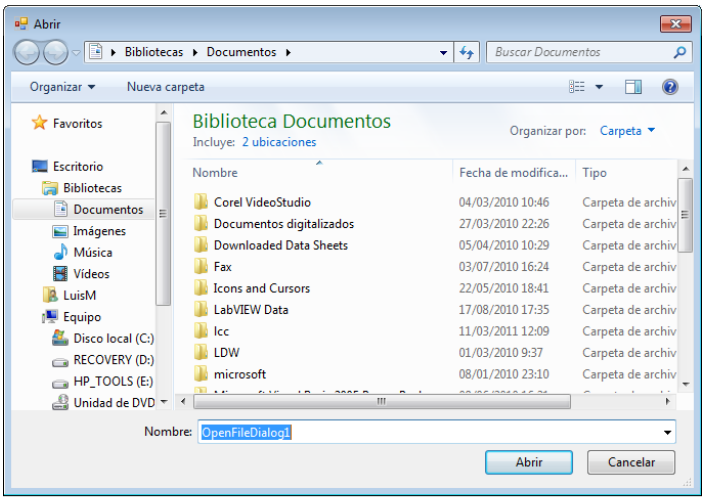
Ejercicios sobre Controles - 5.
Curso 2010-2011

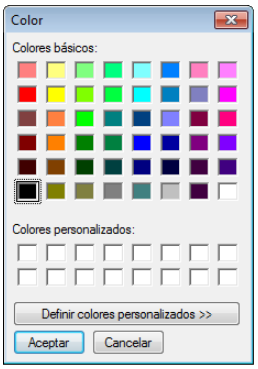
Controles *Common Dialog Box*.

La plataforma .NET proporciona un conjunto de controles que constituyen cajas de diálogo pre-elaboradas, con una gran cantidad de funcionalidades, que permiten realizar algunas de las tareas comunes de una aplicación sin tener que escribir todo el código correspondiente.

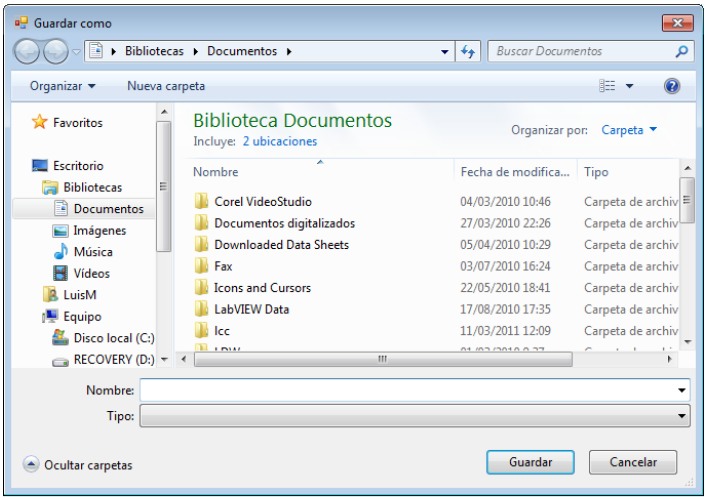
Dichos controles se muestran en la siguiente figura y se explican a continuación.



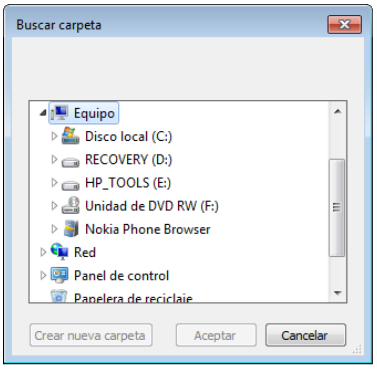
OpenFileDialog



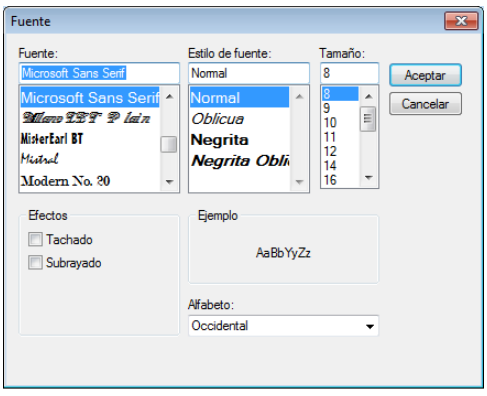
ColorDialog



SaveFileDialog



FileBrowserDialog



FontDialog

Common Dialog Boxes

OpenFileDialog	Permite seleccionar uno o varios archivos para procesar en la aplicación.
SaveFileDialog	Permite establecer el nombre y la ruta de un archivo que se va a guardar en alguno de los medios de almacenamiento.
FolderBrowserDialog	Permite seleccionar una carpeta en la estructura de directorios del ordenador.
ColorDialog	Permite seleccionar un color de los predefinidos o configurar uno a medida.
FontDialog	Permite establecer las características de la letra en el texto seleccionado en ese momento.

