

UNIVERSIDAD NACIONAL DE INGENIERIA

FACULTAD DE INGENIERIA MECANICA

CURSO PROGRAMACION ORIENTADA A OBJETOS

CÓDIGO MB545



Tema: Las Clases MFC y DEVICE CONTEX

2era parte

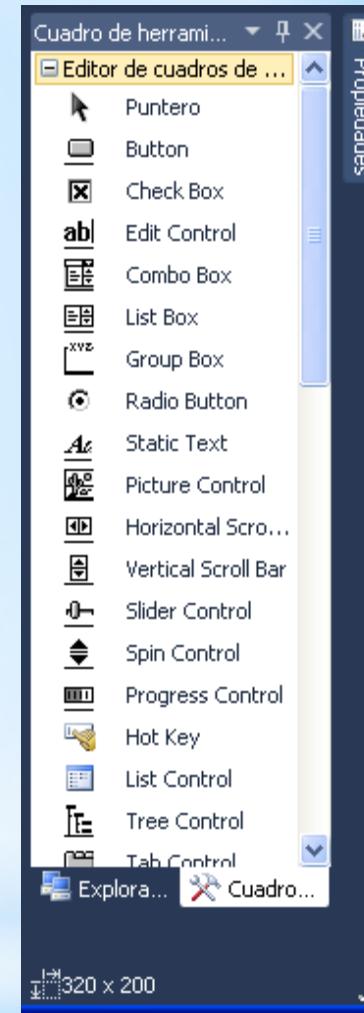
1

Ingeniero Daniel Osorio Maldonado

23/08/2019

Los Controles y sus aplicaciones

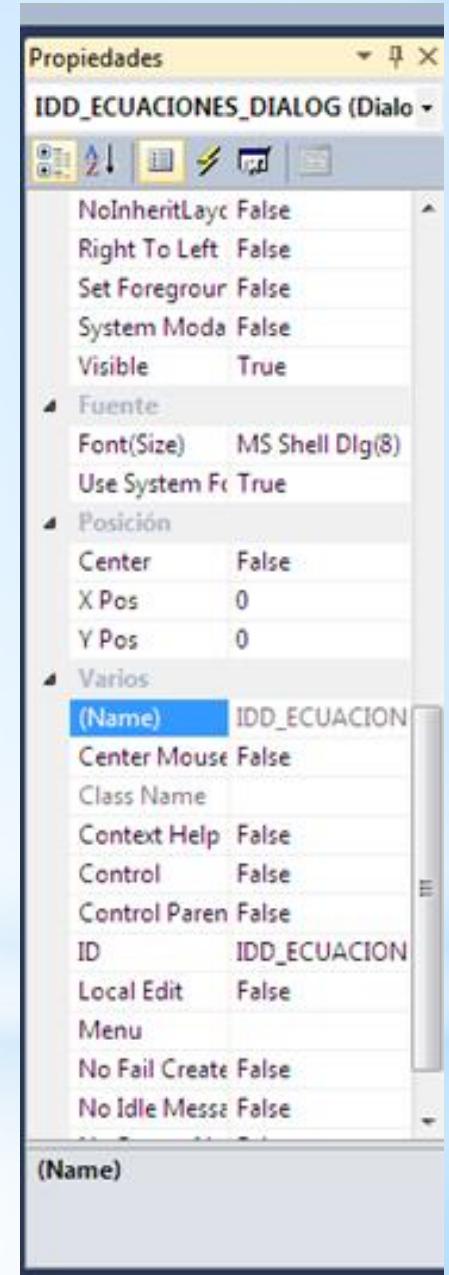
- El control **Static Text Aa**: usados cuando se trata de colocar títulos o cadenas en un cuadro de dialogo.
- **Grupo Box** :reúne en un espacio limitado, un grupo de controles ejemplo Button, radios, cheks, etc.
- **Check Box**: seleccionado ya sea individual o en grupo
- **Edit Box**: crea un rectángulo interactivo
- **Radio Button**: circulo pequeño que generalmente va en Grupo solo se permite activar uno.
- **Button** :permite el acceso a la aplicación ejecutora
- **List Box**:permite elegir un item de varios.
- **Ficha (Tab)**: el usuario usa diferentes paginas dentro del Cdialogo
- **Image**: en un rectángulo coloca una imagen de un mapa de bits



Las Propiedades de los objetos

La

ventana **Propiedades** muestra distintos tipos de campos de edición, según las necesidades de una propiedad determinada. Entre estos campos de edición se incluyen cuadros de edición, listas desplegables y vínculos a cuadros de diálogo de editor personalizado. Las propiedades que se muestran en gris son de sólo lectura.



La Funcion DoDataExchange

- ClassWizard añade una función miembro llamada DoDataExchange al archivo .cpp del dialog de clase.
- La función contiene todas las llamadas a las funciones de intercambio y validación de datos que necesita el dialogo.
- Para un solo control de edición que acepte un numero entero desde 1 hasta 99.
- Ejemplo ClassWizard escribe una función DoDataExchange que funciona así

```
void CdemoDlg::DoDataExchange(CDataExchange *pDX)
{ Cdialog::DoDataExchange(pDX);
//{AFX_DATA_MAP(CdemoGlg)
DDX_Text(pDX, IDC_EDIT,m_nEditval);
DDV_MinMaxInt(pDX,m:nEditVal,1,99);
//}}AFX_DATA_MAP}
```

➤ Se dice que es una función completa, mas no un código matriz al que deba añadirle otros códigos.

➤ Si se invoca al DoDataExchange, cuando el dialogo empieza y termina, es a veces conveniente añadir inicialización y eliminar código para ello, de lo contrario puede olvidar que la función existe.

➤ Si elimina una variable en la pestaña Member Variables, ClassWizard elimina cualquier invocación de intercambio y validación de datos para la variable en la función DoDataExchange.

- Para un control Edit que acepte solo números enteros simple de base decimal(en lugar de base Hexadecimal), seleccione el cuadro de comprobación Number en la pestaña Style del dialogo Edit Properties.
- Si un cuadro de combinación o cuadro de lista acepta sólo datos numéricos formados libremente, acceda al dialogo de control properties y desactive la opción de clasificación. Para un cuadro de combinación y cuadro de lista normal, desactive el cuadro comprobación Sort en la pestaña Style.
- Si el caso es crear un grupo de botones radio, ajuste la propiedad auto para cada botón en el grupo, se excluirán mutuamente.
- Ajuste la propiedad Group solo para el primer botón de radio de un grupo, los demás botones del grupo deben seguir el orden de fabulación secuencial.
- También instale la propiedad Group para el control que sigue inmediatamente al ultimo botón de radio de un grupo. Indicará el final del grupo anterior y el principio de otro.
- La Función **UpdateData** de la clase CWnd permite iniciar los controles

Aplicando un Cuadro de Texto

➤ En esta aplicación se utilizará un cuadro de texto de edición, lo que se pretende es que mientras el usuario escribe algo en el cuadro de edición, de manera simultánea se escribirá lo mismo en el título de la ventana de diálogo.

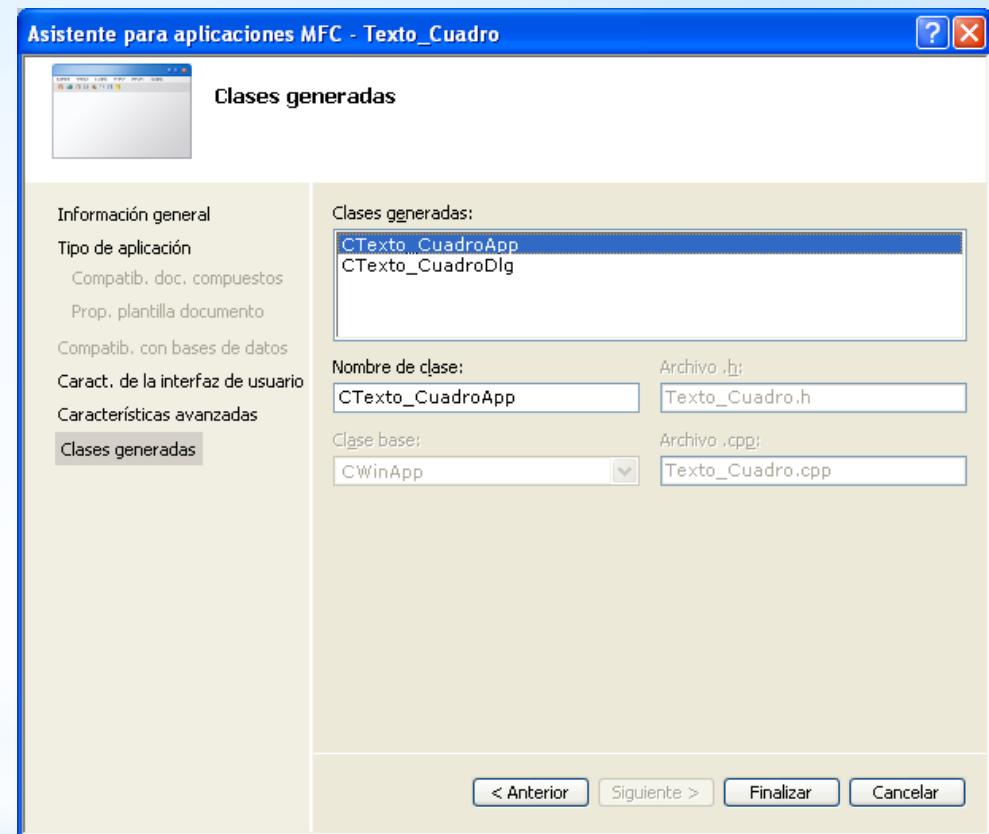
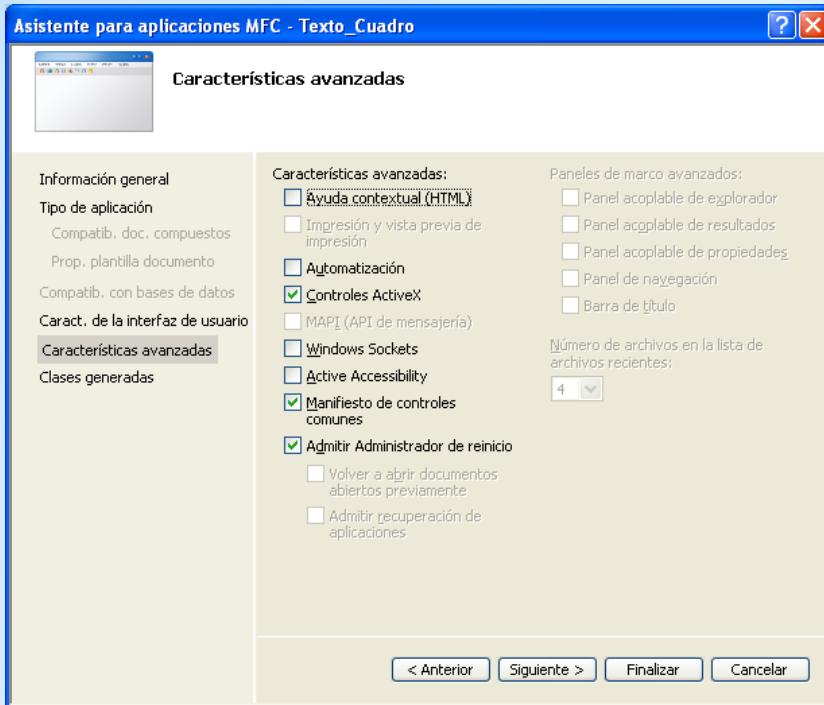
The image displays two side-by-side windows of the 'Asistente para aplicaciones MFC - Texto_Cuadro' (MFC Application Wizard). Both windows have a blue title bar with the application name and standard window controls (minimize, maximize, close).

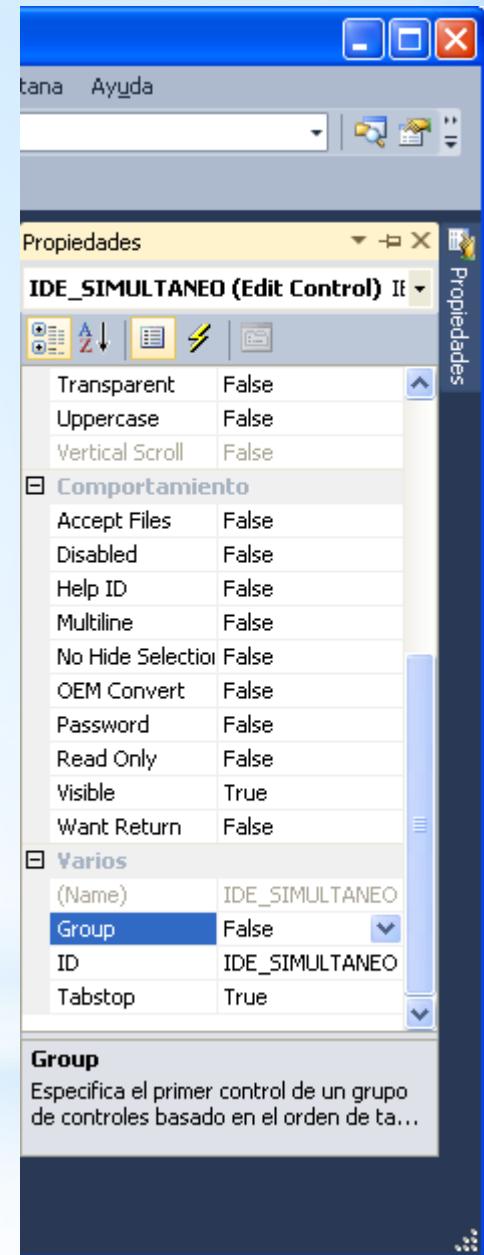
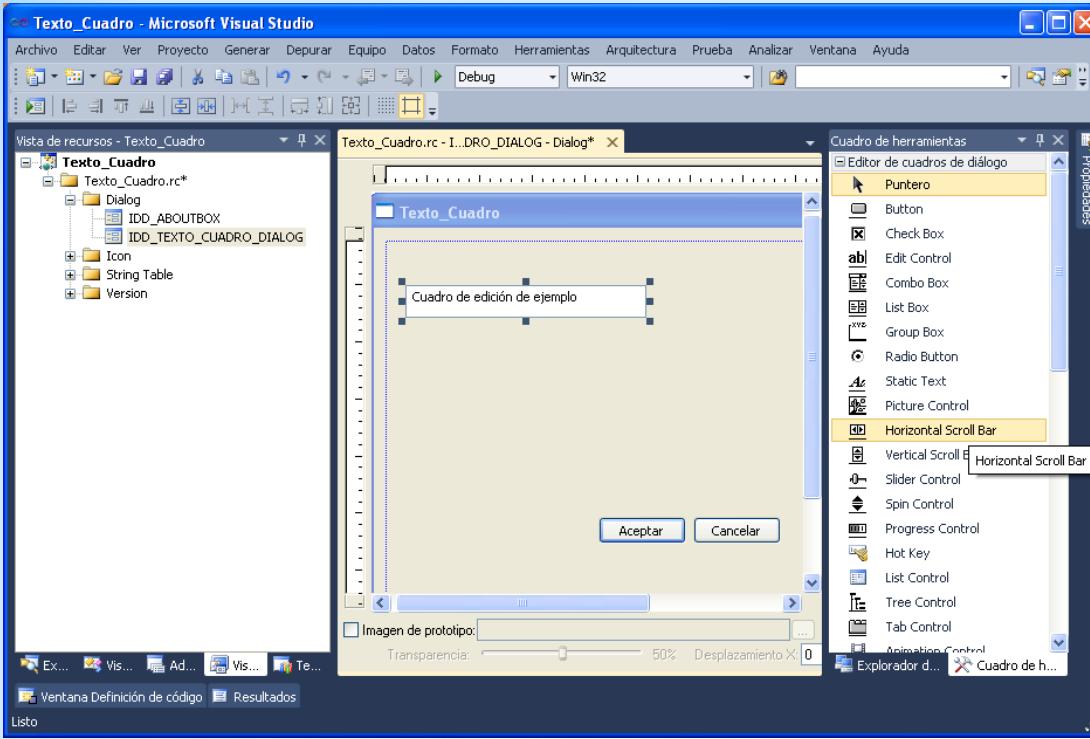
Left Window (Información general):

- Información general:** Contains a list of configuration options:
 - Basado en cuadros de diálogo
 - Sin compatibilidad con bases de datos
 - Sin compatibilidad con documentos compuestos
 - Interfaz de barra de menús y barra de herramientas personalizable
 - Apariencia de aplicación de Visual Studio 2008
 - Estilo de proyecto MFC estándar
 - Admitir el Administrador de reinicio (volver a abrir documentos, recuperación de aplicaciones)
- Botones:** < Anterior, Siguiente >, Finalizar, Cancelar.

Right Window (Tipo de aplicación):

- Tipo de aplicación:** Shows a preview of a dialog box with a text input field.
- Información general:** Contains:
 - Compatib., doc., compuestos
 - Prop., plantilla documento
 - Compatib., con bases de datos
 - Caract. de la interfaz de usuario
 - Características avanzadas
 - Clases generadas
- Tipo de aplicación:** Radio buttons:
 - Documento único
 - Varios documentos
 - Basada en cuadros de diálogo
 - Múltiples documentos de nivel superior
- Estilo del proyecto:** Radio buttons:
 - MFC estándar
 - Explorador de Windows
 - Visual Studi...
 - Office
- Estilo visual y colores:** dropdown menu set to 'Nativo o predeterminado de Windows'
- Uso de MFC:** Radio buttons:
 - Usar MFC en un archivo DLL compartido
 - Usar MFC en una biblioteca estática
- Checkboxes:**
 - Compatibilidad con la arquitectura documento/vista
 - Usar bibliotecas Unicode
- Botones:** < Anterior, Siguiente >, Finalizar, Cancelar.





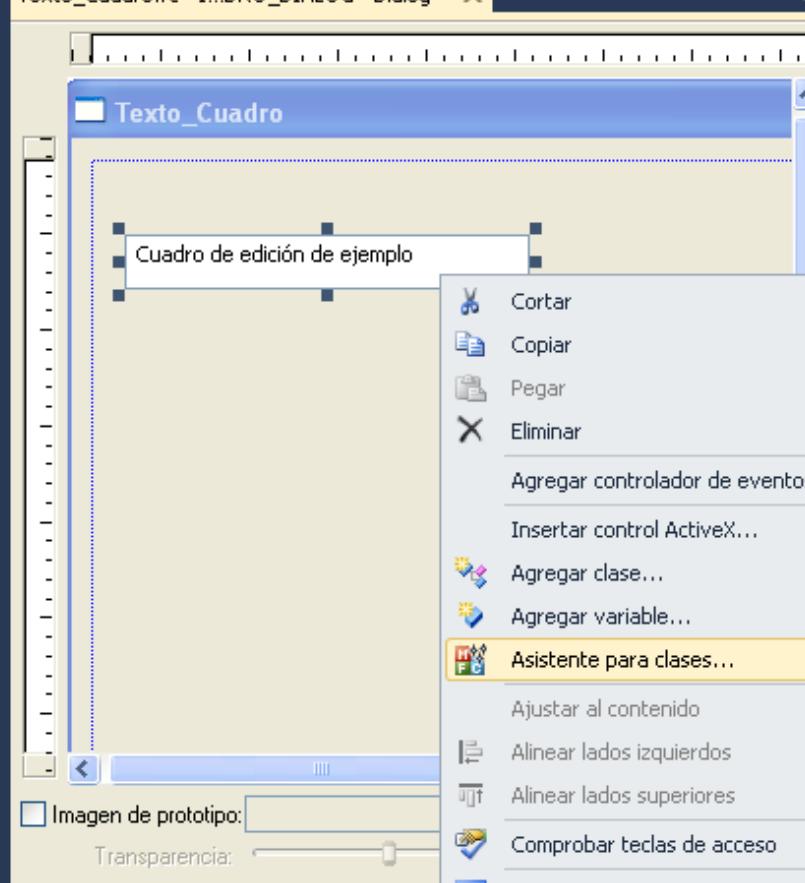
- Agregue un cuadro de edición en la pantalla, tal como se muestra.
- Haciendo clic con el botón derecho active properties y modifique el ID:
IDE_SIMULTANEO



Vista de recursos - Texto_Cuadro

- Texto_Cuadro
 - Texto_Cuadro.rc*
 - Dialog
 - IDD_ABOUTBOX
 - IDD_TEXTO_CUADRO_DIALOG
 - Icon
 - String Table
 - Version

Texto_Cuadro.rc - IDD_TEXTO_CUADRO_DIALOG - Dialog*



Cuadro de herramientas

- Editor de cuadros de diálogo
 - Puntero
 - Button
 - Check Box
 - Edit Control
 - Combo Box
 - List Box

Ctrl+X

Ctrl+C

Ctrl+V

Supr

Agregar controlador de eventos...

Insertar control ActiveX...

Agregar clase...

Agregar variable...

Asistente para clases...

Ctrl+Mayús.+X

Ajustar al contenido

Mayús.+F7

Alinear lados izquierdos

Ctrl+Mayús.+Flecha izquierda

Alinear lados superiores

Ctrl+Mayús.+Flecha arriba

Comprobar teclas de acceso

Ctrl+M

Propiedades



MFC Class Wizard

Proyecto:

Texto_Cuadro

Nombre de clase:

CTexto_CuadroDlg

Agregar clase ▾

Clase base: CDlgEx

Declaración de clase: texto_cuadrodlg.h

Recurso: IDD_TEXTO_CUADRO_DIALOG

Implementación de clase: texto_cuadrodlg.cpp

Comandos

Mensajes

Funciones virtuales

Variables de miembro

Métodos

Buscar comandos

Agregar controlador...

Identificadores de objeto:

- IDABORT
- IDC_EDIT1
- IDC_STATIC
- IDCANCEL
- IDCLOSE
- IDD_ABOUTBOX
- IDD_TEXTO_CUADRO_DIALOG
- IDE_SIMULTANEO
- IDHELP

Mensajes:

- EN_CHANGE
- EN_ERRSPACE
- EN_HSCROLL
- EN_KILLFOCUS
- EN_MAXTEXT
- EN_SETFOCUS
- EN_UPDATE
- EN_VSCROLL

Eliminar controlador

Editar código

Funciones de miembro:

Nombre de función	Identificador de comando	Mensaje

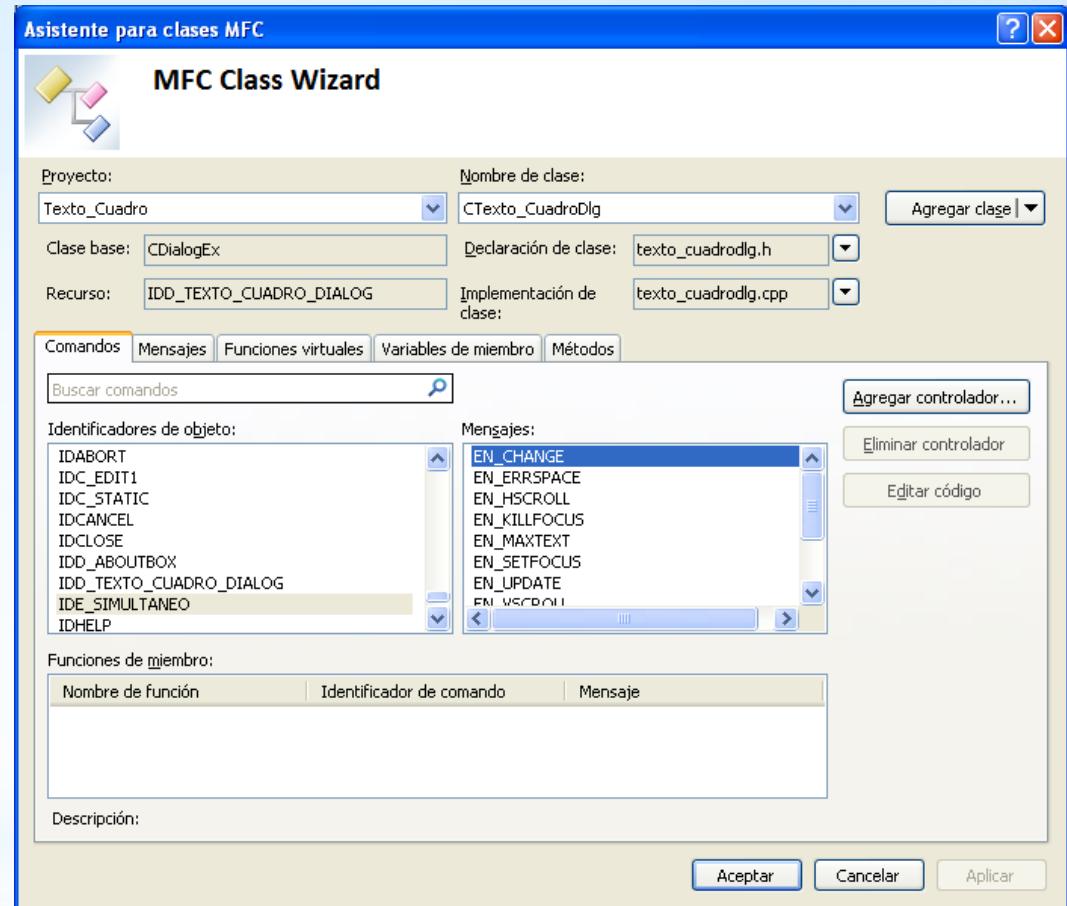
Descripción:

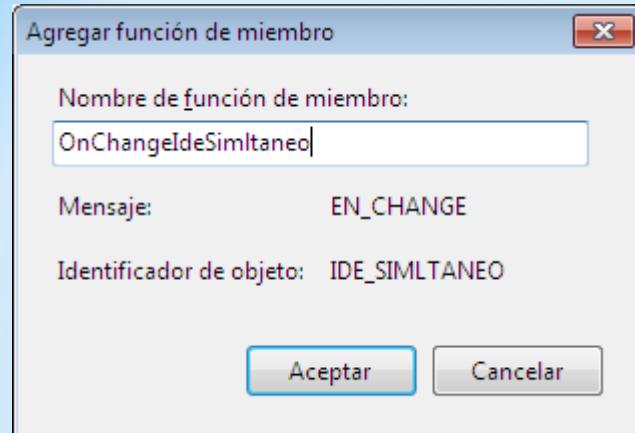
Aceptar

Cancelar

Aplicar

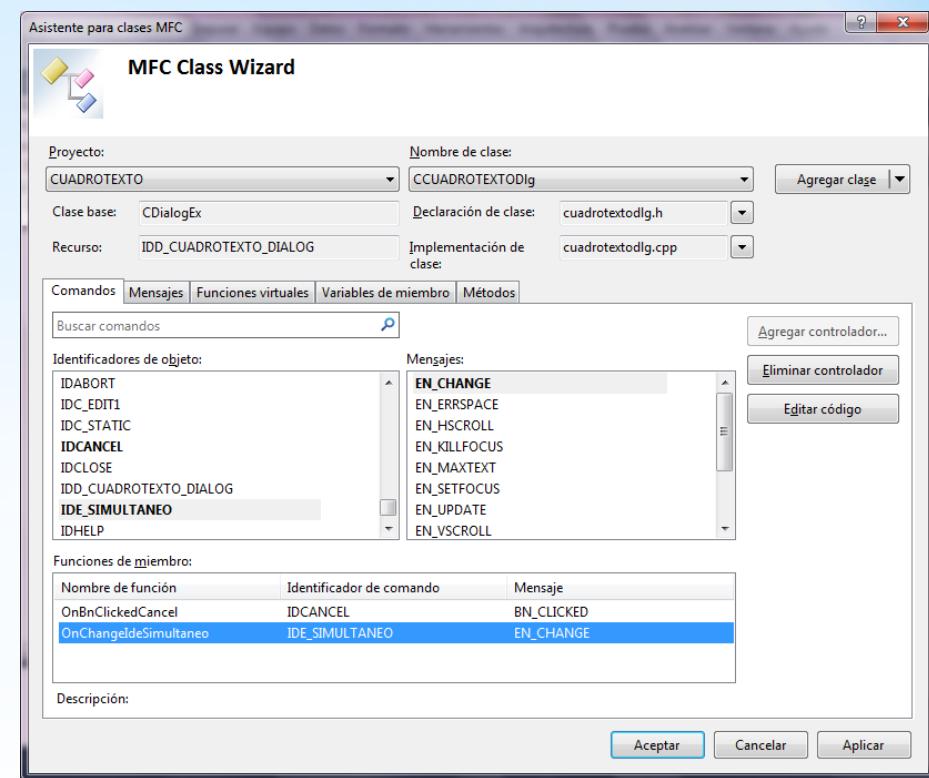
- El siguiente paso es unir el código a los objetos, para ello usaremos la MFC ClassWizar
- Como lo que se quiere es que al momento de estar escribiendo cada carácter del texto que se ingresa en el cuadro de edición, se deberá mostrar en la barra de título de la ventana.
- El objeto IDE_SIMULTANEO utilizara como mensaje EN_CHANGE, y adicione **Agregar controlador**, que pasara a formar parte de la lista Member Functions





➤ Una vez que ha adicionado la función deberá apreciarse la siguiente ventana

➤ En ella y haciendo clic en editar código escriba el siguiente código:



```
void CCUADROTEXTODlg::OnEnChangeSimltaneo(){}
```

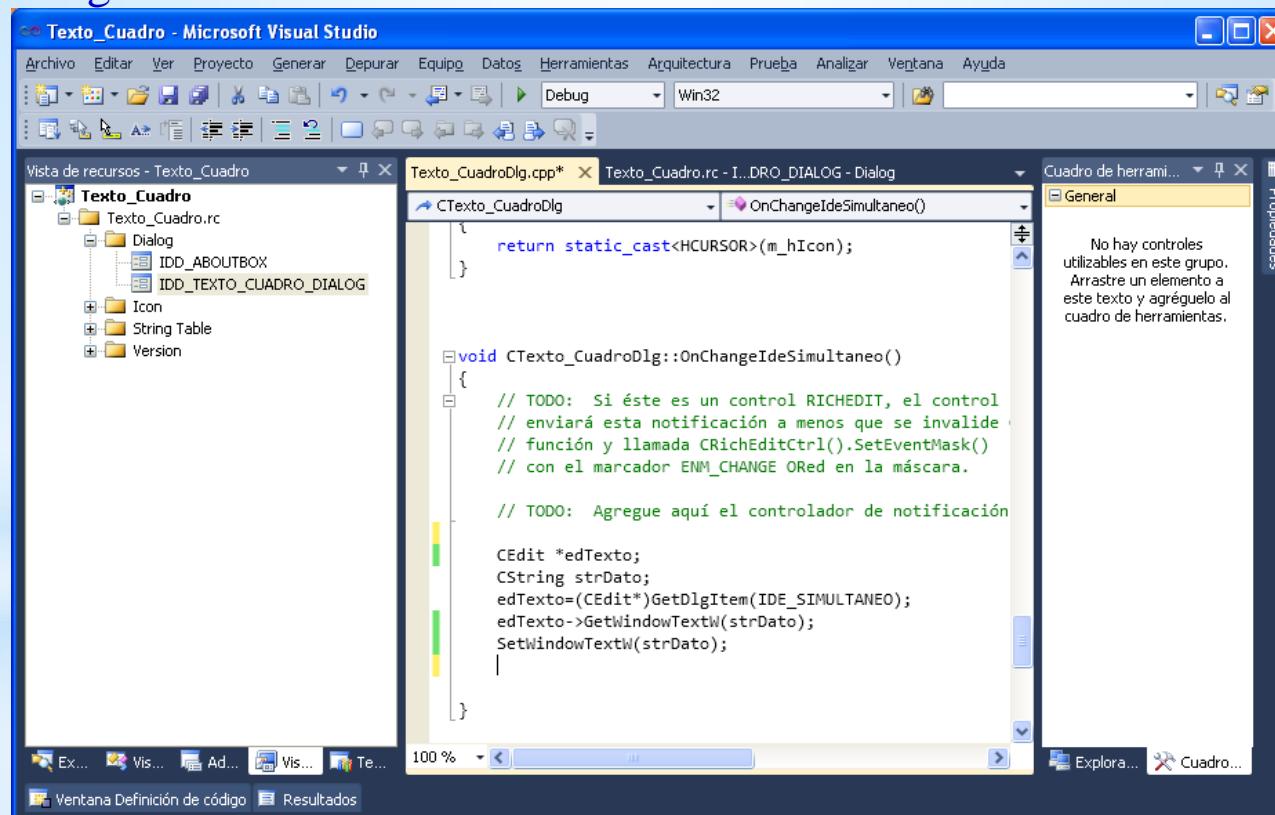
CEdit *edTexto;//La variable **edTexto** es declarada como un puntero de la clase **Cedit** y va hacer uso de las funciones miembro de **CEdit**

CString strData;//Se declara una variable **strData** del tipo **Cstring** para almacenar los caracteres que se van ingresando

edTexto=(CEdit*)GetDlgItem(IDE_EDIT1);//La función **GetDlgItem** pasa el **ID** del objeto a asociar con la variable; el problema es que **GetDlgItem** retorna un puntero a **CWnd** y uno a **Cedit**, por eso usa el operador (**CEdit***) que permite realizar la conversión al tipo deseado

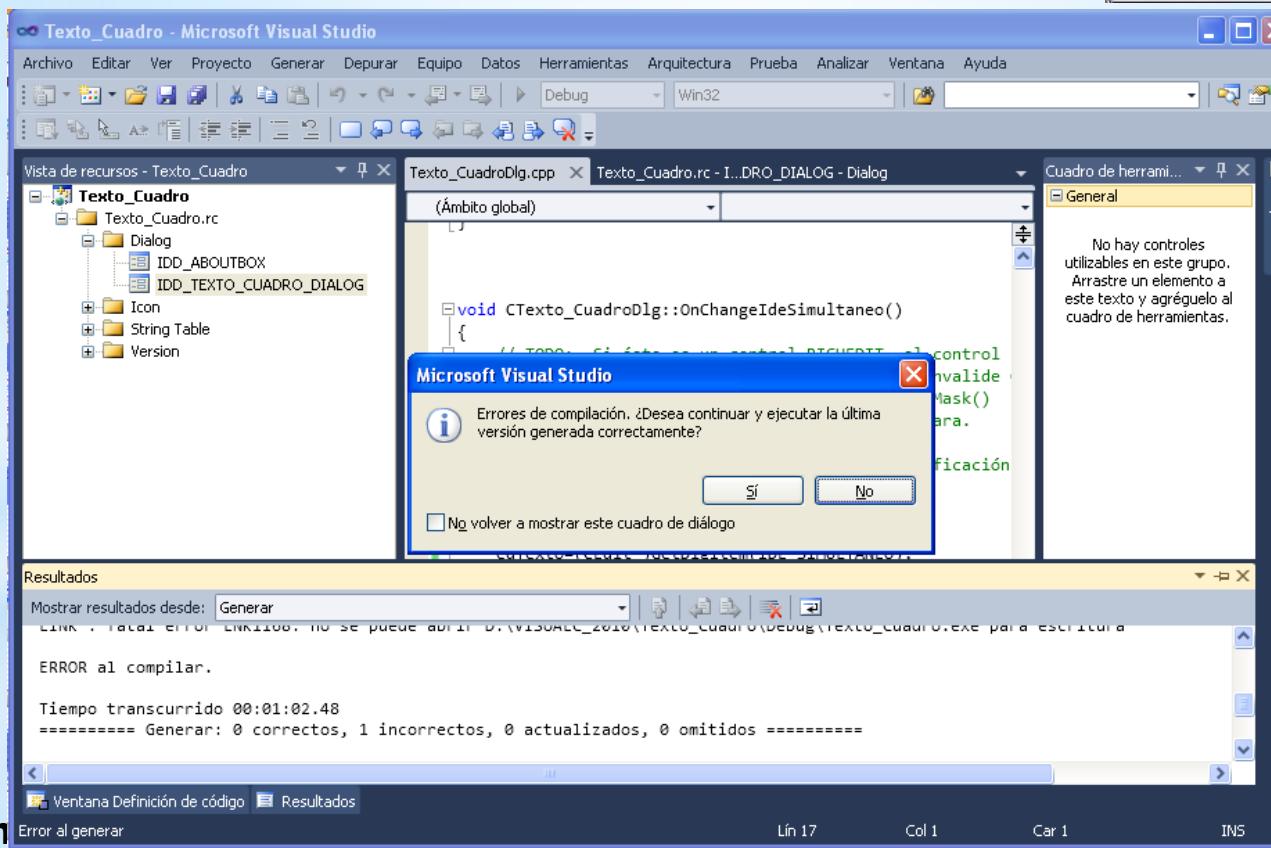
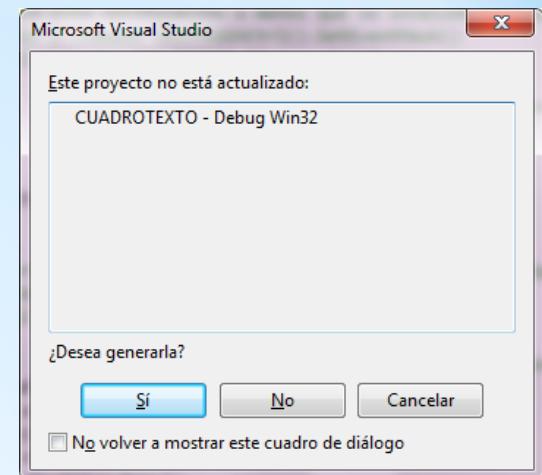
```
edTexto->GetWindowTextW(strData);  
//Obtiene el cuadro de texto de edición con GetWindowTextW a la cual se le pasa como parámetro la variable Cstring. A la función se le llama con el operador -> pues la variable es un puntero a CEdit.
```

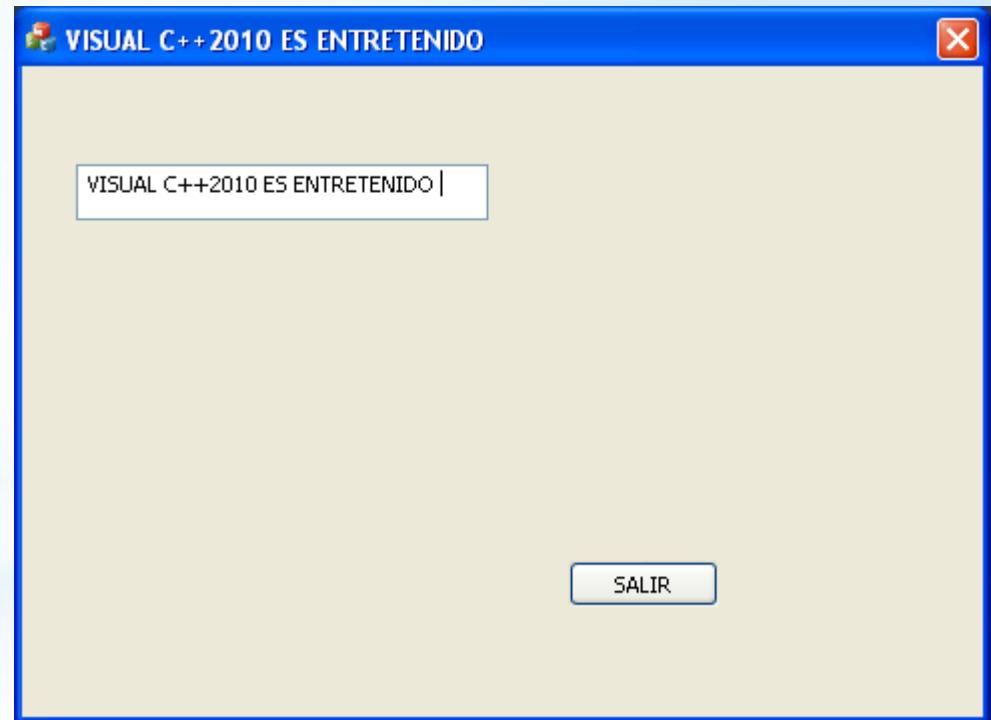
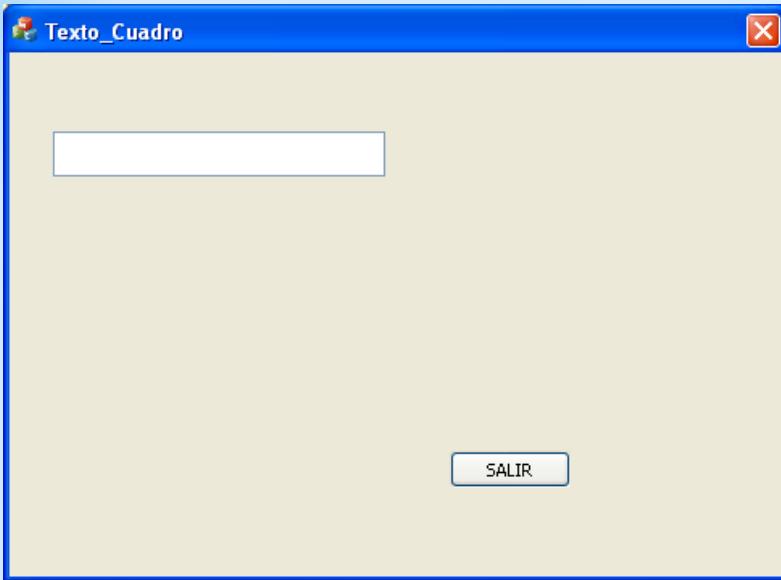
```
SetWindowTextW(strData);}// Se establece el título del cuadro de diálogo por medio de la función SetWindowText a la cual se le pasa la variable Cstring que desde la línea anterior viene cargada con el contenido del cuadro de edición.
```



Al generar la aplicación aparece la siguiente ventanita →

En el supuesto hubiese error tal como se aprecia hay que corregir





Debe quedar en claro que es posible manipular, mediante un código, los datos contenidos en un control, con la idea que después se pueden volver a volcar al control.

- Agregar un control button a la aplicación anterior, cambie su propiedad caption a "Regresar", el ID a "IDB_Regresar".
- Crear una variable miembro para el cuadro de Edición para ello ir al ClassWizard, seleccionando la pestaña "Member Variables"
- Pulse el botón Add Variable, el nombre a crearse será **m_strDato**, **Category Value** y **Variable Type Cstring**.
- Pulse la opción OK (acepte) de la ventana y también de la ventana del ClassWizard

cuando se escribe en la cajita de edicion aparece en la parte superior

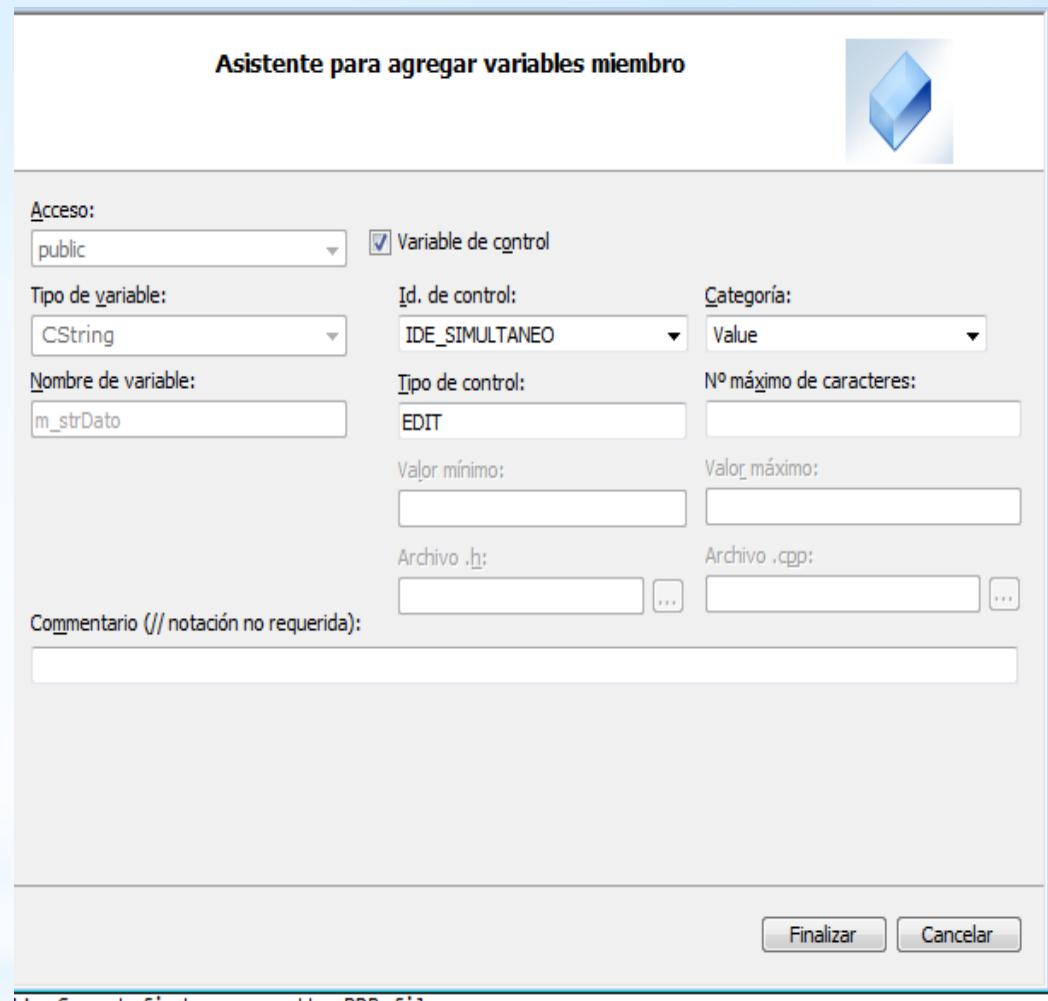
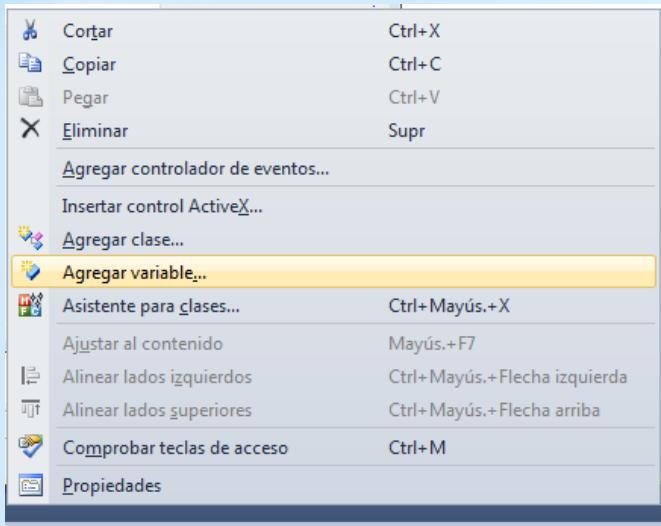
TEXTO QUE SE ESCRIBIRA

escribe en la cajita de edicion aparece en la parte superior

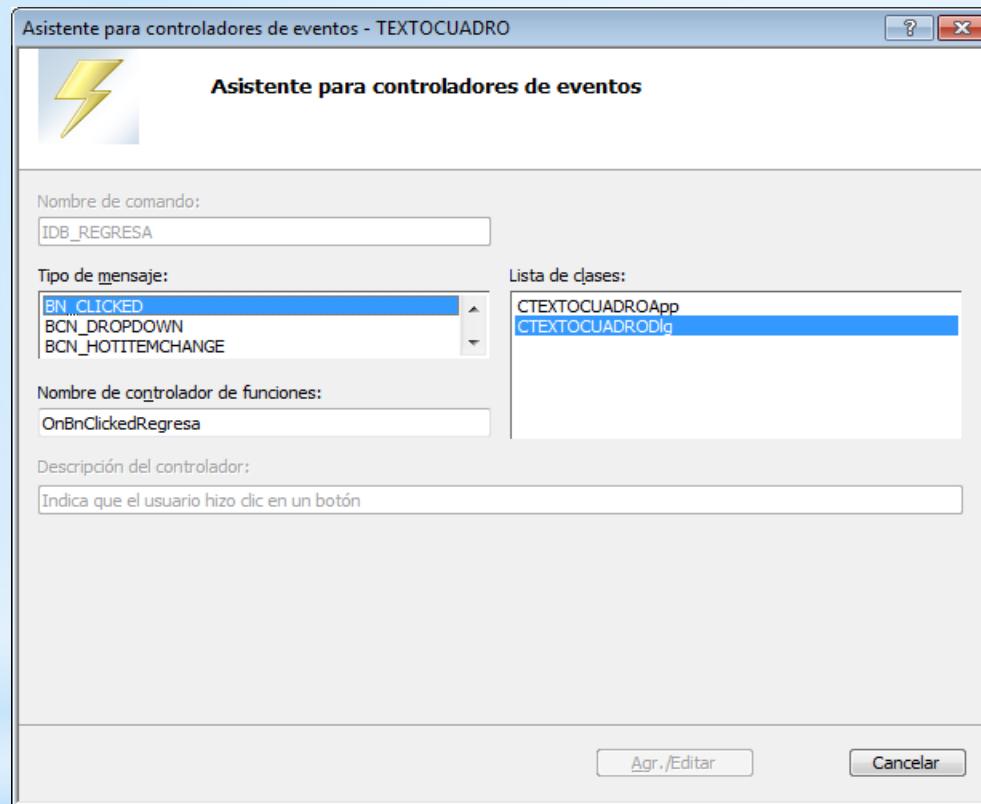
REGRESAR

Aceptar

Cancelar



➤ Ahora actuaremos sobre el botón "Regresar", ir al ClasWizard



- Seleccione de la pestaña Message Map el Object Ids IDB_REGRESAR.
- Message BN_CLICKED.
- Nombre de Controlador de funciones OnBnClickedRegresa

TEXTOCUADRO (Ejecutando) - Microsoft Visual Studio (Administrador)

Archivo Editar Ver Proyecto Generar Depurar Equipo Datos Herramientas VMware Arquitectura Prueba Analizar Ventana Ayuda

Proceso: Subproceso: Marco de pila:

TEXTOCUADRO.cpp x TEXTOCUADRO.rc - [I...acional]) - Dialog

CTEXTOCUADRODlg

```
// TODO: Agregue aqui el controlador de notificación de controles
CEdit *edTexto;
CString strData;
edTexto=(CEdit*)GetDlgItem(IDC_SINTANEO);
edTexto->GetWindowTextW(strData)
SetWindowTextW(strData);
}

void CTEXTOCUADRODlg::OnBnClickedRegresa()
{
    // TODO: Agregue aqui su código
    UpdateData(TRUE);
    if(m_strData.IsEmpty()==FALSE)
    { MessageBox(_T("Ahora se invertira"));
        m_strData.MakeReverse();
        UpdateData(FALSE); }
```

TEXTO QUE SE INVERTIRA
TEXTO QUE SE ESCRIBIRA
REGRESAR

Aceptar Cancelar

100 %

Resultados

Mostrar resultados desde: Depurar

```
'TEXTOCUADRO.exe': se cargó 'C:\Windows\System32\imm32.dll', Cannot find or open the PDB file
'TEXTOCUADRO.exe': se cargó 'C:\Windows\System32\msctf.dll', Cannot find or open the PDB file
'TEXTOCUADRO.exe': se cargó 'C:\Windows\System32\uxtheme.dll', Cannot find or open the PDB file
'TEXTOCUADRO.exe': se cargó 'C:\Windows\System32\dwmapi.dll', Cannot find or open the PDB file
'TEXTOCUADRO.exe': se cargó 'C:\Windows\System32\mfci00esn.dll', El binario no se generó con la información de depuración.
'TEXTOCUADRO.exe': se cargó 'C:\Windows\System32\shell32.dll', Cannot find or open the PDB file
```

Pila de llamadas Puntos de interrupción Resultados Lista de errores

Lista Lín1 Col1 Car1 INS

ES 04:56 p.m. 19/11/2013

Agregue el siguiente código dentro del botón en la función creada.

```
void CTexto_en_CuadroDlg::OnRegresar()
{
    UpdateData(TRUE);
    if(m_strDato.IsEmpty() == FALSE)
    {
        MessageBox("Ahora se invertira:" + m_strDato);
        m_strDato.MakeReverse();
        UpdateData(FALSE);
    }
}
```

TEXTOCUADRO

X

Ahora se invertira: TEXTO QUE SE INVERTIRA

Aceptar

TEXTO QUE SE INVERTIRA

TEXTO QUE SE ESCRIBIRA

ARITREVNI ES EUQ OTXET

REGRESAR

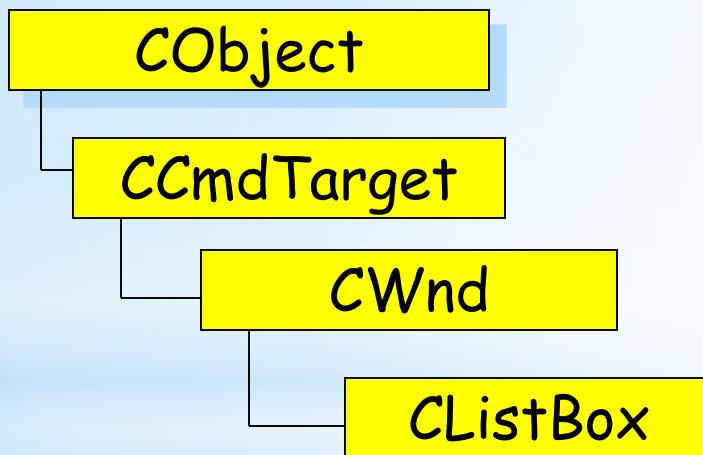
Aceptar

Cancelar

23/08/2019

La Clase CListBox

- La clase Base MFC CListBox define los controles llamados Listas
- Un Control lista es una ventana hija, rectangular, que visualiza una lista de elementos.
- El usuario puede añadir o suprimir elementos de la lista, los cuales forman una columna desplazable dentro del rectángulo.



