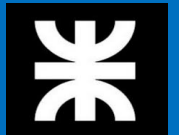


Unidad 3

Desarrollo de Aplicaciones de Escritorio



Contenidos

- ¿Qué es Windows Forms?
- Controles
- Manejo de Layout
- GDI+
- Conceptos Básicos, Proyecto, Controles
- Controles en Tiempo de Ejecución
- Diseñador de Formularios y objeto Form
- Mas Controles
- ClickOnce

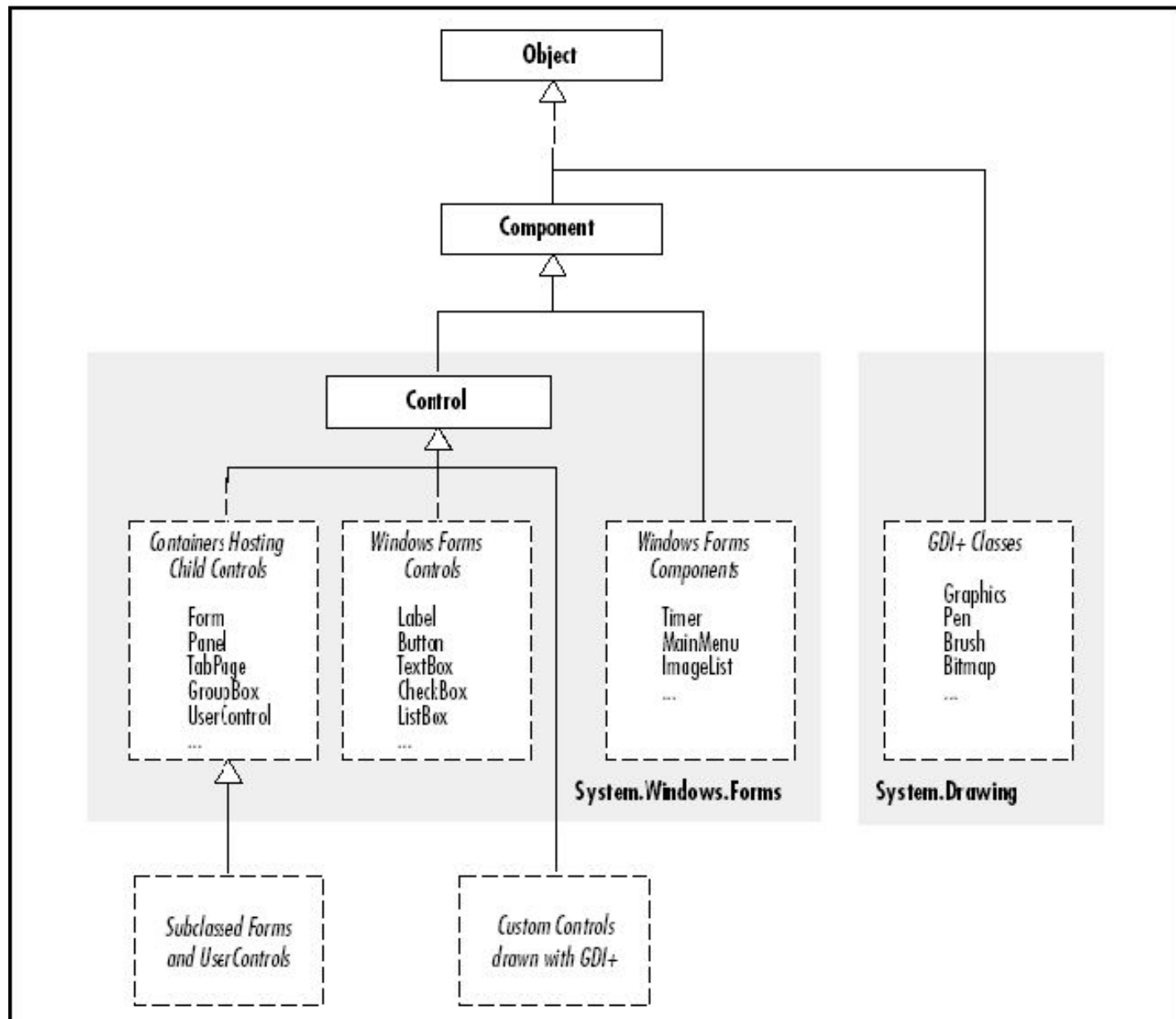
¿Qué es WinForms ?

- Conjunto de clases que permite desarrollar aplicaciones basadas en Windows
- Interfaz de usuario rica
- Incluye
 - Acceso al framework de .NET
 - Ejecución en un entorno manejado
 - Seguridad integrada
 - Diseño OOP
 - Orientado a Eventos
 - Aplicaciones orientadas a datos con ADO.NET
 - Fácil acceso a Servicios Web
 - Elección de lenguaje (VB.Net, C#, etc)

¿Qué es WinForms ?

- Componentes .NET para el desarrollo de aplicaciones de escritorio:
 - Windows Forms (componentes para gestionar ventanas y controles)
 - Interfaz de dispositivos gráficos (GDI+), constituido por las funciones que encapsulan las funcionalidades graficas de bajo nivel)
 - Windows Presentation Foundation (segunda generación de aplicaciones de escritorio)

Jerarquia de Objetos



¿Qué es WinForms ?

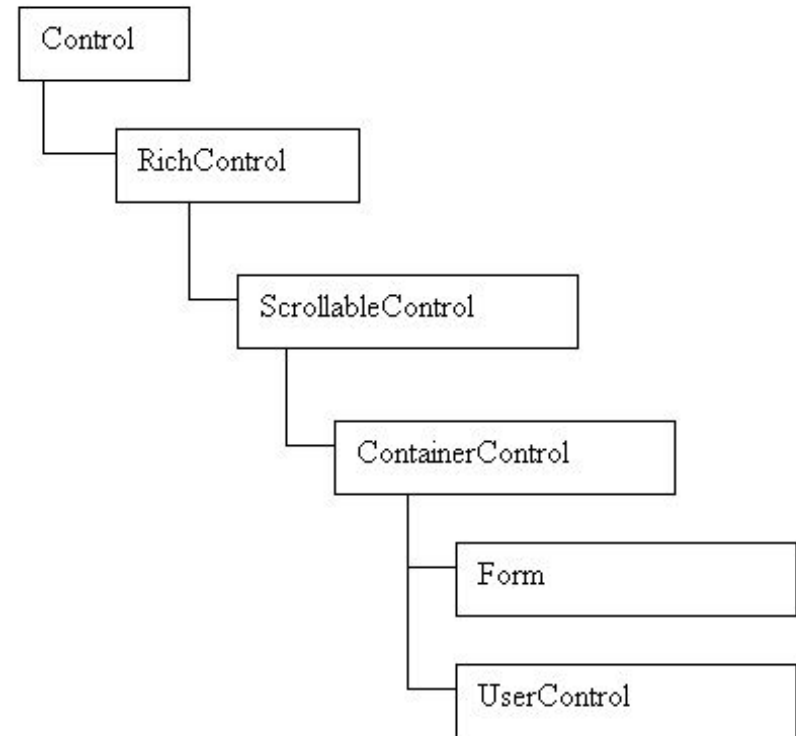
Clase	¿Qué hace?	¿Por qué es necesario?
Object	Clase básica de todos los tipos en el entorno .NET	Proveer funcionalidad básica para todos los objetos, además de ser la base de un sistema único de tipos.
Component	Funcionalidad básica de los contenedores, facilitando la tarea de albergar elementos en un entorno gráfico. Establece un protocolo de disponibilidad de los recursos	Albergar un amplio tipo de controles y componentes de una manera genérica, además de constituir la base para componentes no visuales.
Control	Funcionalidad básica de un control visual que responde a eventos de ratón y teclado, acepta el concepto de foco (ser el foco de acción), además de participar en operaciones drag & drop.	Superclase de todos los controles existentes (ej. TextBox, Label, Button, etc.), además de proveer de la base para componentes personalizados.
Form	Define una clase que representa la ventana Windows a la que podemos añadir controles	Clase base para crear nuestras propias ventanas con una funcionalidad de contenedor ya incluida (que puede ser extendida).

¿Qué es WinForms ?

- Crear una aplicación Windows Forms, es tan fácil como instanciar y extender los formularios de Windows
- Proceso:
 - Crear un nuevo proyecto definiendo la estructura de una aplicación basada en Windows Forms.
 - Definir los Windows Forms necesarios.
 - Mediante el diseñador, personalizarlos añadiendo controles según el propósito, configurando los manejadores para la gestión de los eventos.
 - Agregar nuestro código para personalizar el comportamiento y la apariencia.
 - Diseñar nuestros propios controles.