Para utilizar una caja de diálogo hay que añadir a la aplicación, en tiempo de diseño, una instancia del control correspondiente, de la misma manera que se hace con los demás controles analizados hasta el momento. También se pueden modificar determinadas propiedades según la apariencia o funcionalidad que se necesite.

Sin embargo, estos controles no se colocan directamente sobre el formulario sino en la zona inferior de este (la denominada *Components Tray*) y se observan en forma de íconos. La activación de la caja de diálogo durante la ejecución del programa se realiza mediante una llamada al método ***ShowDialog***.

El método ***ShowDialog*** retorna un valor que indica el botón que fue activado para cerrar la ventana del diálogo. Dicho resultado es de tipo **Windows.Forms.DialogResult** y puede ser ***OK*** o ***Cancel***.

Estos controles se muestran en formularios **modales**, lo que significa que mientras estén activos no se puede interactuar con los demás formularios de la aplicación.

Una vez que se cierra la caja de diálogo, se puede acceder a sus distintas propiedades para obtener los parámetros que fueron establecidos durante su funcionamiento. A continuación se indican las propiedades más importantes de cada una de las cajas de diálogo anteriormente mencionadas. Algunas de estas propiedades son comunes a todas las cajas de diálogo por lo que solo se hará una referencia a ellas.

Propiedades de *OpenFileDialog* y de *SaveFileDialog*.

AddExtension	Indica si la caja de diálogo agrega automáticamente una extensión a un nombre de archivo en caso de que el usuario omita dicha extensión.
CheckFileExists	Indica si la caja de diálogo muestra una advertencia cuando el usuario especifica un nombre de archivo que no existe.
CheckPathExists	Indica si la caja de diálogo muestra una advertencia cuando el usuario especifica una ruta de acceso que no existe.
DefaultExt	Obtiene o establece la extensión de nombre de archivo predeterminada.
FileName	Obtiene o establece una cadena de caracteres que contiene el nombre de archivo seleccionado en la caja de diálogo.
FileNames	Obtiene una matriz de cadenas de caracteres con los nombres de todos los archivos seleccionados en la caja de diálogo.
InitialDirectory	Obtiene o establece la ruta de la carpeta que muestra la caja de diálogo al abrirse.
Multiselect	Indica si se pueden seleccionar varios nombres de archivo en la caja de diálogo. Para seleccionar varios archivos el usuario debe mantener presionada la tecla **Ctrl** (Control)
ShowHelp	Indica si se muestra el botón **Ayuda** en la caja de diálogo.
Title	Obtiene o establece el título que aparecer en la caja de diálogo.

Propiedades de *FolderBrowserDialog*.

SelectedPath	Obtiene o establece la ruta de acceso seleccionada por el usuario.
ShowNewFolderButton	Indica si el botón **Nueva Carpeta** aparece en el cuadro de diálogo explorador de carpetas.

Propiedades de *ColorDialog*.

AllowFullOpen	Indica si el usuario puede utilizar la caja de diálogo para definir colores personalizados.
AnyColor	Indica si la caja de diálogo muestra todos los colores disponibles en el conjunto de colores básicos.
Color	Obtiene o establece el color seleccionado por el usuario.
CustomColors	Obtiene o establece el conjunto de colores personalizados que se muestran en la caja de diálogo.
FullOpen	Indica si los controles utilizados para crear colores personalizados son visibles al abrir la caja de diálogo.
SolidColorOnly	Indica si la caja de diálogo muestra únicamente colores sólidos para elegir.

Los valores establecidos en las propiedades ***AllowFullOpen, AnyColor, CustomColors, FullOpen*** y ***SolidColorOnly*** pueden hacer que cambie el aspecto con que se muestra la caja de diálogo.

Propiedades de *ColorDialog*.