- Un objeto de la clase **CListBox** se puede crear desde una plantilla de dialogo o directamente con código.
- El siguiente paso es colocar cadenas de texto.
- Se usa la función miembro **AddString** esto cuando la lista esta clasificada o bien la función **InsertString**.
- La cantidad de elementos se obtiene con **GetCount**.

- Al seleccionar un elemento de la lista se utiliza la función miembro **SetCurSel** o la función miembro **SelectString**. En cualquier elemento, la función **GetCurSel** devuelve el índice del elemento seleccionado (el primer elemento tiene índice 0).

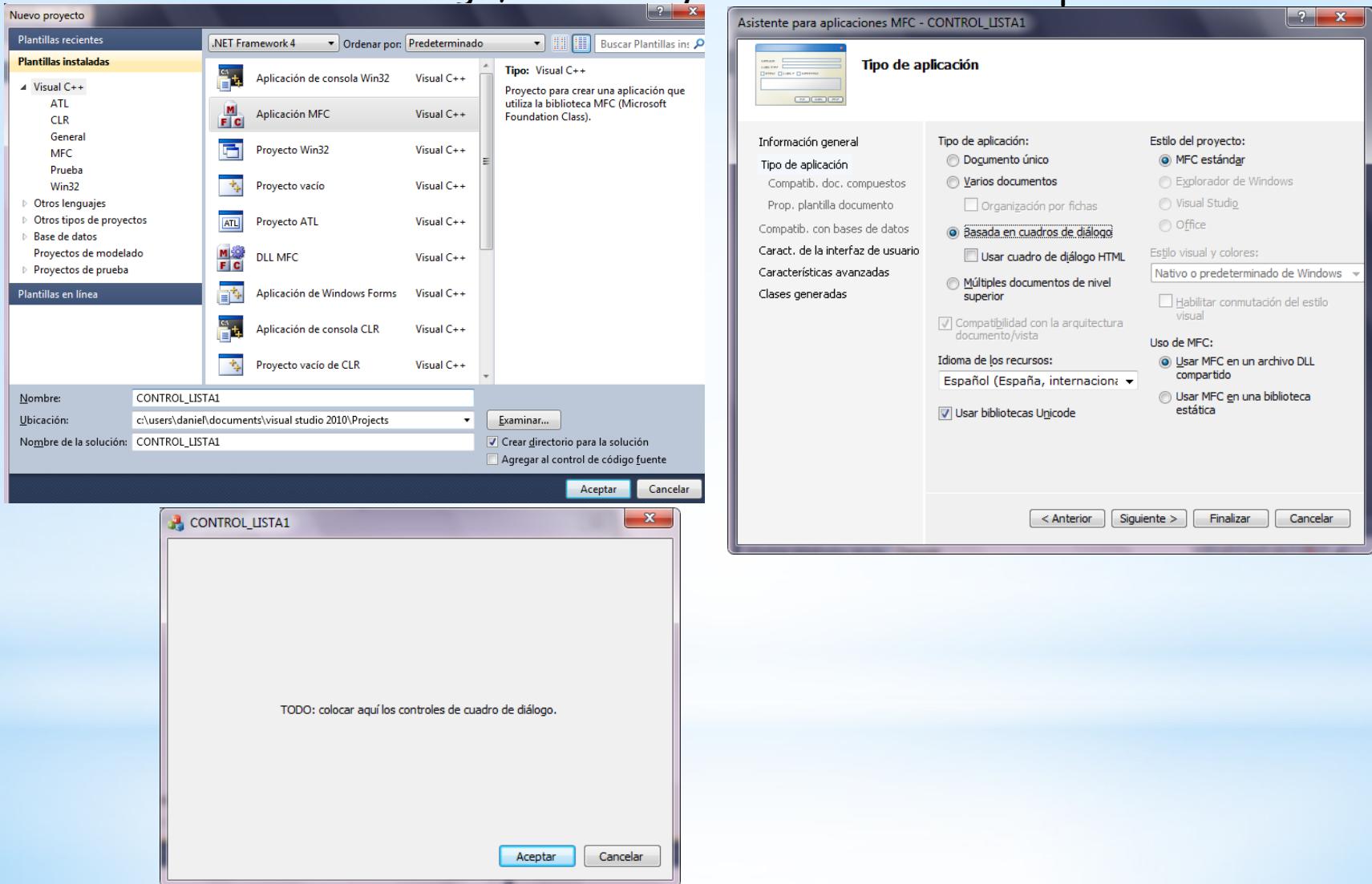
- Un elemento de la lista se puede copiar en un buffer utilizando la función miembro **GetText** y se puede borrar utilizando la función miembro **DeleteString**.
- **ResetContent()** → Remueve todos los elementos de la lista
- Para crear una lista, primero se construye un objeto de la clase **CListBox** o de la clase derivada y después se llama a la función **Create**.

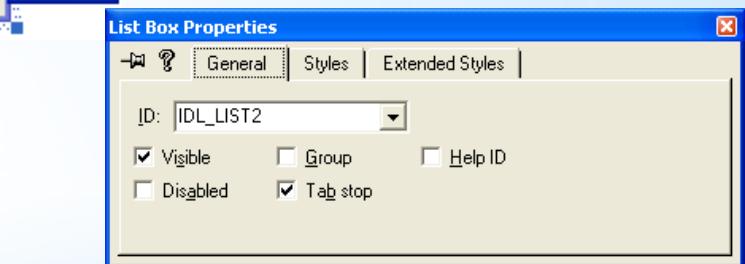
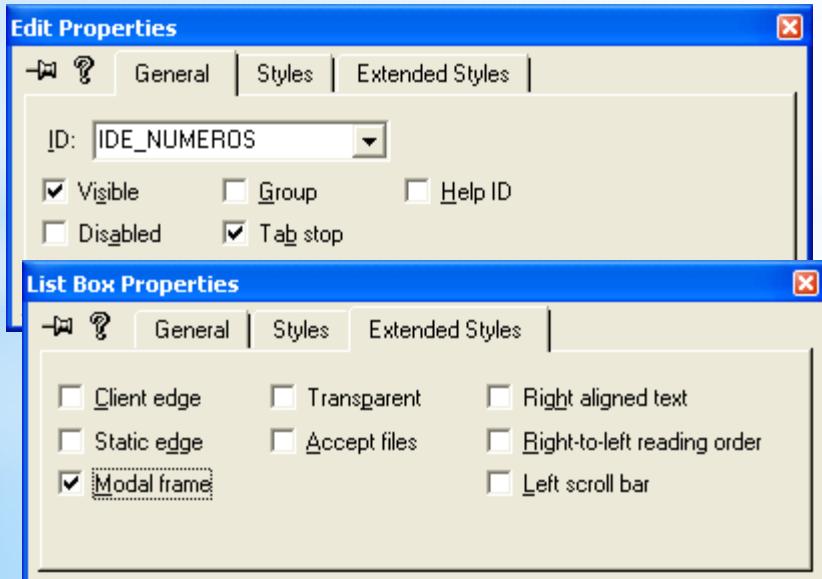
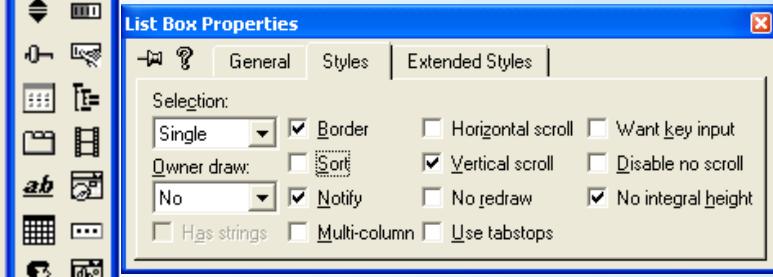
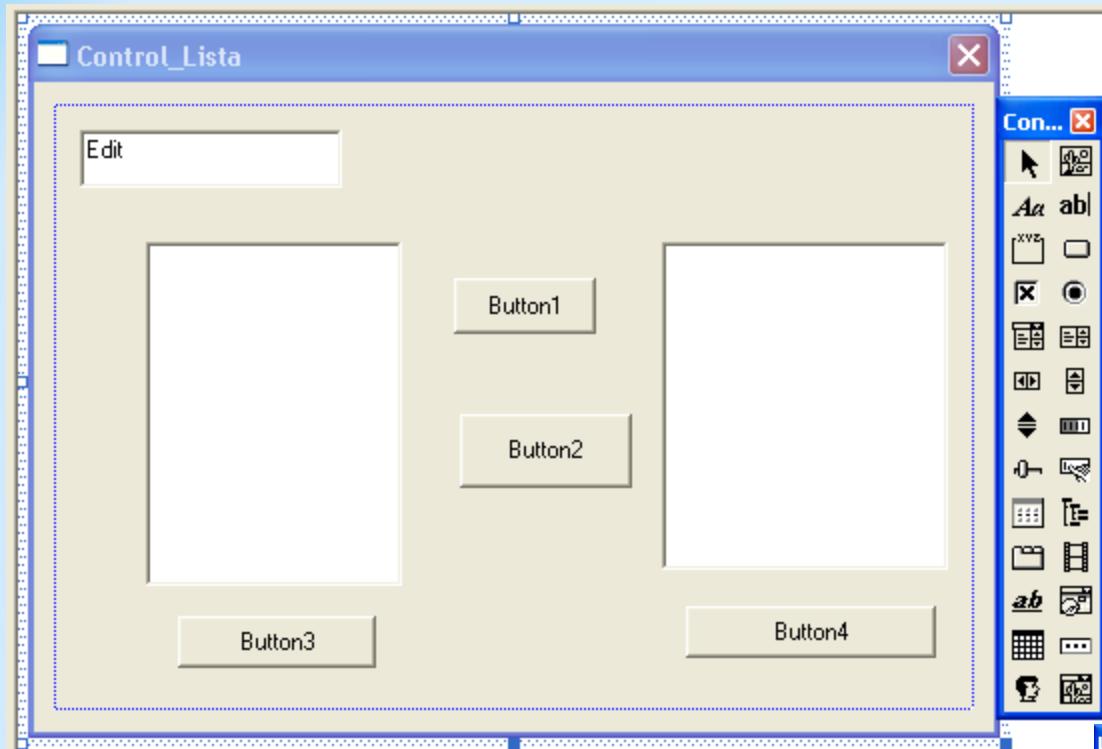
➤ Clase **CListBox**

- La clase **CListBox** define controles llamados listas.
- Un control lista es una ventana hija, rectangular, que visualiza una lista de elementos.
- Un control lista muestra una lista de elementos, que el usuario puede ver y seleccionar.
- En un control lista con selección simple, el usuario puede seleccionar solamente un elemento.
- Si hay selección múltiple, un rango de elementos puede ser seleccionado.
- Al seleccionar un elemento este es iluminado y el control lista manda un mensaje de notificación hacia la ventana padre.

Una aplicación con Listas

Crear la aplicación, para ello use la MFC basada en dialogo, tal como ya se crearon otras aplicaciones.





Control_Lista - Microsoft Visual C++ - [Control_Lista.rc - IDD_CONTROL_LISTA_DIALOG (Dialog)]



File Edit View Insert Project Build Layout Tools Window Help



CControl_ListaDlg

(All class members)

CControl_ListaDlg



Control_Lista resources *

- + Dialog
- + Icon
- + String Table
- + Version

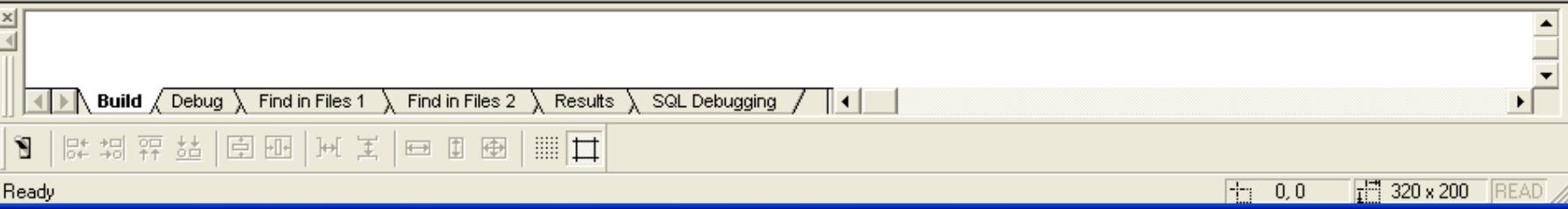
Control_Lista

Digite la cantidad de Numeros

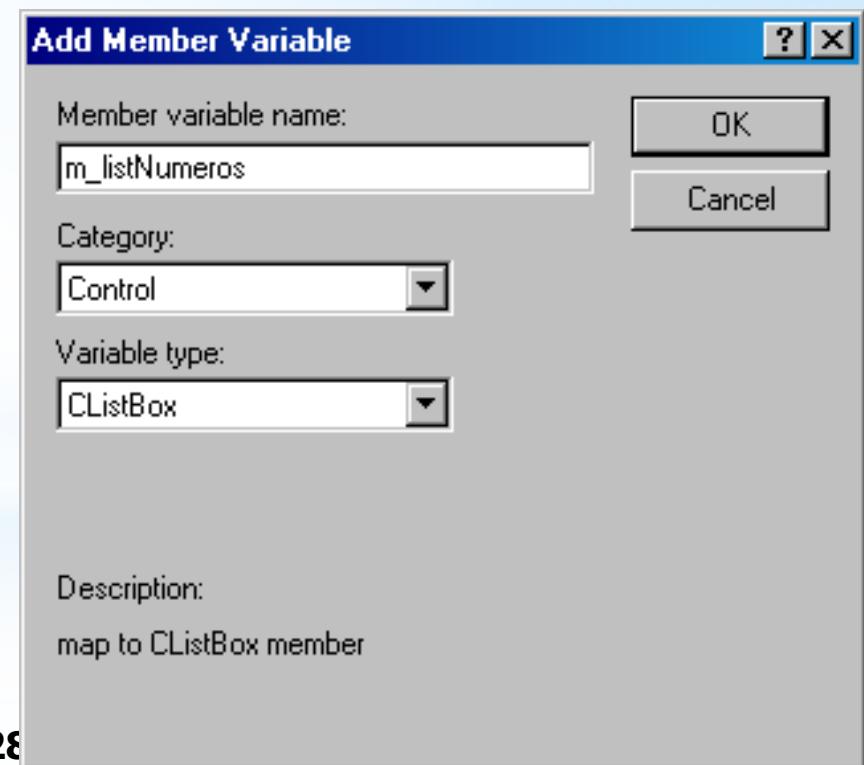
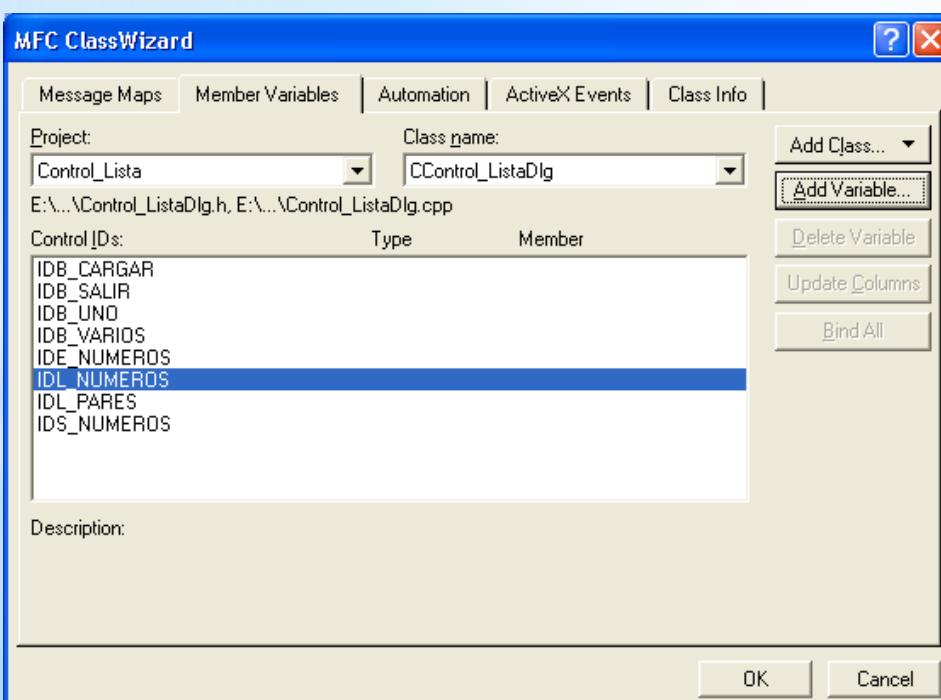
Con...

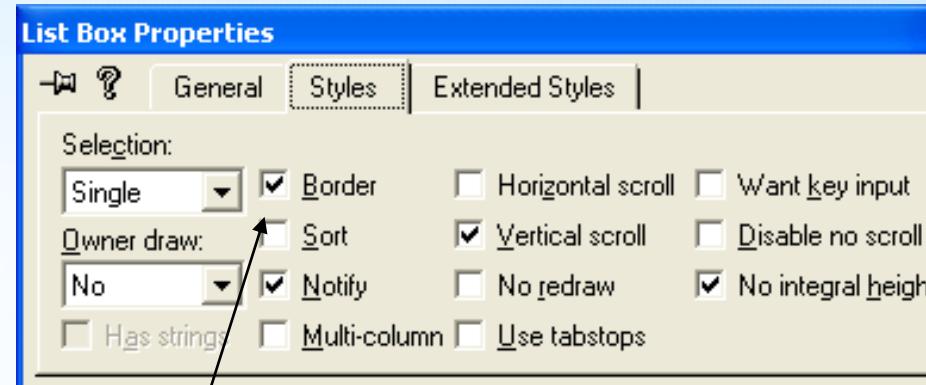
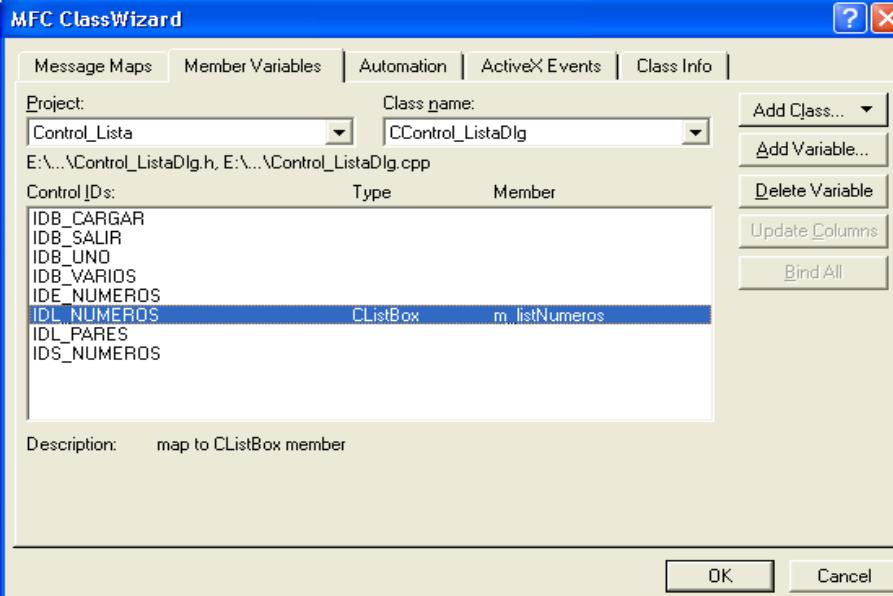


ClassVi... Resour... FileView



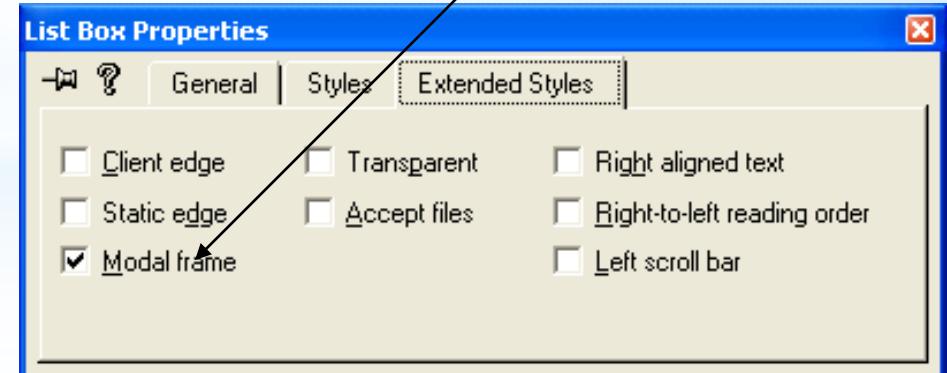
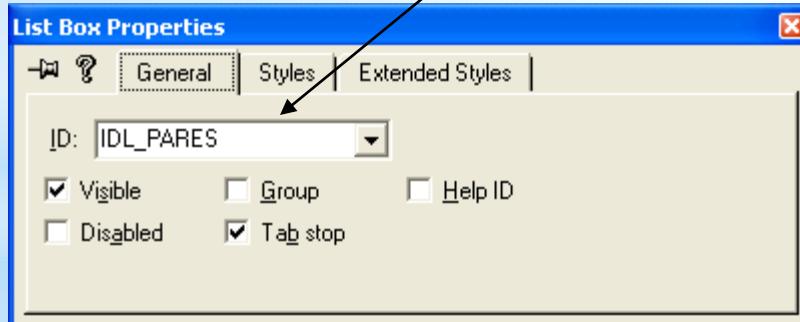
- En la lista de la derecha desmarque la opción Sort.
- En Extended Style marque Modal Frame
- Seleccione la lista de la Izquierda y presione Ctrl W(ClassWizard).
- En Member Variables marque IDL_NUMEROS y presione Add Variable.
- En Name m_lstNumeros, en Category seleccione Control y Tipo CLListBox.





