GRAPHICAL USER INTERFACE DESIGN

Chapter 7

Software Engineering
Computer Science School
DSIC – UPV

Goal

- Understand the principles of visual applications.
- Understand the design of the graphical user interface (use of controls and events).
- Understand the communication between the presentation and the business logic layers.

Contents

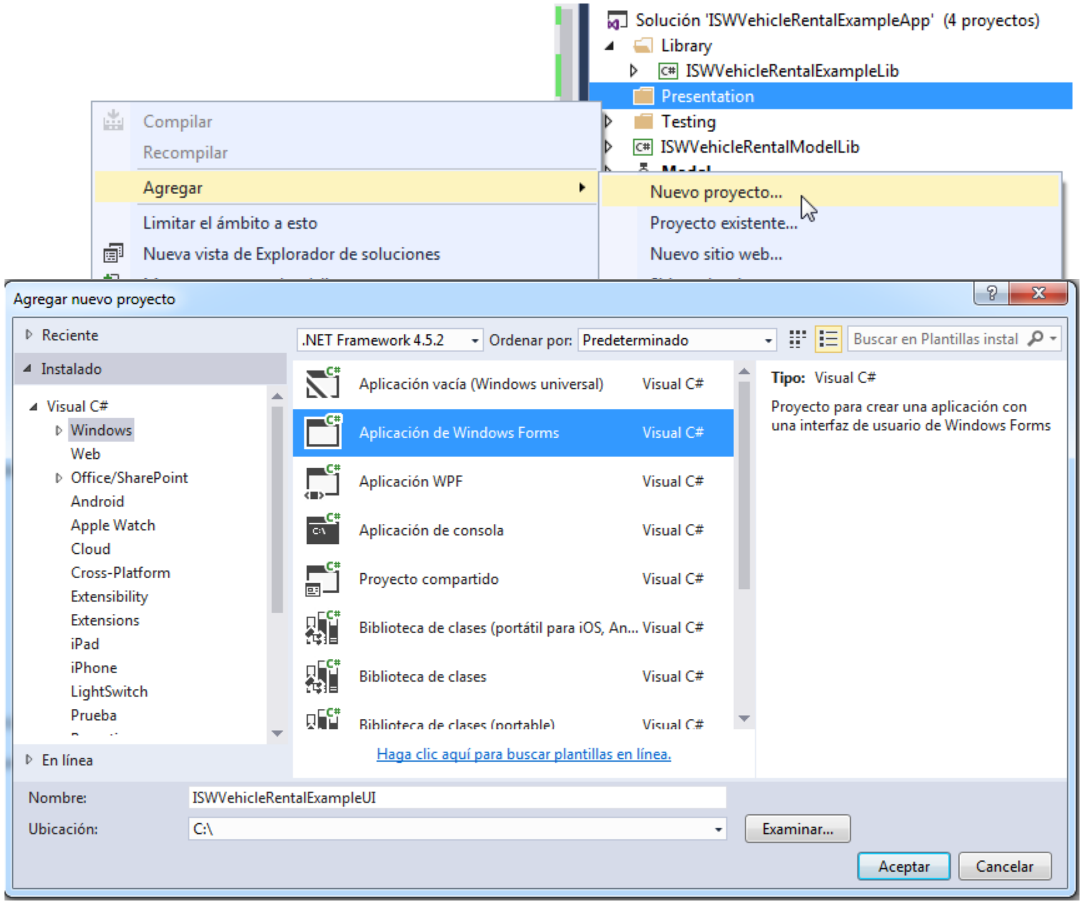
Introduction

- The creation of **Visual Apps for Windows** may be done, among others with the namespace `System.Windows.Forms` which includes classes, structures, interfaces, etc. to develop these types of applications.

- The namespace `System.Windows.Forms` includes the following classes:
  - **`Application`**: The core of a Windows app. Its methods are used to process Windows messages and visual apps are created and destruyed.
  - **`Form`**: Represents a window or a dialog box in a visual application.
  - **`Button`**, **`ListBox`**, **`TextBox`**, **`PictureBox`**, **`Label`**,...: Providing the functionality of common Windows controls.
  - **`StatusBar`**, **`ToolBar`**,...:Windows utilities.
  - **`ColorDialog`**, **`FileDialog`**,...: Standard dialog boxes.
  - **`StripMenu`**, **`StripMenuItem`**,...: Use to create different types of menus.
  - **`ToolTip`**, **`Timer`**,...: To ease the interactivity of applications.

Creating a Windows Application

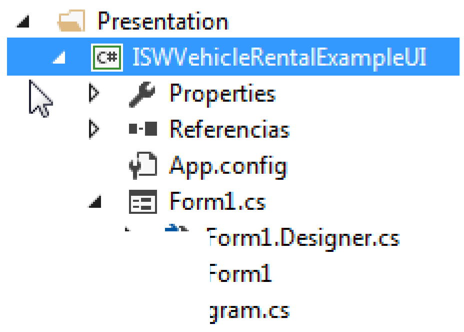
- Add a new project of type **Aplicación de Windows Forms** to the solution folder **Presentation**.


Solución 'ISWVehicleRentalExampleApp' (4 proyectos)
Library
ISWVehicleRentalExampleLib
Presentation
Testing
ISWVehicleRentalModelLib
Compilar
Recompilar
Agregar
Limitar el ámbito a esto
Nueva vista de Explorador de soluciones
Nuevo proyecto...
Proyecto existente...
Nuevo sitio web...
Agregar nuevo proyecto
Reciente
Instalado
Visual C#
Windows
Web
Office/SharePoint
Android
Apple Watch
Cloud
Cross-Platform
Extensibility
Extensions
iPad
iPhone
LightSwitch
Prueba
En línea
.NET Framework 4.5.2
Ordenar por: Predeterminado
Buscar en Plantillas instal
Aplicación vacía (Windows universal) Visual C#
Aplicación de Windows Forms Visual C#
Aplicación WPF Visual C#
Aplicación de consola Visual C#
Proyecto compartido Visual C#
Biblioteca de clases (portátil para iOS, An... Visual C#
Biblioteca de clases Visual C#
Biblioteca de clases (portable) Visual C#
Haga clic aquí para buscar plantillas en línea.
Tipo: Visual C#
Proyecto para crear una aplicación con una interfaz de usuario de Windows Forms
Examinar...
Aceptar
Cancelar

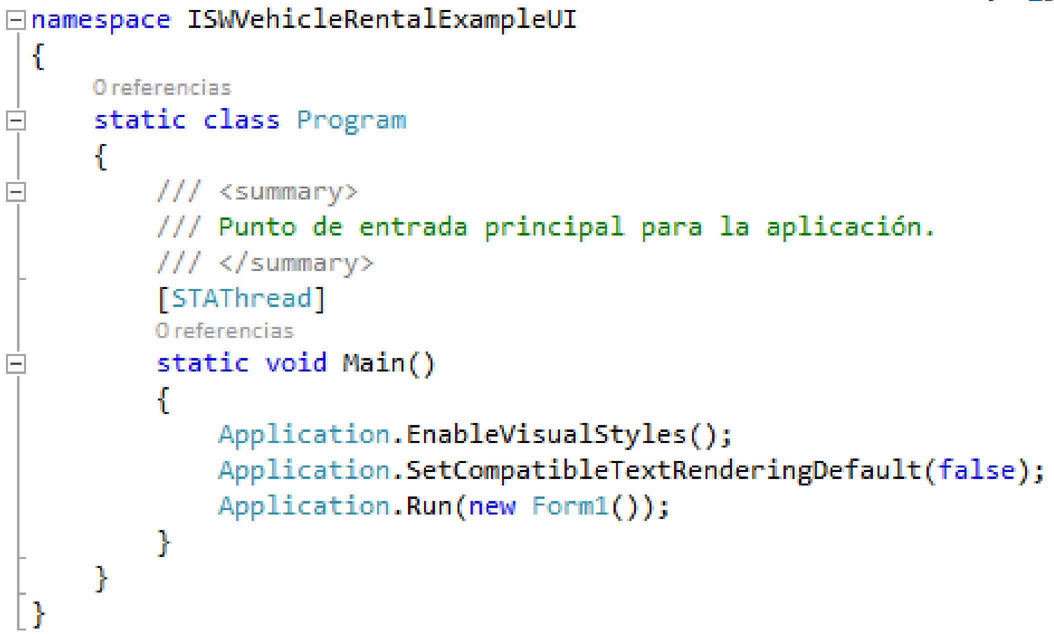
Nombre: ISWVehicleRentalExampleUI
Ubicación: C:\

Creating a Windows Application

- If the app is run, a Windows with the standard basic features is created.
- The files in this Project are:
  - Form1.cs: contains the design of the form. If opened the form may be modified in a visual designer.
    - Form1 has constructo InicializeC
    - Form1.Des generated
  - Program.cs method().