AllowScriptChange	Indica si el usuario puede cambiar el juego de caracteres especificado en el cuadro combinado **Alfabeto** para mostrar un juego de caracteres diferente del que aparece en la caja.
AllowSimulations	Indica si el cuadro de diálogo muestra una vista previa de fuente de la interfaz de dispositivo gráfico (GDI).
Font	Obtiene o establece el tipo de letra seleccionada.
FontMustExist	Indica si el cuadro de diálogo debe especificar una condición de error cuando el usuario intenta seleccionar una fuente o un estilo que no existe.
ShowApply	Indica si el cuadro de diálogo contiene un botón **Aplicar**. La funcionalidad de este botón se establece en el manejador del evento **Apply**.
ShowColor	Indica si el cuadro de diálogo muestra la opción de cambio de color de los caracteres.
ShowEffects	Indica si el cuadro de diálogo contiene controles que permiten al usuario especificar opciones de tachado, subrayado y color del texto.

Los valores establecidos en las propiedades ***AllowSimulations, ShowApply, ShowColor,*** y ***ShowEffects*** pueden hacer que cambie el aspecto con que se muestra la caja de diálogo.

A continuación se muestran fragmentos de código que ejemplifican algunas de las funcionalidades de las cajas de diálogo analizadas con anterioridad.

- Se abre una caja de diálogo ***ColorDialog*** llamada **DlgColor** para cambiar el color de los caracteres de una caja de texto llamada **txtBox**. El proceso se lleva a cabo si el usuario abandona la caja de diálogo presionando el botón **OK**, de lo contrario se muestra un mensaje de advertencia.

```
If DlgColor.ShowDialog = Windows.Forms.DialogResult.OK Then
    txtBox.ForeColor = DlgColor.Color
Else
    MessageBox.Show("No se realizó el cambio")
End If
```

- Se abre una caja de diálogo ***ColorDialog*** llamada **DlgColor.** Previamente se asigna el color de fondo actual del formulario a la propiedad **Color** de dicha caja para que se muestre seleccionado al abrirla. Si se selecciona un nuevo color y se activa el botón **OK**, se establece el color seleccionado como color de fondo del formulario.

```
DlgColor.Color = Me.BackColor
If DlgColor.ShowDialog = Windows.Forms.DialogResult.OK Then
    Me.ForeColor = DlgColor.Color
End If
```

- Se abre una caja de diálogo ***ColorDialog*** llamada **DlgColor.** Previamente inicializa una matriz con tres valores que se utilizan como colores personalizados.

```
Dim colors() As Integer = {222663, 35453, 7888}

DlgColor.CustomColors = colors
DlgColor.ShowDialog()
```

- Se abre una caja de diálogo ***FontDialog*** llamada **DlgFuente** para cambiar el tipo de letra de una caja de texto llamada **txtBox**. El proceso se lleva a cabo si el usuario abandona la caja de diálogo presionando el botón **OK**.

```
If DlgFuente.ShowDialog = Windows.Forms.DialogResult.OK Then
    txtBox.Font = DlgFuente.Font
End If
```

- El mismo ejemplo anterior pero previamente se inicializa una variable de tipo **Font** para que, al abrir la caja de diálogo, al usuario le quede pre-establecido el tipo de letra Arial de 10 puntos en negrita. Para conseguir la funcionalidad del botón **Aplicar** es necesario definir un procedimiento manejador para el evento **Apply** de este control. De todas maneras el usuario puede elegir otro tipo de fuente. De todas maneras el usuario puede elegir otro tipo de fuente.

```
Dim nuevaFuente As New Font(”Arial”, 10, FontStyle.Bold)

DlgFuente.Font = nuevaFuente
If DlgFuente.ShowDialog = Windows.Forms.DialogResult.OK Then
    txtBox.Font = DlgFuente.Font
End If
```

Para conseguir la funcionalidad del botón **Aplicar** es necesario definir un procedimiento manejador para el evento **Apply** de este control, tal como se muestra a continuación.

```
Private Sub DlgFuente_Apply(...) Handles DlgFuente.Apply
    txtBox.Font = DlgFuente.Font
End Sub
```

- Se abre una caja de diálogo ***FileOpenDialog*** llamada **DlgArchivo** para obtener un nombre de archivo. Se asigna por defecto la extensión “.BMP” al archivo y se muestran todos los archivos de la carpeta que tengan esta extensión. También se da la posibilidad al usuario de abrir los archivos que tengan las extensiones “.GIF” y “.JPG”. La ruta inicial es “C:\IMAGENES\APLICACION\” . El nombre de archivo se muestra en una caja de texto denominada txtBox si el usuario abandona la caja de diálogo presionando el botón **OK**.