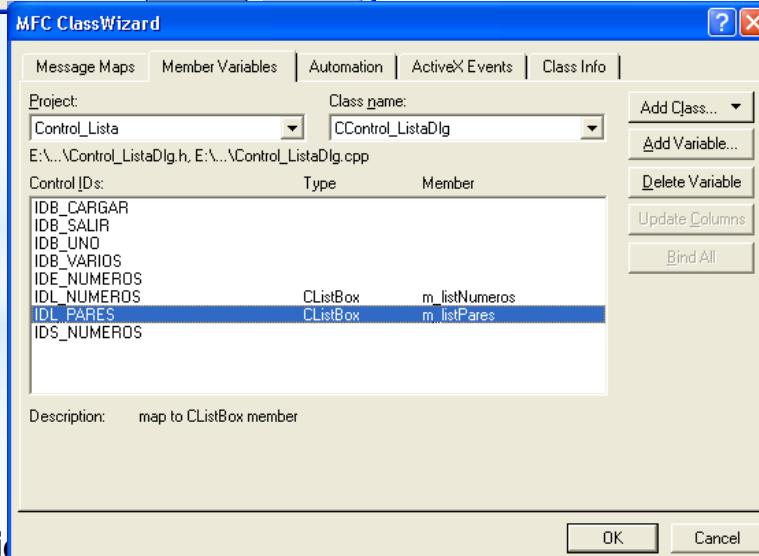
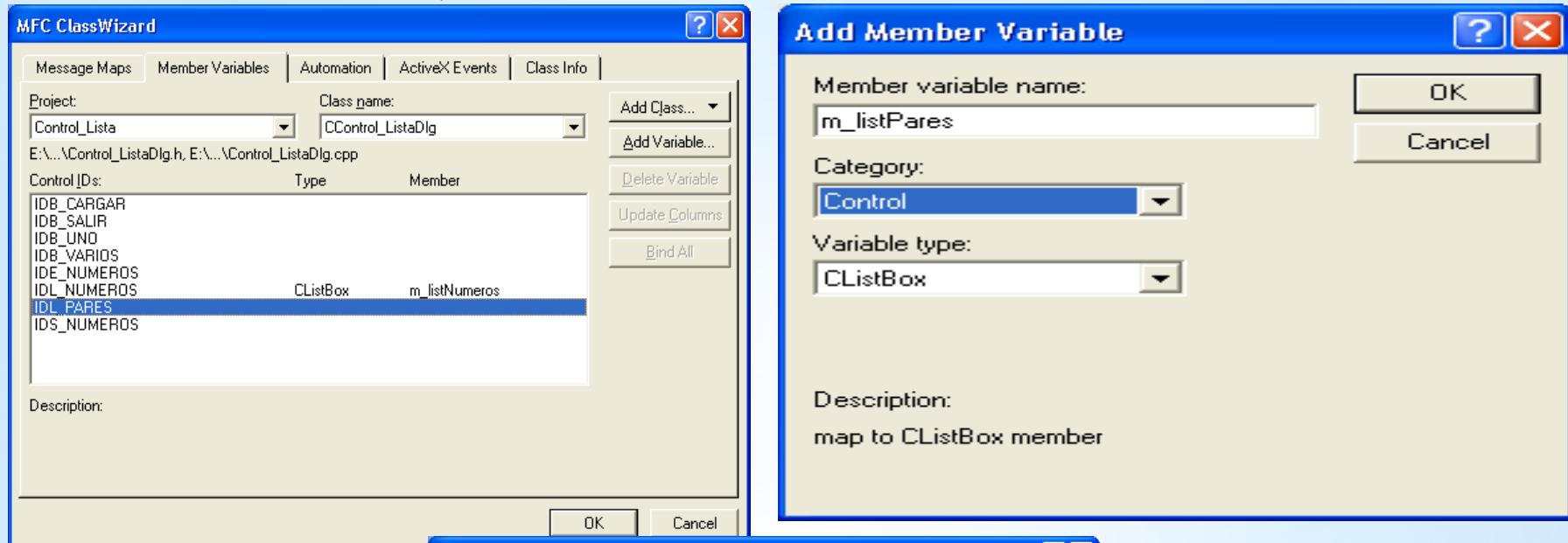
Propiedades de la lista de la derecha:

➤ El ID debe tener . En Style desmarque Sort. En Extended



➤ Nuevamente como en la lista anterior active Ctrl+W

➤ Ir a la solapa Member Variables, marque IDL_PARES



➤ Active ahora el botón >> cambie las propiedades



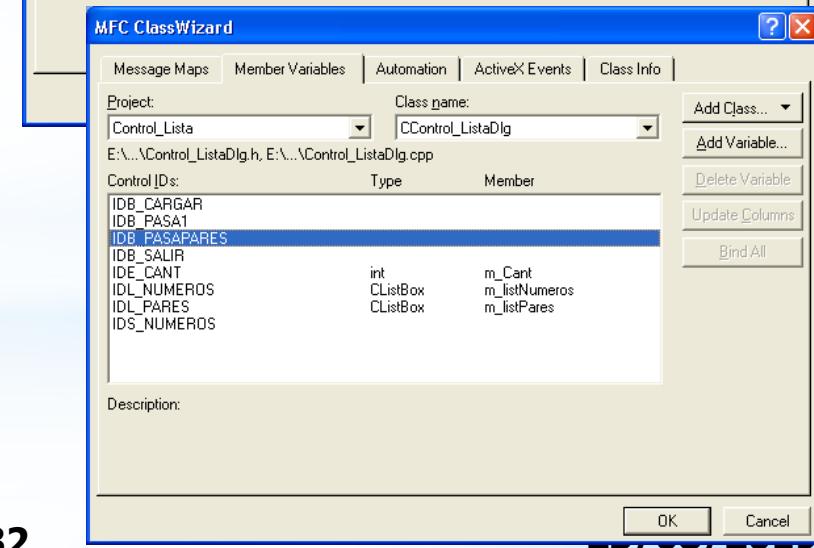
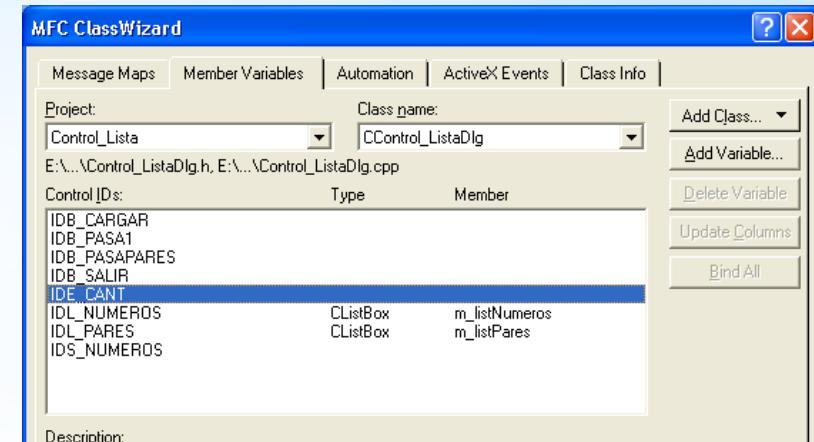
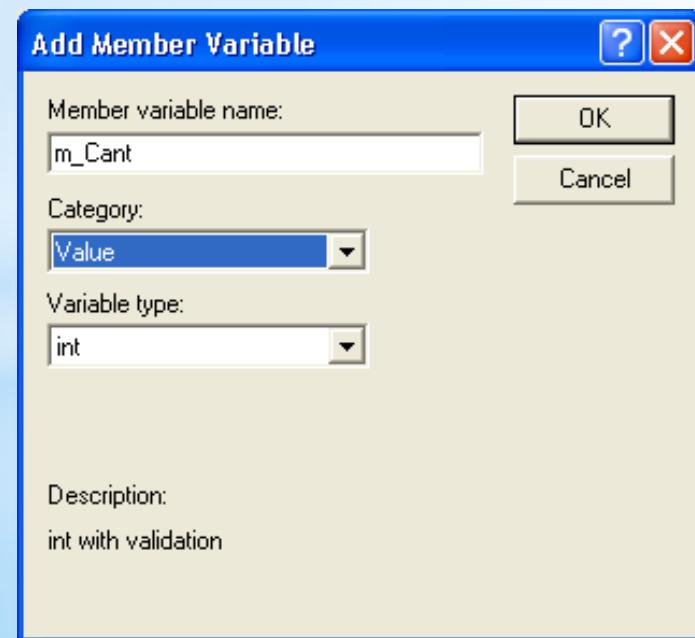
➤ Active ahora el botón > de abajo y modifique las propiedades.



➤ Ahora ir a la ventanita del cuadro de edición Edit.

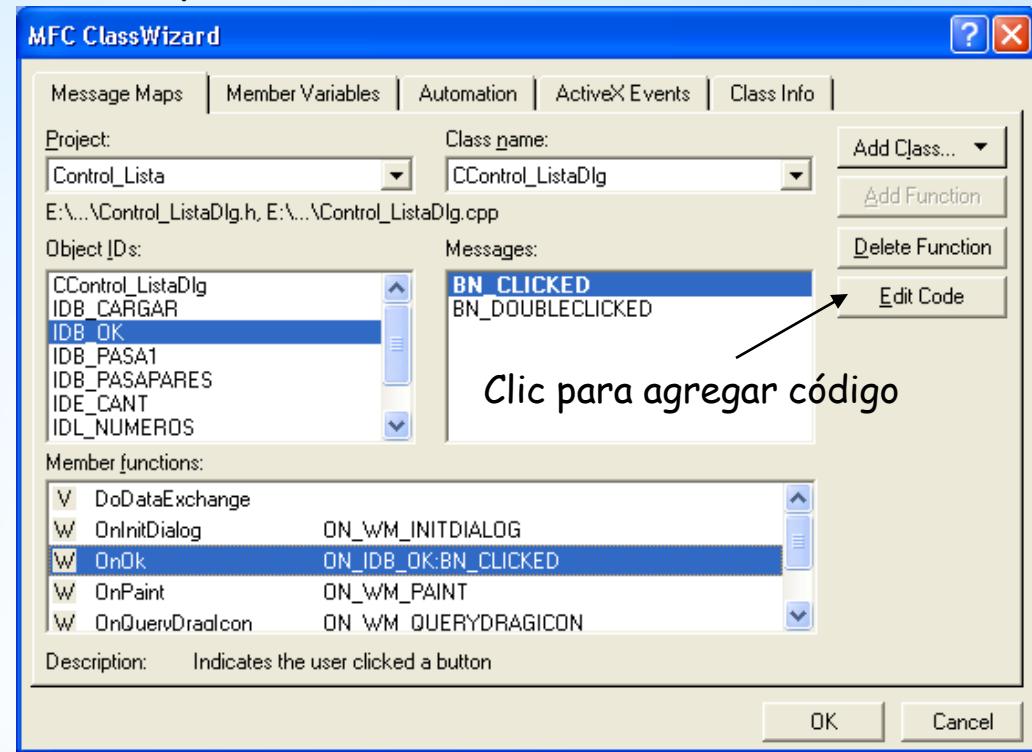
➤ Escriba el ID IDE_CANT.

➤ Cree una variable miembro para el cuadro de Edición de tipo int Category value y de nombre m_Cant, vea la secuencia.



Introduciendo El Código de la Aplicación

➤ Modifique el Botón OK(Aceptar)



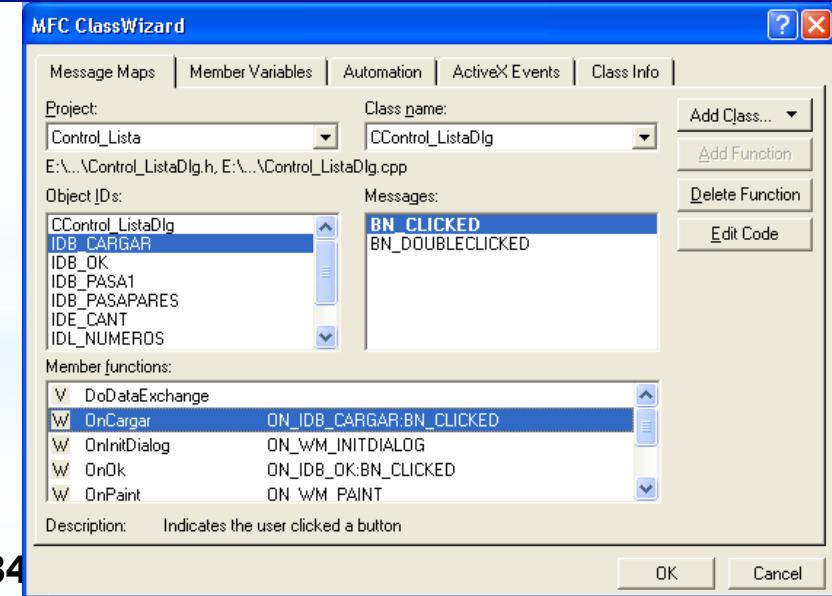
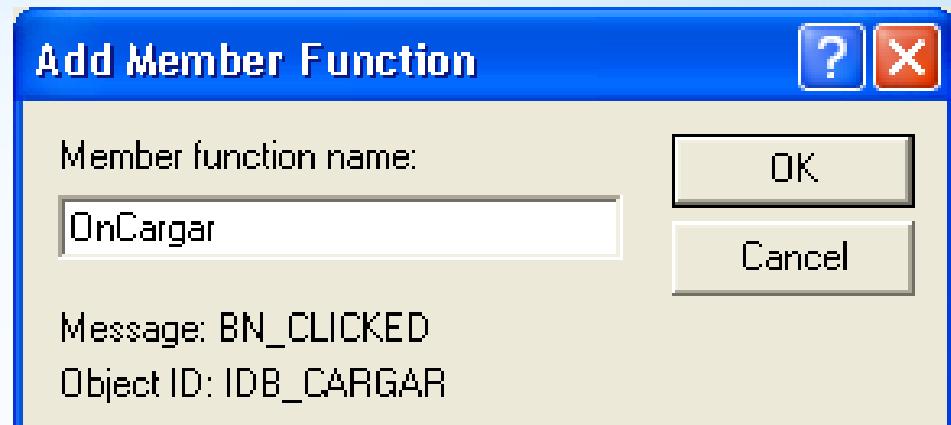
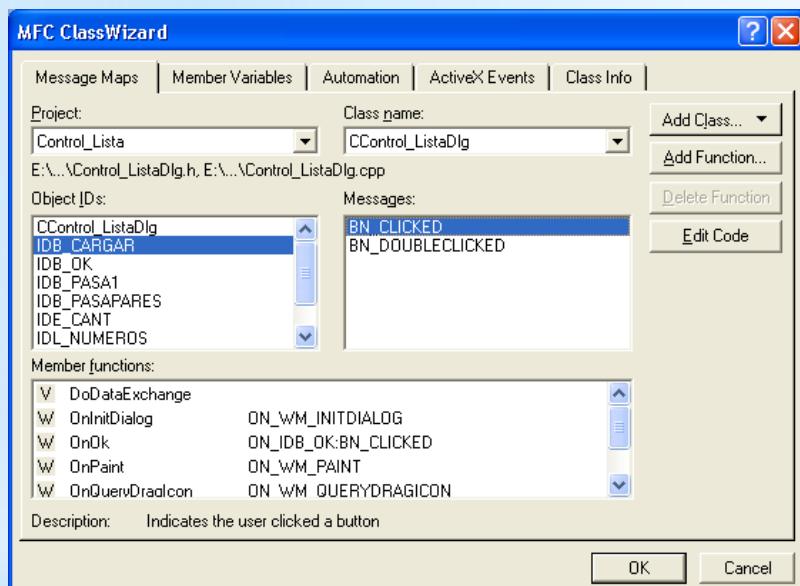
➤ Encontrara la siguiente función, donde agregara código

```
void CControl_ListaDlg::OnOk()
{if(MessageBox("Desea salir?","Salir",MB_ICONQUESTION+MB_YESNO)==IDYES)
CDialog::OnOK();}
```

➤ Active el botón Cargar luego acceda al ClassWizard.

➤ Seleccione Message Box, luego en Objects ID's marcar IDB_CARGAR.

➤ En Message marcar BN_CLICKED y entonces presione Ad Function luego edit Code



➤ Al editar la Función se aprecia el siguiente código, esta lista para agregar la rutina.

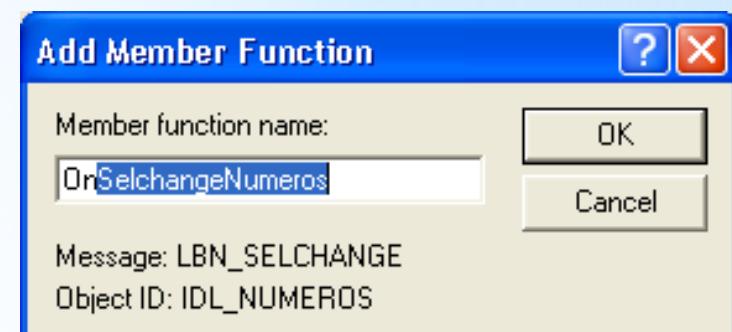
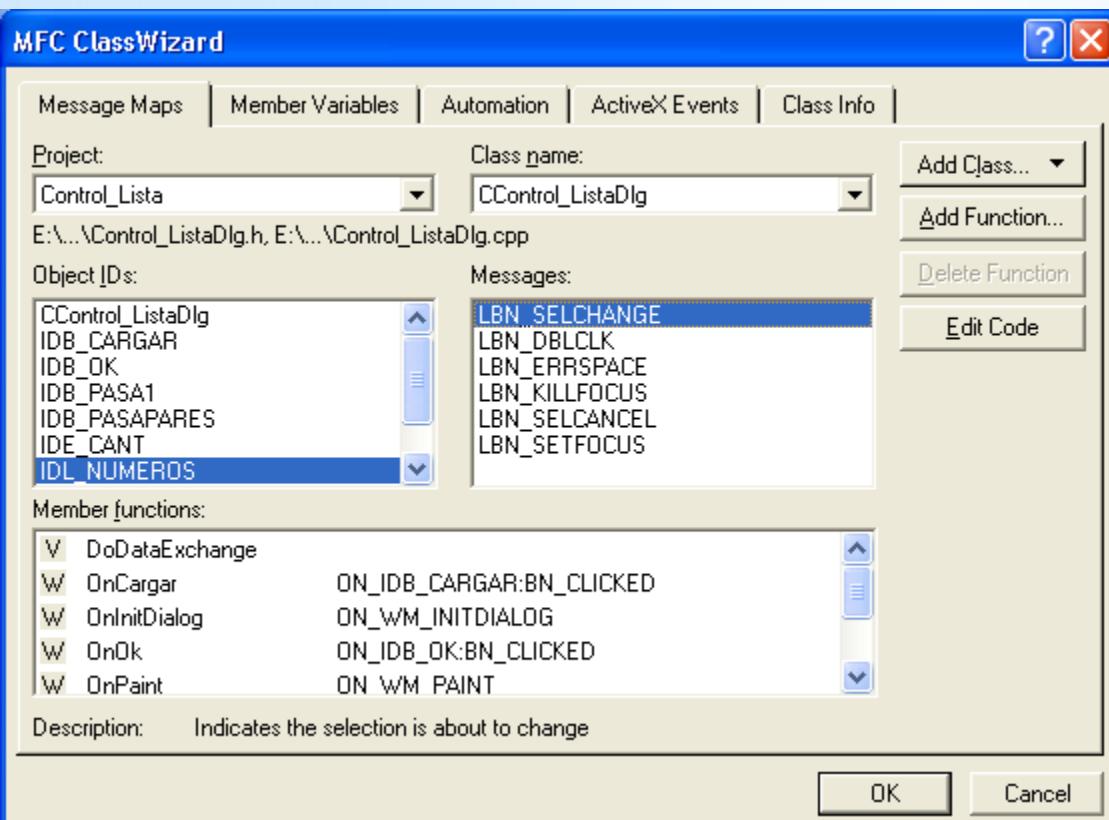
```
void CControl_ListaDlg::OnCargar()
{
CString c; int i;
    UpdateData(TRUE);
    m_listNumeros.ResetContent();
    for(i=1;i<=m_Cant;i++) {
        c.Format("%s%i",c,i);//formatea i como String
        m_listNumeros.AddString(c);//Se agrega a la lista
        c=" ";
    }
}
```

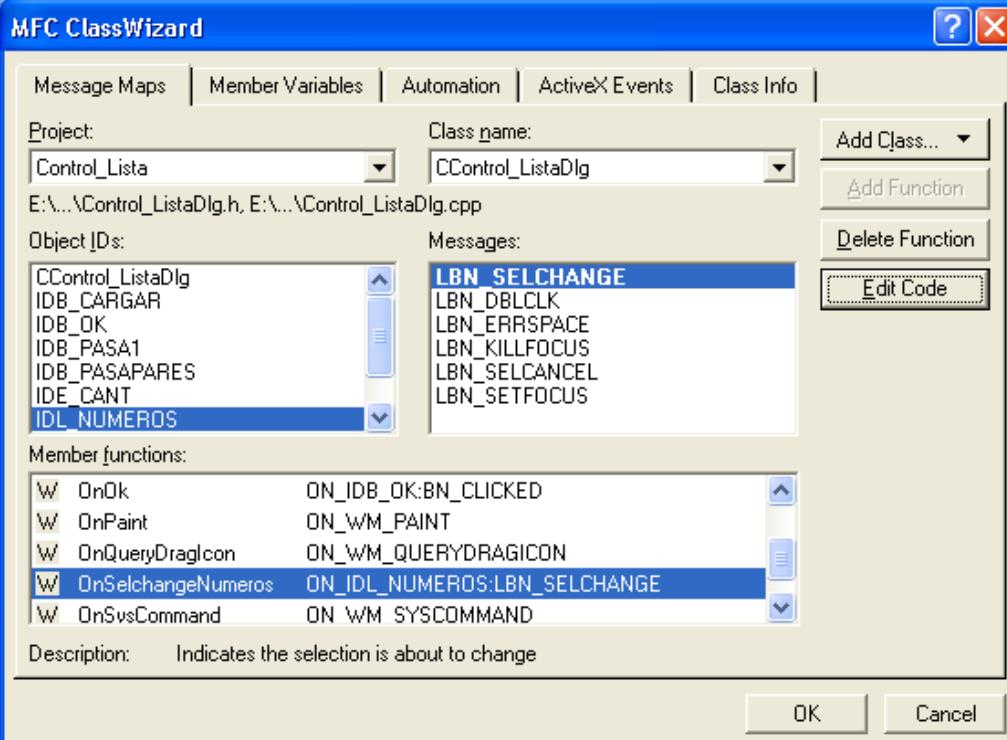
➤ Acceda a lista nuevamente Ctrl +W luego Message Maps con IDL_NUMEROS

➤ En Object ID's Seleccione LBN_SELCHANGE en Message.

➤ Pulse Add Function y luego Edit Code

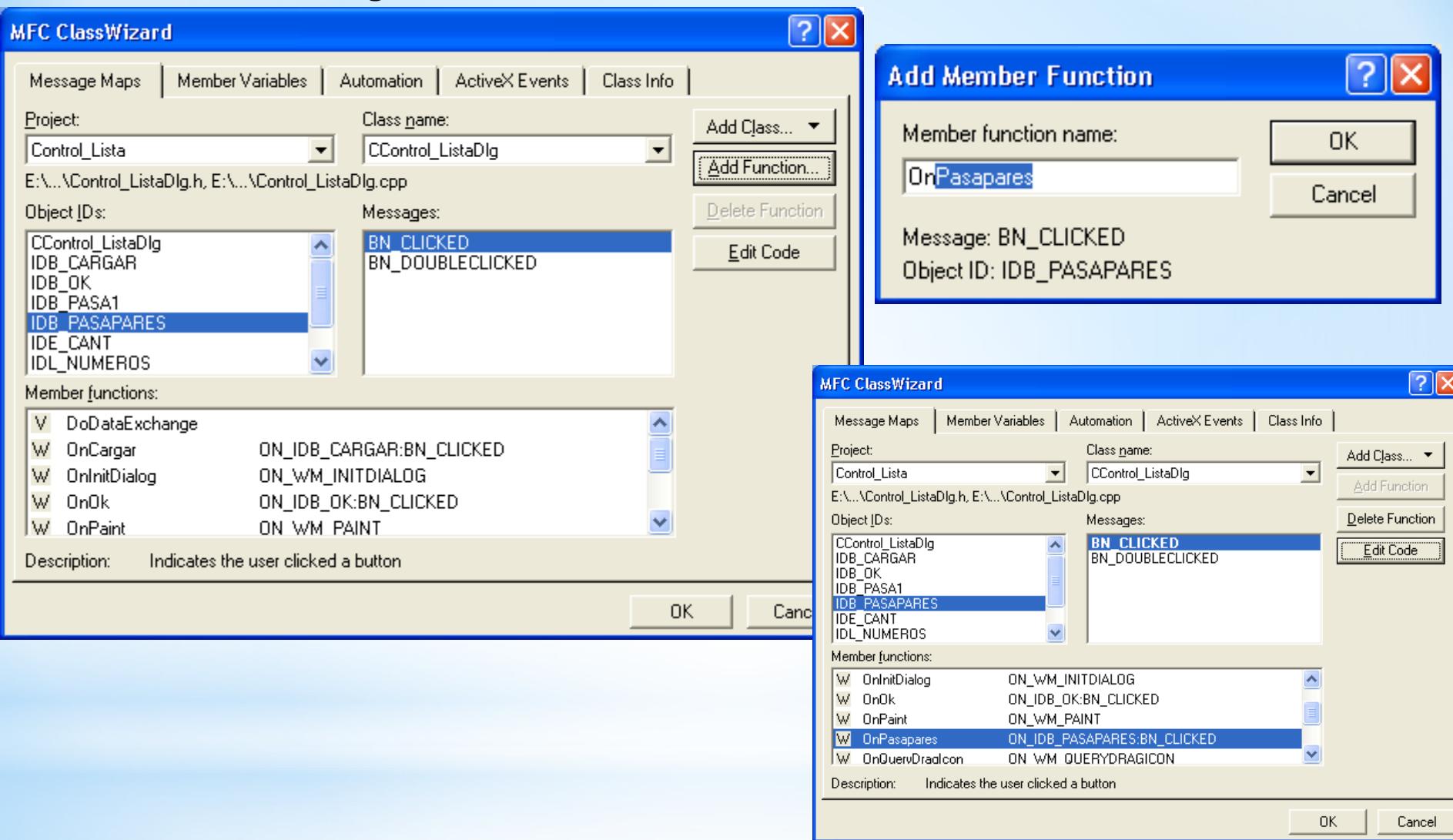
➤ Escriba el siguiente código





```
void CControl_ListaDlg::OnSelchangeNumeros()
{
    // TODO: Add your control notification handler code here
    CString strTexto;
    m_listNumeros.GetText(m_listNumeros.GetCurSel(),strTexto);
    MessageBox(strTexto);
}
```