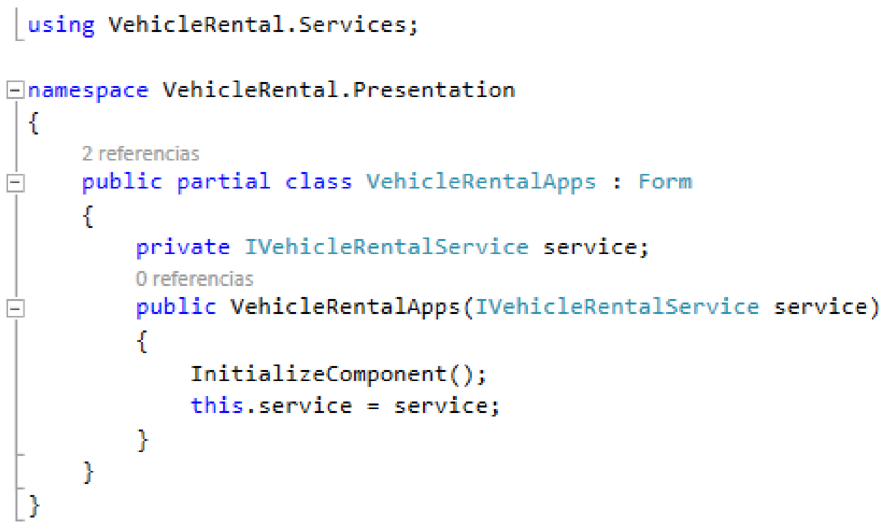
```
Presentation
  ISWVehicleRentalExampleUI
    Properties
    Referencias
    App.config
    Form1.cs
      Form1.Designer.cs
      Form1
    gram.cs
```



```
namespace ISWVehicleRentalExampleUI
{
    0 referencias
    static class Program
    {
        /// <summary>
        /// Punto de entrada principal para la aplicación.
        /// </summary>
        [STAThread]
        0 referencias
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

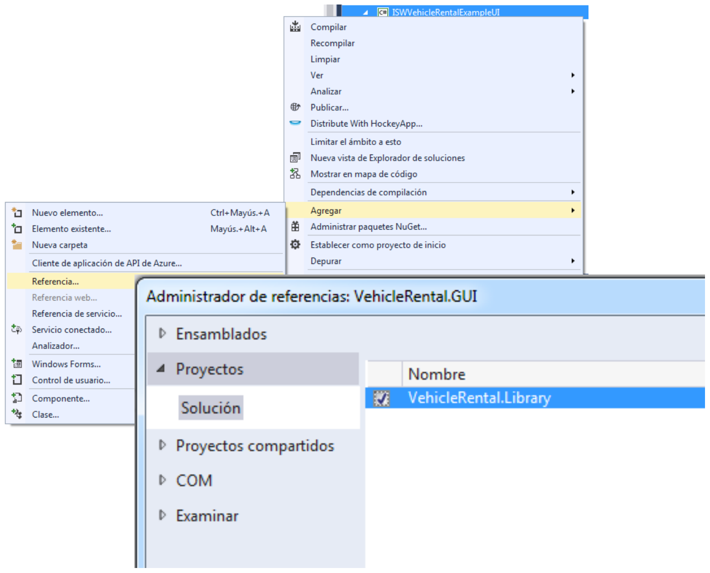
Dependencies Management

- This Project will depend on `IVehicleRentalService` and on the domain clases located at `VehicleRental.Services`. Thus, a reference has to be added



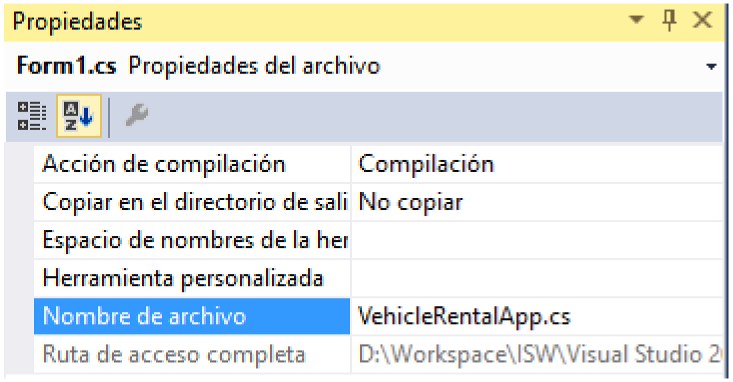
```
using VehicleRental.Services;

namespace VehicleRental.Presentation
{
    2 referencias
    public partial class VehicleRentalApps : Form
    {
        private IVehicleRentalService service;
        0 referencias
        public VehicleRentalApps(IVehicleRentalService service)
        {
            InitializeComponent();
            this.service = service;
        }
    }
}
```


ISWVehicleRentalExampleUI
Compilar
Recompilar
Limpiar
Ver
Analizar
Publicar...
Distribute With HockeyApp...
Limitar el ámbito a esto
Nueva vista de Explorador de soluciones
Mostrar en mapa de código
Dependencias de compilación
Agregar
Administrar paquetes NuGet...
Establecer como proyecto de inicio
Depurar
Nuevo elemento... Ctrl+Mayús.+A
Elemento existente... Mayús.+Alt+A
Nueva carpeta
Cliente de aplicación de API de Azure...
Referencia...
Referencia web...
Referencia de servicio...
Servicio conectado...
Analizador...
Windows Forms...
Control de usuario...
Componente...
Clase...
Administrador de referencias: VehicleRental.GUI
Ensamblados
Proyectos
Solución
Proyectos compartidos
COM
Examinar
Nombre
VehicleRental.Library

First steps…

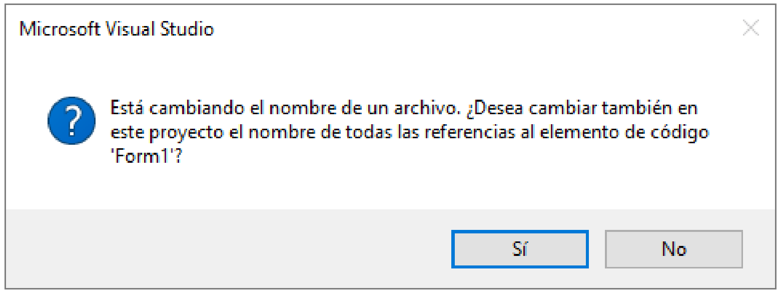
- Give an appropriate name to the elements in the Project (e.g. change the name of the file **Form1.cs** to **VehicleRentalApp**).



Propiedades

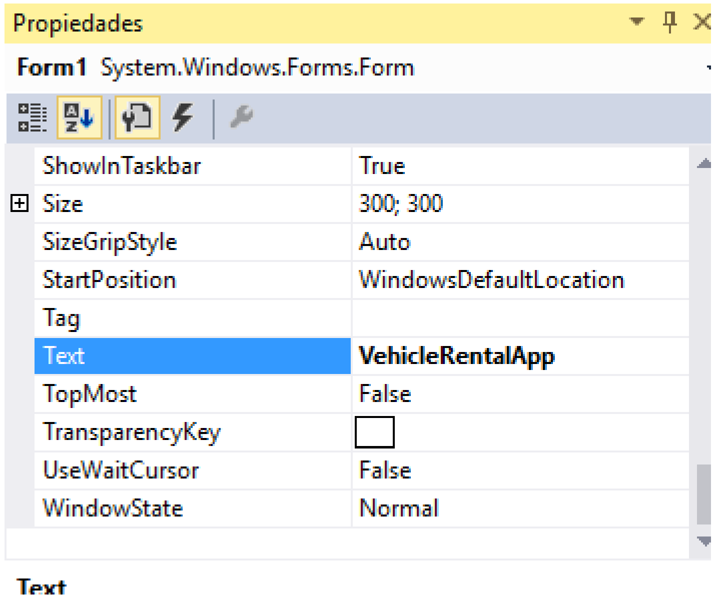
Form1.cs Propiedades del archivo

Acción de compilación	Compilación
Copiar en el directorio de sali	No copiar
Espacio de nombres de la her	
Herramienta personalizada	
Nombre de archivo	VehicleRentalApp.cs
Ruta de acceso completa	D:\Workspace\ISW\Visual Studio 2



Microsoft Visual Studio

Está cambiando el nombre de un archivo. ¿Desea cambiar también en este proyecto el nombre de todas las referencias al elemento de código 'Form1'?



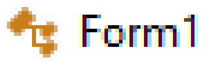
Propiedades

Form1 System.Windows.Forms.Form

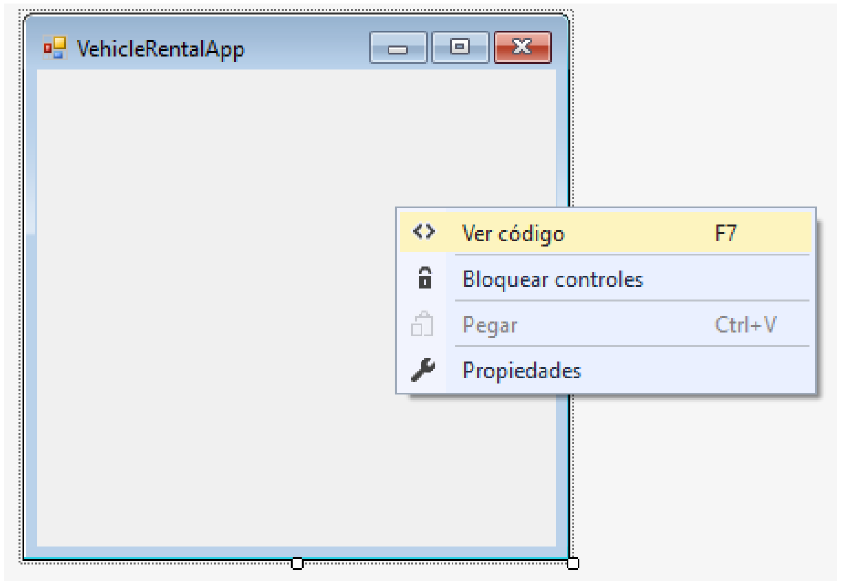
ShowInTaskbar	True
Size	300; 300
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	**VehicleRentalApp**
TopMost	False
TransparencyKey	⬜
UseWaitCursor	False
WindowState	Normal

Text

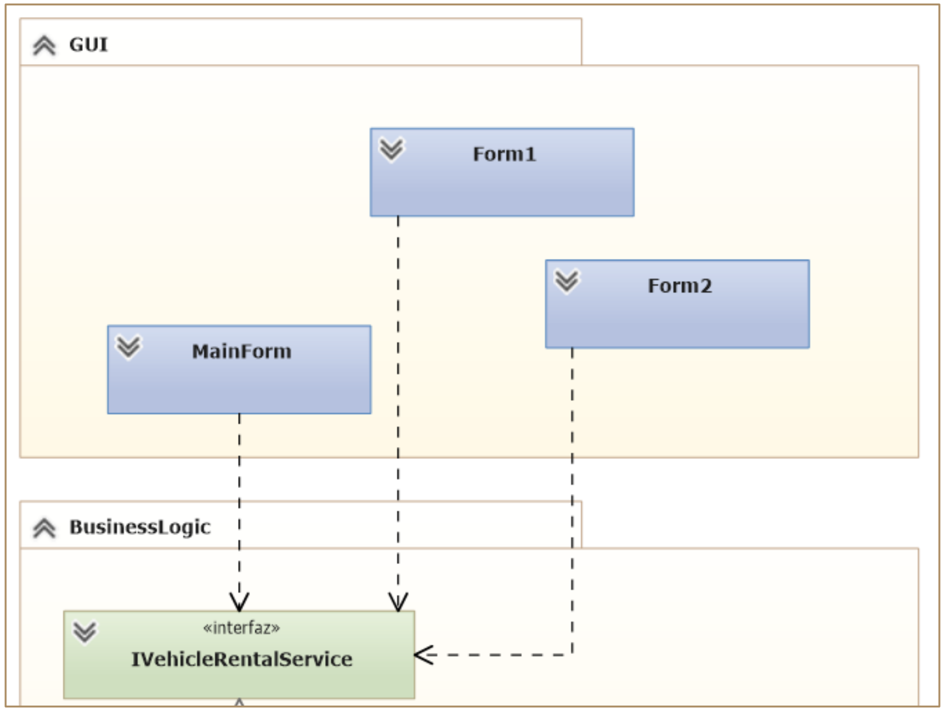
Code Inspection…

- Two ways to Access C# editable code of the form:
  - Double click on 

  - Select the form *right button click* > **Ver código**, or **F7**



Connect with Business Logic Layer



Connect with Business Logic Layer

Modify class **VehicleRentalApp** to have an attribute of type **IVehicleRentalService,** which is passed as a parameter in the constructor.