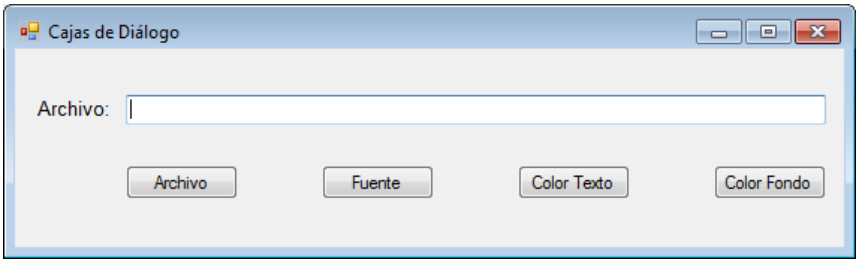
```
DlgArchivo.DefaultExt = “.BMP”
DlgArchivo.AddExtension = True
DlgArchivo.Filter = “Bitmaps|*.BMP|Imágenes GIF |*.GIF|” & _
                    ”Imágeners JPEG|*.JPG|Todas|*.BMP;*.GIF;*.JPG”
DlgArchivo.InitialDirectory = “C:\IMAGENES\APLICACION”
If DlgArchivo.ShowDialog = Windows.Forms.DialogResult.OK Then
    txtBox.Text = DlgArchivo.FileName
End If
```

- Se abre una caja de diálogo ***FileOpenDialog*** llamada **DlgArchivo**, en la que se pueden marcar varios archivos a la vez. Los nombres de los archivos marcados se muestran en una lista llamada **lbxArchivos**.

```
DlgArchivo.Multiselect = True
DlgArchivo.InitialDirectory = "C:\Users\Alumnos"
DlgArchivo.ShowDialog()
lbxArchivos.Items.Clear()
For Each files As String In DlgArchivo.FileNames
    lbxArchivos.Items.Add(files)
Next
```

Ejercicio

1. Desarrollar una aplicación con la apariencia mostrada a continuación:



   - Mediante el botón **Archivo** se puede seleccionar un archivo en la estructura de directorios del ordenador y mostrar su nombre (incluyendo la ruta) en la caja de texto de la aplicación. Probar las diferentes opciones de filtro que tiene la caja de diálogo **OpenFileDialog**, es decir, hacer que solo se puedan seleccionar archivos de un determinado nombre o extensión.
   - Si se abandona la caja de diálogo sin seleccionar algún archivo debe mostrarse un mensaje de advertencia.
   - Permitir la caja de diálogo muestre la carpeta en la que se encuentra el archivo, cuyo nombre (incluyendo la ruta) se ha especificado en la caja de texto. Obviamente se debe especificar un archivo que exista en la carpeta.
   - Mediante el botón **Fuente** se puede cambiar el tipo de letra de la caja de texto donde se muestra el nombre del archivo.
   - Mediante el botón **Color Texto** se puede cambiar el color de los caracteres de la caja de texto donde se muestra el nombre del archivo.
   - Mediante el botón **Color Fondo** se puede cambiar el color del fondo de la caja de texto donde se muestra el nombre del archivo.
   - En las opciones de color, probar con colores estándar, sólidos solamente y/o personalizados.

Control *RichTextBox*.

El control **RichTextBox** constituye un procesador de textos en toda regla. Posee la mayoría de las funcionalidades básicas de cualquier aplicación profesional diseñada con este propósito.

La funcionalidad de este control es la misma de un **TextBox** en cuanto a escritura de texto se refiere. Además permite:

- Varios tipos y tamaños de letras en el mismo texto.
- Varios colores de letras en el mismo texto.
- Control preciso sobre los márgenes de edición de texto.
- Insertar imágenes dentro del texto.

Es por todo ello que este control se puede utilizar para mostrar o escribir texto "enriquecido", es decir, texto que contiene información de formato además de los caracteres.

El texto se almacena en formato RTF (***R**ich **T**ext **F**ormat*), cuya descripción se puede encontrar en http://es.wikipedia.org/wiki/Rich_Text_Format.