Todos son Controles

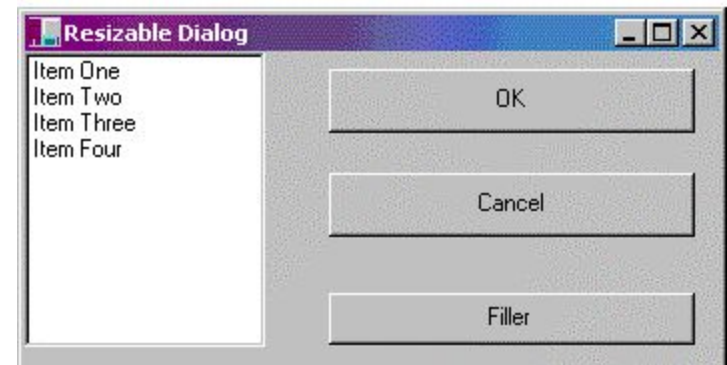
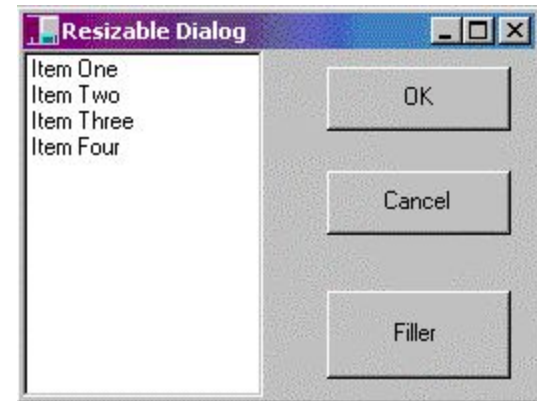
- Todos los controles derivan de Control
- La mayoría de los controles estan en System.Windows.Forms



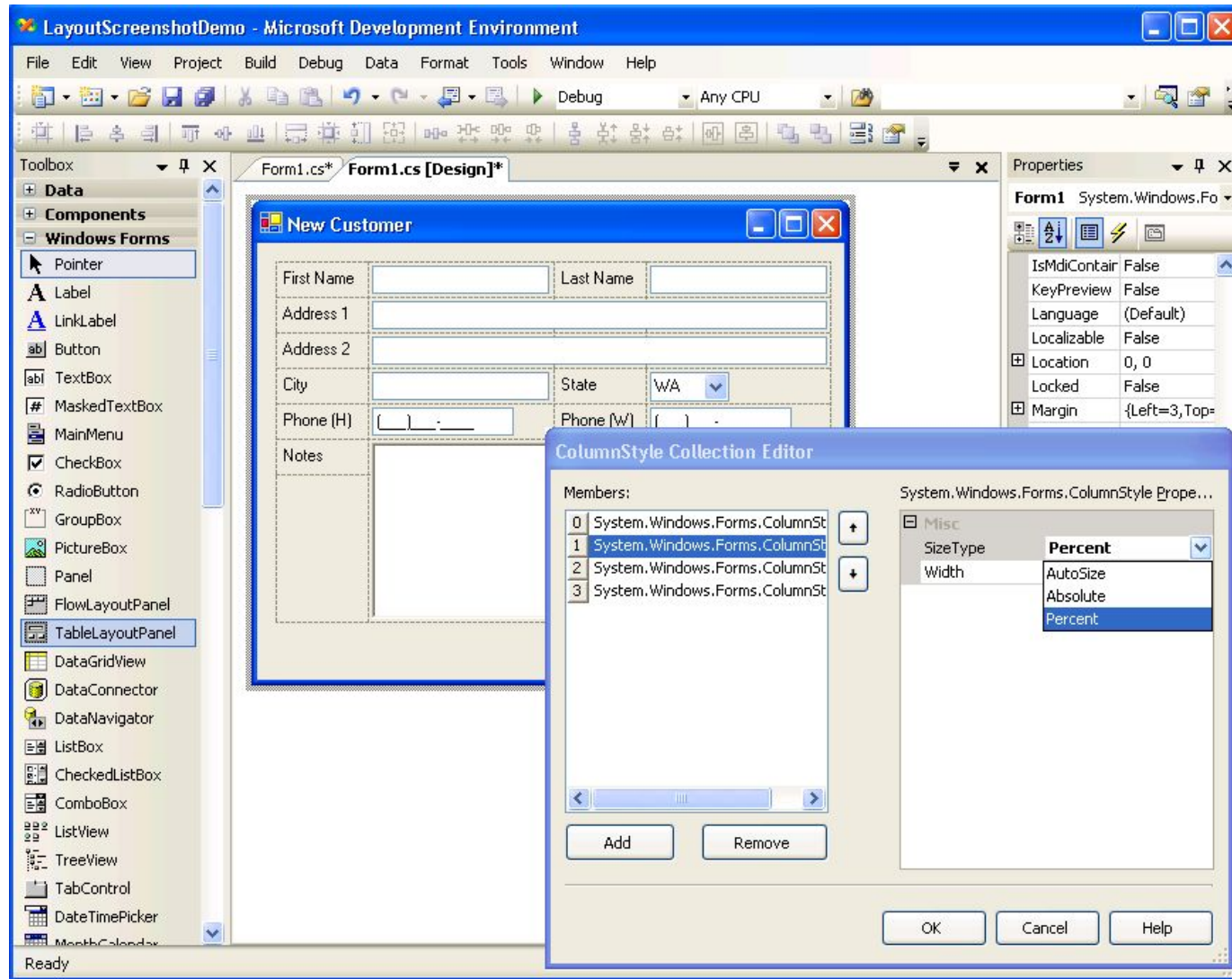
Manejo de Layout

- Layout difícil de manejar en otras plataformas (Swing por ejemplo) se hace muy sencillo en Windows Forms
- Inclusive WPF es más complejo (aunque también más avanzado)

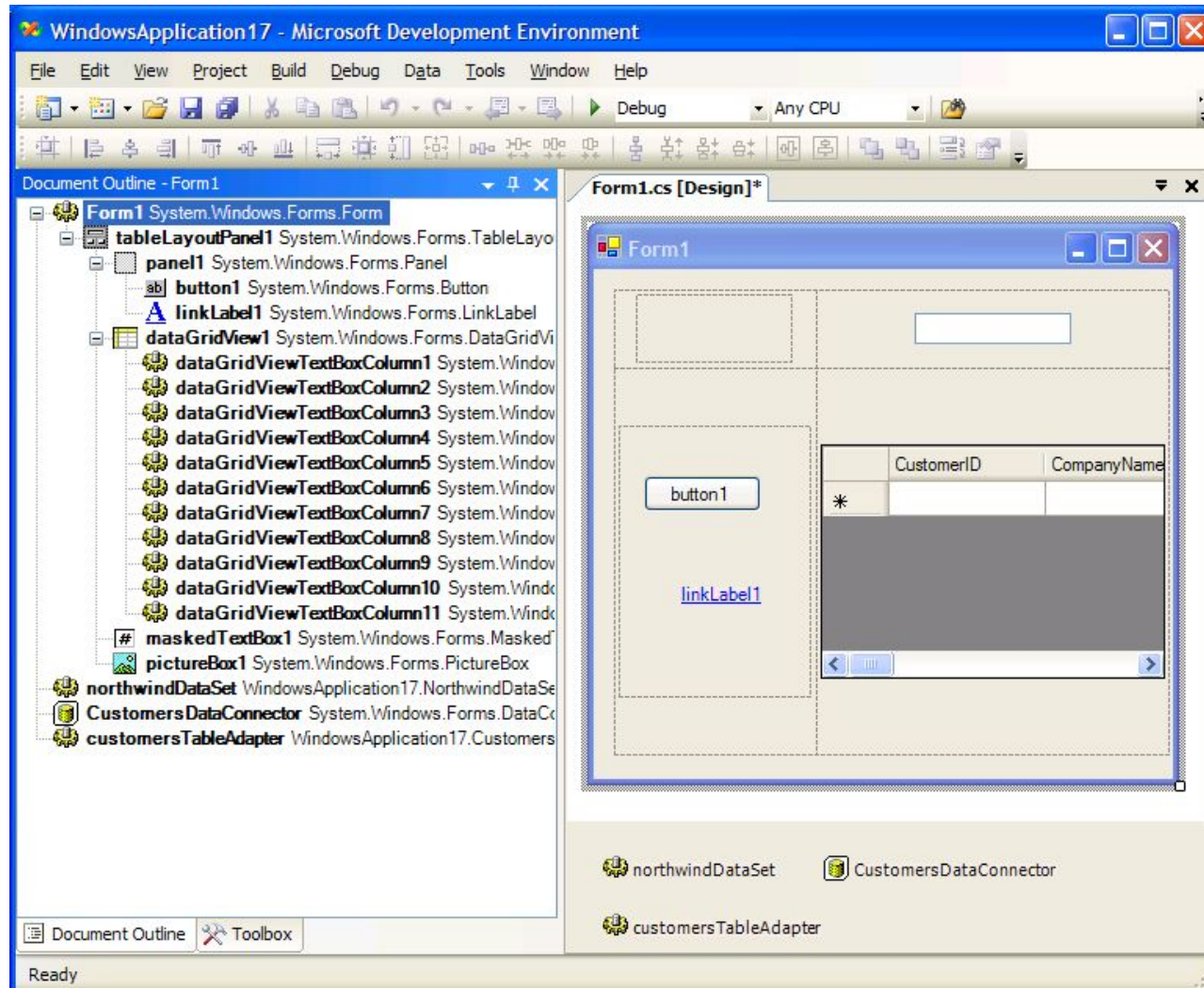
- Windows Forms provee
 - Anchoring
 - Docking
 - Soporte total en tiempo de diseño