```
using VehicleRental.Services;

namespace VehicleRentalUI

 public partial class VehicleRentalApp:Form
    {
        private IVehicleRentalService service;

        public VehicleRentalApp(IVehicleRentalService service)
        {
            InitializeComponent();
            this.service = service;        }
   }
}
```

Connect with Business Logic Layer

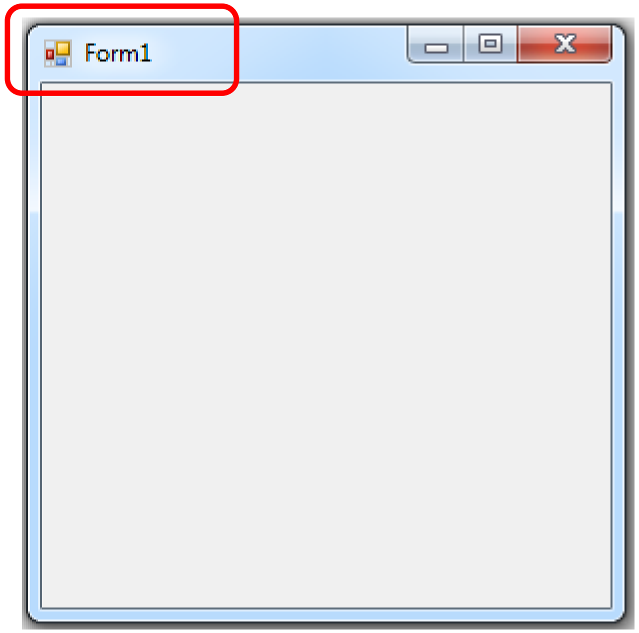
Modify the Main method (Program class) to create an object **IVehicleRentalService** and pass it to the main form.

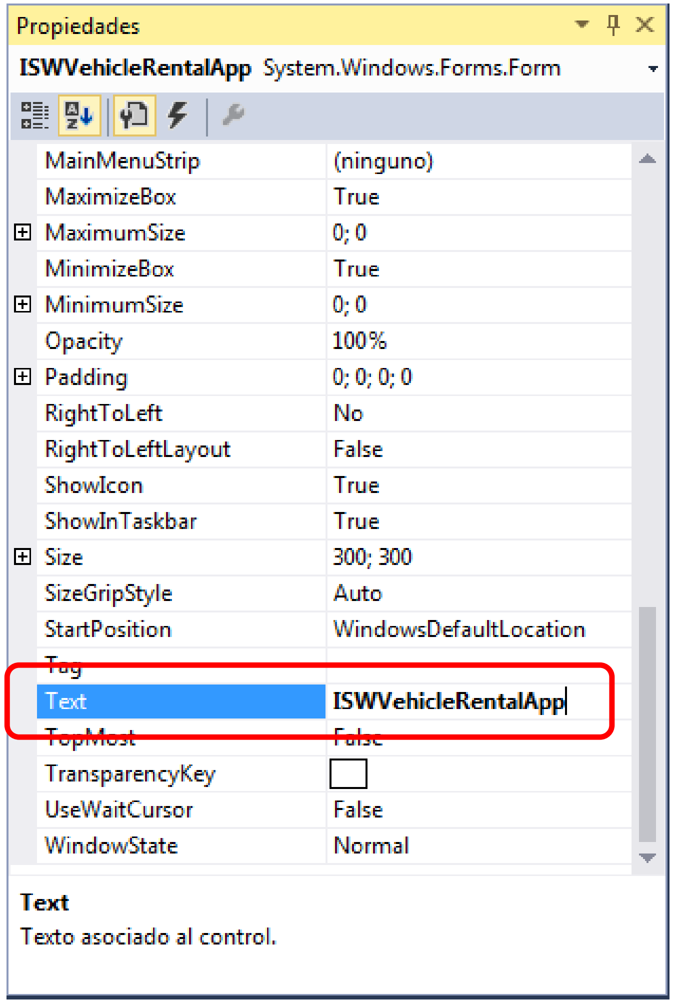
```
static void Main()
{
    IVehicleRentalService service = new VehicleRentalService(new
EntityFrameworkDAL(new VehicleRentalDbContext()));

