Capítulo 8
Creación de formularios

1. Utilizar los formularios

Los formularios representan la interacción del usuario con una aplicación. Se utilizan tanto para la presentación como para la introducción de datos. El diseño de la interfaz de usuario es una etapa importante, ya que una interfaz visualmente coherente y lógica, y por tanto usable, es más sencilla de utilizar.

1.1 Añadir formularios al proyecto

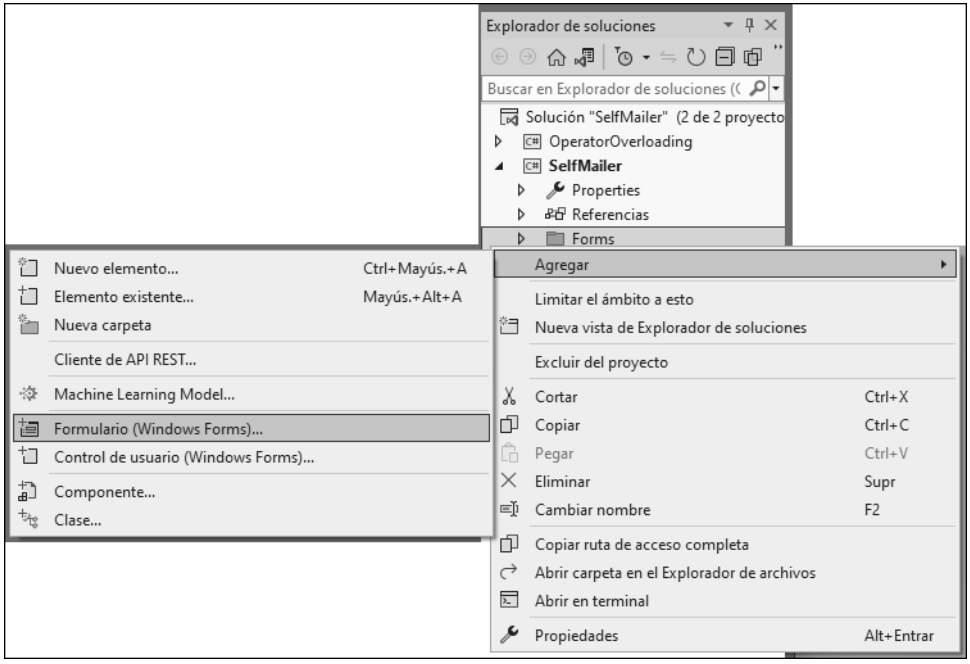
Durante la creación del proyecto de aplicación Windows se crea un formulario por defecto, **Form1**. Este formulario representa la clase que se instanciará durante la ejecución. Esta clase parcial está compuesta por dos archivos **Form1.designer.cs** y **Form1.cs**:

– **Form1.designer.cs**: los archivos de formularios con extensión **.designer.cs** se generan automáticamente en el diseñador de pantallas. Cuando se ubica un control en el formulario, Visual Studio inserta el código de inicialización y los parámetros por defecto en este archivo. El diseñador de pantallas es, en este caso, un mecanismo para crear código de manera visual.

– **Form1.cs**: este archivo permite escribir la lógica del formulario, los tratamientos de las acciones de usuario y todo el código necesario para el correcto funcionamiento lógico del formulario.

Antes de crear un nuevo formulario vamos a crear una nueva carpeta llamada **Forms** en la raíz del proyecto para almacenar en ella todos los formularios. Separar los elementos dentro de un proyecto permite tener una visualización de conjunto mejor y encontrar con mayor facilidad el elemento deseado, sobre todo en caso de mantenimiento.

Haciendo clic con el botón derecho del ratón en la nueva carpeta **Forms** y seleccionando después **Agregar** y **Formulario (Windows Forms)**, se abre el cuadro de diálogo que permite agregar un elemento en Visual Studio, preseleccionando el tipo **Windows Forms**. Basta con introducir el nombre del formulario, **MailServerSettings.cs**, y validar.



