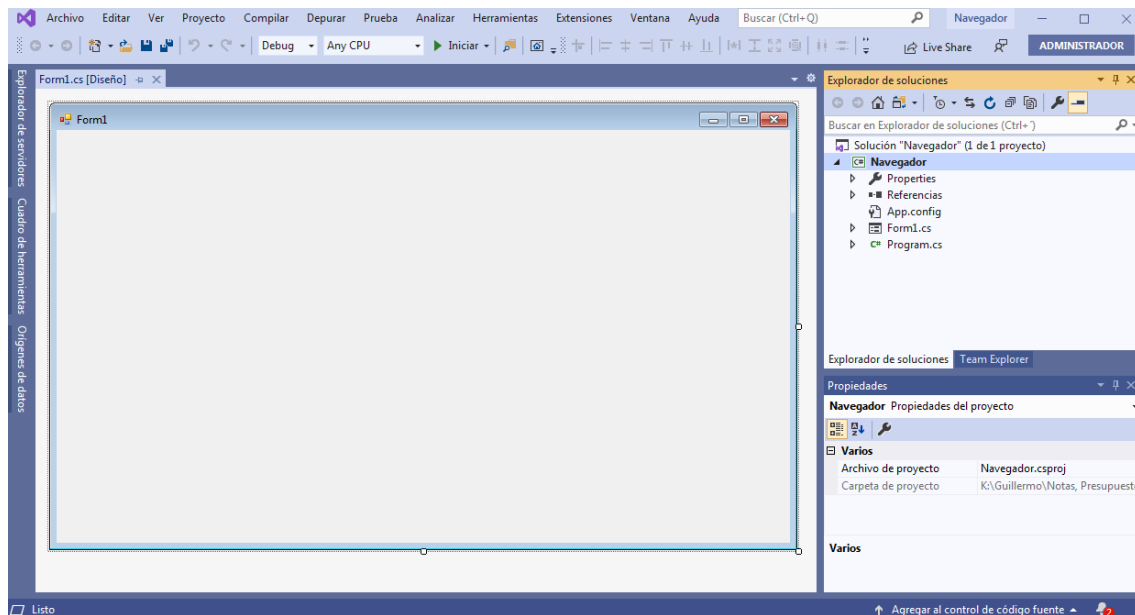
Siguiente

**ATENCIÓN CON ESTE PASO!. Si elegís otra plantilla, aún si “es parecida”, cambia todo el ambiente de desarrollo, y posiblemente no te ande lo que hagamos, o encuentres diferencias significativas. DEBE ser: Aplicación de Windows Forms (.NET Framework), tanto para C# como para VB .NET.**

Coloquen la ruta en la que quieren guardarlo ustedes

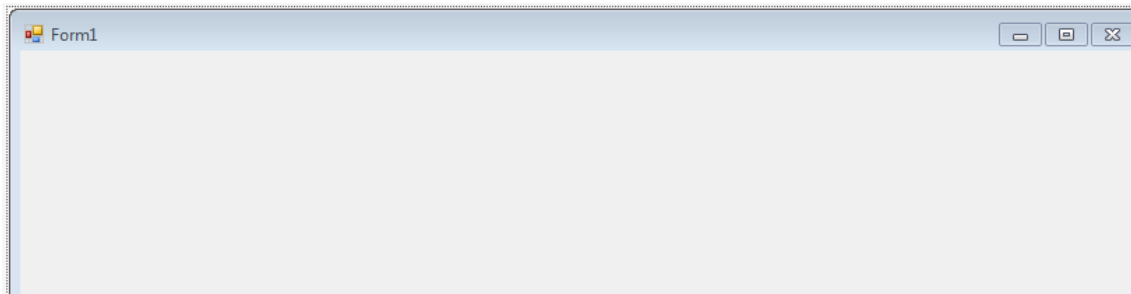
Elegí .NET Framework 4.5 para que me deje ejecutar el programa que hagamos en Windows 7, sino... anda sólo en W 10. La mayoría de la gente tiene 7, si usamos 4.5 vamos a tener de clientes a todos los que tengan 7 y 10, sino.... (muy pocas empresas tienen 10 aún).