    Application.EnableVisualStyles();
    Application.SetCompatibleTextRenderingDefault(false);
    Application.Run(new VehicleRentalApp(service));
```

First steps…

- Modify the properties of the form elements:




Propiedades

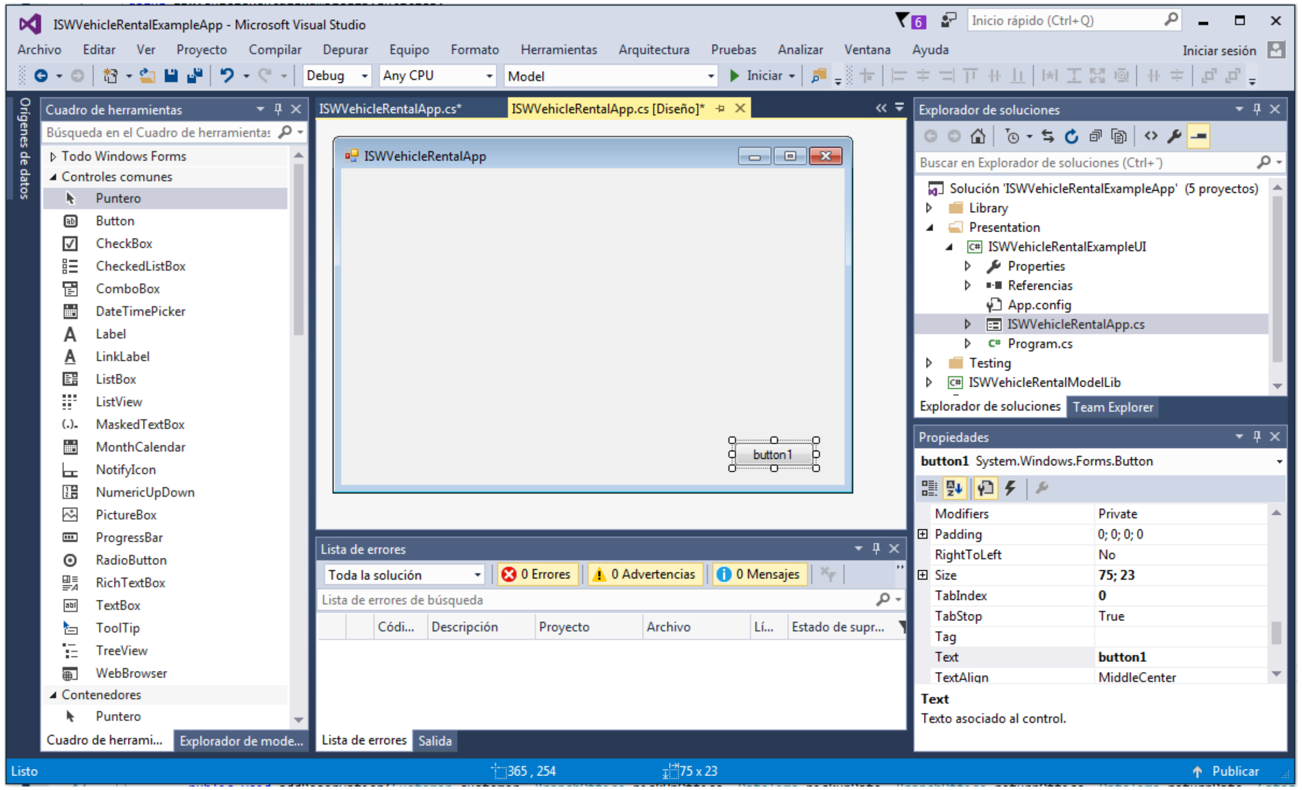
ISWVehicleRentalApp System.Windows.Forms.Form

MainMenuStrip	(ninguno)
MaximizeBox	True
⊞ MaximumSize	0; 0
MinimizeBox	True
⊞ MinimumSize	0; 0
Opacity	100%
⊞ Padding	0; 0; 0; 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
⊞ Size	300; 300
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	**ISWVehicleRentalApp**
TopMost	False
TransparencyKey	☐
UseWaitCursor	False
WindowState	Normal

Text
Texto asociado al control.

Forms with controls

- Controls are objects of the Control class: buttons, textboxes, …
- Can be added at design time (visual editor and toolbox) or at execution time.



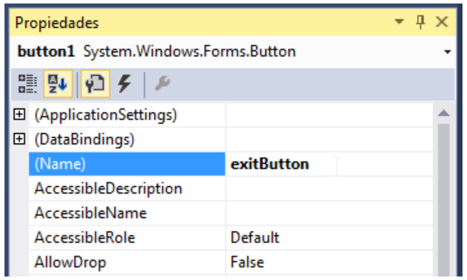
button1 System.Windows.Forms.Button

Modifiers	Private
Padding	0; 0; 0; 0
RightToLeft	No
Size	75; 23
TabIndex	0
TabStop	True
Tag	
Text	button1
TextAlign	MiddleCenter

Text
Texto asociado al control.

Controls: Properties

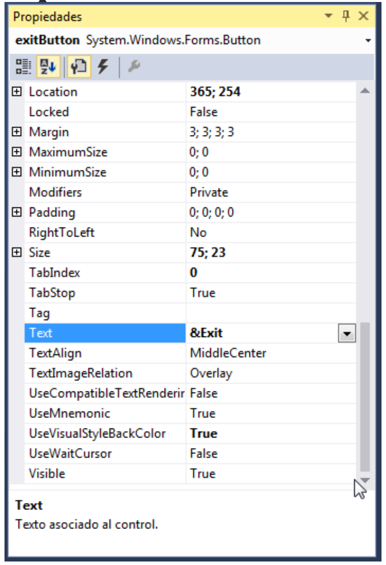
- **Name**: The name of the control. It is important to select a meaningful name.



Propiedades
button1 System.Windows.Forms.Button

(ApplicationSettings)	
(DataBindings)	
(Name)	exitButton
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False

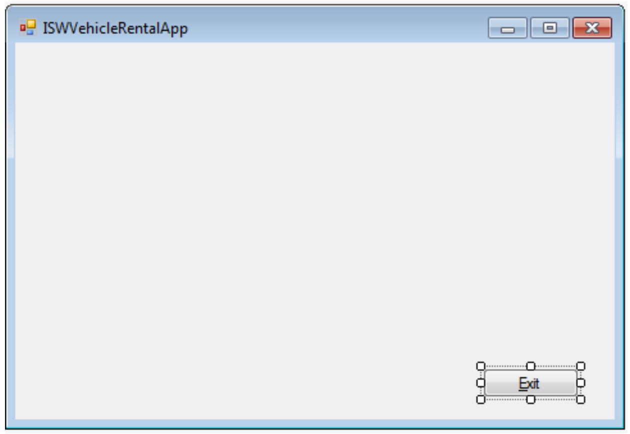
- **Text**: Represents the title of the control



Propiedades
exitButton System.Windows.Forms.Button

Location	365; 254
Locked	False
Margin	3; 3; 3; 3
MaximumSize	0; 0
MinimumSize	0; 0
Modifiers	Private
Padding	0; 0; 0; 0
RightToLeft	No
Size	75; 23
TabIndex	0
TabStop	True
Tag	
Text	&Exit
TextAlign	MiddleCenter
TextImageRelation	Overlay
UseCompatibleTextRenderir	False
UseMnemonic	True
UseVisualStyleBackColor	True
UseWaitCursor	False
Visible	True

Text
Texto asociado al control.

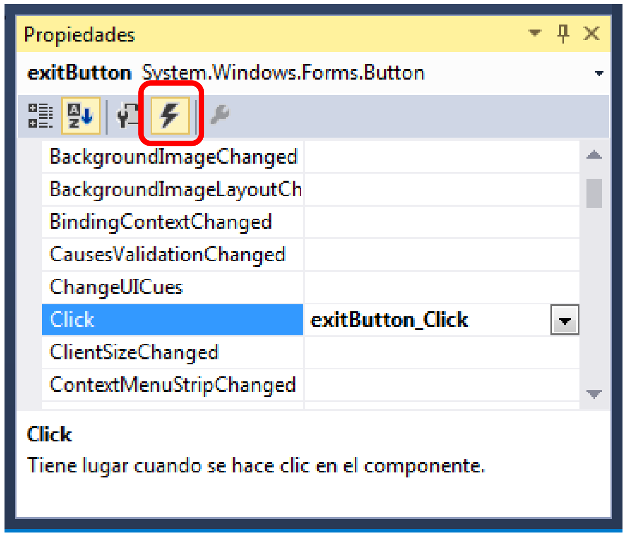


Events in forms

- An event describes a situation to which the application must respond.
- Events are generated by:
  - A user action (click a mouse button, hit a key, etc.)
  - The app code.
  - The operating system.
- Windows apps are event-driven:
  - When an event occurs the app may specify methods (event handlers) to process the event and execute the corresponding actions
- Every control exhibits events to which a handler can be associated.

Events: handlers

- When an event occurs the associated handler is executed
- The events that may be raised by a control appear in the properties window.
- A handler may be associated as follows:
  - Writing the name of the handler method.
  - Selecting a handler method from the dropdown list.
  - double *click*, and Visual Studio creates a default handler definition.

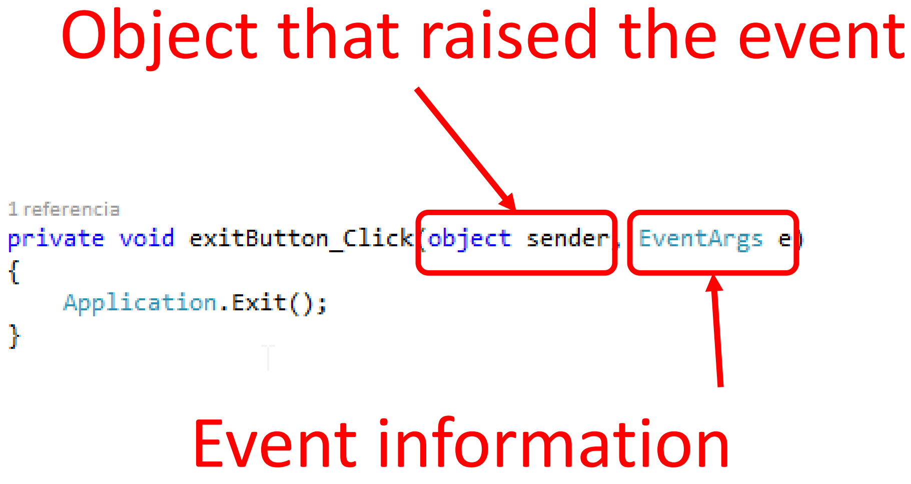


Propiedades

exitButton System.Windows.Forms.Button

BackgroundImageChanged	
BackgroundImageLayoutCh	
BindingContextChanged	
CausesValidationChanged	
ChangeUICues	
Click	exitButton_Click
ClientSizeChanged	
ContextMenuStripChanged	

Click
Tiene lugar cuando se hace clic en el componente.



Object that raised the event

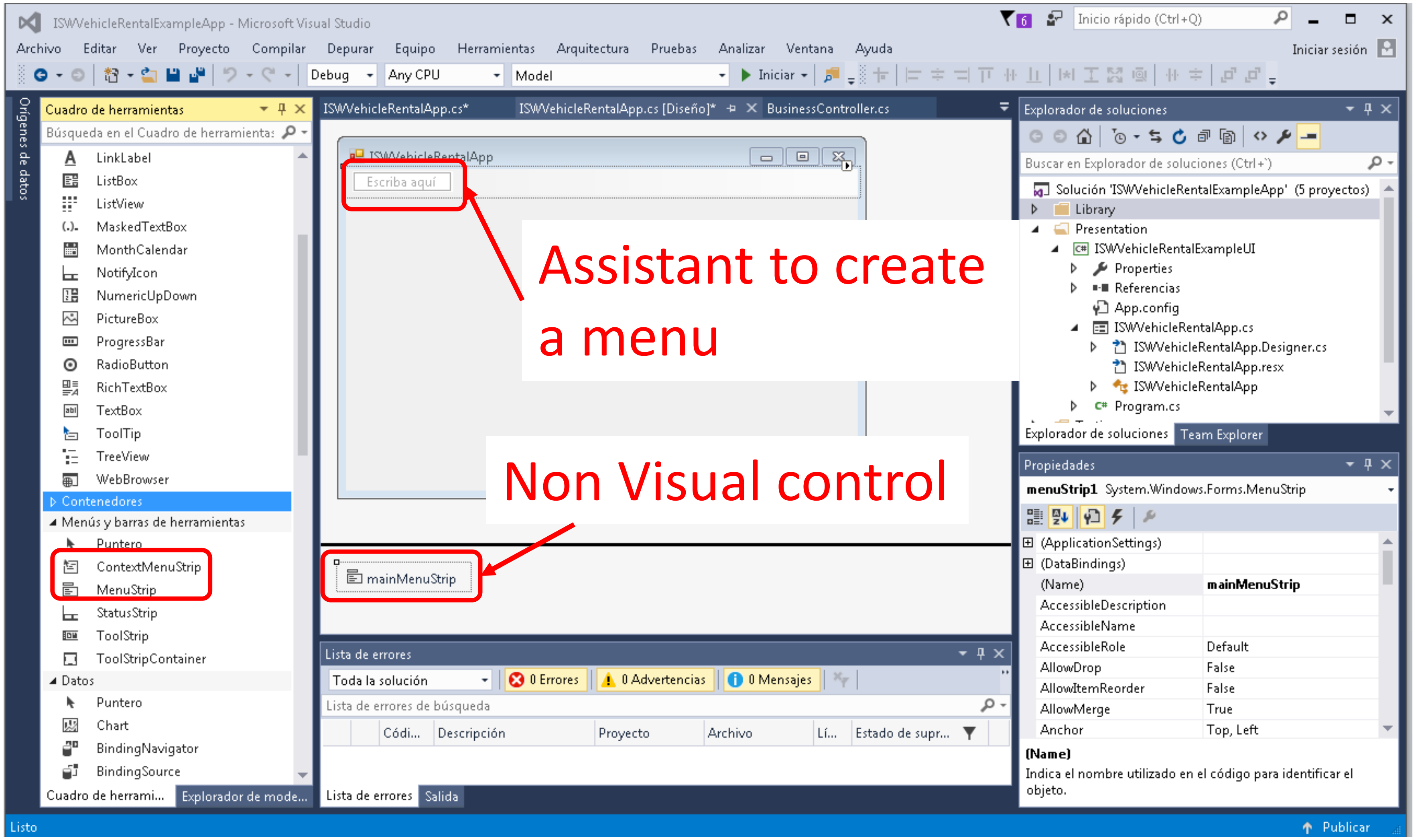
```
1 referencia
private void exitButton_Click(object sender, EventArgs e)
{
    Application.Exit();
}
```

Event information

Designing and using menus

- Most Windows applications have menus
- There are two types of menus:
  - MenuStrip: a main menu
  - ContextMenuStrip: a contextual menu
- All the elements of a menu are stored in the Item property which is a collection of objects belonging to the class ToolStripMenuItem. These elements may contain other submenus.

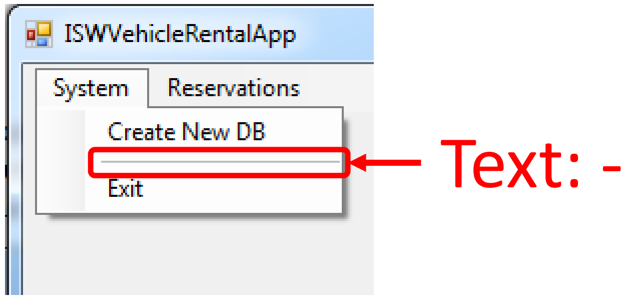
Designing and using menus

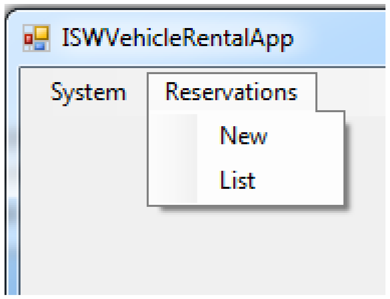


(ApplicationSettings)	
(DataBindings)	
(Name)	mainMenuStrip
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AllowItemReorder	False
AllowMerge	True
Anchor	Top, Left

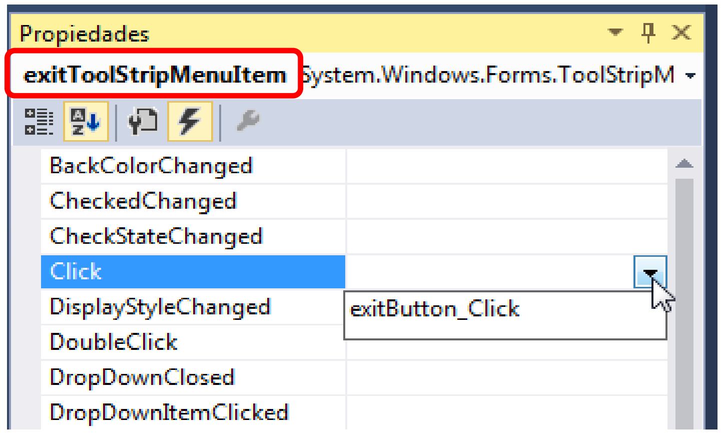
(Name)
Indica el nombre utilizado en el código para identificar el objeto.

Example Menu





- Assigning a handler is done in the same way as with other controls.



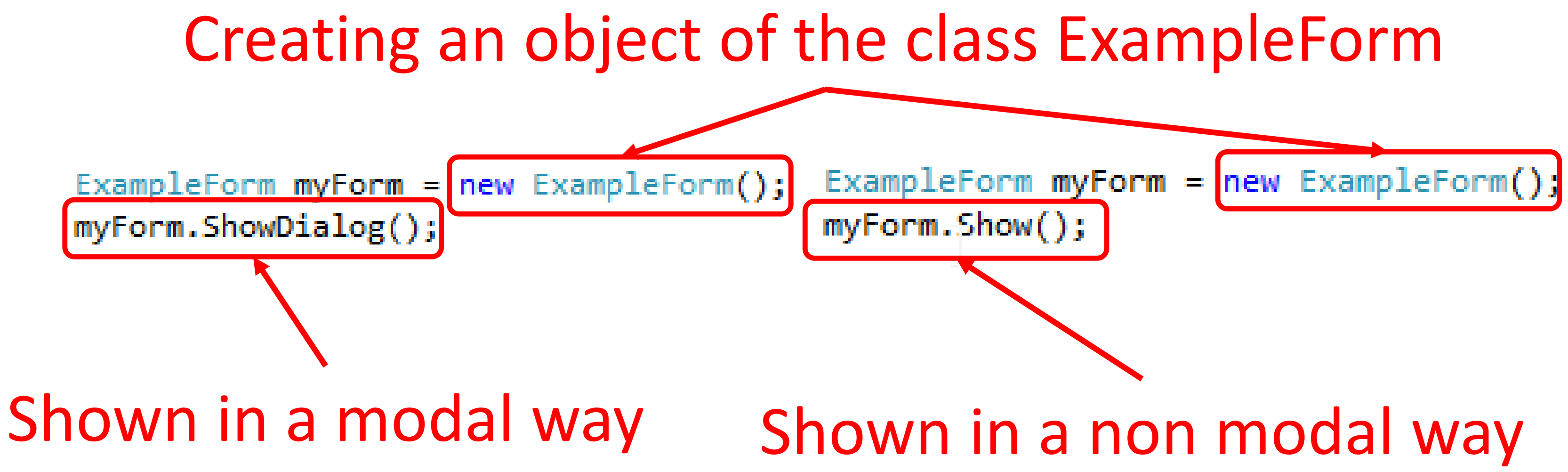
BackColorChanged	
CheckedChanged	
CheckStateChanged	
Click	
DisplayStyleChanged	exitButton_Click
DoubleClick	
DropDownClosed	
DropDownItemClicked	

Applications with several forms

- Usually several forms are used.
- The predefined aspect of a form is defined by the property FormBorderStyle.
- There are several types of forms:
  - User designed: added to the Project with Proyecto|Agregar Windows Forms.
  - Predefined in the environment: dialog box.

User Defined Forms

- Modal: It must be closed to return to the main form. It is shown using the method ShowDialog().
- Non Modal: several forms may be used simultaneously. Shown using the method Show().

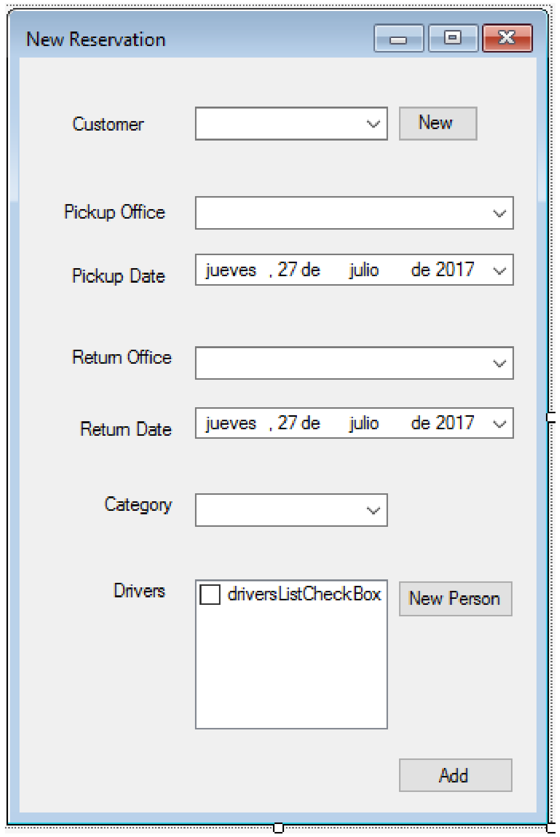