La posibilidad de tener diferentes formatos en el mismo texto se implementa mediante varias propiedades que permiten cambiar dicho formato a un bloque de texto previamente **marcado**. La marca del texto la puede realizar el usuario mediante las herramientas de la interfaz o mediante instrucciones del programa. A continuación se indican los miembros más importantes de este control.

Propiedades.

BulletIndent	Obtiene o establece la sangría que se utiliza en el control cuando se aplica el estilo de viñeta al texto.
DetectURL	Indica si el control debe aplicar formato a una URL cuando se escriba en el control.
Rtf	Obtiene o establece el texto del control, incluidos todos los códigos con formato de texto enriquecido (RTF).
SelectedRtf	Obtiene o establece el texto con formato de texto enriquecido (RTF) actualmente seleccionado en el control.
SelectedText	Obtiene o establece el texto actualmente seleccionado en el control.
SelectionAlignment	Obtiene o establece la alineación de los caracteres seleccionados en el control.
SelectionBackColor	Obtiene o establece el color de fondo de los caracteres seleccionados en el control.
SelectionColor	Obtiene o establece el color de los caracteres seleccionados en el control.
SelectionFont	Obtiene o establece el tipo de letra de los caracteres seleccionados en el control.
SelectionStart	Obtiene o establece la posición del primer carácter del bloque seleccionado en base al inicio del texto.
SelectionLength	Obtiene o establece el número de caracteres seleccionados en el control.

Métodos.

AppendText	Añade texto al contenido actual del control.
Clear	Borra el texto del control.
Copy	Copia en el portapapeles el texto seleccionado del control.
DeselectAll	Asigna cero a la propiedad **SelectionLength** para que no se seleccione ningún

carácter en el control.

Find	Busca una cadena en el texto del control.
LoadFile	Carga el contenido de un archivo en el control.
SaveFile	Guarda el texto del control en un archivo.
SelectAll	Selecciona todo el texto del control.
Paste	Pega el contenido del portapapeles en el control.

Ejercicio

2. Desarrollar un editor de texto apoyado en las facilidades que brinda el control **RichTextBox**. Añada las siguientes funcionalidades:
    a. Cambiar el tipo de letra de toda o de una parte del texto.
    b. Cambiar el color de toda o de una parte del texto.
    c. Cambiar el color de fondo de toda o de una parte del texto.
    d. Cambiar el margen derecho del texto seleccionado mediante una barra de desplazamiento convenientemente colocada encima del control.
    e. Copiar texto al portapapeles
    f. Pegar texto desde el portapapeles

    Pueden obviarse las funciones de cargar y guardar el texto en disco.

    Nota: Este ejercicio no pretende remplazar ninguno de los editores de texto profesionales, sino ejercitar algunas de las facilidades de programación de Visual Basic.

Control *ToolTip* .

Un aspecto importante en el diseño de la interfaz de usuario de una aplicación es lograr que dicha interfaz sea intuitiva para el usuario final. Ello se logra, entre otras cosas, proporcionando información rápida durante la ejecución de la aplicación sobre la función de cada uno de los controles que componen la interfaz.

Una herramienta que proporciona esta facilidad es el control **ToolTip** (*Sugerencia*). Dicho control permite que se muestre una pequeña ventana rectangular, con un texto explicativo cuando, el ratón pasa sobre un área específica.

El control **ToolTip** se añade al formulario de la misma forma que se hace con los demás controles proporcionados por Visual Studio .NET. Este control no se muestra visible en el formulario durante la ejecución de la aplicación, sino que aporta la funcionalidad antes descrita.

Una vez que se añade un control **ToolTip** a un formulario, todos los demás controles adquieren una propiedad especial denominada **ToolTip**. En dicha propiedad se especifica el texto que debe mostrarse en la ventana explicativa.

Propiedades.

AutoPopDelay	Obtiene o establece el período de tiempo que estará visible el texto explicativo a partir de que el puntero del ratón se detiene sobre el área que ocupa el control.
InitialDelay	Obtiene o establece el tiempo que demora en mostrarse el texto explicativo a partir de que el puntero del ratón se detiene sobre el área que ocupa el control.
IsBalloon	Indica si el texto explicativo se muestra en una ventana de globo.
ShowAlways	Indica si el texto explicativo se muestra aunque el control no esté activo.
ToolTipTitle	Obtiene o establece el título que aparece en la ventana del texto explicativo.

Ejercicio

3. Para el ejercicio 2, agregar ventanas de texto explicativo en todos aquellos controles que sea conveniente.

Control *ToolStrip* .