Crear



El Explorador de Soluciones te muestra los archivos que dependen de tu Solución. Una solución puede tener dentro varios programas, por eso de la Solución cuelga **C# Navegador**

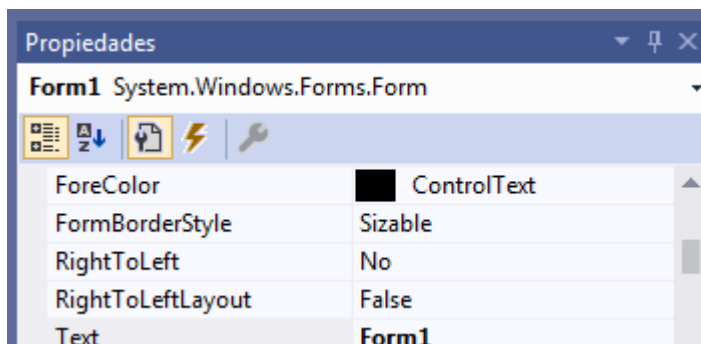
Del programa Navegador, cuelgan varias cosas, nos centramos en Form1.cs, que es el formulario windows que vamos a usar. Al ejecutar, formará la ventana de Windows del programa. En VB va a decir Form1.vb




Este es tu formulario, lo que será tu programa. Dale el tamaño que querés que tenga la ventana de Windows de tu EXE cuando lo compilemos. Por ahora vamos a hacer un buscador de Internet con la potencia de Google (medio trampa pero cumple...). En las próximas clases podemos hacer un navegador REAL con todas las de la ley y a nuestro gusto completo. Este ejemplo es más para ver como funcionan las cosas básicas del diseñador, pero andar va a andar...

Como vimos en la clase 1, todos los objetos son clases, y hay que instanciar una clase para poder usarla. El form1, es la instancia del objeto form, y ya aparece instanciado. Como cualquier objeto, tiene propiedades, métodos y eventos como repasamos en la clase anterior.

Para ver las propiedades del objeto Form1, primero le hacemos click para seleccionarlo, y luego nos fijamos en la Ventana de Propiedades.



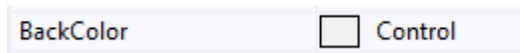
Desplazate entre las propiedades, y vas a notar que están agrupadas. Personalmente no recomiendo que estén así, porque es difícil encontrar algo si no sabés primero a qué grupo pertenece, es mejor buscarlo alfabético. Apretá  que está en la barra.

Ahora van a estar ordenados alfabéticamente, menos la propiedad Name, que es el nombre de la instancia del objeto. Dejemos Form1 como está (sino hay que hacer varias cosas extra... para otra clase...)

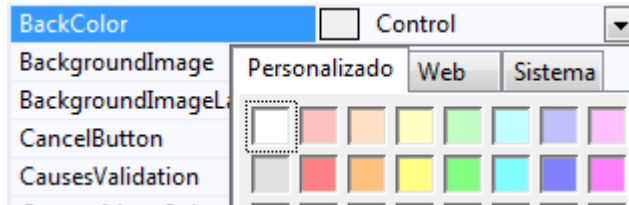
(Name)	<b>Form1</b>
AcceptButton	(ninguno)
AccessibleDescription	

Vamos a cambiar Propiedades en Tiempo de Diseño, lo que significa que lo que cambiemos se verá cuando ejecutemos y cuando volvamos. También podemos cambiar las propiedades en tiempo de ejecución, por ejemplo, el formulario está gris, pero aprieto un botón que hace que el backcolor sea rojo, y lo vemos rojo.

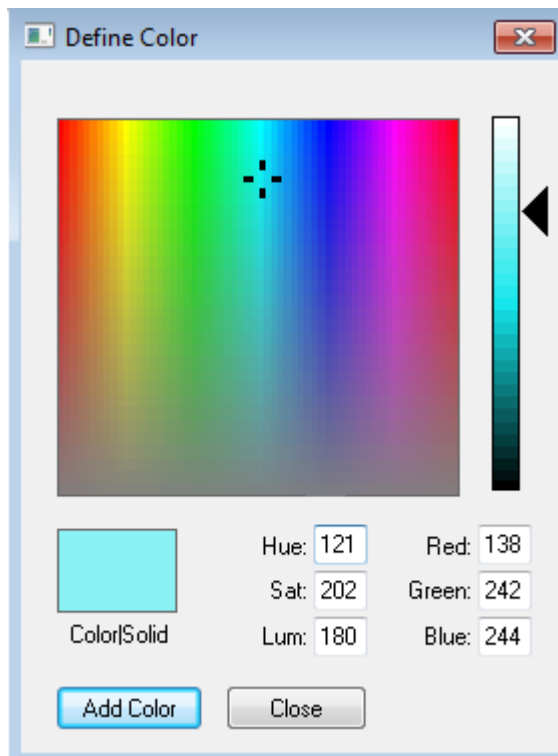
Al cortar la ejecución queda como definimos en diseño... gris. Si le hubiéramos cambiado el color en diseño, iba a seguir de este color. Bueno... cambiemos el color, buscá Backcolor