```
ExampleForm myForm = new ExampleForm();
myForm.ShowDialog();
```

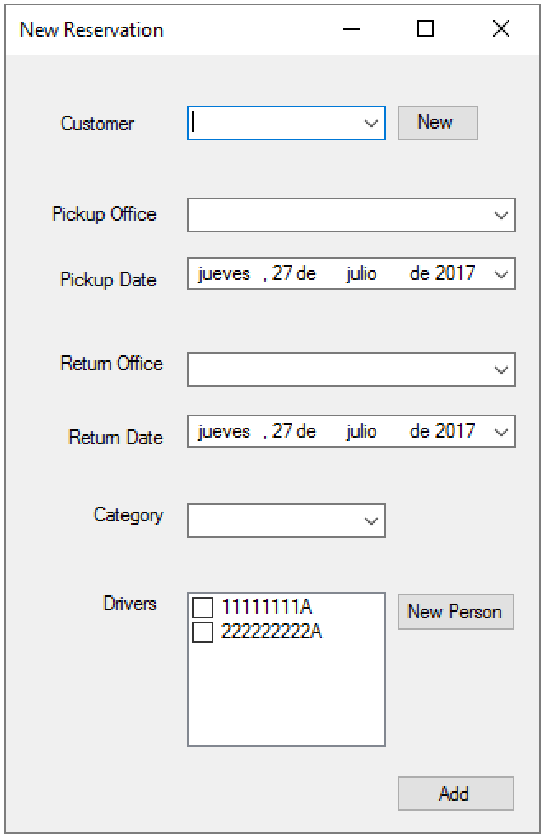
```
ExampleForm myForm = new ExampleForm();
myForm.Show();
```

Forms: Example

- Design view



- Run time view



Forms: Example of Main Form