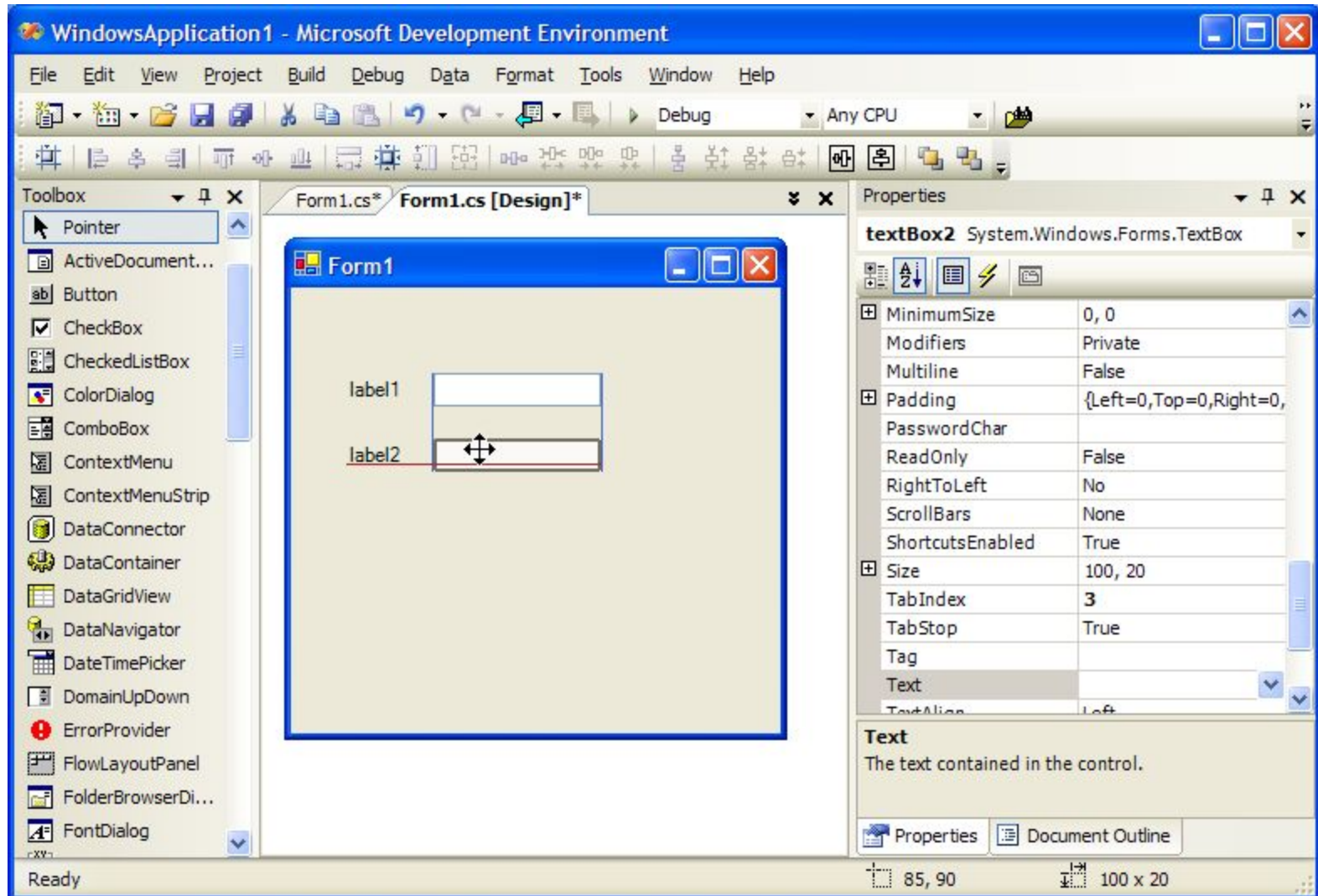
Manejo de Layout



Document Outline



SnapLines



- Sistema grafico 2-D de Microsoft
- Windows Forms saca provecho 100% de GDI+
- Incluye
 - Posibilidad de dibujo (System.Drawing)
 - Pens, Brushes, Images
 - Alpha blending
 - Color gradients
 - Imagenes en varios formatos a parte de Bitmap



Conceptos Básicos

- Creando un proyecto
- ¿Con que nos encontramos?
- InitializeComponent
- Tipos de aplicaciones
- Abriendo Forms
- Controles de Winforms
- App.config

Creando Proyecto WinForm

- Creando un proyecto:
 - File -> New -> Project -> Windows Application
- Nos encontramos con:
 - Se crea un formulario principal
 - Con ese form empieza la aplicacion
 - No es un setting mágico del proyecto
 - `Application.Run(new Form1())`

IntializeComponent

- Metodo agregado por VS.Net
- Toda la creación de los controles sobre la superficie de diseño, el posicionamiento, etc.
- Es código generado y se mantiene totalmente sincronizado

Tipos de aplicaciones

- SDI (single document interface)
 - Un formulario solo activado
- MDI (multiple document interface)
 - Un formulario padre y muchos hijos
 - Setear la propiedad IsMDIContainer en el formulario padre
 - Setear IsMDIChild a true en los formularios hijos

Abriendo Forms

- Dos maneras:
 - `form.Show()` no-modal
 - `form.ShowDialog()` modal

Controles Windows Form

- Categorias

- Action Controls

- Button, Toolbar, MenuBar, ContextMenu, etc

- Value Controls

- Label, TextBox, PictureBox, etc

- List Controls

- ListBox, DataGrid, TreeView, ComboBox, etc

- Container Controls

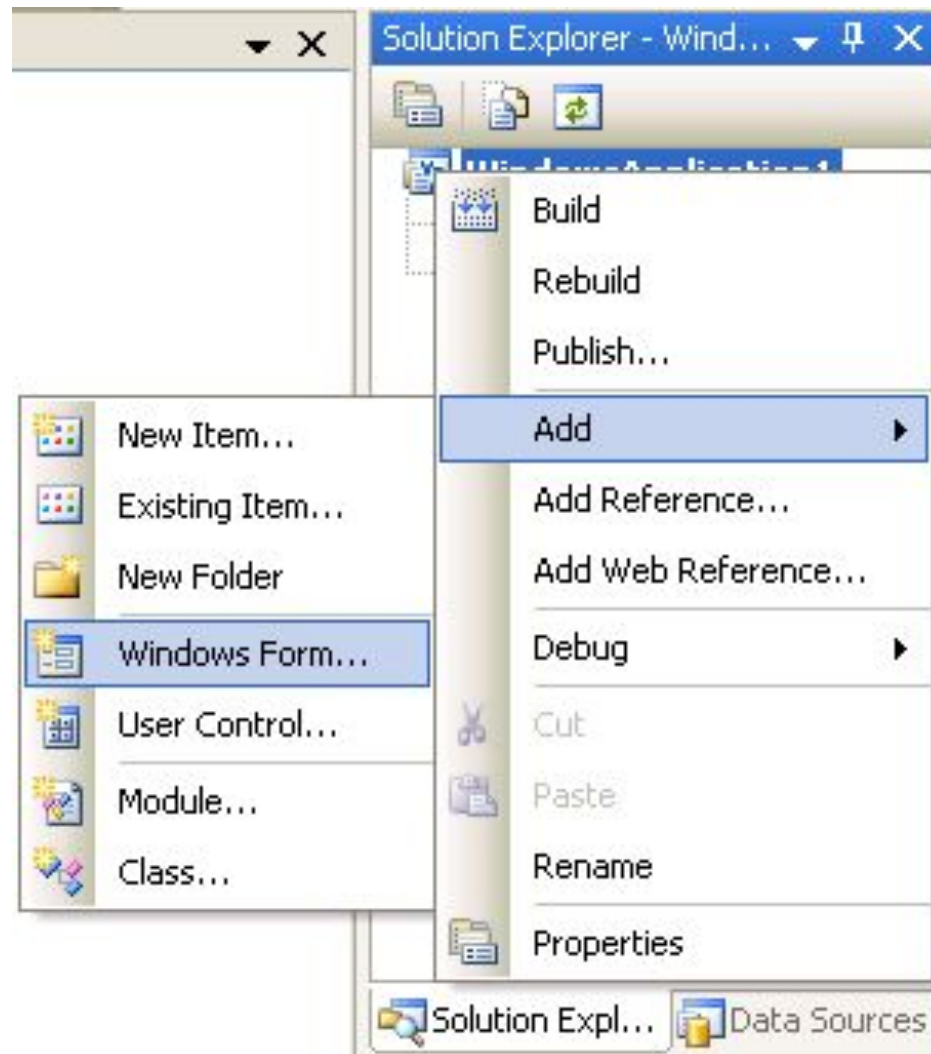
- GroupBox, Panel, ImageList, etc

- Dialogs

- OpenFileDialog, SaveFileDialog, PrintDialog, etc

- Archivo de configuración
- Evolucion del archivo INI
- Formato XML
 - <appSettings />: parametros generales
 - <connectionStrings />: connection strings