hacé click sobre Control y aparece una flechita desplegable

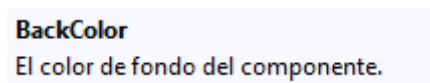


tocá Personalizado y elegí un color. Inmediatamente se cambia "Control" por el RGB (Red, Green, Blue) del color que elegiste. Yo elegí celeste 192, 255, 255. Poné el que quieras. Si elegis en vez de un color, los cuadros blancos de abajo (botón derecho sobre el cuadrado), vas a ver una "cascada" de colores para elegir "más fino"

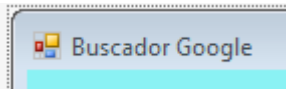


 para agregarlo

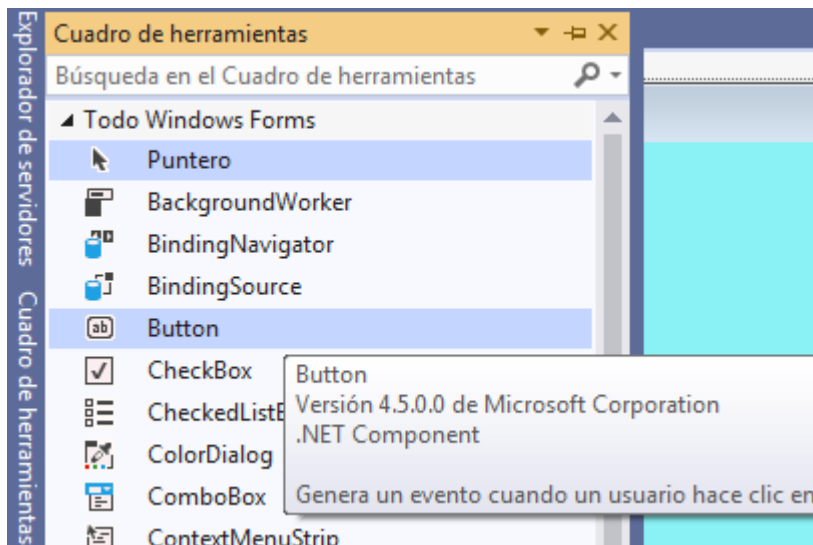
Notá que al tocar cualquier propiedad, abajo te dice lo que es...



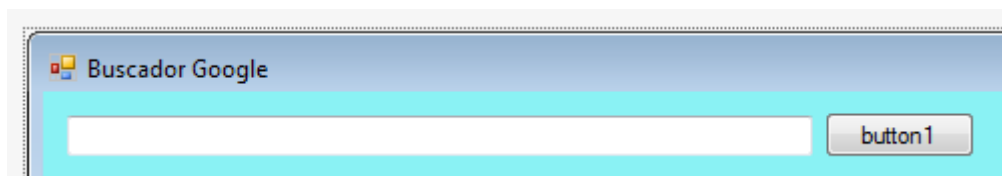
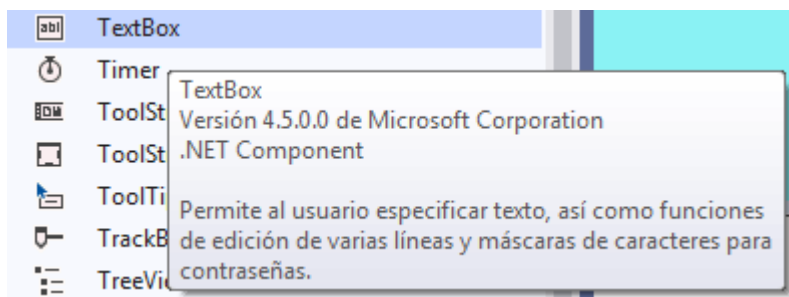
Elegí ahora la propiedad Text, que dice Form1, y poné allí "Buscador Google". Esto cambia el text, el texto que muestra el formulario, NO el nombre. Si vos te ponés una remera que dice Adidas, te seguís llamando igual que antes... Con un click en cualquier otra propiedad ya cambia.