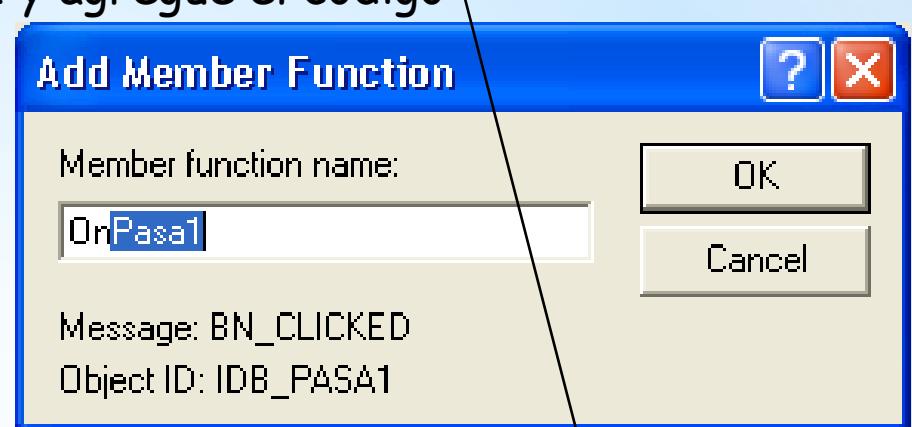
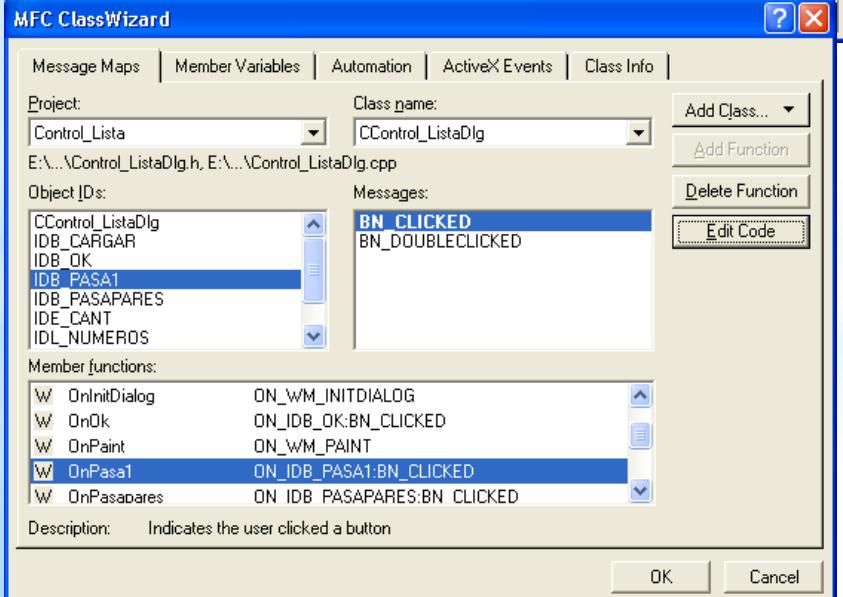
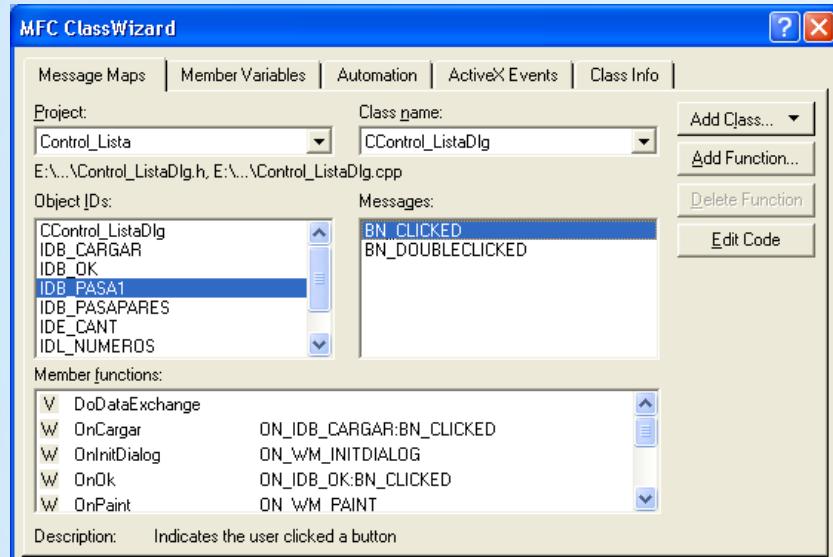
➤ El código del Botón >> (IDC_PASAPARES)



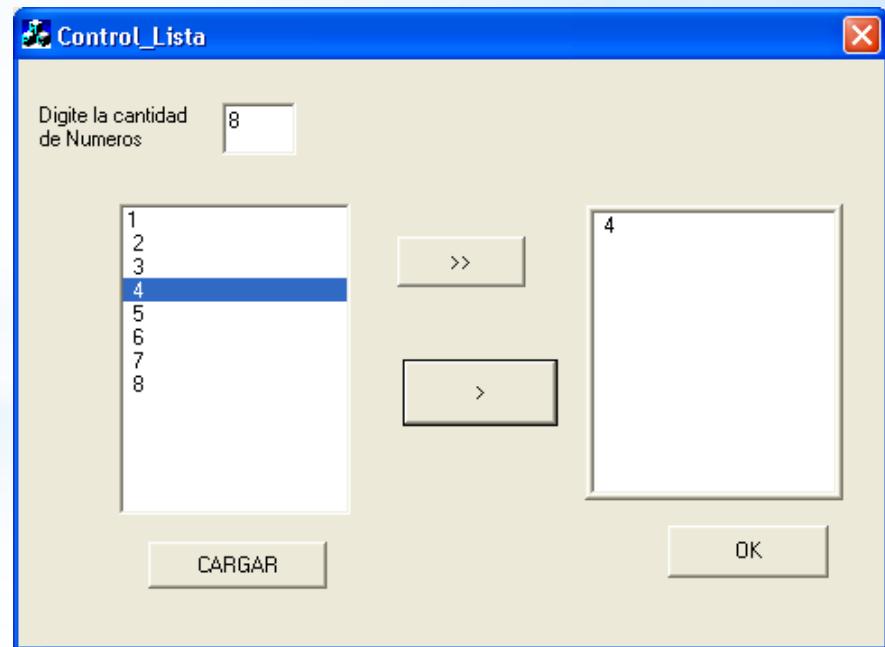
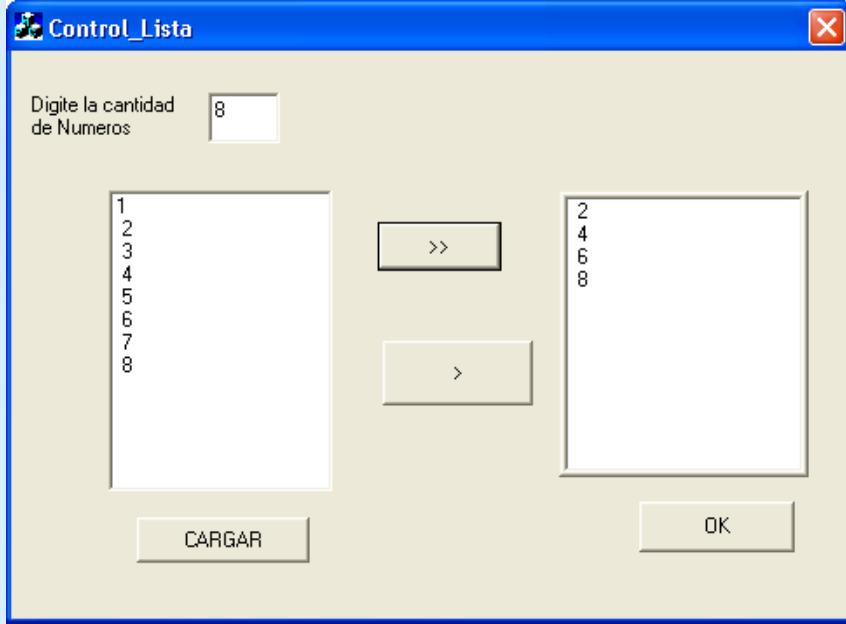
```
void CControl_ListaDlg::OnPasapares()
{ int i,n,ce;
  CString strC;
  ce=m_listNumeros.GetCount();
  if(ce>0)
    for(i=0;i<ce;i++){
      m_listNumeros.GetText(i,strC);//obtiene el elemento
      n=atoi(strC);// convierte a numerico
      if(n%2==0) //ver si es par
        m_listPares.AddString(strC); //;pasa a la otra lista}
}
```

➤ Activar el boton de > (pasa1)

➤ Crear una funcion OnPasa1 y agregue el codigo



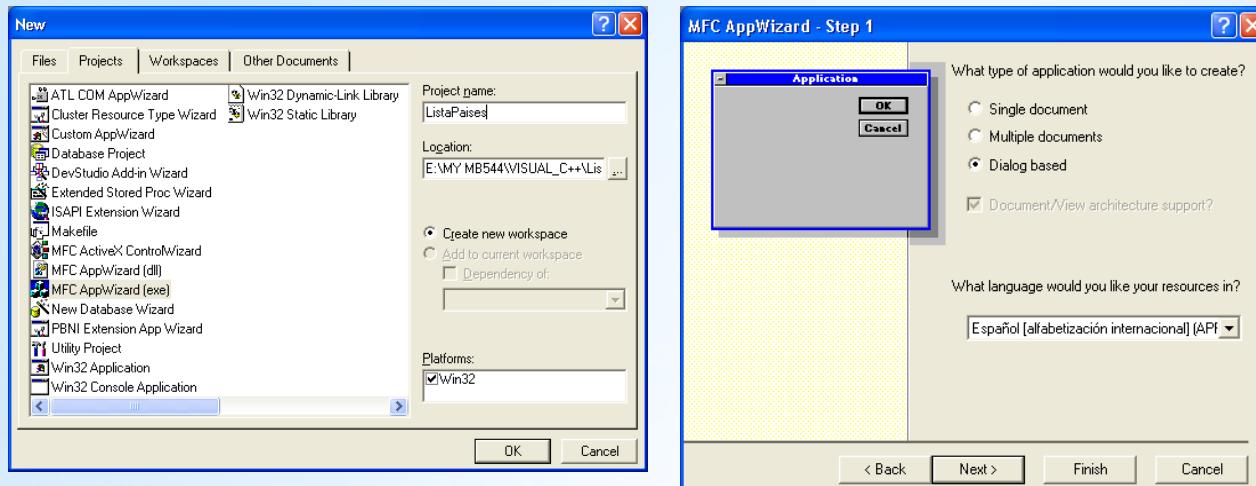
```
void CControl_ListaDlg::OnPasa1()
{
    int n;
    CString strC;
    m_listNumeros.GetText(m_listNumeros.GetCurSel(),strC);
    n=atoi(strC);
    if(n%2==0)
        m_listPares.AddString(strC);
}
```



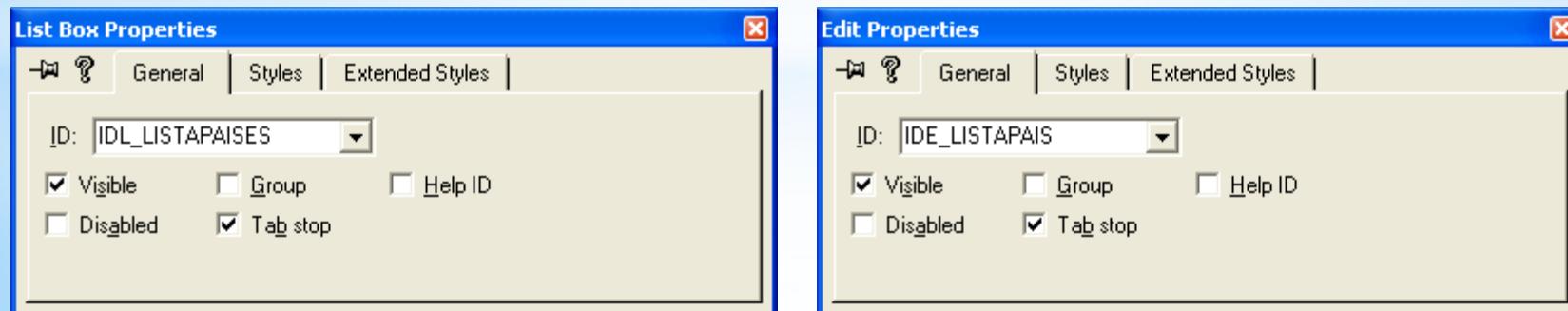
Ingeniero Daniel Osorio Maldonado

Una aplicación que utiliza a una lista para seleccionar un elemento y enviarlo a un cuadro de texto

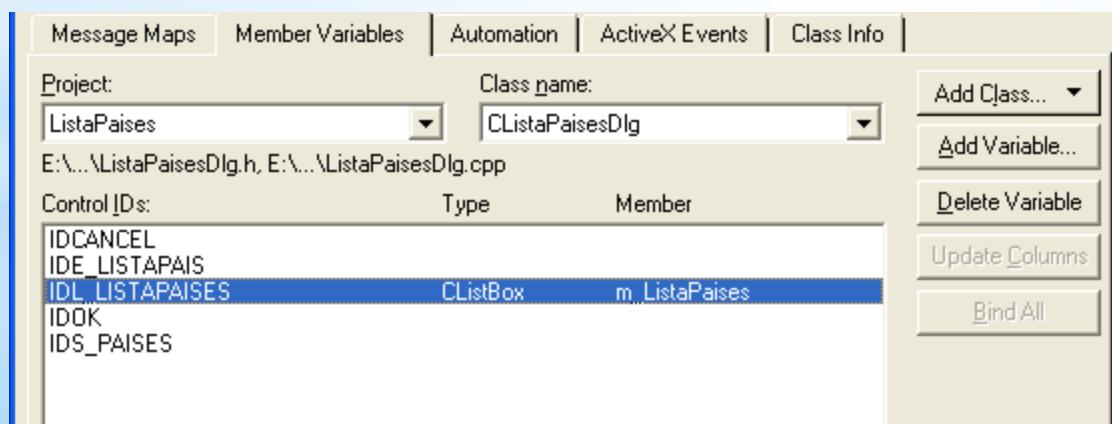
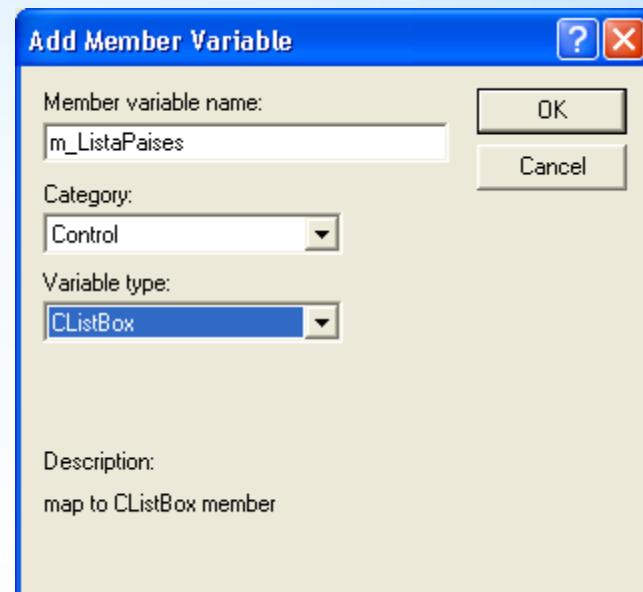
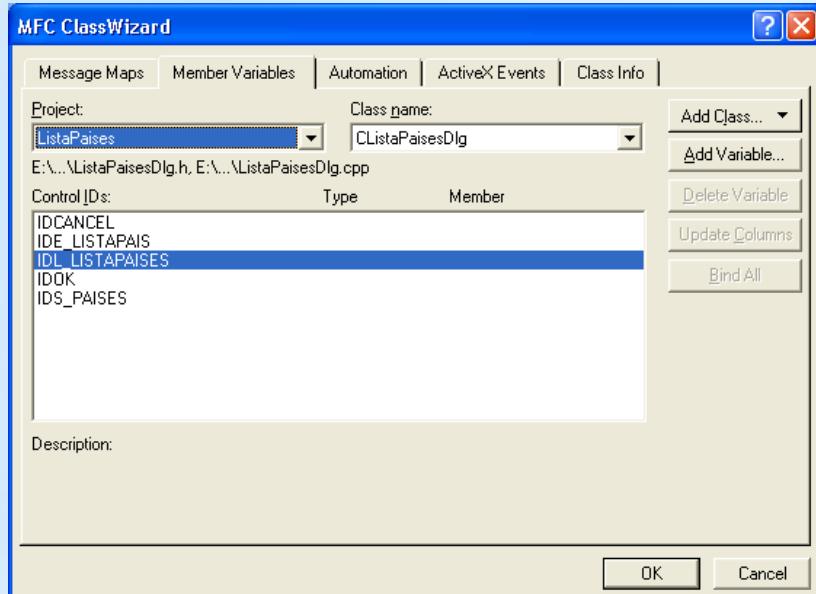
➤ Elegir MFC AppWizard(exe) y como nombre del proyecto ListaPaises



Colocar un control List Box en la Ventana de Dialogo Principal y otro control Edit Box



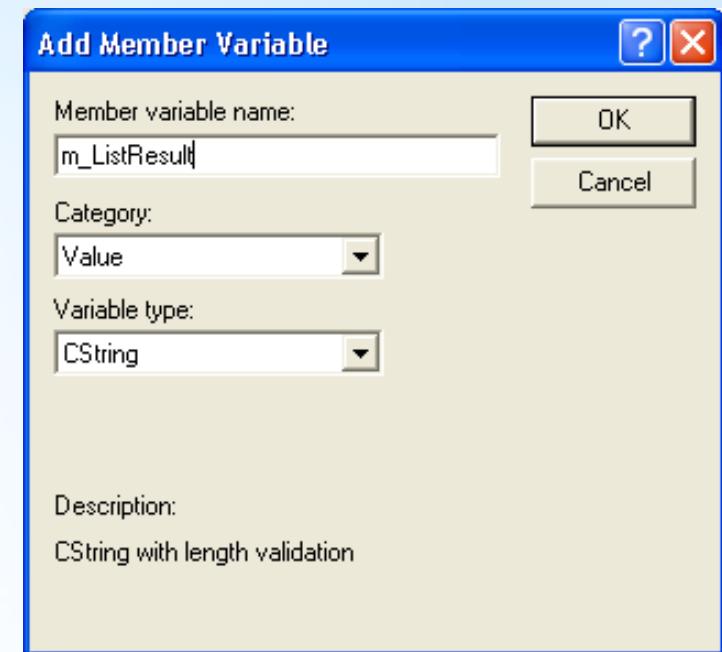
➤ Agregar variable a Add Member Variable debe quedar tal como se observa



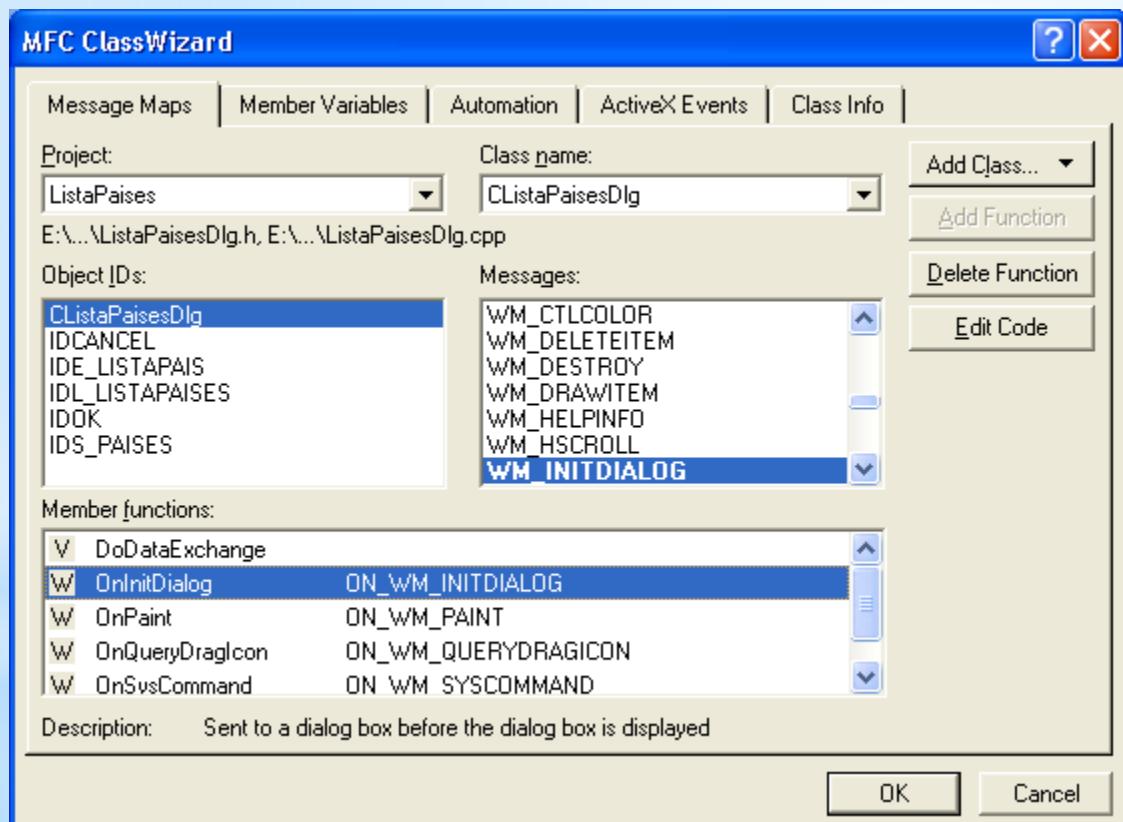
➤ Agregar una variable para el control IDE_LISTAPAISES, tal como se aprecia

The screenshot shows the 'Member Variables' tab of the Class View window. The project is set to 'ListaPaises'. The class name is 'CListaPaisesDlg'. The control IDs listed are IDCANCEL, IDE_LISTAPAISES, IDL_LISTAPAISES, IDOK, and IDS_PAISES. The member variables are m_ListResult (CString type) and m_ListaPaises (CListBox type). The 'IDE_LISTAPAISES' row is selected.

Control IDs:	Type	Member
IDCANCEL		
IDE_LISTAPAISES	CListBox	m_ListaPaises
IDL_LISTAPAISES		
IDOK		
IDS_PAISES		

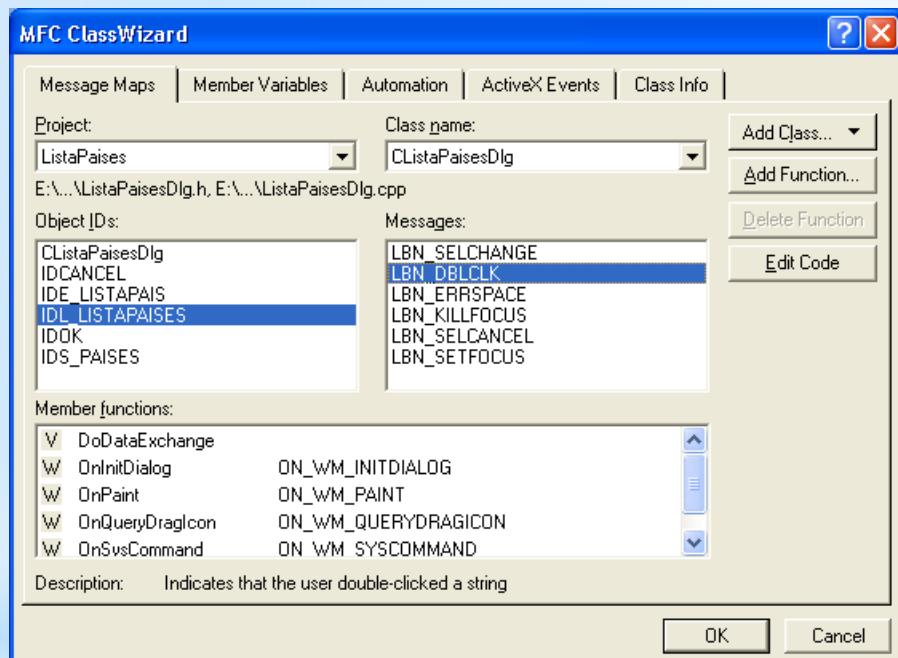


➤ Adicione elementos a la lista es decir los países, para ello debe ir a Members Function y activar OnInitDialog(), luego Edit Code y agregar los items

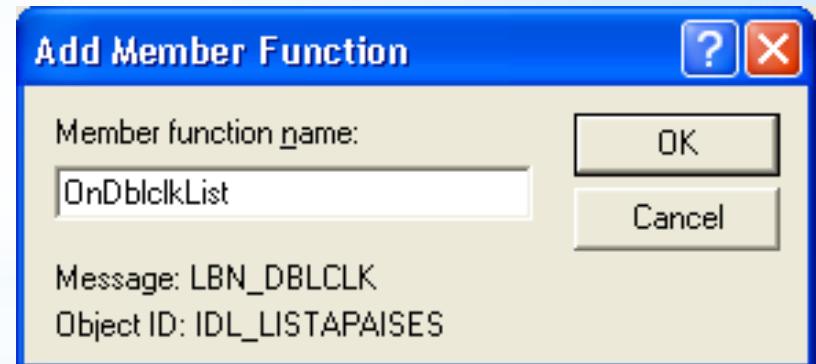
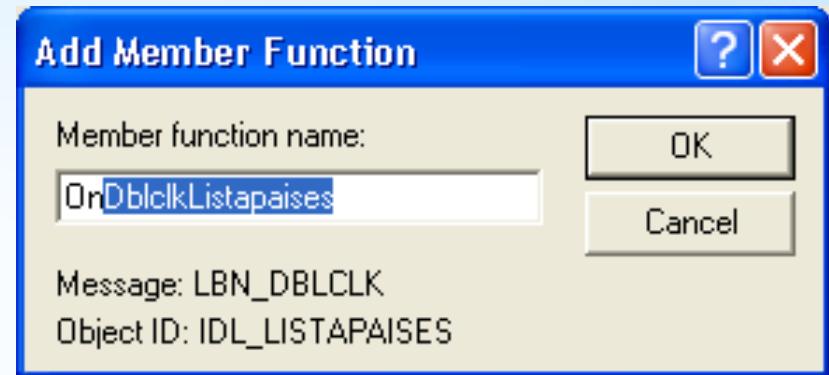