Cómo Crear un Formulario



Cómo Asignar Propiedades

Nombre del Objeto

Botón para Lista por Categorías

Botón para Lista Alfabética

Panel de Descripción

Properties

Form1 System.Windows.Forms.Form

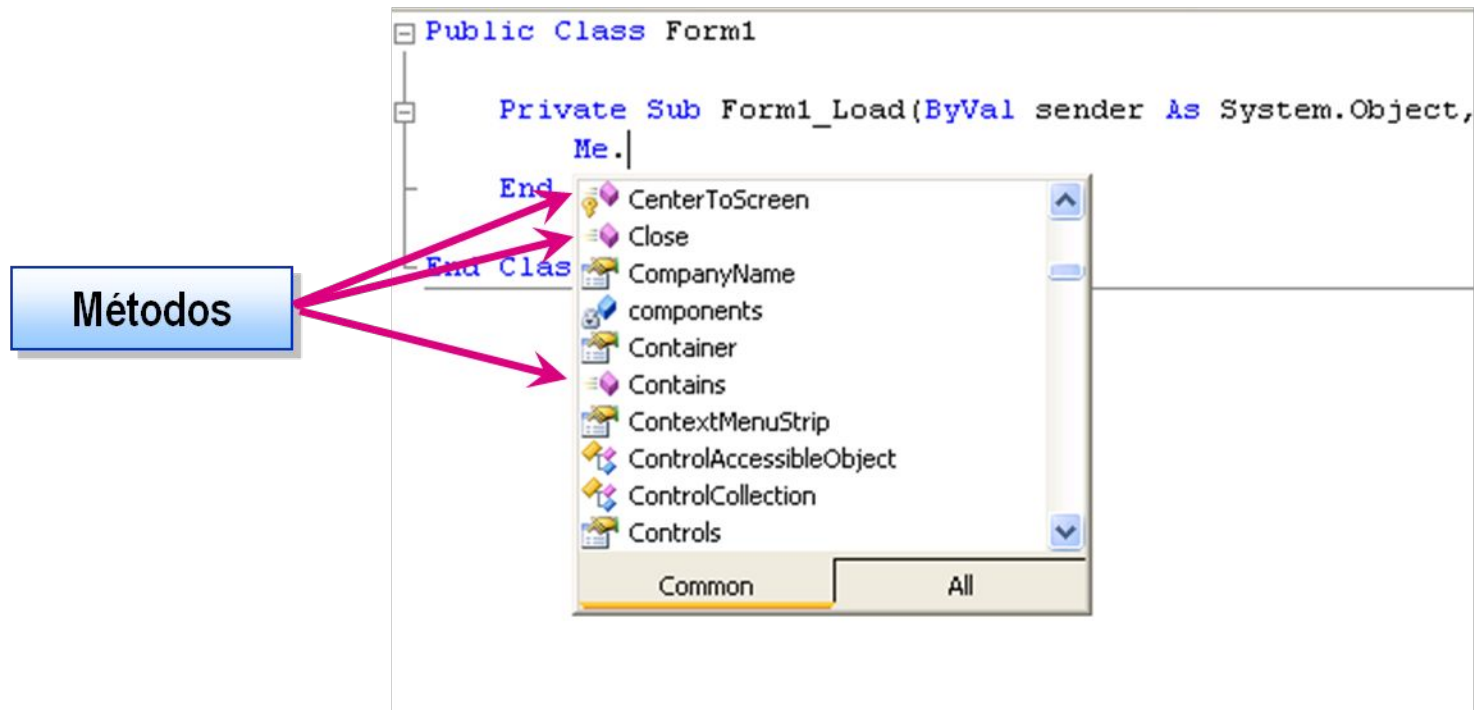
(ApplicationSettings)

(DataBindings)

(Name)	Form1
AcceptButton	(none)
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AutoScaleMode	Font
AutoScroll	False
AutoScrollMargin	0; 0
AutoScrollMinSize	0; 0
AutoSize	False
AutoSizeMode	GrowOnly
AutoValidate	EnablePreventFocusChange
BackColor	<input type="color"/> Control
BackgroundImage	<input type="image"/> (none)
BackgroundImageLayout	Tile
CancelButton	(none)

(Name)
Indicates the name used in code to identify the object.

Cómo Invocar Métodos

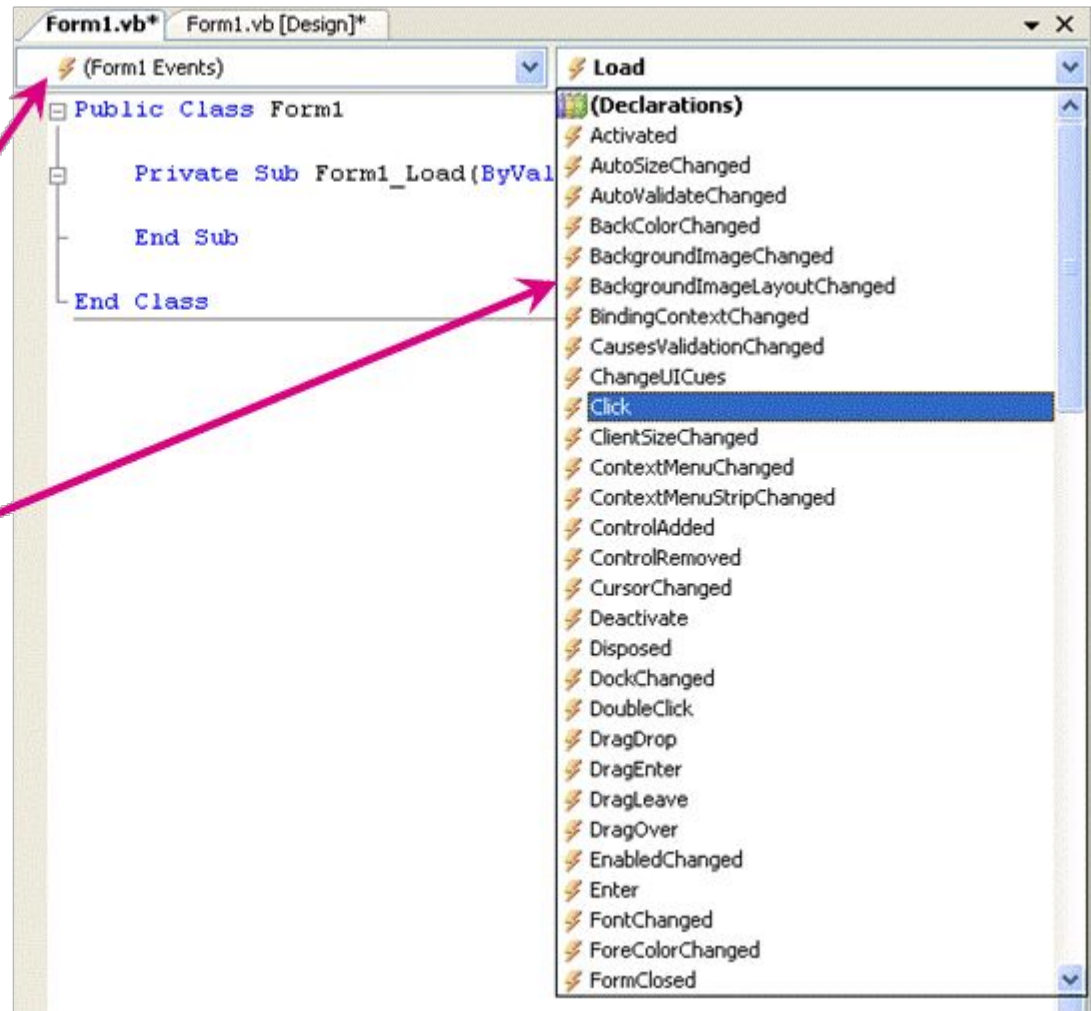


```
Sub Form1_Click  
    Me.CenterToScreen( )  
End Sub
```

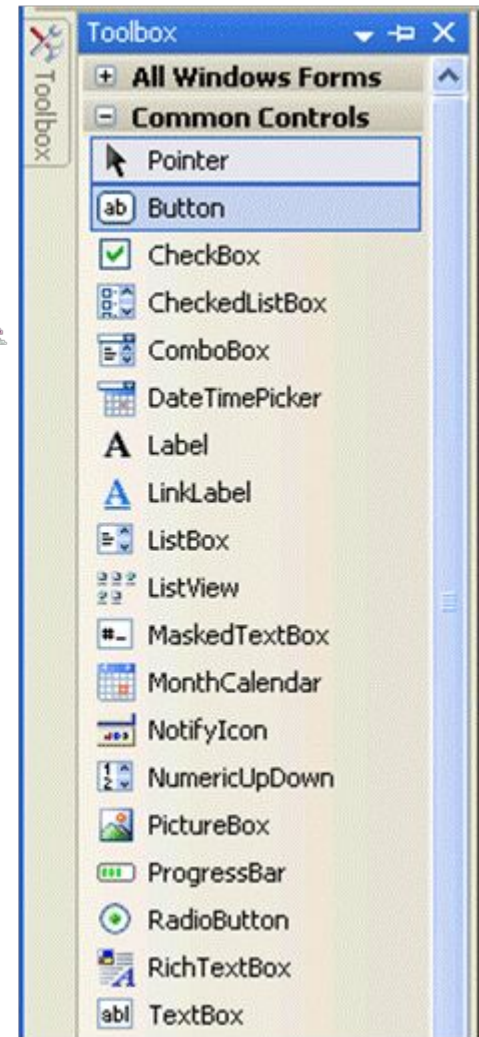
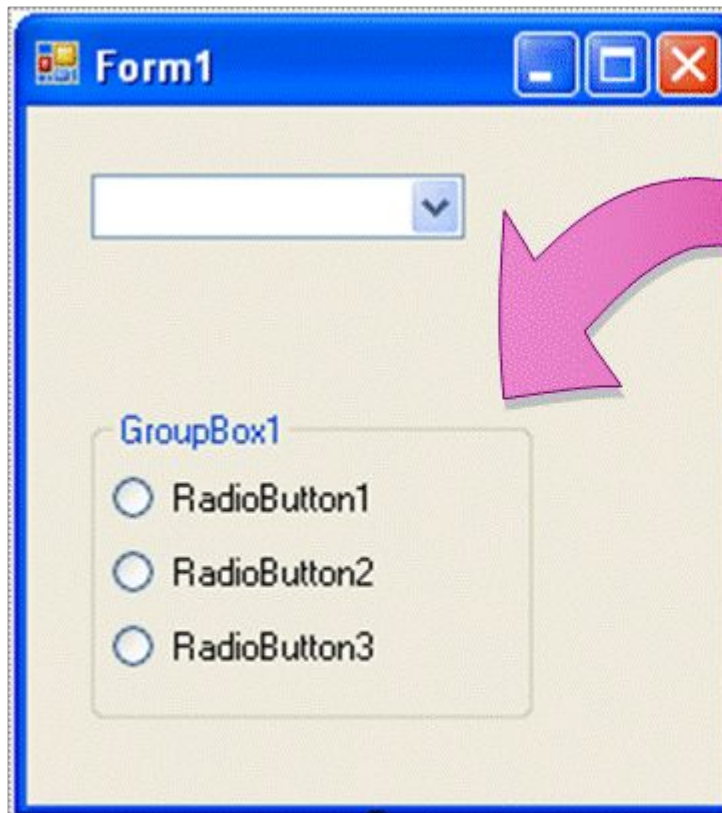
Cómo Manejar Eventos