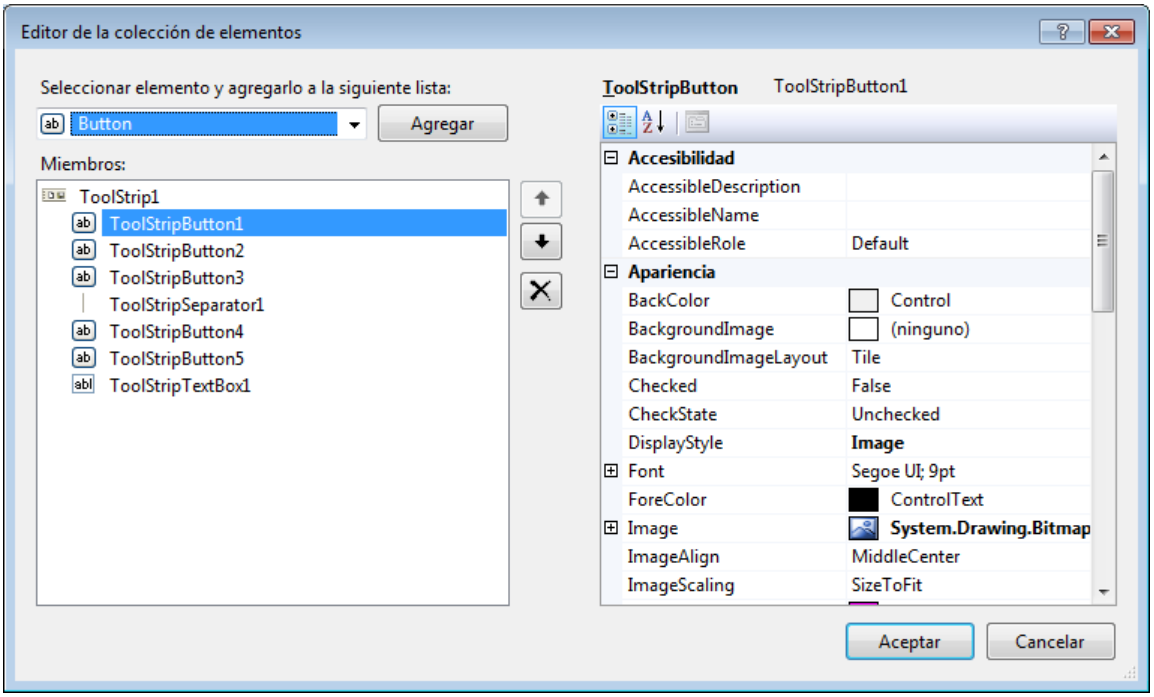
Otro de los elementos que aparece a menudo en una aplicación Windows es la barra de herramientas, en la que se sitúan controles que permiten ejecutar tareas que el usuario realiza frecuentemente.

Para insertar una barra de herramientas en un formulario se utiliza el control **ToolStrip**, el cual actúa como contenedor de otros controles que forman parte de la barra de herramientas.

Al insertar un control **ToolStrip** en el formulario, la barra de herramientas se sitúa en la parte superior de este, justo por debajo del menú (si lo hubiera). Ésta es la ubicación por defecto.Sin embargo, se puede cambiar mediante su propiedad **Dock**.

Se pueden definir dos tipos de barras de herramientas. La barra estándar, mostrada en la figura, es la que tiene los controles básicos que se incluyen en la mayoría de las aplicaciones Windows.

La barra personalizada permite definir los controles que se incluirán en ella. Esto se efectúa mediante la propiedad **Items** del control **ToolStrip**, que da acceso al cuadro de diálogo mostrado en la siguiente figura, donde se pueden añadir botones y otros tipos de controles. Para añadir controles se elije el control en la lista superior izquierda y se pulsa el botón **Agregar**.



Las propiedades de los controles de la barra de herramientas se configuran en el cuadro de diálogo anterior, de la forma que se ha explicado en las prácticas anteriores. También se les puede incluir procedimientos manejadores para los diferentes eventos que se necesiten controlar. Con todo ello logra la funcionalidad deseada para cada uno de ellos.

Ejercicio

4. Para el ejercicio 2, agregar una barra de herramientas con los controles que realizan las operaciones básicas del editor de texto. Asegúrese de realizar los cambios necesarios para asignar los procedimientos manejadores de evento a los nuevos controles.