```
public partial class VehicleRentalApp : Form
{
  private IVehicleRentalService service;
  private NewReservationForm newReservationForm;
  private ListReservationsForm listReservationForm;

  public VehicleRentalApp(IVehicleRentalService service)
  {
    InitializeComponent();
    listReservationForm = new ListReservationsForm(service);
    newReservationForm = new NewReservationForm(service);

  }

  private void newToolStripMenuItem_Click(object sender, EventArgs e)
  {
    newReservationForm.ShowDialog();
  }
…
```

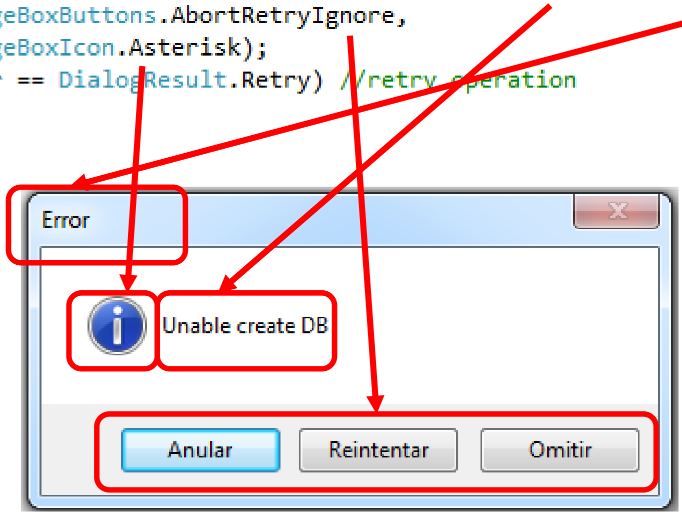
Passing parameters in constructor

New form is shown “Modal”

Dialog boxes

- The class **MessageBox** provides simple dialog boxes and modal behavior.
- The title, the descriptive message and the icon may be customized using the Show method