Ahora necesitamos un cuadro de texto: TextBox para poder escribir allí lo que vamos a buscar y un botón. Los objetos que queremos **instanciar**, están en el Cuadro de Herramientas, colgado a la izquierda.



Buscá Button, y arrastralo al formulario, lo mismo que un Textbox



Agrandá el textbox y poné el botón a la derecha.

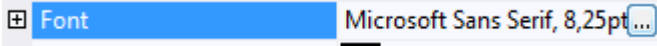
Ahora cambiale las propiedades a los dos objetos (hacele click a uno, y anda a la ventana de propiedades).

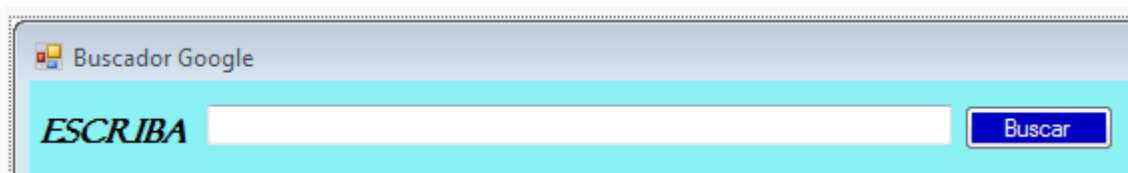
Al botón cambiale el color de fondo a azul, y el color de la letra (forecolor) a blanco (o lo que quieras experimentar). Ponene de nombre bBuscar. En el text ponele "Buscar".

Microsoft recomienda ponerle a cada objeto algo que lo identifique por su tipo, para los botones dice "cmd", yo prefiero "b" de botón, entonces "bBuscar" es botón Buscar. Pongo la B en mayúsculas para que resalte el nombre por sobre el tipo.

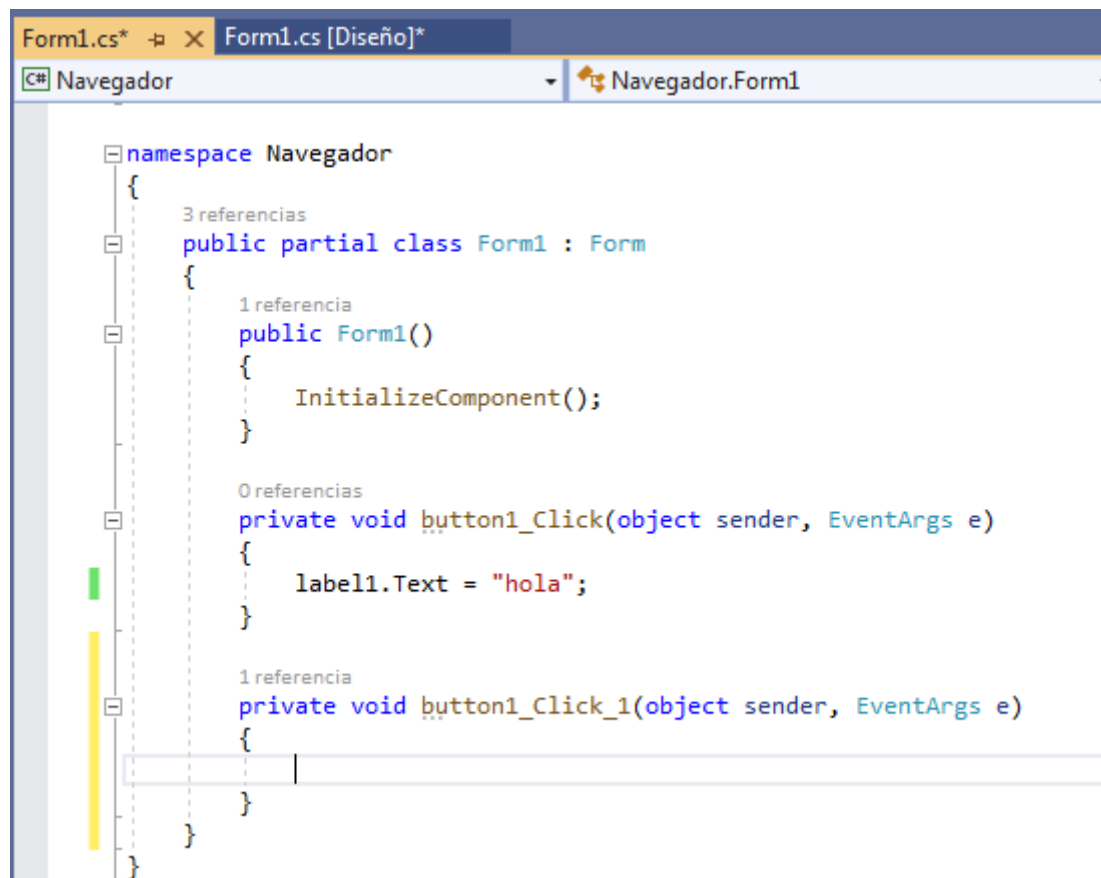
Al textbox probá cambiándole propiedades a tu gusto, pero que el nombre sea "tBuscar" por texto a Buscar.