El archivo **MailServerSettings.cs** que crea Visual Studio, tiene este aspecto:

```
namespace SelfMailer.Forms
{
    public partial class MailServerSettings: Form
    {
        public MailServerSettings()
        {
            InitializeComponent();
        }
    }
}
```

Observe que el nombre de la carpeta que contiene (**Forms**) se ha añadido al espacio de nombres del formulario. Además de clasificar los elementos visualmente, las carpetas permiten a Visual Studio organizar la jerarquía de clases dentro de los espacios de nombres durante la creación.

La clase parcial, identificada por la palabra clave `partial`, deriva de la clase `Form`, y permite al formulario heredar todas las propiedades básicas de un formulario Windows desde el Framework .NET. Es posible hacer que un formulario herede de otro formulario cuando, por ejemplo, el aspecto es el mismo pero el tratamiento es diferente.

La última observación sobre esta clase es que el constructor tiene una llamada al método `InitializeComponent` que se encuentra en el archivo **MailServerSettings.designer.cs**:

```
private void InitializeComponent()
{
    this.components = new System.ComponentModel.Container();
    this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
    this.Text = "MailServerSettings";
}
```

Este método se encarga de ejecutar la inicialización de los elementos del formulario y de definir las propiedades de los mismos. Se desaconseja modificar este archivo con el editor de texto, ya que el diseñador podría eliminar los cambios durante la generación de este código.

1.2 Modificar el formulario de inicio

El formulario de inicio es el que se abre por defecto cuando se ejecuta la aplicación. Se trata de la clase que se instancia en el archivo **Program.cs**.

Para modificar el formulario de inicio, que actualmente es **Form1**, hay que empezar creando uno nuevo en la carpeta **Forms** y llamarlo **Main.cs**. En el archivo **Program.cs**, basta con sustituir la instrucción:

```
Application.Run(new Form1());
```

por

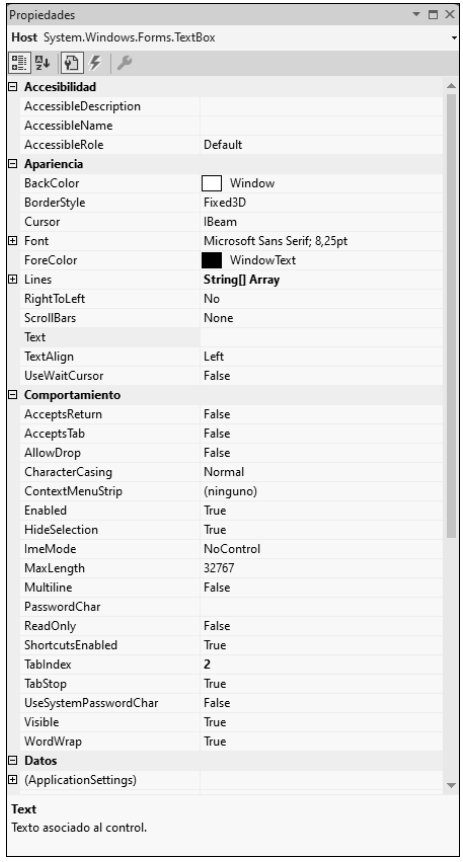
```
Application.Run(new SelfMailer.Forms.Main());
```

Como el formulario **Form1** ya no es necesario, se puede eliminar.

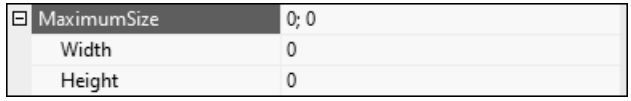
Ejecutando la depuración ([F5]) comprobamos que se muestra el formulario **Main**.

1.3 Las propiedades de los formularios

En modo diseñador de pantallas, Visual Studio permite definir las propiedades de los objetos gracias a la ventana **Propiedades** ([Alt][Intro]).