```
DialogResult answer = MessageBox.Show(this, "Unable create DB", "Error",
    MessageBoxButtons.AbortRetryIgnore,
    MessageBoxIcon.Asterisk);
if (answer == DialogResult.Retry) //retry operation
{ }
```

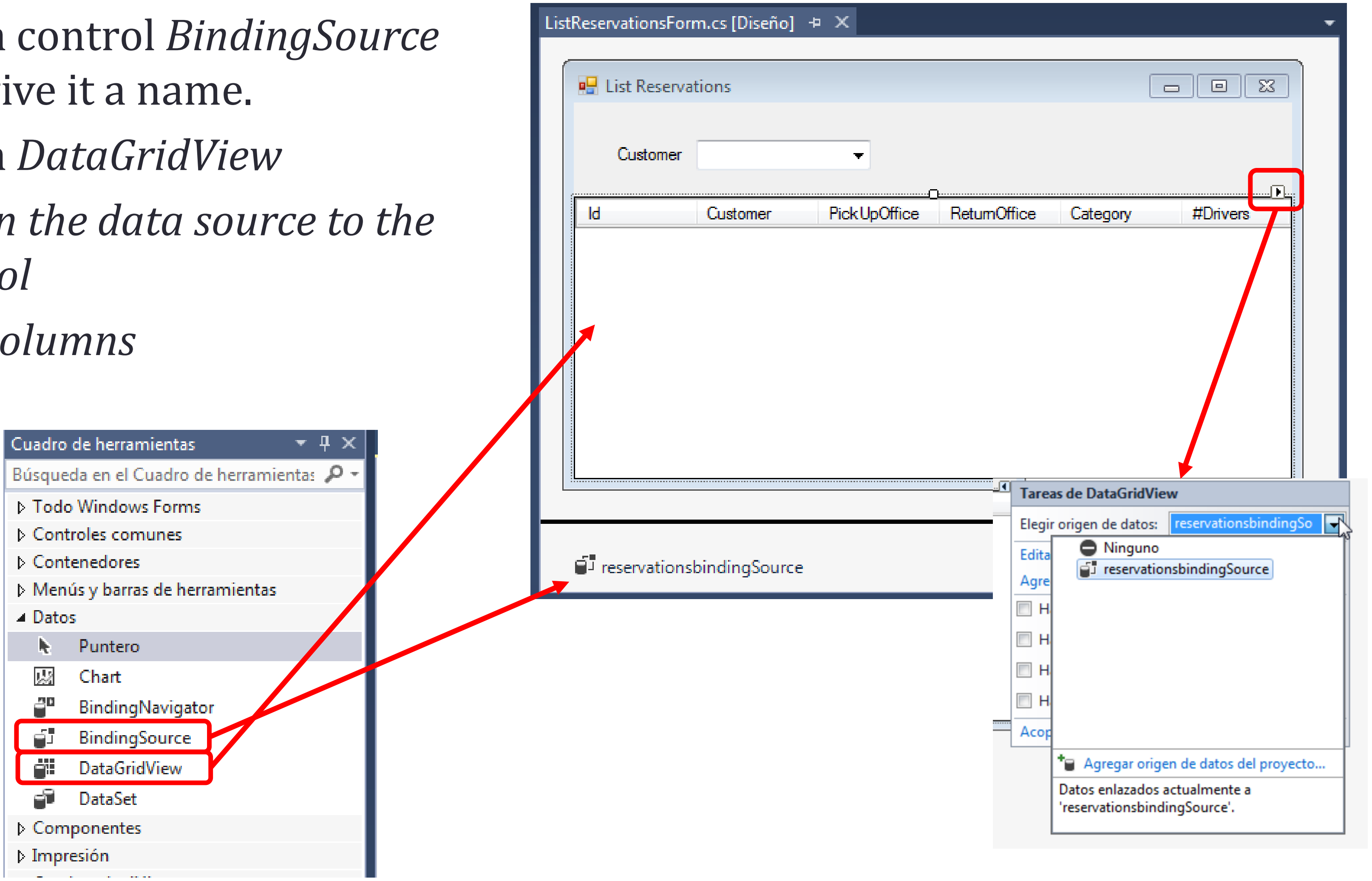


Dialog Boxes

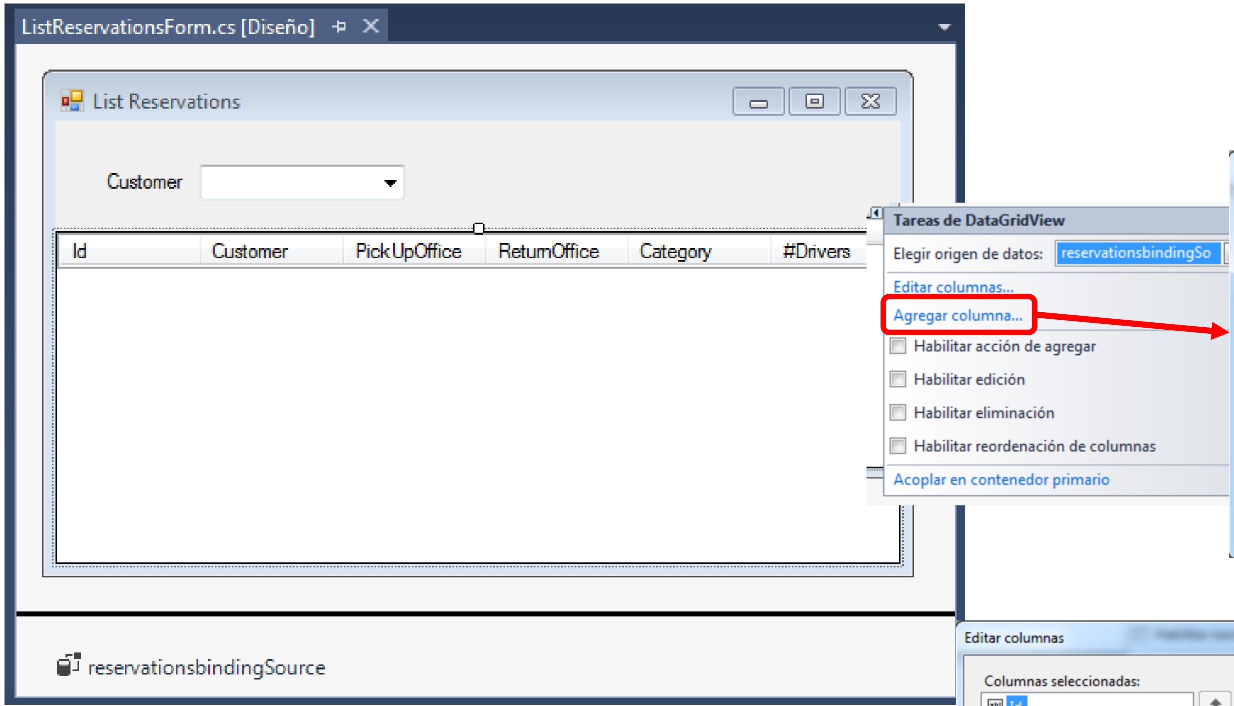
- **Standard Dialog Boxes**
  - These allow carrying out operations such as opening and storing files, printing, selecting colors, etc: ***OpenFileDialog, SaveFileDialog, FolderBrowserDialog, ColorDialog, FontDialog, PageSetupDialog*** and ***PrintDialog***.
  - Inherit from the class CommonDialog. The most imporant method is ShowDialog(), that shows the form and returns an object DialogResult :
    - DialogResult.OK if the user clicks the OK button
    - DialogResult.CANCEL otherwise.

Displaying Data Sets

1. Add a control *BindingSource* and give it a name.
2. Add a *DataGridView*
3. *Assign the data source to the control*
4. *Add columns*

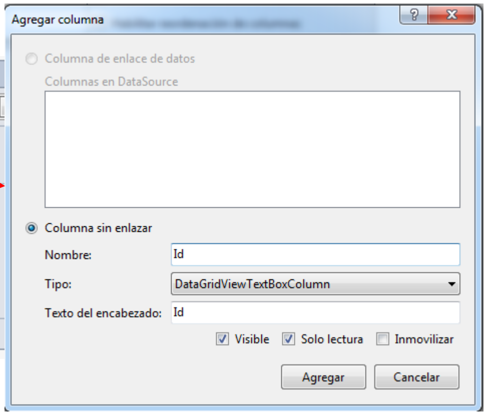

ListReservationsForm.cs [Diseño]
List Reservations
Customer
Id
Customer
PickUpOffice
ReturnOffice
Category
#Drivers
reservationsbindingSource
Tareas de DataGridView
Elegir origen de datos: reservationsbindingSo
Ninguno
reservationsbindingSource
Agregar origen de datos del proyecto...
Datos enlazados actualmente a 'reservationsbindingSource'.
Cuadro de herramientas
Búsqueda en el Cuadro de herramientas
Todo Windows Forms
Controles comunes
Contenedores
Menús y barras de herramientas
Datos
Puntero
Chart
BindingNavigator
BindingSource
DataGridView
DataSet
Componentes
Impresión

Displaying data sets



Elegir origen de datos: reservationsbindingSo

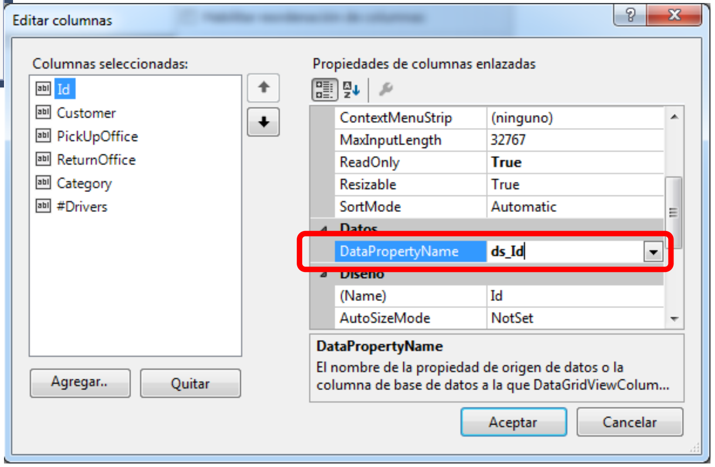
- [ ] Habilitar acción de agregar
- [ ] Habilitar edición
- [ ] Habilitar eliminación
- [ ] Habilitar reordenación de columnas



● Columna sin enlazar

Nombre:	Id
Tipo:	DataGridViewTextBoxColumn
Texto del encabezado:	Id

☑ Visible ☑ Solo lectura ☐ Inmovilizar

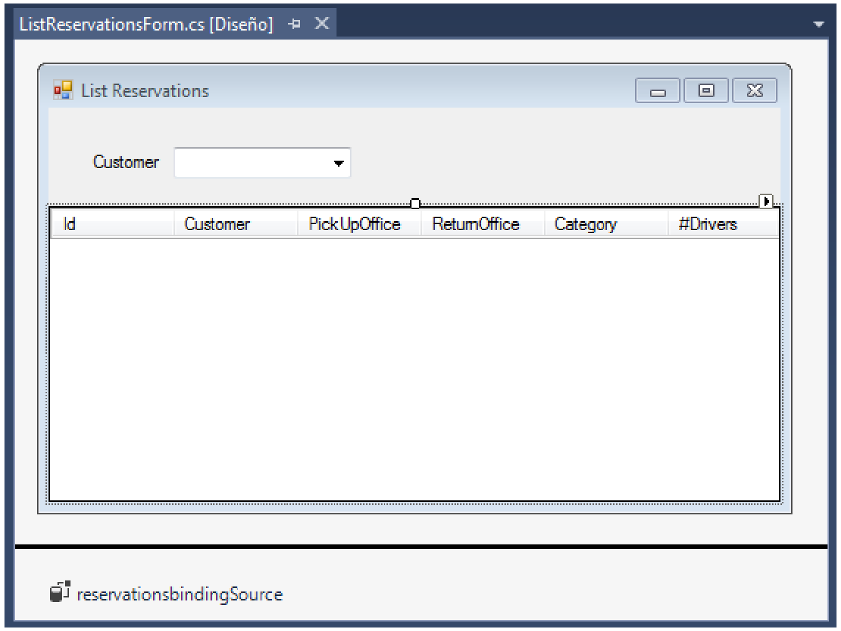


ContextMenuStrip	(ninguno)
MaxInputLength	32767
ReadOnly	**True**
Resizable	True
SortMode	Automatic
Datos	
DataPropertyName	ds_Id
Diseño	
(Name)	Id
AutoSizeMode	NotSet

DataPropertyName
El nombre de la propiedad de origen de datos o la columna de base de datos a la que DataGridViewColum...

After adding columns they must be edited to assign the name of the property in the data source.

Displaying data sets



Functionality

1. When the form is shown a Customer may be selected.
2. After selecting the customer the information is displayed in the *DataGridView.*

Displaying data sets

- When the form is created the *ComboBox* is populated.

  The method *LoadData* populates the ComboBox *customersComboBox:*