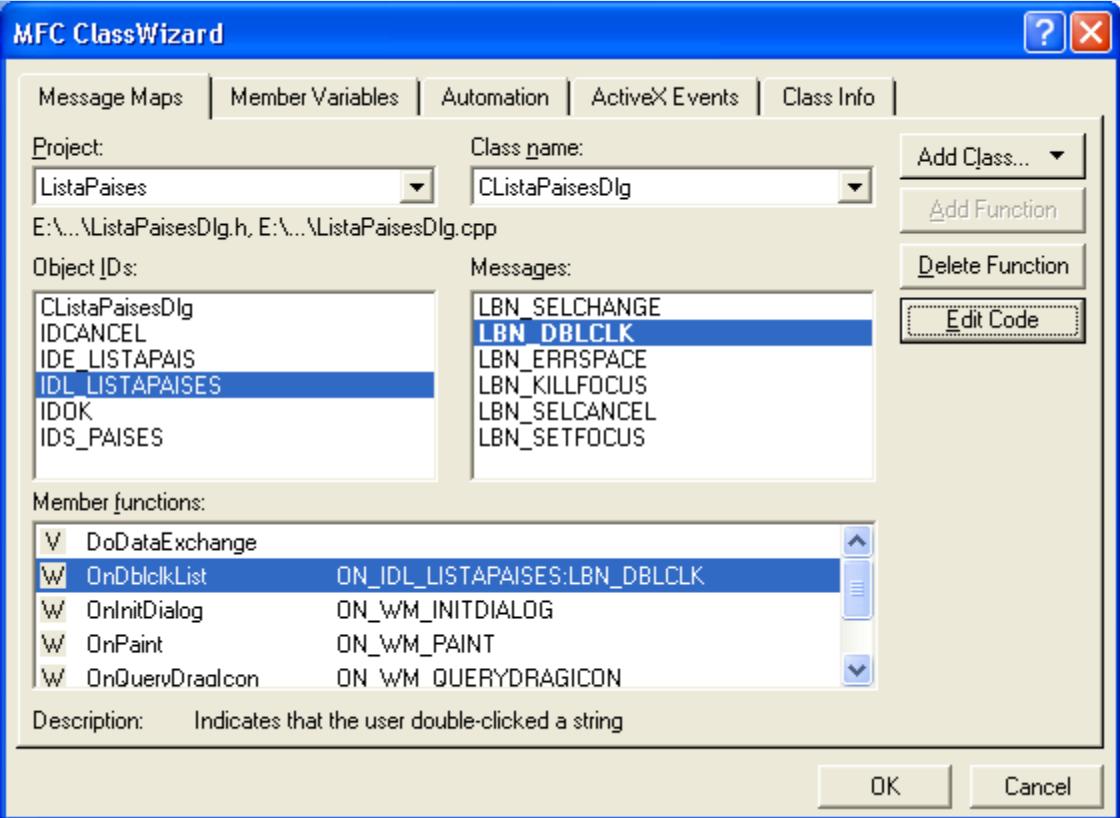
```
// TODO: Add extra initialization here
m_ListaPaises.AddString("Italia");
m_ListaPaises.AddString("Grecia");
m_ListaPaises.AddString("Egipto");
m_ListaPaises.AddString("Portugal");
m_ListaPaises.AddString("Canada");
m_ListaPaises.AddString("Peru");
m_ListaPaises.AddString("Colombia");
m_ListaPaises.AddString("Bolivia");
m_ListaPaises.AddString("Alemania");
m_ListaPaises.AddString("Japon");
m_ListaPaises.AddString("China");
```

- Se Trata de Transferir un elemento hacia un Edit Box.
- Dar doble clic dentro de la lista



Cambie el nombre a `OnDblclkList`





```
void CListaPaisesDlg::OnDbclkList()
{
    // TODO: Add your control
    // notification handler code here

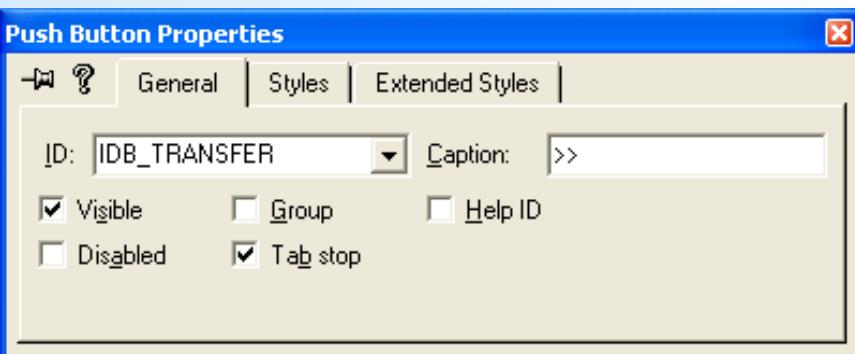
    UpdateData(TRUE);

    m_ListaPaises.GetText(m_ListaPaises.
    GetCurSel(),m_ListResult);

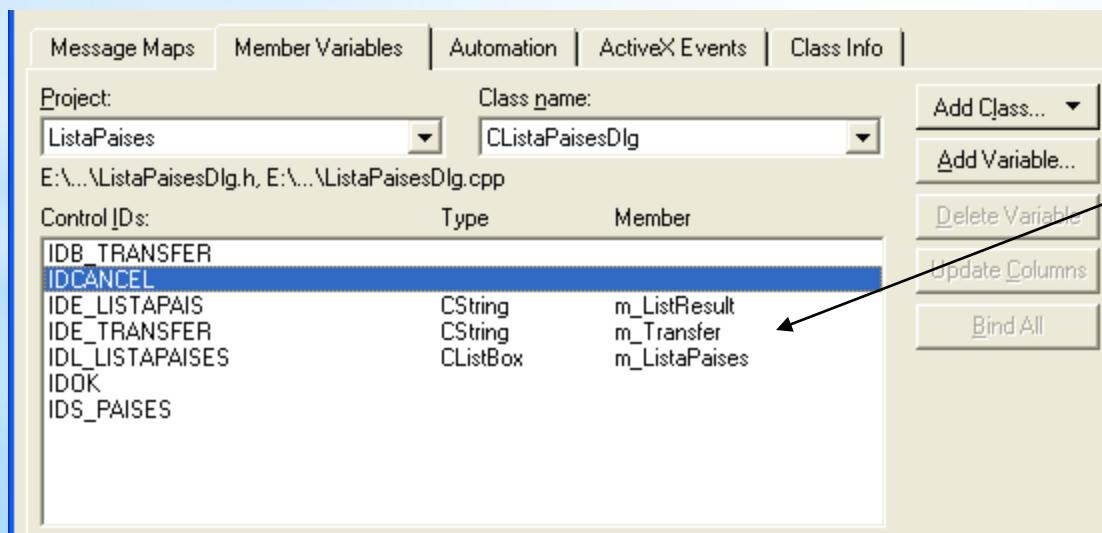
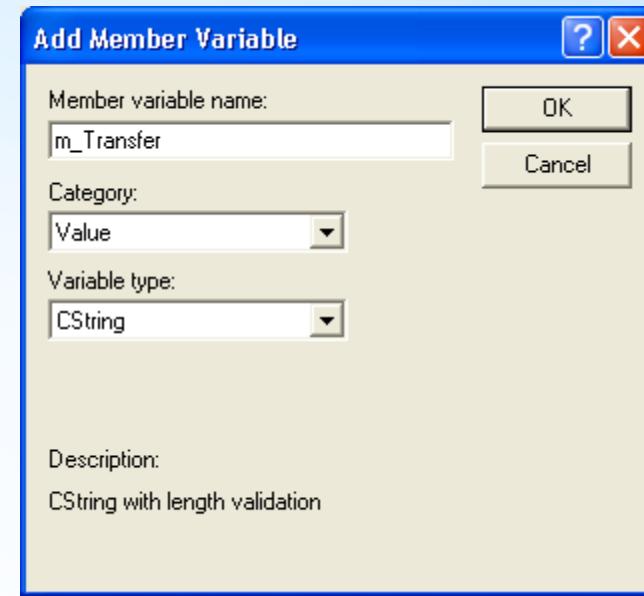
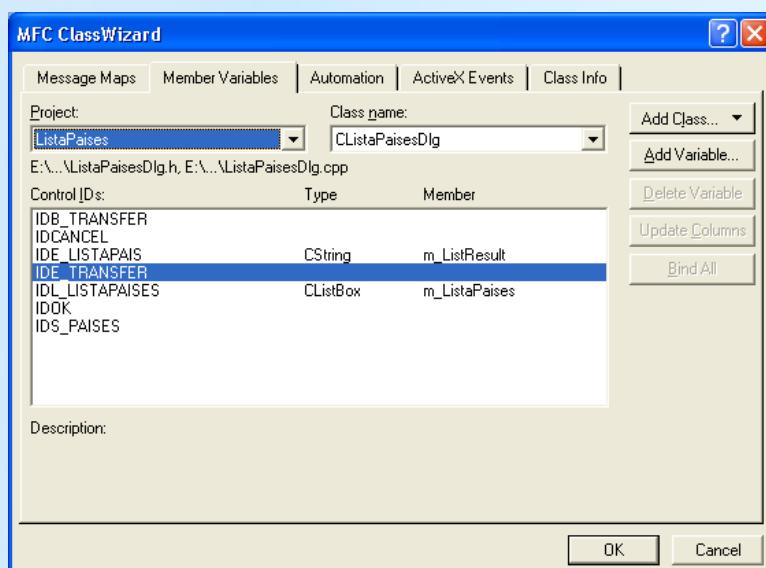
    UpdateData(FALSE)
}
```

Transfiere el contenido de un Edit Box a un List Box

- Adicione un botón Comand, sobre la Ventana principal.
- Seleccione un Ventana de Edicion

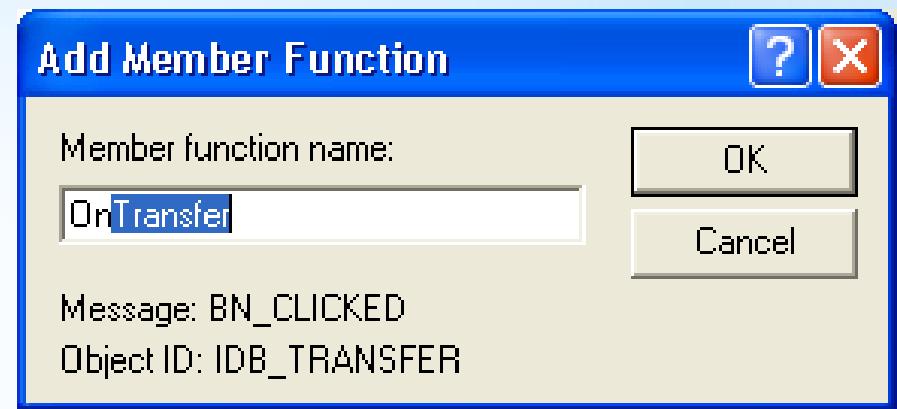
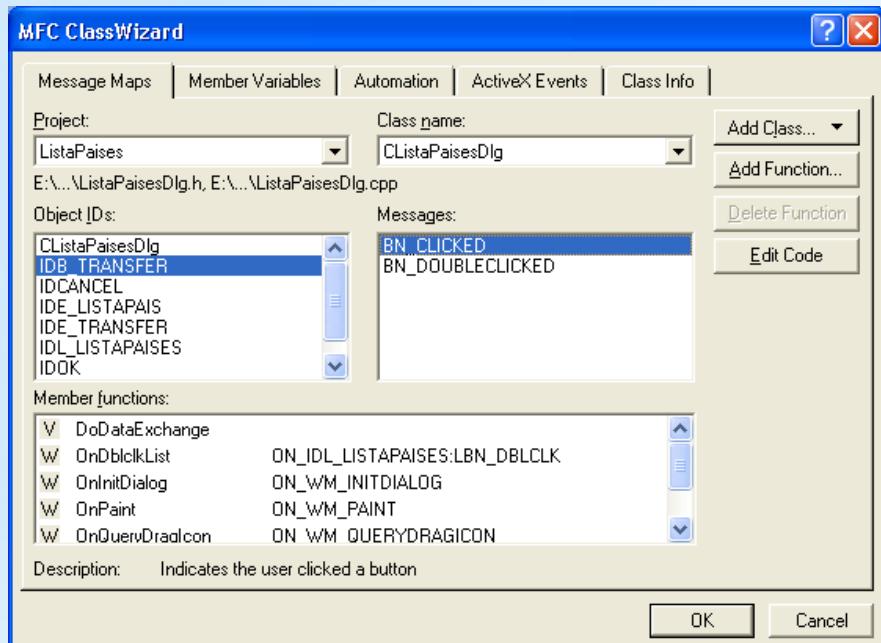


➤ Debe crearse un variable en la caja de Edición

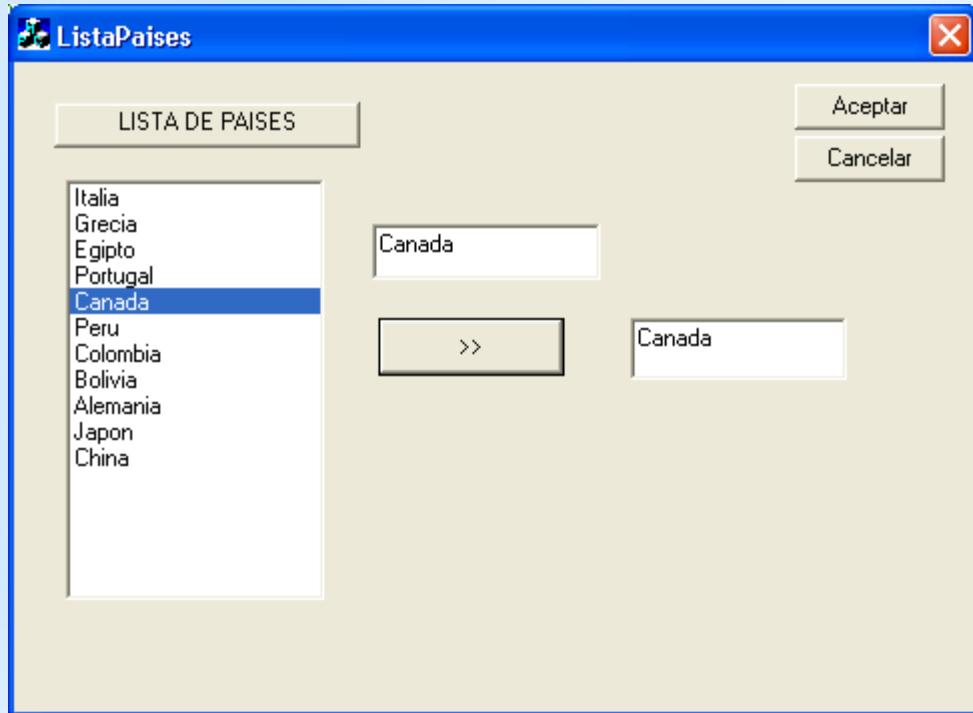


➤ Aprecie las variables
creadas de los controles

➤ Añadir código al botón >> , tal como se muestra, pruebe la aplicación



```
void CListaPaisesDlg::OnTransfer()
{
    // TODO: Add your control notification handler code here
    UpdateData(TRUE);
    m_ListaPaises.GetText(m_ListaPaises.GetCurSel(),m_Transfer);
    UpdateData(FALSE);
}
```

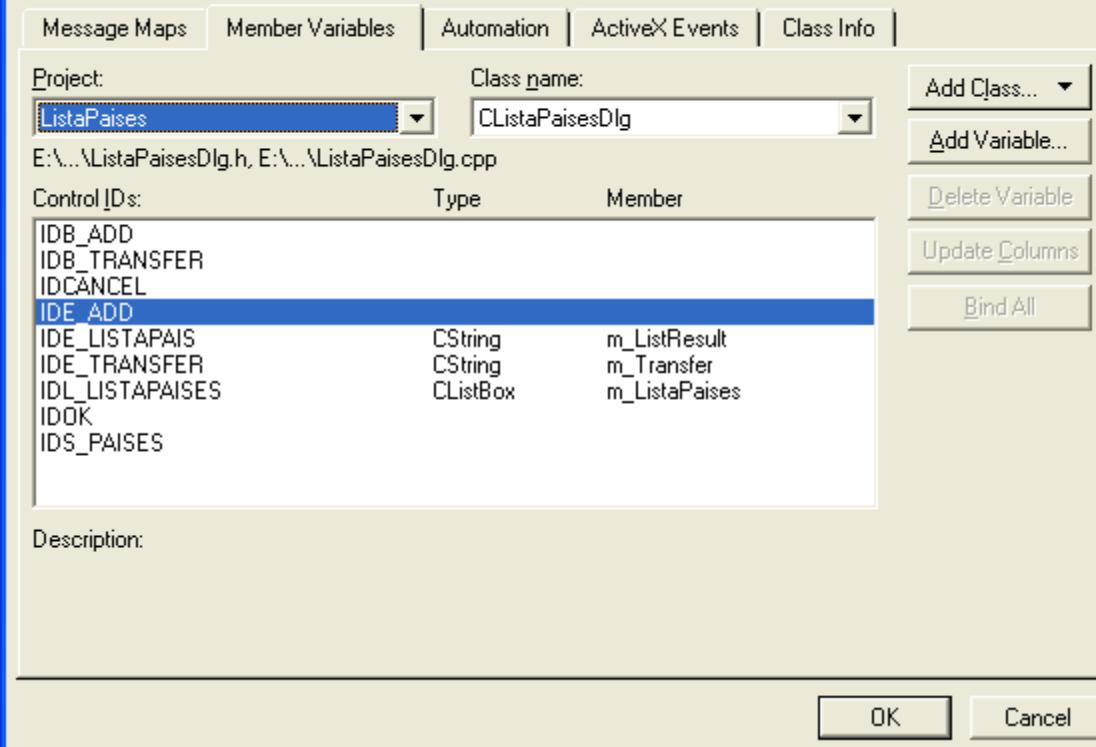


Agregar el Contenido de un Edit Box a un List Box

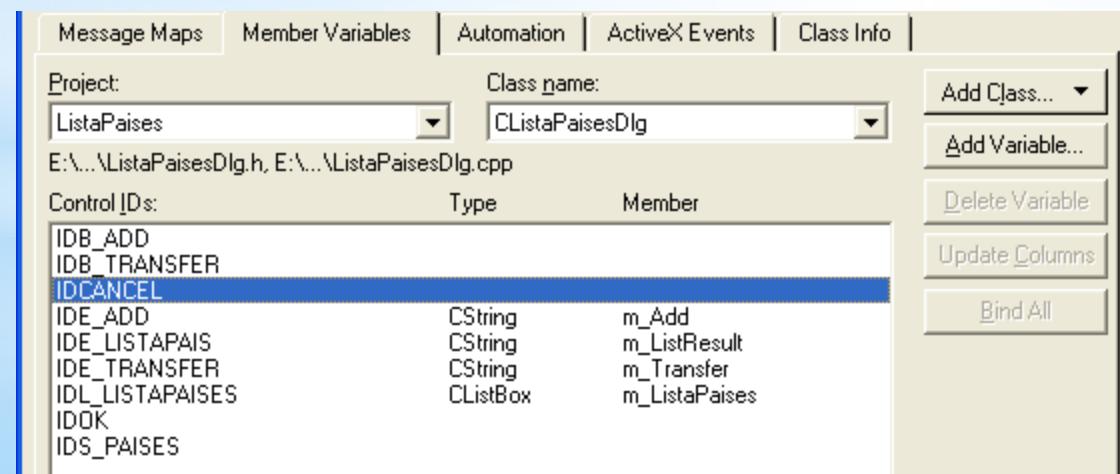
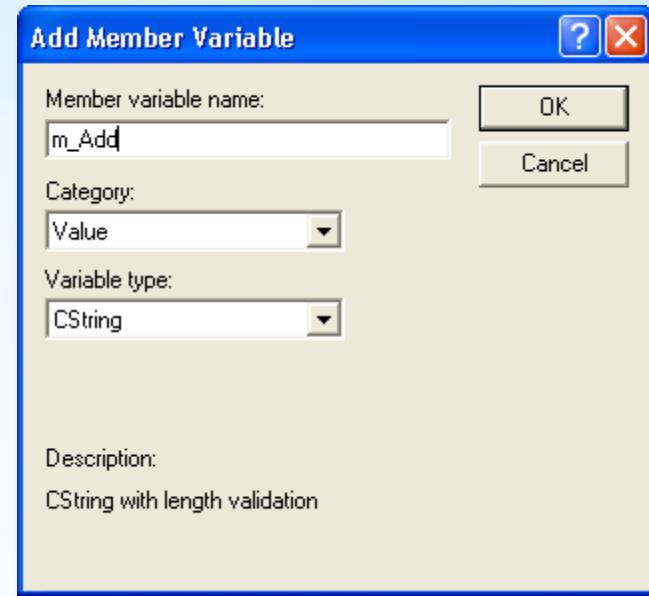
- Adicione sobre la ventana principal un control button.
- Adicione una Ventana Edit Text



MFC ClassWizard

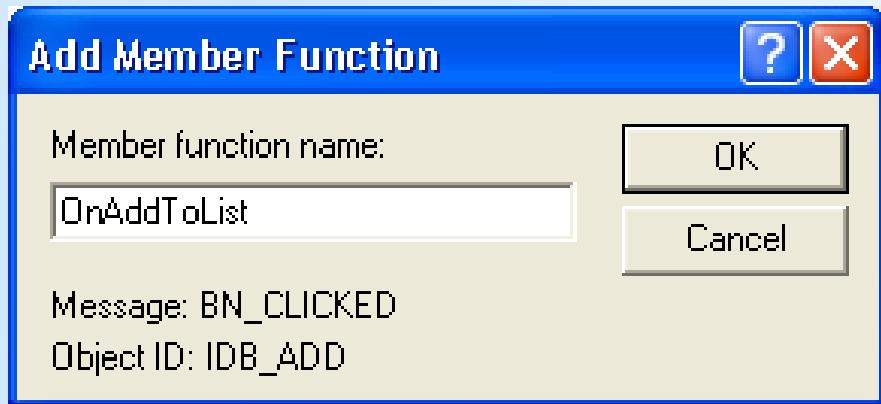


Adicione la variable m_Add,
tal como se aprecia.



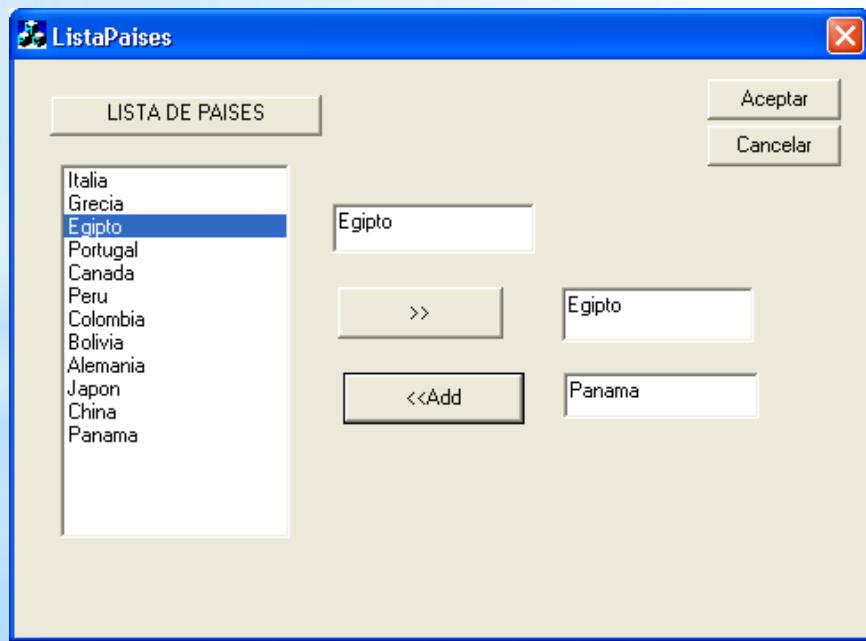
23/08/2019

➤ Active el botón Add con Ctrl +W para agregar código



```
void CListaPaisesDlg::OnAddToList()
{
    // TODO: Add your control notification handler
    // code here

    UpdateData(TRUE);
    m_ListaPaises.AddString(m_Add);
    UpdateData(FALSE);
}
```



➤ Pruebe la Aplicación y debe tener la misma ventana.

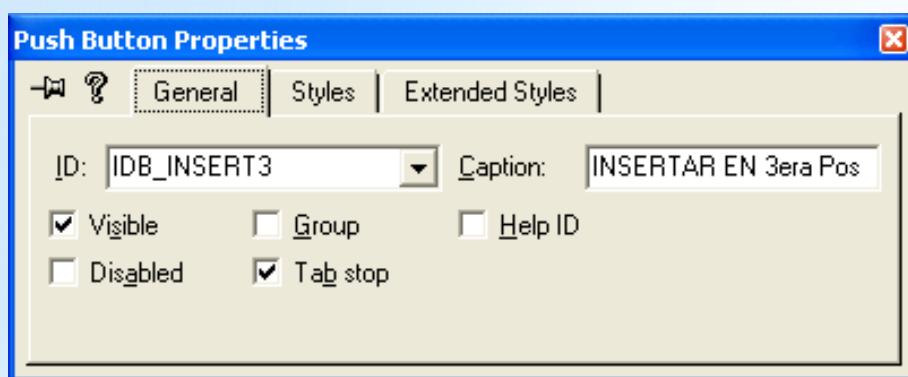
➤ No olvide hacer doble clic sobre uno de los elementos de la lista de la izquierda.

➤ Hacer clic sobre el botón >> deberá aparecer el país por ejem Egipto.

➤ Escriba un nombre de País en la ventanita de Edición , luego clic en <>Add debiendo aparecer el país en la lista.pe Panamá.