Las propiedades se clasifican en grupos lógicos. Cada una de ellas afecta al aspecto o al comportamiento del formulario. Algunas propiedades, como la propiedad `MaximumSize`, son estructurales. Haciendo clic en la flecha que hay al lado, es posible modificar cada propiedad:



Las propiedades también se pueden modificar a nivel de código:

```
this.Text = "Argumentos del servidor mail";
```

Esta línea, ubicada en el constructor del formulario, tendrá el mismo efecto que modificar la propiedad en la ventana de propiedades del diseñador de pantallas, con la diferencia de que Visual Studio habrá situado esta instrucción en el archivo designer. En el archivo con extensión `.designer.cs`, se transcriben todas las modificaciones realizadas en el constructor de formularios en código por Visual Studio.

La asignación de las propiedades sigue la misma sintaxis que para otros miembros de la clase. El operador de asignación (=) se utiliza para asignar un valor a la propiedad referenciada con su nombre.

Asigne las siguientes propiedades al formulario **MailServerSettings**:

Propiedades	Valor	Efecto
`FormBorderStyle`	`FixedToolWindow`	El formulario ya no se redimensiona. Los botones para agrandar o reducir la ventana y el icono ya no se ven más.
`ShowInTaskbar`	`False`	Cuando el formulario se abre, no se ve ninguna pestaña en la barra de tareas de Windows.
`Size`	`412; 231`	Fija el tamaño del formulario.
`StartPosition`	`CenterParent`	El formulario se muestra centrado en relación a su padre.
`Text`	Parámetros del servidor mail	Se modifica el texto en la barra del título del formulario.

Con la asignación de los valores anteriores en el diseñador de pantallas, Visual Studio completa el archivo designer con las líneas siguientes:

```
this.ClientSize = new System.Drawing.Size(412, 213);
this.FormBorderStyle =
            System.Windows.Forms.FormBorderStyle.FixedToolWindow;
this.ShowInTaskbar = false;
this.StartPosition =
```

```
            System.Windows.Forms.FormStartPosition.CenterParent;
this.Text = "Argumentos del servidor mail";
```

Todo lo que se hace en el diseñador de pantallas, repercute en el código del archivo designer.

1.4 Los métodos de los formularios

Los formularios derivan de la clase base `System.Windows.Forms.Form` y heredan varios métodos que permiten gestionar la visualización de los formularios.

Una vez instanciado, el objetivo de un formulario es permitir al usuario interactuar con él. Los métodos `Show` y `ShowDialog` de un formulario hacen que el formulario se pueda ver. El método `Show` muestra el formulario en la pantalla y le asigna el foco. La propiedad `Visible` del formulario toma automáticamente el valor `true`. El método `Show` no crea el formulario. Si existe en memoria pero no es visible, es decir si su propiedad `Visible` es `false`, la llamada del método cambia este valor a `true`. El método `ShowDialog` es idéntico, excepto que el formulario es modal, es decir, se debe cerrar antes de que el foco se pueda asignar a otro formulario de la aplicación. Este método obliga al usuario a realizar una acción.

Después de que el usuario ha realizado las acciones que desee en el formulario, éste se debe cerrar. Existen dos maneras de cerrar un formulario. La primera consiste en ocultarlo con el método `Hide`. Esto equivale a asignar el valor `false` a la propiedad `Visible` del formulario. Con este método, el formulario siempre está en memoria y la llamada del método `Show` lo vuelve visible. La segunda manera de cerrar un formulario es con el método `Close`, que elimina la instancia en memoria y librera los recursos. Es imposible llamar al método `Show` de un formulario después llamar al método `Close`, pues el formulario ya no existe y se debe instanciar de nuevo. La llamada al método `Close` para el formulario de inicio implica cerrar la aplicación.

Observación

Estos métodos se usan durante la implementación de administradores de eventos.