```
 public ListReservationsForm(IVehicleRentalService service) : base(service)
{
    InitializeComponent();
    LoadData();
}

public void LoadData()
{
    ICollection<Customer> customers = service.findAllCustomers();
    customersComboBox.Items.Clear();
    if (customers!=null)
    foreach (Customer c in customers)
           customersComboBox.Items.Add(c.Dni);
    customersComboBox.SelectedIndex = -1;
    customersComboBox.ResetText();
    reservationsbindingSource.DataSource = null;
}
```

Displaying data sets

When an element is selected in the *ComboBox* the *DataGridView* is populated.

The event handler *SelectedIndexChanged* of the *ComboBox* object is executed.

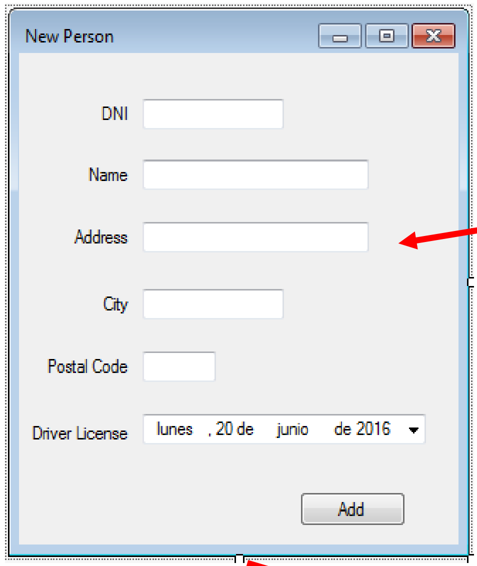
```
private void customersComboBox_SelectedIndexChanged(object sender, EventArgs e)
{
    string dni = (string) customersComboBox.SelectedItem;
    ICollection<Reservation> reservations = service.findReservationsbyCustomerID(dni);

//A BindingList of anonymous objects is used to provide the data model to the DataGrid

    BindingList<object> bindinglist = new BindingList<object>();
    foreach (Reservation r in reservations)
    //Adding one anonymous object for each reservation obtained
    bindinglist.Add(new
        {
          //ds_... are DataPropertyNames defined in the DataGridView object
          //see DataGridView column definitions in Visual Studio Designer
            ds_Id = r.Id,
            ds_Customer = r.Customer.Name,
            ds_PickUpOffice = r.PickUpOffice.Address,
            ds_ReturnOffice = r.ReturnOffice.Address,
            ds_Category = r.Category.Name,
            ds_NumDrivers = r.Drivers.Count
        });
    reservationsbindingSource.DataSource = bindinglist;
}
```

Advanced Operations: Visual Inheritance

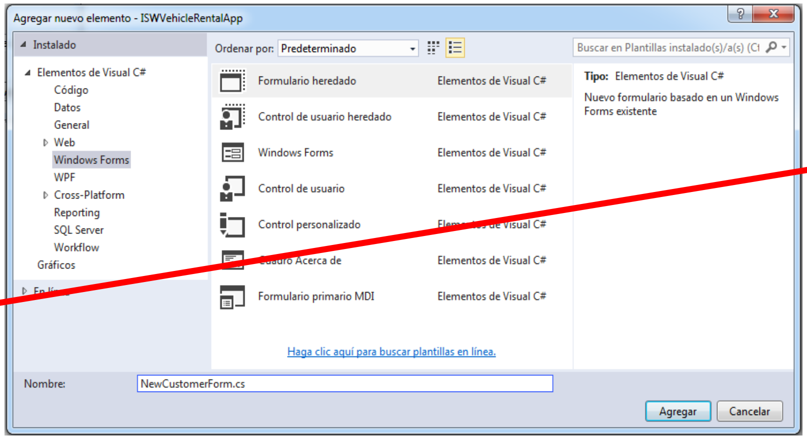
Forms may inherit from other forms so that the behavior and visual appeareance is reused



New Person

DNI	
Name	
Address	
City	
Postal Code	
Driver License	lunes , 20 de junio de 2016

After compiling a form



Agregar nuevo elemento - ISWVehicleRentalApp

Instalado
- Elementos de Visual C#
  - Código
  - Datos
  - General
  - Web
  - Windows Forms
  - WPF
  - Cross-Platform
  - Reporting
  - SQL Server
  - Workflow
- Gráficos

Ordenar por: Predeterminado

Buscar en Plantillas instalado(s)/a(s) (Ct

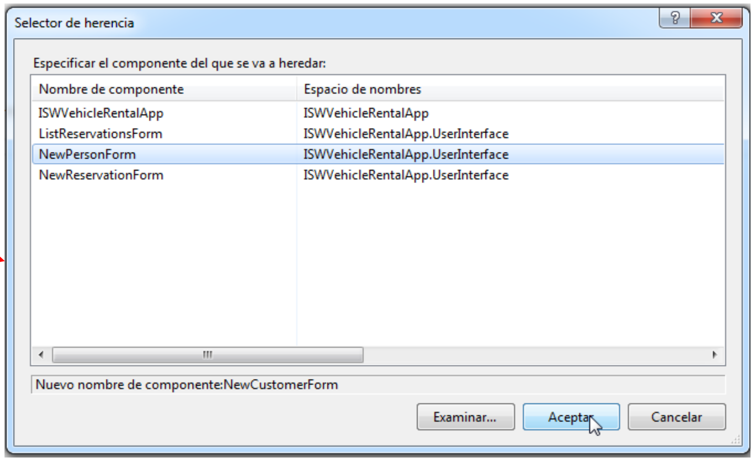
Formulario heredado	Elementos de Visual C#
Control de usuario heredado	Elementos de Visual C#
Windows Forms	Elementos de Visual C#
Control de usuario	Elementos de Visual C#
Control personalizado	Elementos de Visual C#
Cuadro Acerca de	Elementos de Visual C#
Formulario primario MDI	Elementos de Visual C#

Tipo: Elementos de Visual C#

Nuevo formulario basado en un Windows Forms existente

Haga clic aquí para buscar plantillas en línea.

Nombre: NewCustomerForm.cs

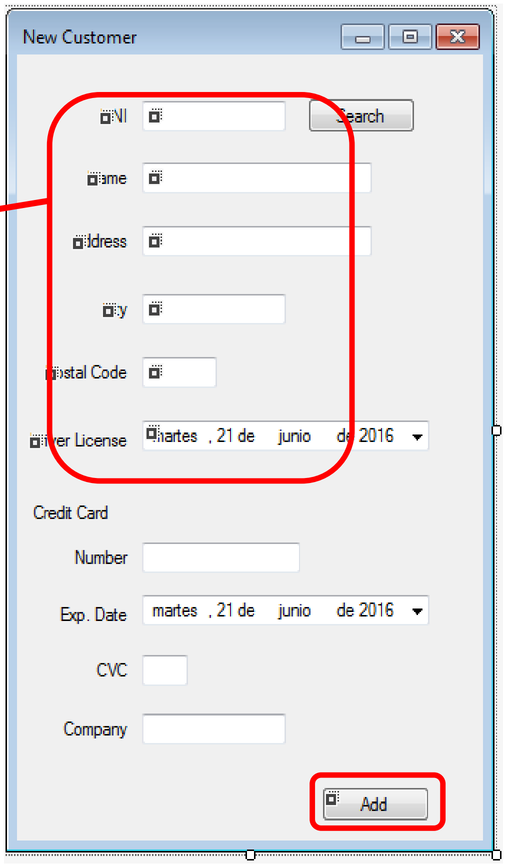


Selector de herencia

Especificar el componente del que se va a heredar:

Nombre de componente	Espacio de nombres
ISWVehicleRentalApp	ISWVehicleRentalApp
ListReservationsForm	ISWVehicleRentalApp.UserInterface
NewPersonForm	ISWVehicleRentalApp.UserInterface
NewReservationForm	ISWVehicleRentalApp.UserInterface

Nuevo nombre de componente:NewCustomerForm



New Customer

DNI	
Name	
Address	
City	
Postal Code	
Driver License	martes , 21 de junio de 2016

Credit Card

Number	
Exp. Date	martes , 21 de junio de 2016
CVC	
Company	