Agreguemos una Label, que son etiquetas para que diga "Escriba" o algo así para que el usuario sepa que debe escribir en el textbox lo que va a buscar. No importa el nombre porque no le vamos a dar funcionalidad, es solo un cartelito. Probá cambiar la propiedad Font, que es muy parecido al panel de formato de Word (mismo Framework...)

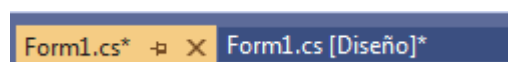
 tocá los 3 puntitos y cambiale la fuente a la label por la que quieras.



Ahora hacele doble click al botón. Eso te va a llevar a la ventana de Código



si querés moverte entre Diseño y Código, apretá las solapas de arriba



```
private void button1_Click_1(object sender, EventArgs e)
{
}
}
```

Este es el código para el evento click. Lo que escribas acá se va a ejecutar cuando aprietes el botón.

Esto es una sub, que empieza y termina con llaves. En VB, empiezan con **Sub** y terminan con **end sub**. En C# hay que terminar cada línea con el pezado punto y coma; o te dá error; en VB no se pone nada...

Necesitamos ahora acceder al namespace que tiene la clase que usa Windows para abrir el navegador. Te acordás que los namespace se separan con puntos?, primero la clase más primitiva, y al final la que querés:

empezá a escribir System, pero vas a notar que a las primeras letras ya te aparece escrito System, entonces apretá el punto que es lo que apretarías al terminar de escribir pero el **intellisense** lo hace por vos. Acostumbrate si o si al intellisense para que escriba solo las instrucciones y comandos, sino vas a sufrir bastante porque algunos son largos y arrevezados, aparte la idea es trabajar lo menos posible no?....

Andá usando el intellisense hasta que completes:

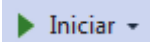
System.Diagnostics.Process.Start

Esta clase no abre sólo el navegador, en realidad es la misma que al hacer doble click sobre un nombre de archivo, reconoce el tipo de archivo y lo abre. En este caso vamos a poner la url de google para que lo abra. Y al final el punto y coma...

Te queda:

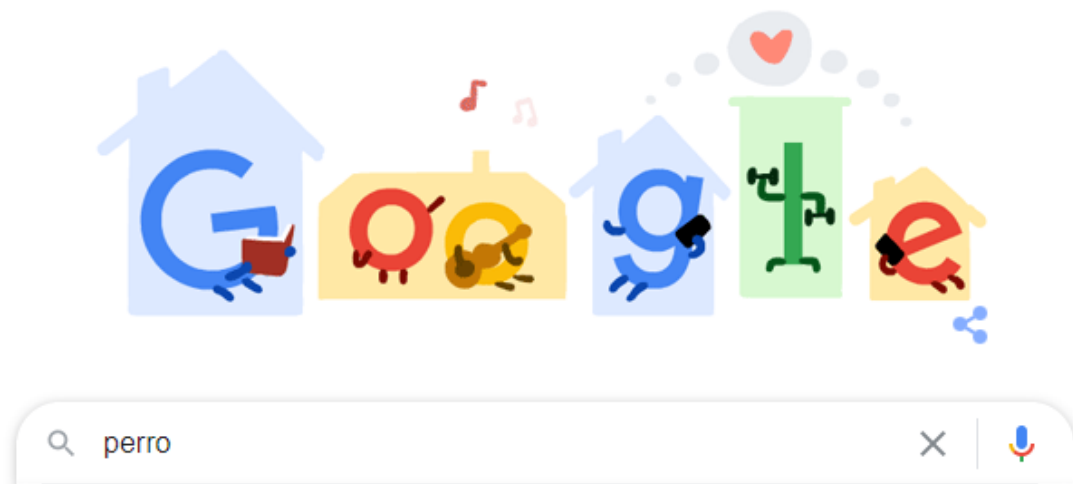
```
private void button1_Click_1(object sender, EventArgs e)
{
    System.Diagnostics.Process.Start("www.google.com");
}
```