Agregando un Elemento en una posición específica dentro de la lista

- Agregue un control button sobre la ventana de Dialogo.
- Agregue una ventanita de Edición a la derecha del button.
- Observe el obj creado y en Mamber Var deberá adicionar la variable a usar



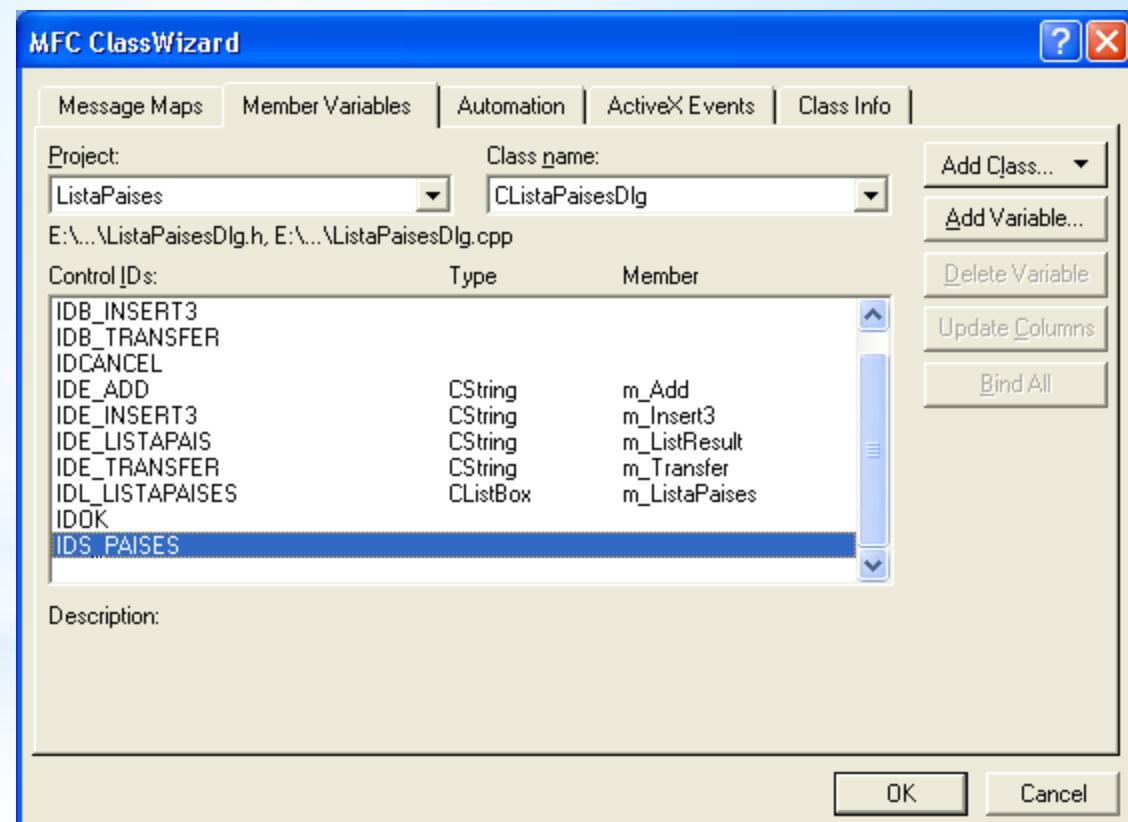
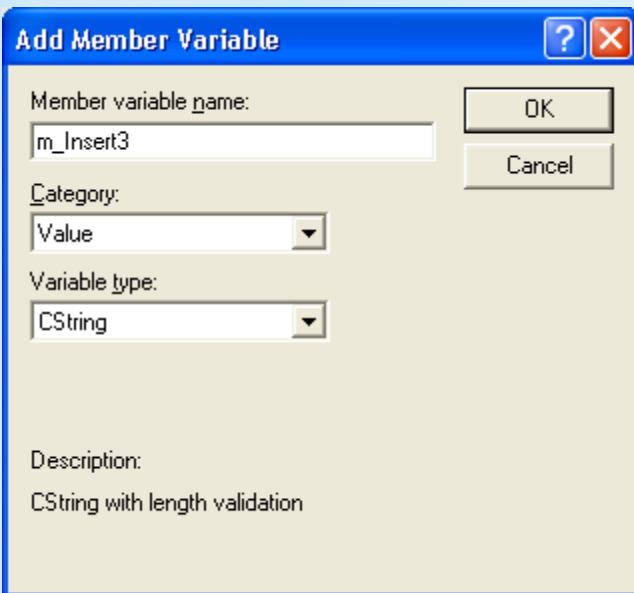
The Message Maps tab of the Class View dialog shows the following message mappings:

Object IDs:	Messages:
CListaPaisesDlg IDB_ADD IDB_INSERT3 IDB_TRANSFER IDCANCEL IDE_ADD IDE_INSERT3	EN_CHANGE EN_ERRSPACE EN_HSCROLL EN_KILLFOCUS EN_MAXTEXT EN_SETFOCUS EN_UPDATE

The Member Variables tab of the Class View dialog shows the following member variables:

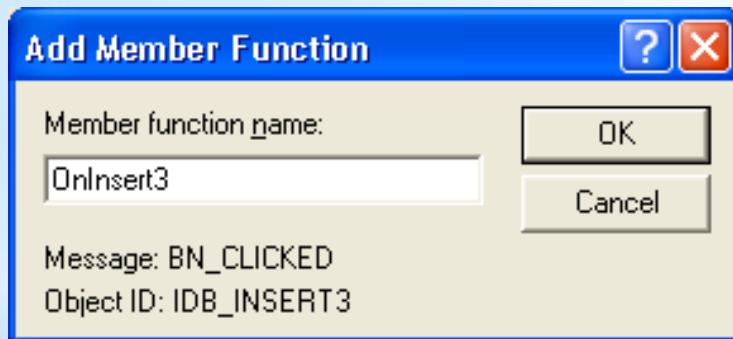
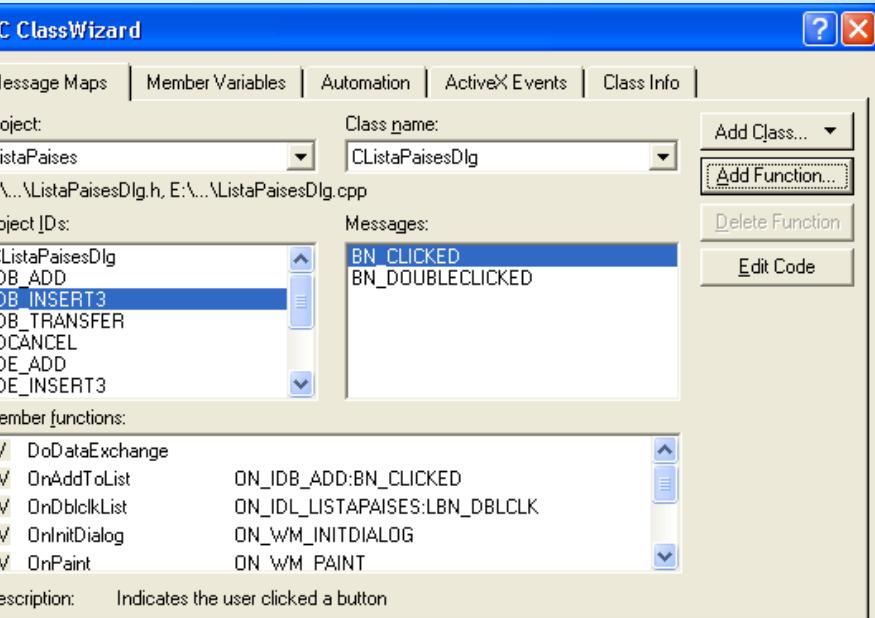
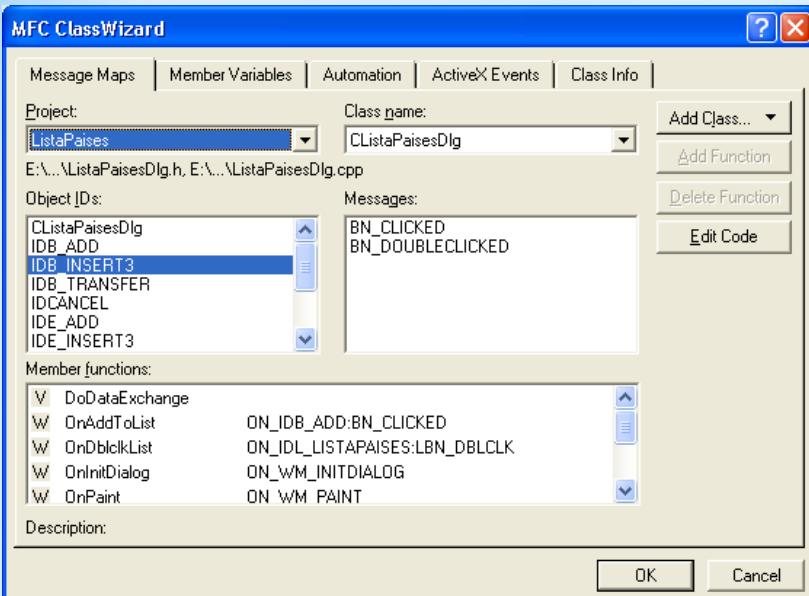
Control IDs:	Type	Member
IDB_ADD		
IDB_INSERT3		
IDB_TRANSFER		
IDCANCEL		
IDE_ADD	CString	m_Add
IDE_INSERT3	CString	m_ListResult
IDE_LISTAPAIIS	CString	m_Transfer
IDE_TRANSFER	CString	m_ListaPaises
IDL_LISTAPAISES	CListBox	
IDOK		
IDS_PAISES		

➤ Adicione la variable y debe quedar tal como se observa en la figura

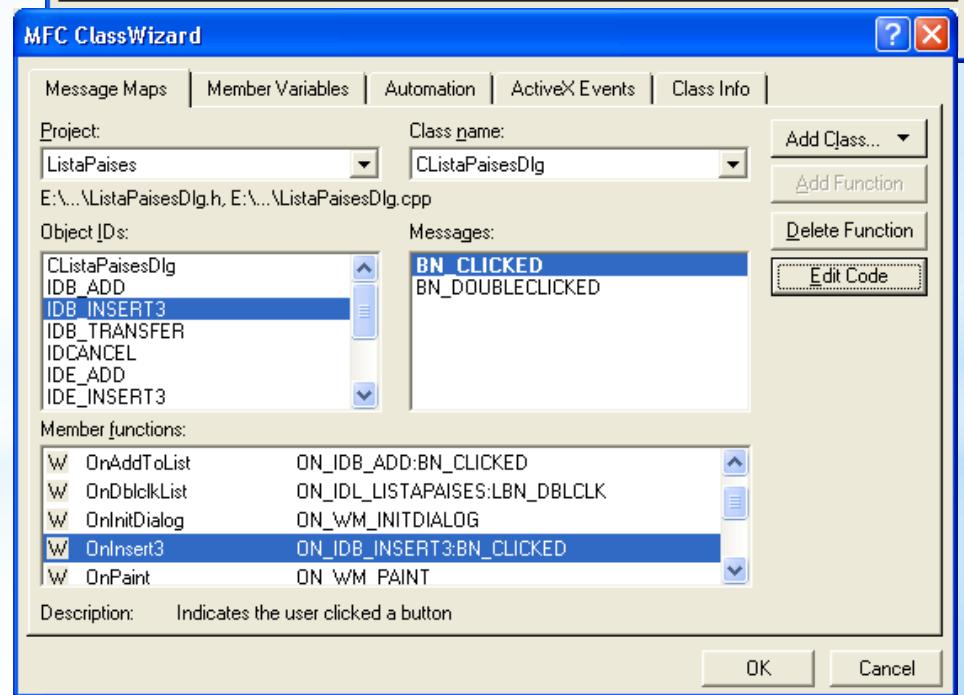


➤ Observe la variable creada y también el resto de variables.

➤ Agregaremos código al button Insertar, Ctrl+W



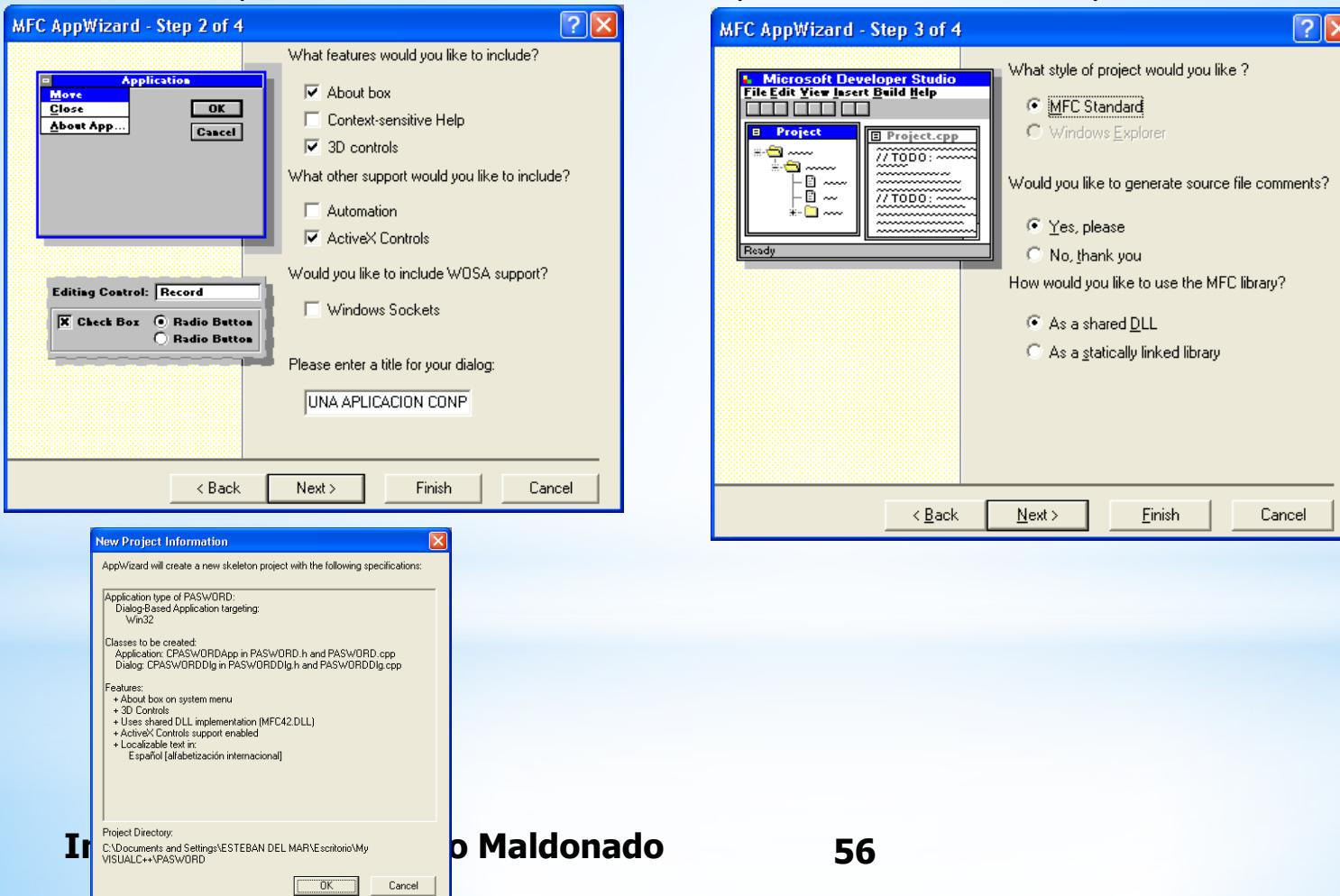
Si todo sale bien debe tener la función creada OnInsert3 y en la MFC ClassWizard el detalle

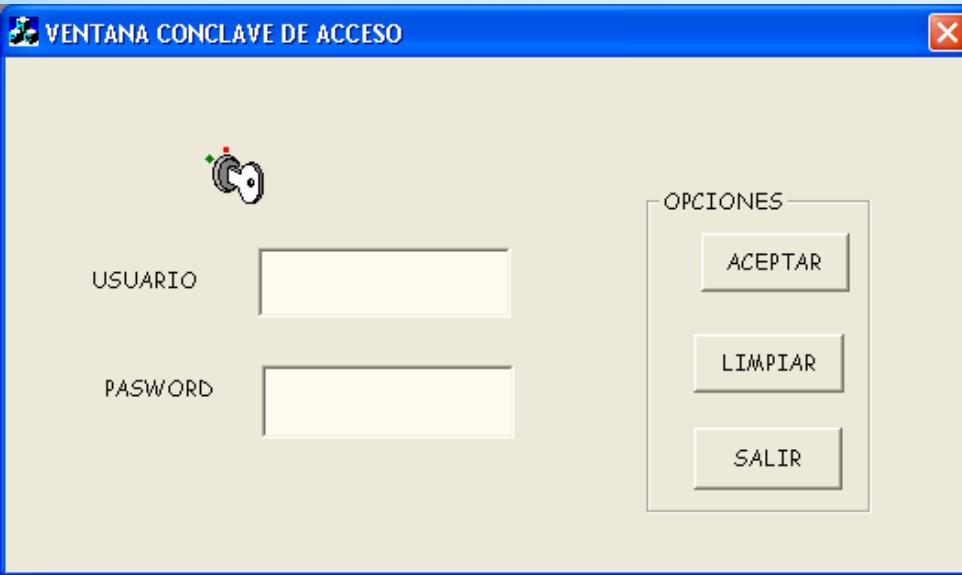


Ingeniero Daniel Osorio Maldonado

UNA SIMPLE APLICACION

- Aplicaremos una sencilla pero significativa aplicación con un gráfico.
- Para ello haremos uso de proyecto **basado en dialogo**, suponemos que el usuario ya conoce la mecánica y estamos en el paso 2/4 del asistente.





- En esta aplicación agregaremos un grafico.
- El usuario debe ingresar con la clave correcta.
- El botón aceptar generará el evento clic creando la función OnAceptar.
- El código deberá es el siguiente

```
void CPASWORDDDlg::OnAceptar() {  
    // TODO: Add your control notification handler code here  
    UpdateData(TRUE);  
    CString pasw ="SUERTE";  
    if(strcmp(m_password,pasw)==0)  
        MessageBox("Ingrese al Sistema ","Usuario Confirmado",MB_OK);  
    else  
        MessageBox("Error al Ingresar ","Error Usuario No Confirmado",MB_ICONSTOP);  
    UpdateData(FALSE);}
```

Gráficos en Visual C++ 2010

- Para realizar gráficos, Windows utiliza fundamentalmente las funciones incluidas en el modulo **GDI.EXE**.
- Las cuales se encargan de llamar a las rutinas de los distintos gestores de dispositivo(**drivers de video, de impresora y de trazadores gráficos**)que son los que directamente actúan sobre el dispositivo.
- Por consiguiente, el **modulo GDI** proporciona una interfaz para dibujar sobre dispositivo gráficos(**GDI**) que tiene como meta principal dar soporte a la realización de gráficos.
- Debe quedar claro que es independiente del dispositivo que se utilice para su visualización.

El Contexto de Dispositivo

- Windows no permite un acceso directo al Hardware de visualización o de impresión
- Se comunica a él por medio de una abstracción denominada contexto de dispositivo(device context-DC).
- Por lo tanto, cuando se quiera dibujar en un dispositivo grafico como una pantalla o una impresora,
 - a) Primero hay que obtener un handle de un contexto de dispositivo, que represente al dispositivo físico.
 - b) Que delimite el área donde se puede pintar.

Se puede obtener un contexto de dispositivo:

- a) Para pantalla entera.
- b) Para el área del usuario de una ventana.
- c) Para la ventana entera o para una impresora.
- d) También se puede crear un contexto de dispositivo no referido a un dispositivo real.

- Un **DC** es una estructura de datos que contiene los atributos necesarios para pintar el gráfico o el texto.
- **Windows** utiliza estos atributos con las funciones del **GDI** para determinar como se pinta el grafico o el texto.
- Esto permite que las funciones **GDI** incluyan:
 - a) Solo el handle del DC.
 - b) Las coordenadas iniciales y el tamaño si es preciso.
Por tanto del destino como del origen, sin que para pintar sea necesario especificar otros atributos como el color de fondo, el color del primer plano, el tipo de letra, el espaciado entre caracteres , etc.
- Por ejemplo cuando en la aplicación de bienvenida dibujado en una pantalla llamamos a la función **DrawTextW**, donde solo indicaremos el contexto de dispositivo, el texto y las coordenadas donde se quiere pintar.

Resumiendo lo anunciado

- Para dibujar gráficos, Windows utiliza las funciones incluidas en el modulo GDI.EXE, las cuales se encargan de llamar a las rutinas de los distintos gestores de dispositivo(drivers: video, de impresora, y de trazadores gráficos), que son los que directamente actúan sobre el dispositivo.
- Por consiguiente el modulo GDI.EXE proporciona una interfaz para dibujar sobre dispositivos graficos(GDI).
- **EL CONTEXTO DE DISPOSITIVO:** es una abstracción que utiliza Windows para el acceso al Hardware de visualización o de impresión, ya que directamente no lo puede hacer, por consiguiente esta abstracción recibe el nombre de **contexto de dispositivo**(device context-DC).
- Por consiguiente cuando se quiera dibujar en un dispositivo grafico como una pantalla o una impresora, primero hay que obtener un **handle** de un contexto de dispositivo que represente al dispositivo físico y que delimite el área donde se puede pintar.

- Se puede obtener un contexto de dispositivo para la pantalla entera, para el área del usuario de una ventana, para la ventana entera o para una impresora
- DC de Pantalla para acceder a los objetos existentes en pantalla, existen 02 funciones para crear: BeginPaint() y GetDC(); existiendo otras 02 funciones complementarias para la liberación: EndPaint() y Release();
- DC de Impresora o printer Device Context es el DC que se utiliza para acceder a cualquier tipo de dispositivo que funcione como impresora, matricial, inyección, láser o vectorial.
- DC Memoria o Memory Context crean mapas de bits compatibles con un determinado hardware, se hace necesario la función CreateCompatibleDC() cuya definición es HDC CreateCompatible(HDC hdc).
- DC de Información una aplicación puede obtener rápidamente información acerca de los valores por defecto del dispositivo en cuestión.

La clase CDC

- Contiene todas las funciones miembro que se necesitan para realizar dibujos en una aplicación de Windows.

CpaintDC

{ Para dibujar dentro de un método en respuesta al mensaje **WM_PAINT** }

CClientDC

{ Para dibujar en el área cliente de forma sincrónica }

CWindowDC

{ dibuja en toda la ventana(no solo en el área del cliente de forma sincrona) }

CMetafileDC Para dibujar sobre un archivo.

- Para crear un DC, primero se crea un objeto de la clase base CDC y luego se llama las funciones miembros que usan contextos dispositivos.

Dibujar Líneas

BOOL LineTo(int x,int y);

BOOL LineTo(POINTpoint);

La función CDC::LineTo pinta una Línea desde la posición actual de la pluma hasta el punto lógico especificado por los argumentos x e y por el argumento punto.

La Clase Crect

La clase CRect es similar a la estructura RECT de Windows. La estructura RECT esta definida en windows.h y es de forma siguiente:

```
Typedef struct tagRECT{  
Int left;  
Int top;  
Int right;  
Int bottom;  
}RECT;
```

Los miembros left y top son las coordenadas(x,y) de la esquina superior izquierda del rectángulo que define una estructura y los miembros right y bottom son las coodenadas(x,y) de la esquina inferior derecha. Algunas funciones de windows requieren objetos de esta clase como parámetro. Un objeto *Crect* puede ser pasado como parametro a una función donde quiera que se utilice RECT o LPRECT. La funcionalidad es soportada por varios conjuntos de funciones miembros:

- Un constructor; operadores sobrecargados:(LPCRECT,LPRECT,
Ingeniero Daniel Osorio Maldonado **64** **23/08/2019**

La Clase CPoint

La clase CPiont es similar a la estructura **POINT** de windows. La estructura POINT esta definida en windows.h y es de la forma siguiente

```
Typedef struct tagPoint{  
Int x;  
Int y;  
} POINT;
```

La clase Cpoint define las coordenadas(x,y) de un punto y proporciona para manipularlos los operadores sobrecargados operator==operator:

=operator+operator=operator+y operator -, la funcion miembro

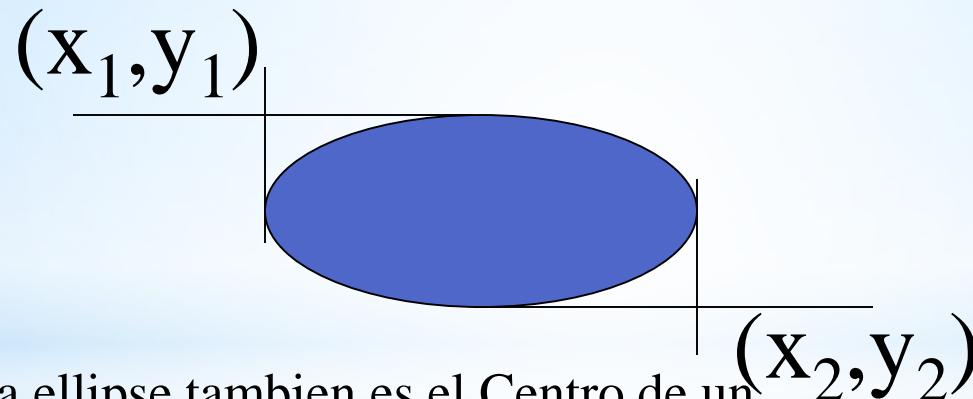
Offset y el constructor CPOINT

DIBUJO DE UNA ELIPSE

BOOL Ellipse(int x1,int y1,int x2,int y2);

La Funcion CDC::Ellipse

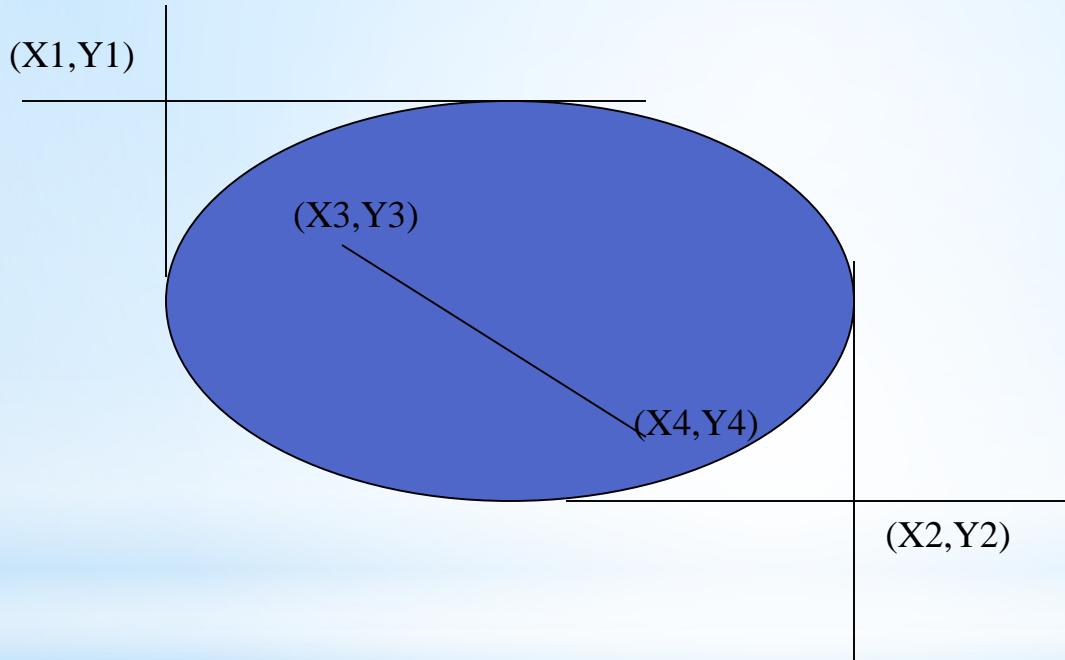
se utiliza para dibujar una elipse o circulo. La forma general es:



El Centro de la ellipse tambien es el Centro de un Rectangulo imaginario descrito por los puntos (x_1, y_1) y (x_2, y_2) , tal como se muestra.