Lista de
Nombres de Clase

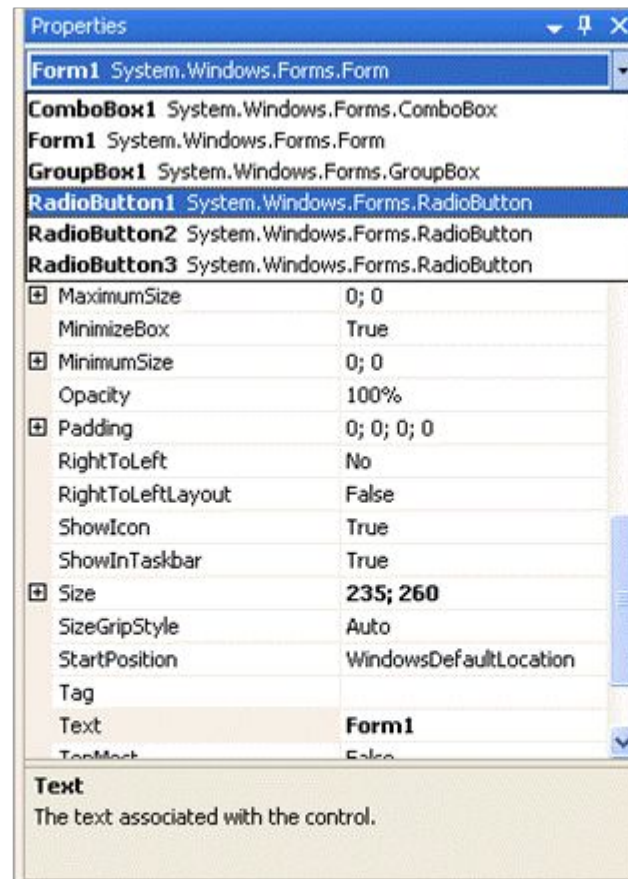
Eventos



Cómo Agregar Controles

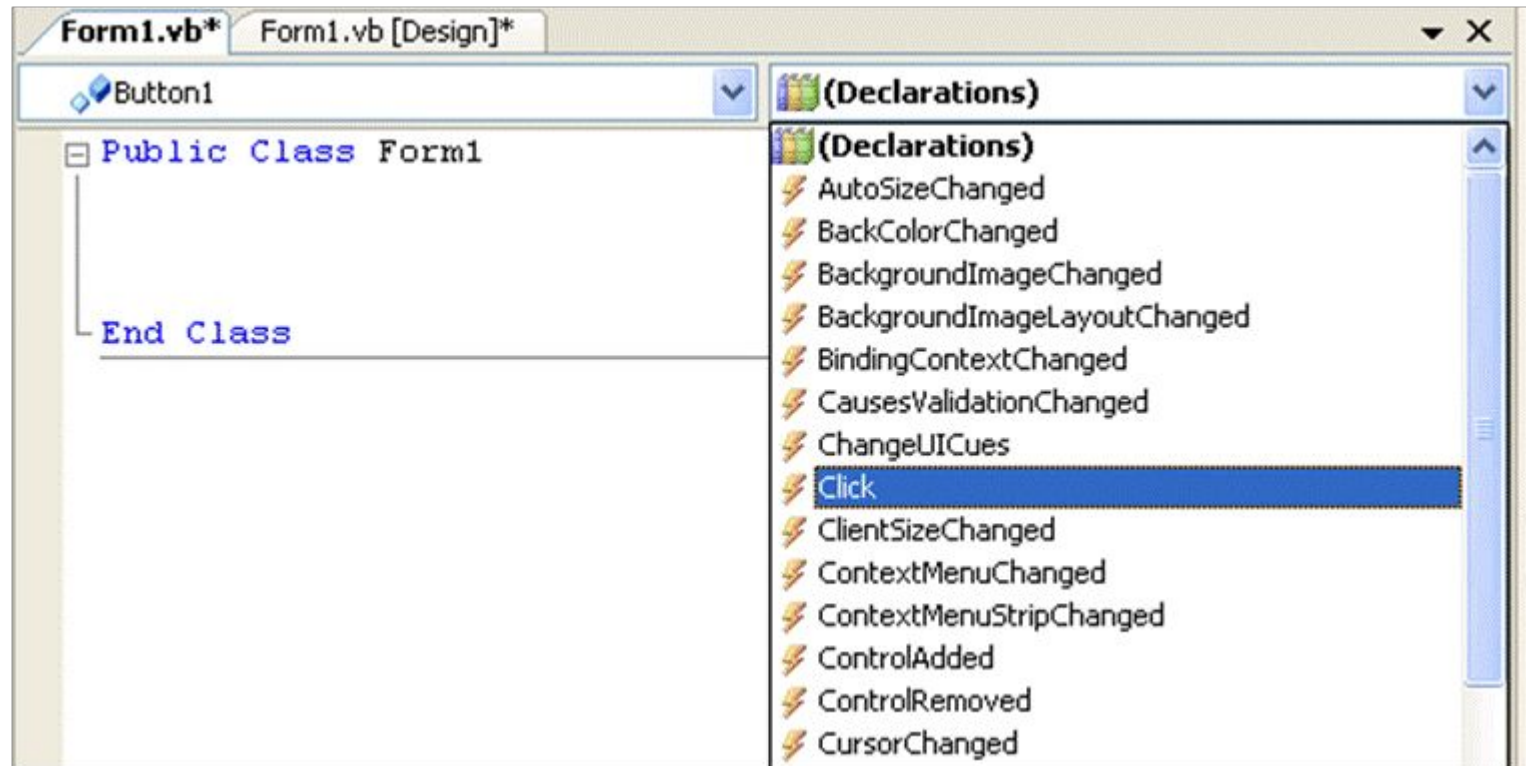


Cómo Asignar las Propiedades

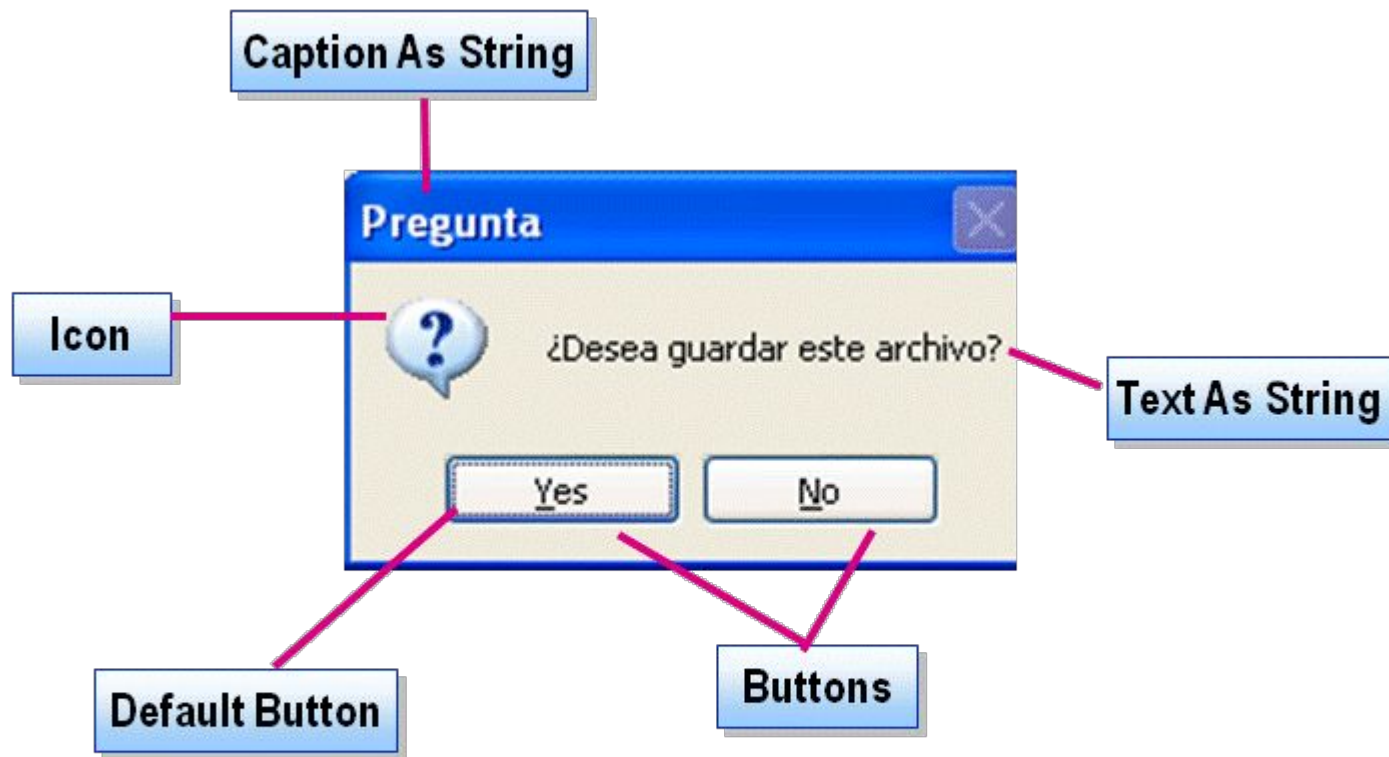


- **Puede asignar el mismo valor de propiedad para múltiples controles al mismo tiempo**