Ahora ejecutamos. Apretá "el play" (porque hay uno que parece el stop...)



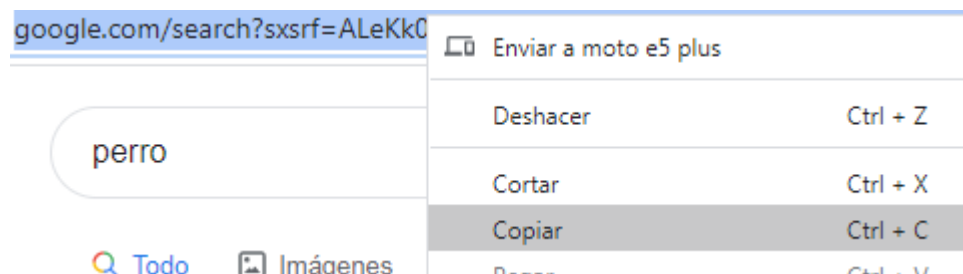
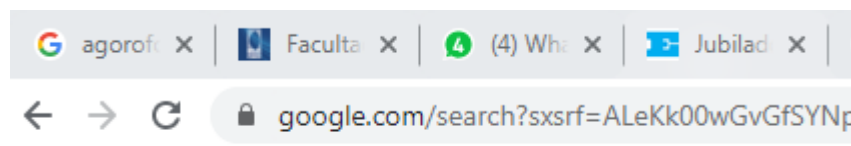
Ahora tenés tu ventana que será tu exe (de hecho se acaba de compilar un exe que luego vamos a buscar)

No vale la pena que escribas nada en el textbox, porque aún no lo usamos. Apretá el botón Buscar y se tiene que abrir Google. (no te decepciones... hay un poco más para hacer...). Escribí "perro" en el buscador de google. En serio... seguime ahora paso a paso por más raro que te parezca lo que sigue...




buscá

Ahora mirá la barra de dirección, y copió con mucho cuidado TODO lo que hay allí



En mi caso (que debería ser muy parecido al tuyo), es :

[https://www.google.com/search?sxsrf=ALeKk00wGvGfSYNp7KYDf9mFIAV90oY-YA%3A1585961134753&source=hp&ei=rtiHXqvRK-uu5OUPh62fyAQ&q=perro&oq=perro&gs\\_lcp=CgZwc3ktYWIQAzIFCAAQgwEyAggAMgIIADIFCAAQgwEyAggAMgIIADICCAyAggAMgIIADIECAAQAzHCCMQ6gIQJzoECCMQJzoECAAQ0oWCBcSEjBnODdnMTI5ZzEwNmc4N2c4MEoPCBgSCzBnMWcxZzFnMWcxUKjeBVif5wVgrqUHaANwAHgAgAF5iAHCA5IBAzQuMZgBAKABAaoBB2d3cy13aXgwAQo&sclient=psy-ab&ved=0ahUKEwir4L\\_Qxc3oAhVrF7kGHYfWB0kQ4dUDCAc&uact=5](https://www.google.com/search?sxsrf=ALeKk00wGvGfSYNp7KYDf9mFIAV90oY-YA%3A1585961134753&source=hp&ei=rtiHXqvRK-uu5OUPh62fyAQ&q=perro&oq=perro&gs_lcp=CgZwc3ktYWIQAzIFCAAQgwEyAggAMgIIADIFCAAQgwEyAggAMgIIADICCAyAggAMgIIADIECAAQAzHCCMQ6gIQJzoECCMQJzoECAAQ0oWCBcSEjBnODdnMTI5ZzEwNmc4N2c4MEoPCBgSCzBnMWcxZzFnMWcxUKjeBVif5wVgrqUHaANwAHgAgAF5iAHCA5IBAzQuMZgBAKABAaoBB2d3cy13aXgwAQo&sclient=psy-ab&ved=0ahUKEwir4L_Qxc3oAhVrF7kGHYfWB0kQ4dUDCAc&uact=5)

Cortamos en programa con el cuadradito de Stop  (si bien podés usar la X del formulario, muchas veces no garantiza que el programa se haya detenido, y vas a notar que "desapareció" la ventana de propiedades entre otras cosas... no es un bug... es que aún está andando. Si te pasa tocá el Stop.