Cómo Agregar Eventos en Control



Cómo usar MessageBox.Show()



```
MessageBox.Show( )
```

Controles en Tiempo de Ejecución

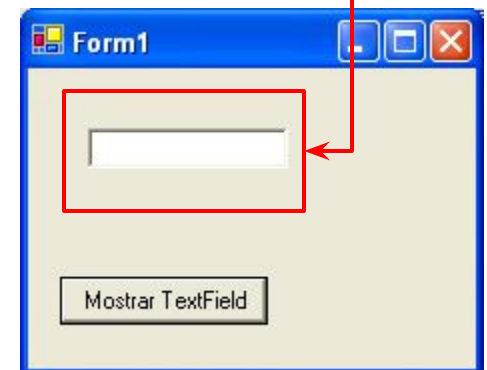
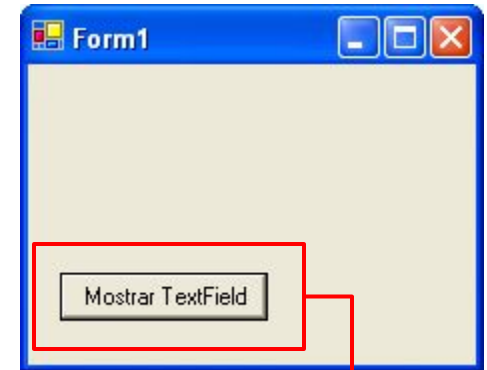
- Los controles también pueden agregarse de manera manual (sin la interfaz del Designer).
- Necesario cuando se debe incluir un componente como resultado de la ejecución de un evento.
- Utilizar como modelo el código generado automáticamente en el Editor de Código.
Concretamente en el método `InitializeComponent`.

Controles en Tiempo de Ejecución

- Agregar de manera manual un componente puede resumirse básicamente en 4 pasos:
 - Añadir un nuevo campo de declaración para el nuevo componente.
 - Instanciar el control.
 - Configurar el control (establecimiento de los valores de las propiedades) y añadir eventos (cuando sean necesarios).
 - Añadirlo a la colección de controles del formulario o a un contenedor que tenga que gestionarlo.

Controles en Tiempo de Ejecución

```
private void button1_Click(object sender, System.EventArgs e)
{
    //Lo instanciamos
    this.nuevoTextBox = new TextBox();
    //lo posicionamos
    this.nuevoTextBox.Location = new Point(30,30);
    //Lo agregamos a la colección de controles del frame
    this.Controls.Add(nuevoTextBox);
}
```



Diseñador de Formularios

- Genera el código del formulario diseñado y de los componentes incluidos en él.
- Partial Class permite separar la implementación de una clase en varios archivos fuente -> separar la lógica de la aplicación (organización)

Objeto Form

- Opacity: transparencia (int 0 - 1).
- Size: tamaño (Size).
- Location: ubicación (Location).
- Show(): mostrar. Permite especificar su propietario.
- ShowDialog(): permite mostrar un diálogo de su propiedad, incluyendo uno modal (no permite tomar el foco por otro formulario mientras se esté mostrando).

Manejo de Eventos

- Por cada evento generado es posible implementar varios métodos manejadores.
- Un método manejador de eventos puede recibir notificaciones de eventos generados en varios objetos.
- Es posible cancelar el tratamiento por defecto del evento manejado con `e.Handled = true`.

Eventos del Form (Ciclo de Vida)

- **Load:** se encuentra recién cargado, aún no se muestra al usuario.
- **Paint:** es dibujado incluyendo a sus controles recursivamente.
- **Activated:** recibe foco.
- **FormClosing:** el proceso de cierre a sido iniciado. En este punto es posible manipularlo y si es necesario, cancelar el cierre.
- **FormClosed:** ha sido cerrado.
- **Disposed:** ha sido destruído.

Eventos del Mouse

- **MouseClicked:** se hizo click con el ratón.
- **MouseEnter:** el puntero del ratón entró a un componente.
- **MouseMove:** el puntero del ratón ha sido movido dentro del componente.

Eventos del Teclado

- **KeyPress:** tecla presionada. Representa una pareja de eventos Up y Down sucesivos. e.KeyChar especifica cual.
- **KeyDown:** tecla hundida.
- **KeyUp:** tecla liberada.
- **HelpRequested:** se presionó F1.

Manipulación del Foco

- **CanFocus:** (true/false) puede el control tomar el foco?
- **Focused:** (true/false) tiene el control el foco?
- **Focus(int):** asigna el foco al componente especificado.
- **TabIndex:** especifica el orden de movimiento del foco.

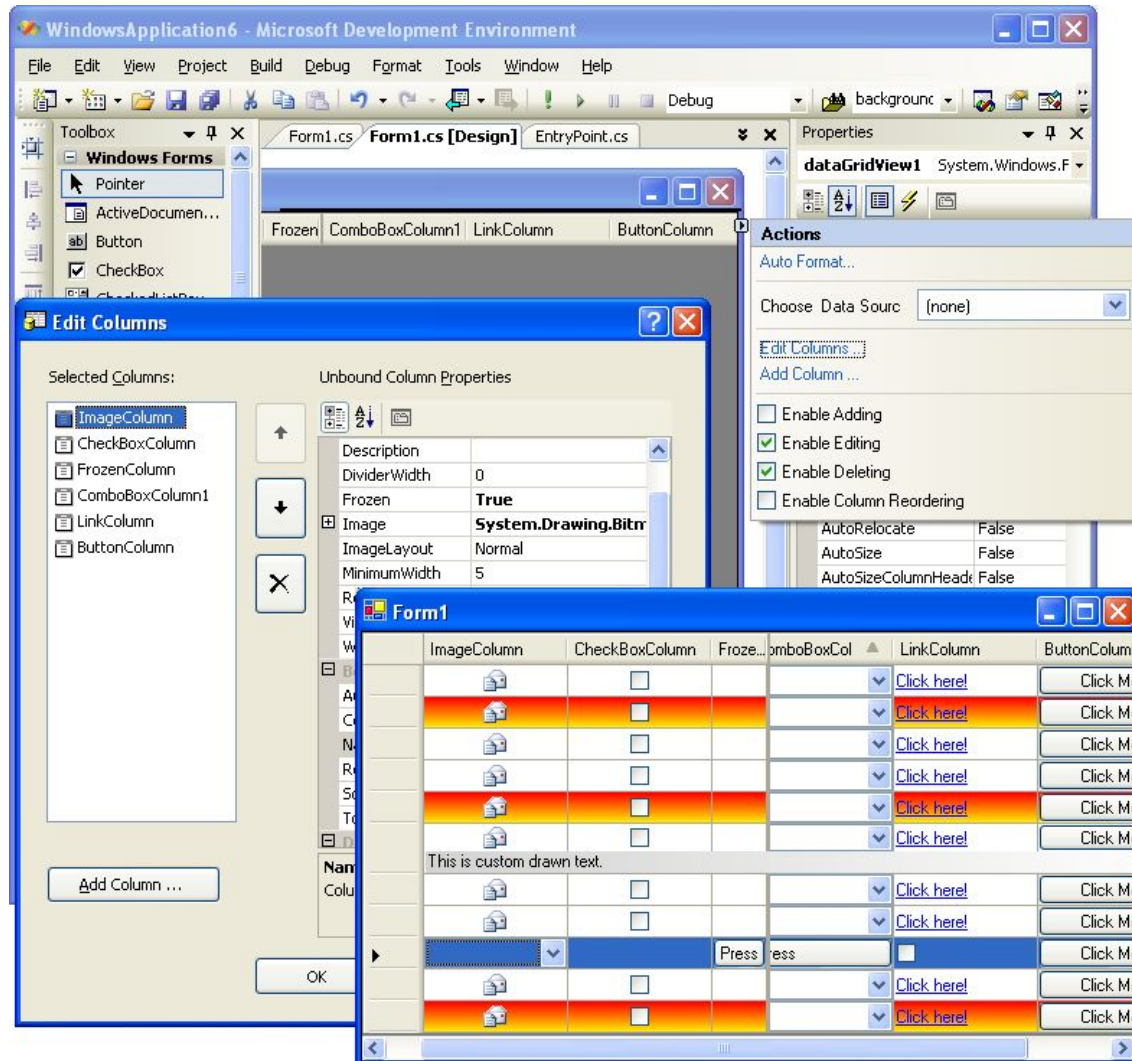
Mas Controles WinForms

- **Datagridview:** tabla de datos. Permite enlazar información de una base de datos de manera dinámica.
- **ToolStrip**
- **Masked Textbox**
- **WebBrowser**
- **Menus**
- **Contenedores**

DataGridView

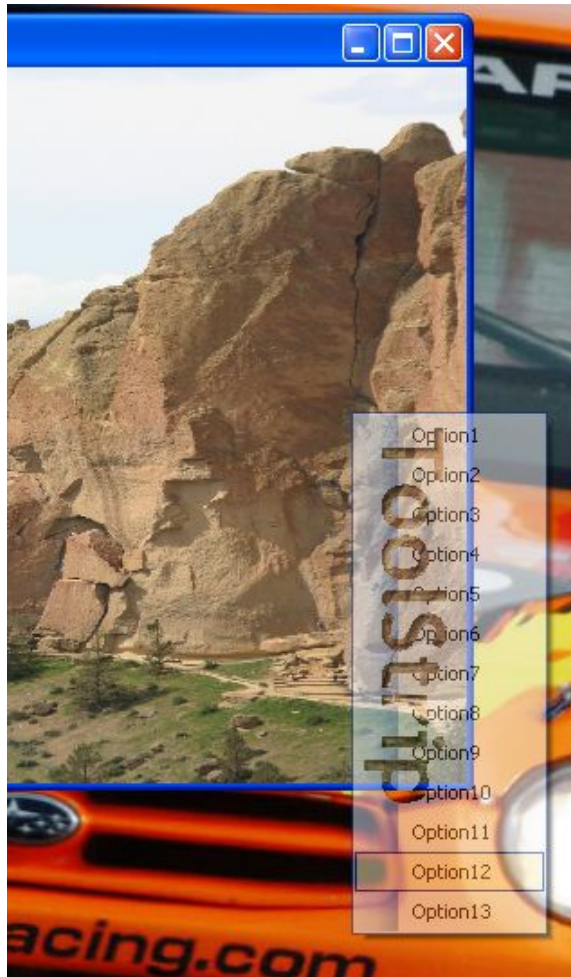
- Evolución del control
System.Windows.Forms.DataGrid (.NET 1.0 y 1.1)
- La vinculación (“binding”) de datos es simple e intuitiva
- Soporta el modelo estandar Windows Form de vinculación (“binding”) de datos.
- Puede ser utilizado en modo estático desconectado (“unbound”) sin dependencia del origen de datos

DataGridView



- Genera barras de herramientas y otros elementos de interfaz de usuario.
- Los ToolStrip reemplazan y agregan funcionalidad a los controles de versiones anteriores.
- Soporta la apariencia y el comportamiento del sistema operativo.
- Crea controles drop-down y editores de tipos con interfaces de usuario avanzadas.

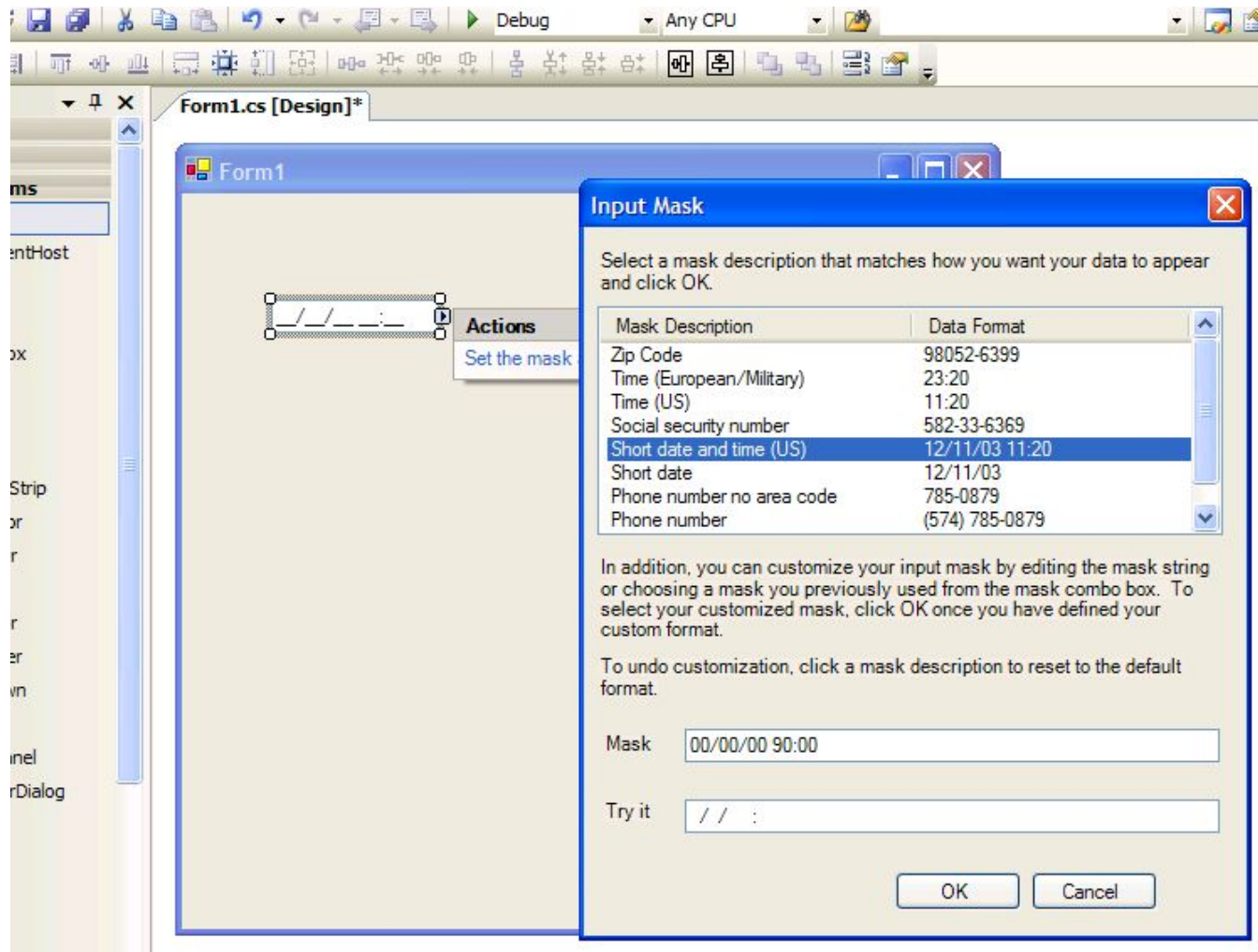
ToolStrip



Masked Textbox

- Mejora sobre el control TextBox
- Soporta sintaxis declarada para aceptar o rechazar un ingreso de datos.
- Se puede especificar sin escribir código:
 - Número de caracteres en un string
 - Caracteres opcionales
 - Caracter espejado en una máscara
- Muestra el formato del texto en tiempo de ejecución

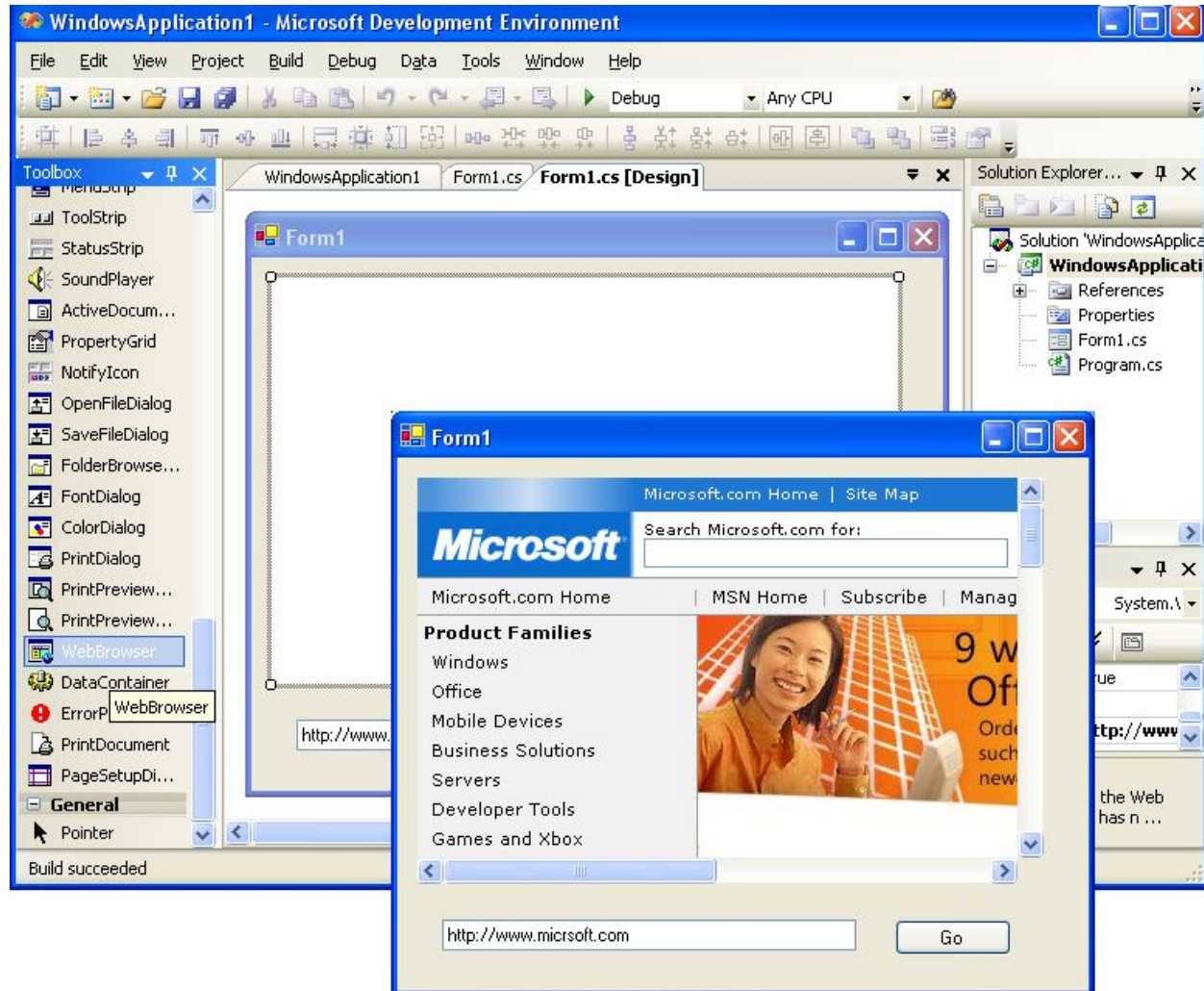
MaskedTextBox



Control WebBrowser

- Proporciona un “wrapper” administrado para el control ActiveX del WebBrowser.
- Reproduce la navegación de Internet Explorer.
- Es un visor de documentos HTML.