Ahora tenés que pegar esto reemplazando el "www.google.com" en el código. te quedaría:

```
private void button1_Click_1(object sender, EventArgs e)
{
    System.Diagnostics.Process.Start("https://www.google.com/search?sxsrf=ALeKk00w(
}
```

y se me va de la pantalla.

Lo que sigue es trabajo muy fino...

Mirá con cuidado la url que pusiste, y vas a notar que en 2 partes dice "perro". La idea es cortar el texto antes y después de perro, para reemplazarlo por el textbox concatenado, así que si pusiste "gato" en el textbox, al concatenarse, se vuelve a armar toda la url pero buscando gato. Y así con cualquier cosa que busques.

En mi caso la parte de los perros es:

[fyAQ&q=perro&oq=perro&gs\\_lcp=](#)

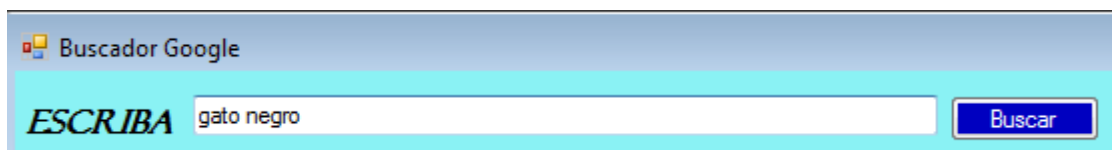
con mucho cuidado tenés que reemplazar cada "perro" por:

" + textbox.text + "

las comillas dobles van... , te quedaría:

[2fyAQ&q=" + tBuscar.Text + "&oq=" + tBuscar.Text + "&gs\\_lcp=](#)

Ejecutá y probá buscando algo...



Tenés ahora un buscador con la "potencia de Google", porque usa Google... no mentí. Cortá la ejecución del programa. Con el explorador de Windows, abrí la carpeta en dónde guardaste el proyecto. En mi caso "K:\Guillermo\Notas, Presupuestos y CV\Universidad\Navegador"

Nombre	Fecha de modifica...
.vs	03/04/2020 08:36 ...
bin	03/04/2020 08:36 ...
obj	03/04/2020 08:36 ...
Properties	03/04/2020 08:36 ...
App.config	03/04/2020 08:36 ...
Form1.cs	03/04/2020 10:06 ...
Form1.Designer.cs	03/04/2020 10:06 ...
Form1.resx	03/04/2020 10:06 ...
Navegador.csproj	03/04/2020 08:42 ...
Navegador.sln	03/04/2020 08:36 ...
Program.cs	03/04/2020 08:36 ...

Abrí la carpeta bin y luego Debug

Navegador.exe	03/04/2020 10:06 ...
Navegador.exe.config	03/04/2020 08:36 ...
Navegador.pdb	03/04/2020 10:06 ...

Copió el navegador.exe y pegalo en el escritorio o en cualquier otra carpeta, para que notes que no tiene dependencia de todo el entorno .NET

Luego hacele doble click, y se abre el programa!. Podés mandárselo a un amigo (metelo en un rar para que lo mande el correo), y tu amigo si tiene W7 o W10, no necesita instalar, le hace doble click y anda.

Armame ahora el mismo proyecto pero en VB, te debería resultar más fácil. Recordá usar & en vez de + (podés con el + pero & es mejor) y guardá ambos proyectos.

Vas a notar que no escribiste código C# ni VB... que el entorno cambia pero lo que vos escribiste es Framework...

Practicá todo lo que vimos en C# y VB. Experimentá con objetos y propiedades. La próxima clase vemos más código y más equivalencias C# / VB

En la Clase 3 vamos a hacer un verdadero buscador integrado en la aplicación. Que no requiera de un navegador, porque está integrado, y que puedas mejorarlo, usarlo y pasárselo a tus amigos.

Nos leemos en el foro!.

Guillermo