WebBrowser



Menús de un Form

- **MenuStrip:** es un contenedor de otros objetos: ToolStripMenuItem, ToolStripComboBox, ToolStripSeparator, ToolStripTextBox.
- **ContextMenuStrip:** menú contextual (botón derecho del ratón). Se puede componer de igual manera que el MenuStrip.

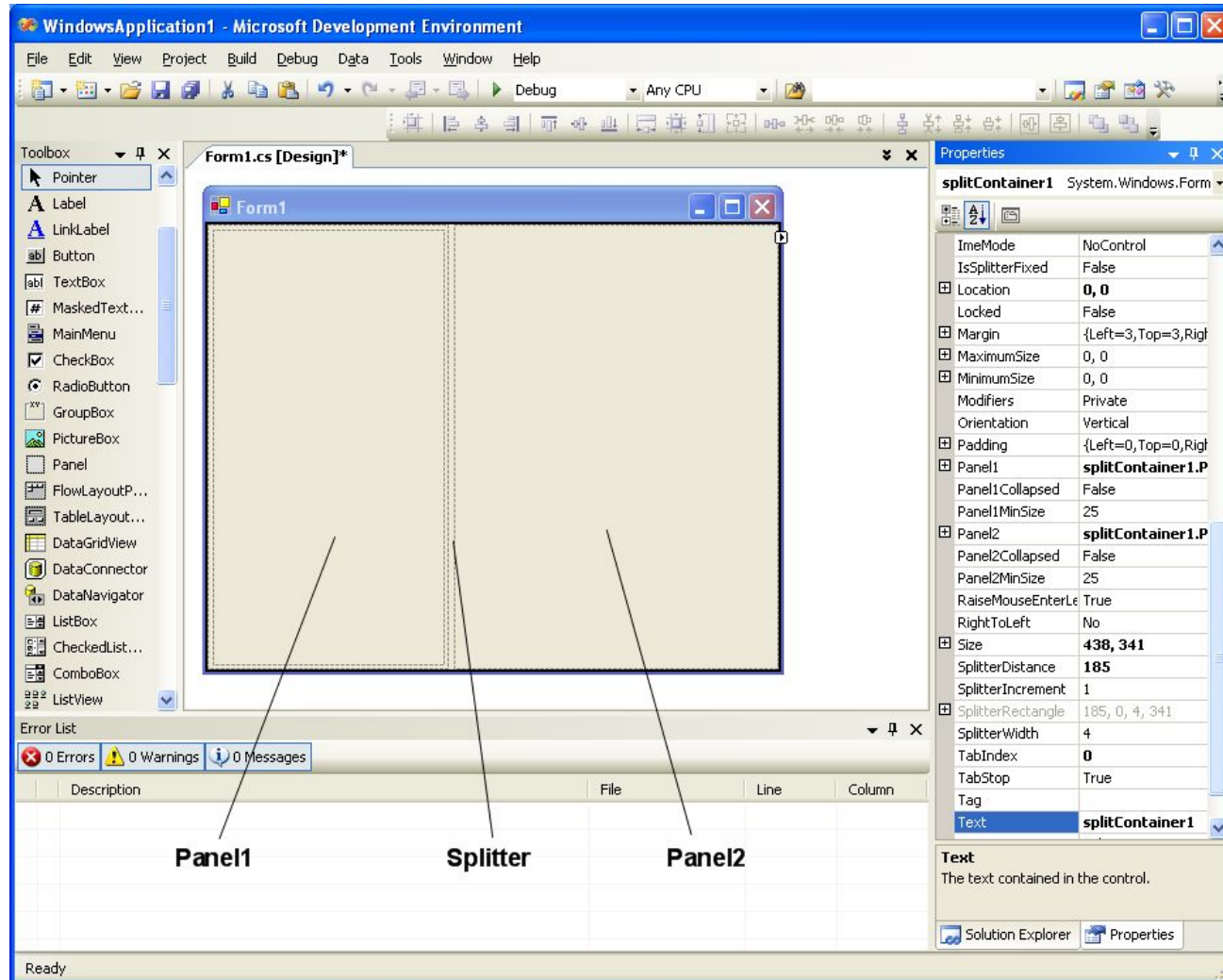
Controles Contenedores

- Pueden contener a otros controles a su vez.
- Heredan de `ContainerControl`.
- Tienen acceso directo a los controles de primer nivel únicamente (acceso recursivo).
- Ejemplos: `Form`, `Panel`, `GroupBox`, `LayoutPanels`

Propiedades

- Anchor: propiedad que permite 'anclar' a un componente a uno o varios bordes del formulario para modificar su presentación de acuerdo con el redimensionamiento de este último.
- Dock: permite fijar un componente a uno de los bordes de la ventana para modificar su ubicación de acuerdo con el redimensionamiento de este último.

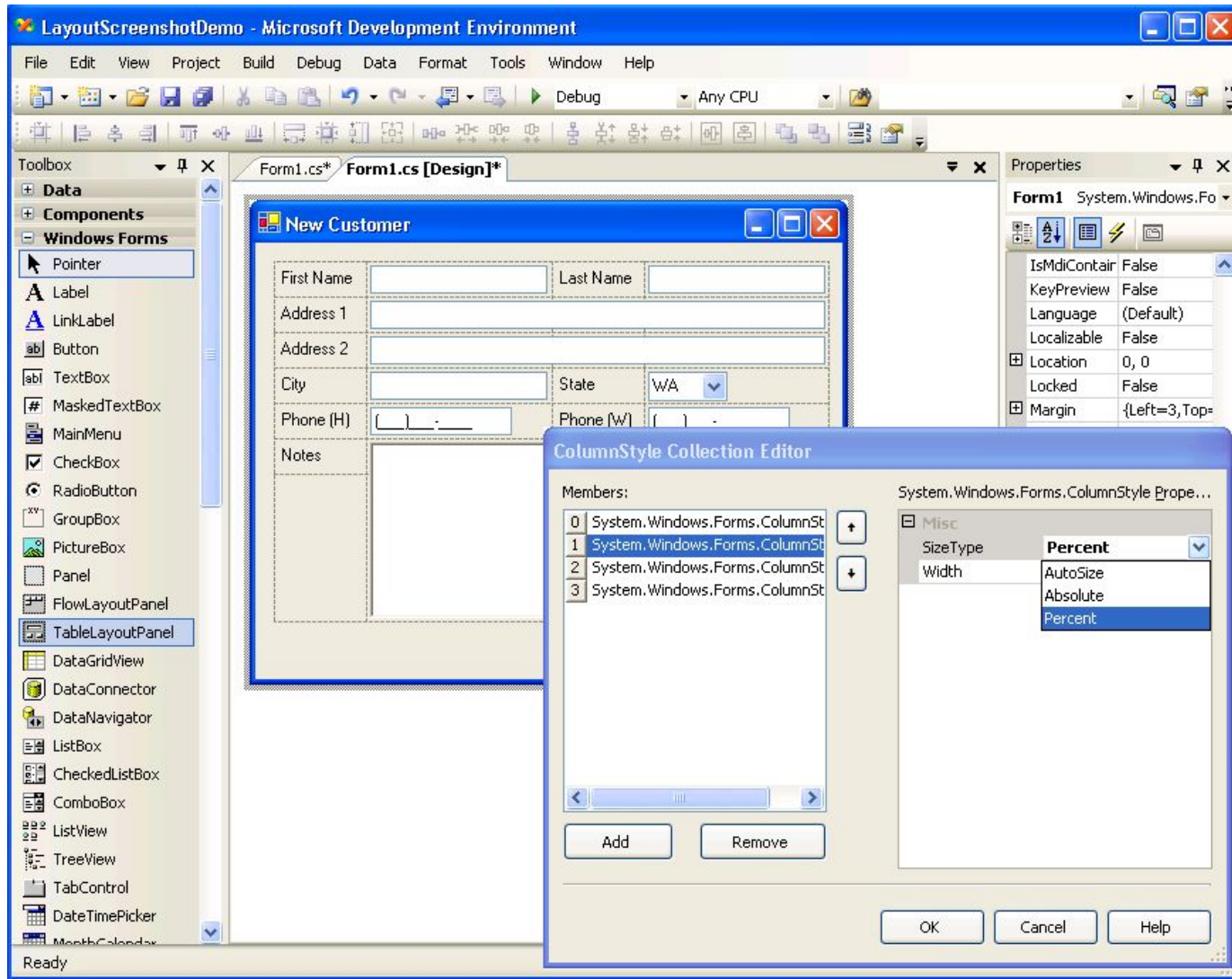
SplitContainer



Layout Panels

- Controla la disposición de elementos sobre el formulario.
- `TableLayoutPanel`: ubica elementos al estilo tabla. Redimensionable.
- `FlowLayoutPanel`: los controles fluyen a medida que se van agregando.

TableLayoutPanel



- Permite actualizaciones automáticas de aplicaciones Windows que pueden ser instaladas con mínima interacción del usuario.
- Supera tres obstáculos inherentes a la distribución de software:
 - Dificultades en las actualizaciones de las aplicaciones.
 - Impacto en el equipo del usuario.
 - Permisos

- Cualquier aplicación Windows Form o modo Consola es publicada a través de la tecnología “ClickOnce”
- Modos de publicación:
 - Como una página web (Internet)
 - Desde una dirección de red local
 - Desde un dispositivo de almacenamiento cualquiera
- Las aplicaciones “ClickOnce” pueden verificar la existencia de nuevas versiones y reemplazar automáticamente los componentes actualizados.

Laboratorios

Laboratorio 1

Login

Laboratorio 2

DataGridView (usa Base de Datos)

Laboratorio 3

Graficos