* DIBUJO DE UNA CUERDA

*BOOL Chord(int x1,int y1,int x2,int y2,int x3,int y3,
int x4,int y4);



La funcion CDC::Chord dibuja una figura cerrada con una
línea entre dos puntos del arco (x3,y3) y (x4,y4)

* DIBUJO DE UN SECTOR CIRCULAR

- BOOL Pie(int x1,int y1,int x2,int y2,int x3,int x4,int y4);
- La función CDC::Pie dibuja sectores circulares. El centro del arco elíptico es también el centro de un rectángulo imaginario descrito por los puntos (x1,y1) y (x2,y2).

DIBUJO DE UN RECTANGULO

- La Función CDC::Rectangle dibuja un rectángulo o cuadro descrito por (x1,y1) y (x2,y2).

Clase CPen

La clase Cpen permite definir la pluma a utilizar con las funciones de dibujo.

Esta herramienta afecta la forma en la que son dibujadas Las lineas pueden ser lineas sólidas lineas discontinuas, etc, también puede afectar los bordes de las figuras cerradas ya sean rectángulos, elipses,etc.

Miembros Principales de la Clase CPen

Constructor : Cpen // crea un Objeto Cpen

Incializacion : CreatePen //Crea una Pluma Logica geométrica con estilo específico, ancho y atributos de pincel y une todo esto al objeto CPen.

Operaciones: FromHandle // Retorna un puntero a un Objeto CPen cuando es dado un handle a un Objeto HPEN

La Clase CPaintDC

La Clase CPaintDC se utiliza con la función OnPaint y permite visualizar texto o gráficos en el área de trabajo de una ventana en respuesta a un mensaje WN_PAINT. Constructor: CPaintDC //crea un objeto Cpaint conectado al CWnd.

Sistema de Coordenadas

Escala	Unidad logica	Eje x	Eje y
		Los valores	aumentan hacia
MM_TEXT	1pixcel	La derecha	abajo
MM_TWIPS	1twip	La derecha	arriba
MM_LOMETRIC	0.1 mm	La derecha	Arriba
MM HIMETRIC	0.01mm	La derecha	Arriba
MM LOENGLISH	0.01pulgadas	La derecha	Arriba
MM HIENGLISH	0.0001pulgadas	La derecha	arriba
MM_ISOTROPIC	0.001pulgadas(X=Y)	Seleznable	Seleznable
MM_ANISOTROPIC	Arbitrario (X!=Y)	Seleznable	Seleznable

La Clase CBrush

La clase Cbrush permite definir el pincel a utilizar con las funciones de dibujo.

Esta herramienta afecta a la forma en la que es pintado el interior de las figuras cerradas.

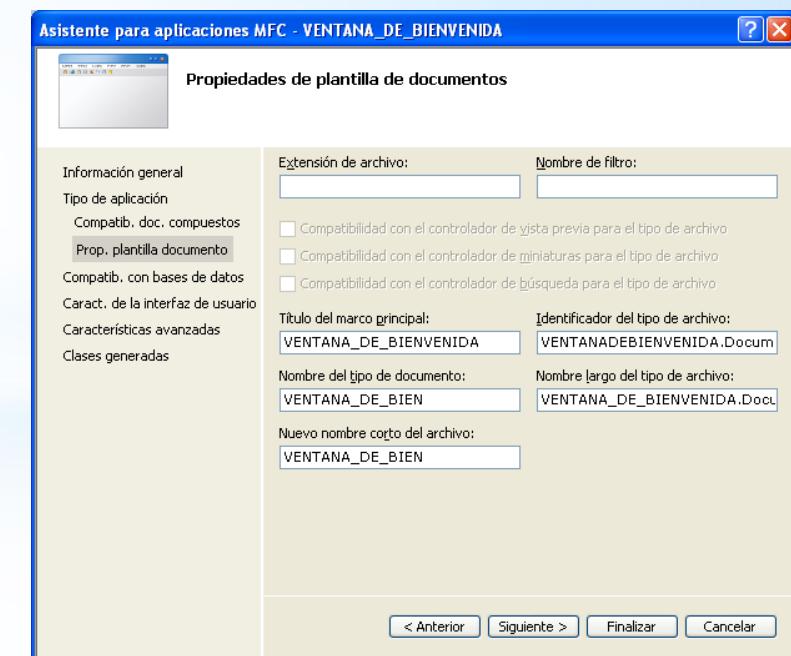
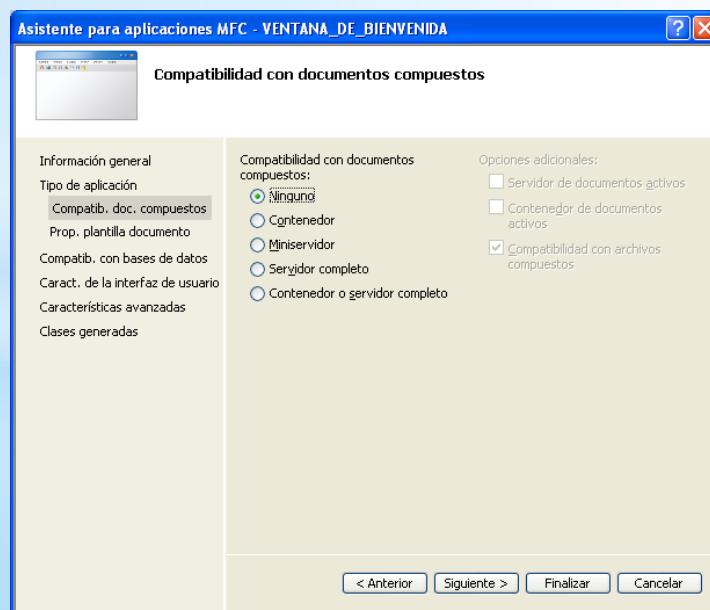
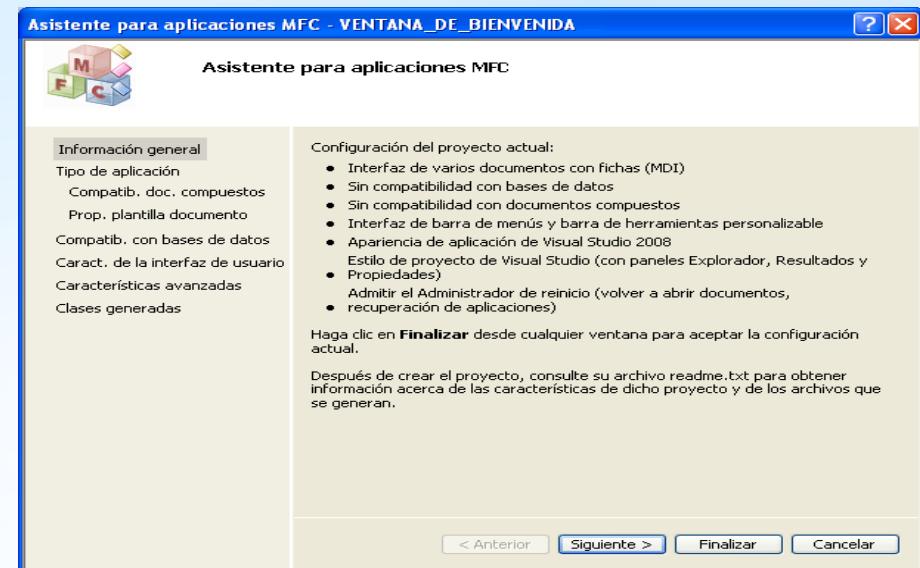
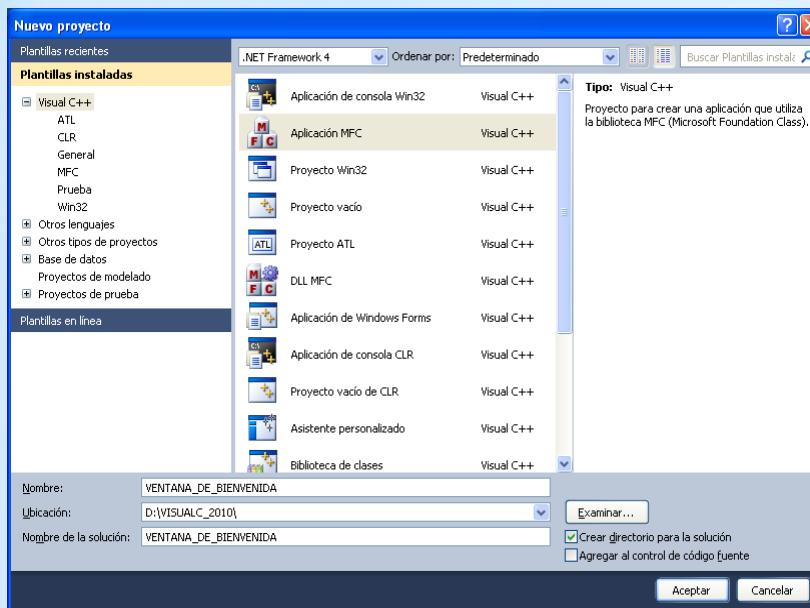
Cuando se crea un contexto de dispositivo, existe un pincel por defecto que pinta de blanco el interior de las figuras cerradas para cambiar el pincel actual puede elegir un pincel de los que Windows tiene en stock o puede crear un pincel y Seleccionarlo para el contexto de dispositivo

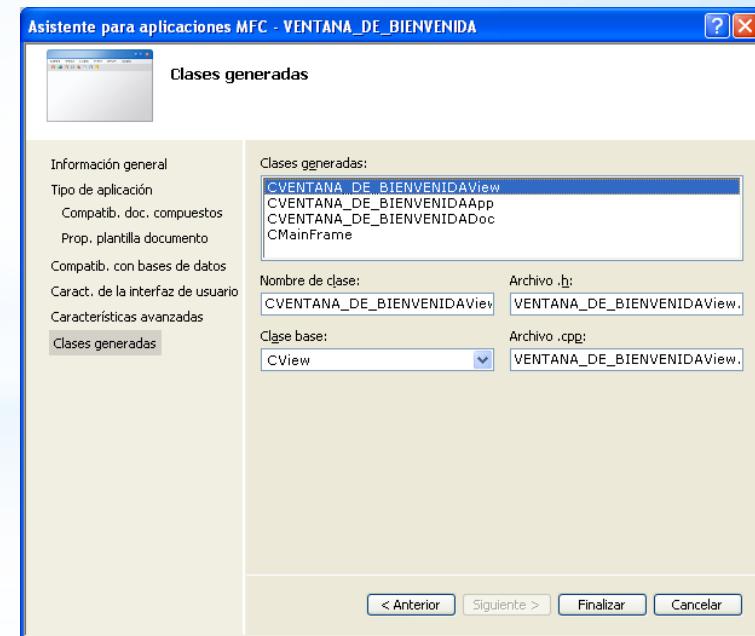
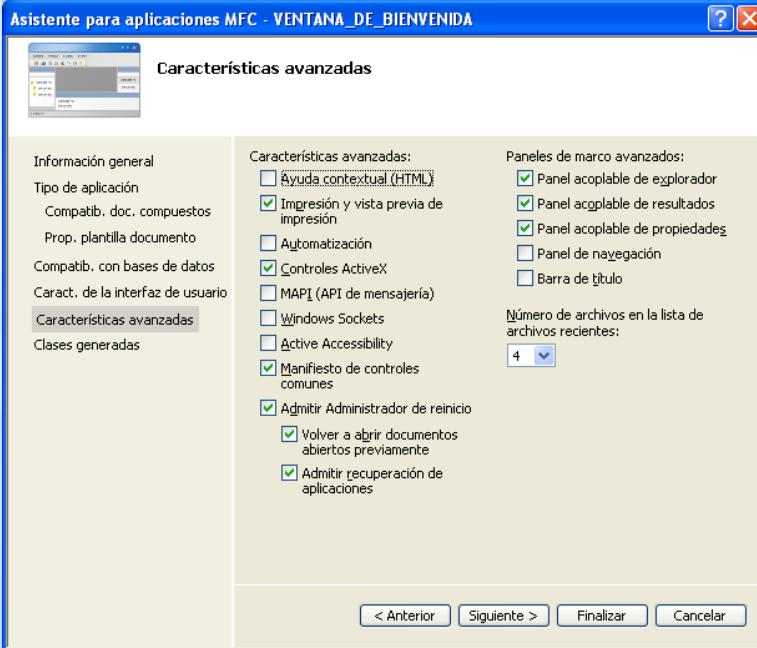
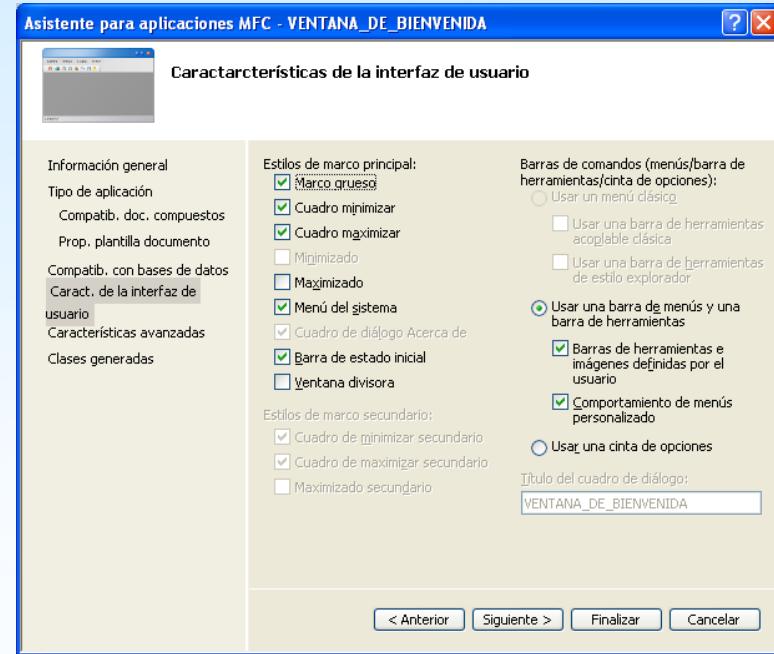
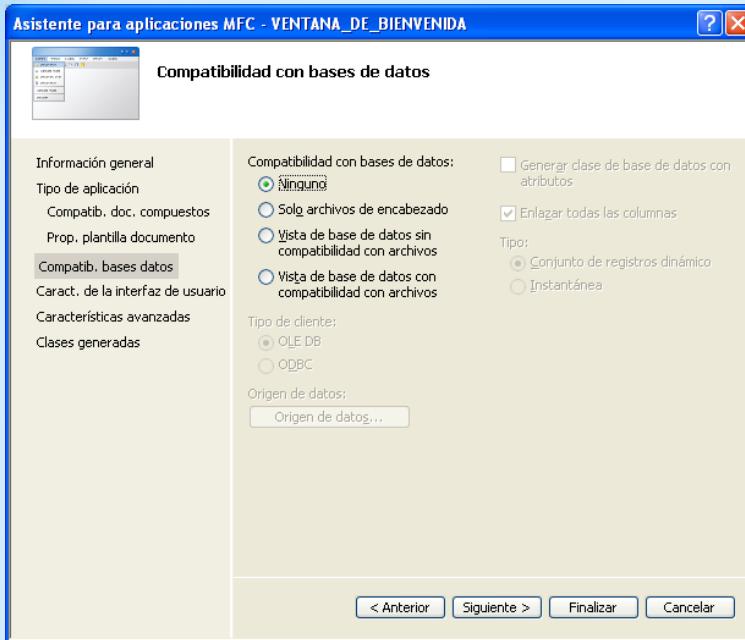
CPaintDC PintarDC(this);

PintarDC.SelectStockObject(LTGRAY-BRUSH); La funcionalidad de esta clase esta soportada por varios conjuntos de funciones miembros:

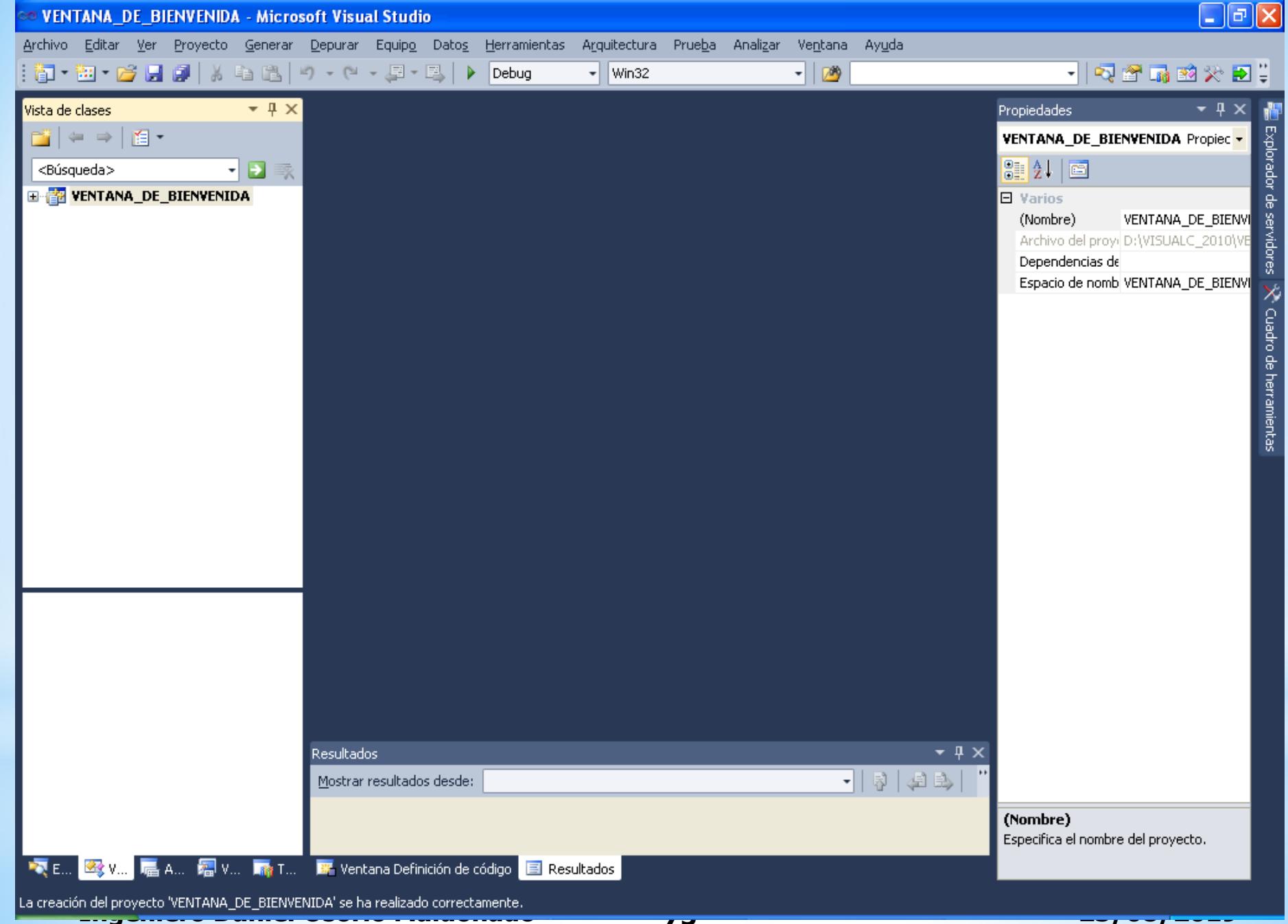
- Un constructor
- Funciones de inicialización(CreateBrushIndirect, CreatePateernBrush,
- La función FromHandle

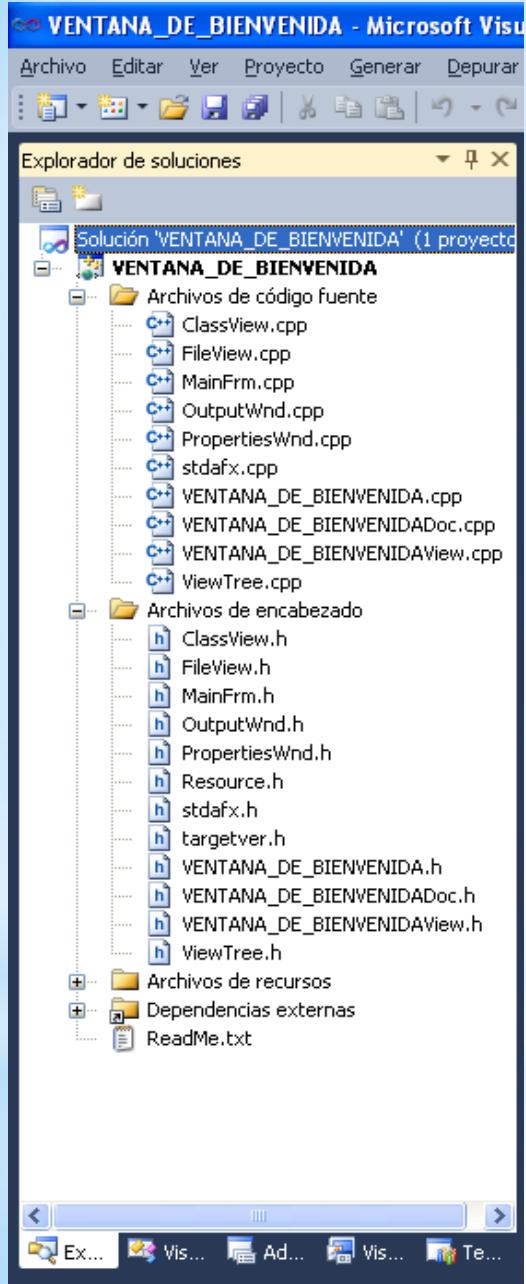
APLICACIONES DOCUMENTO UNICO(SINGLE DOCUMENT)



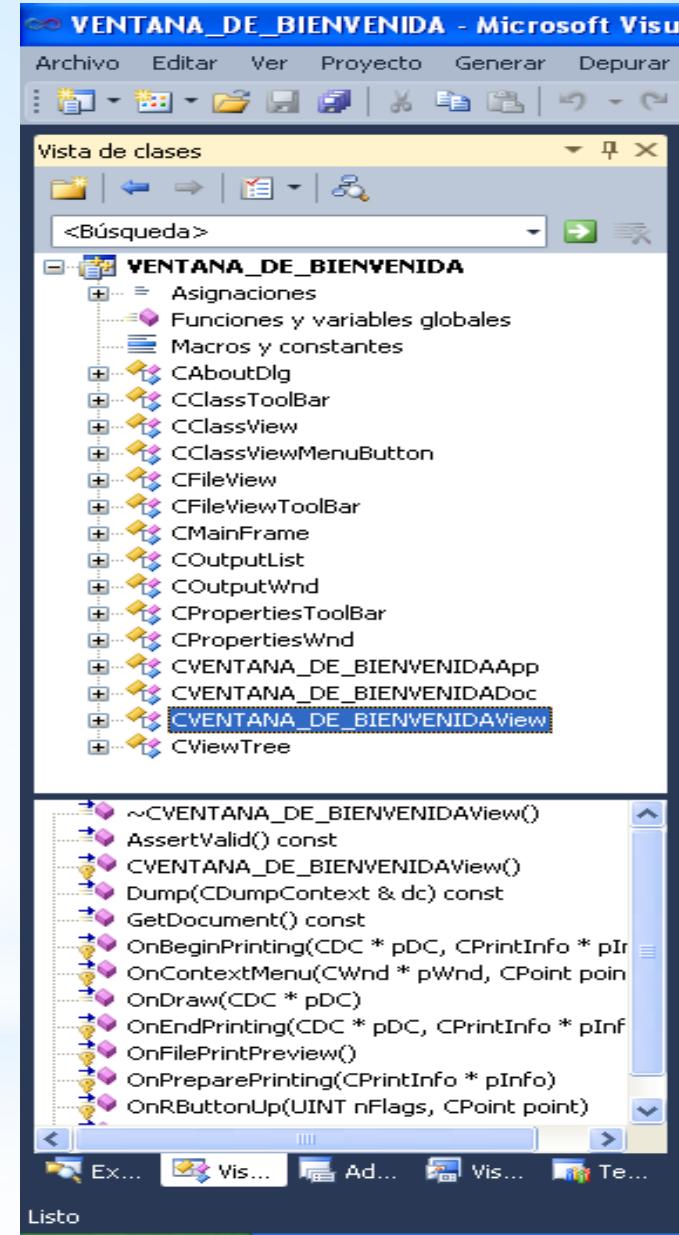


/2019





- Se aprecia en la pestaña **explorador de soluciones** Código fuente, Headers Files, etc.
- En la Pestaña Vista de clases (ClassView) aprecie las clases creadas → CVENTANA_DE_BIENVENIDA.
- Hacer DOBLE clic en la función OnDraw(CDC *pDC)





Vista de clases



VENTANA_DE_BIENVENIDA

- + Asignaciones
- + Funciones y variables globales
- + Macros y constantes
- + CAboutDlg
- + CClassToolBar
- + CClassView
- + CClassViewMenuItem
- + CFileView
- + CFileViewToolBar
- + CMainFrame
- + COutputList
- + COutputWnd
- + CPropertiesToolBar
- + CPropertiesWnd
- + CVENTANA_DE_BIENVENIDAApp
- + CVENTANA_DE_BIENVENIDAView
- + CVentanaDeBienvenidaView
- + CViewTree

- + ~CVentanaDeBienvenidaView()
- + AssertValid() const
- + CVentanaDeBienvenidaView()
- + Dump(CDumpContext & dc) const
- + GetDocument() const
- + OnBeginPrinting(CDC * pDC, CPrintInfo * pInfo)
- + OnContextMenu(CWnd * pWnd, CPoint point)
- + OnDraw(CDC * pDC)
- + OnEndPrinting(CDC * pDC, CPrintInfo * pInfo)
- + OnFilePrintPreview()
- + OnPreparePrinting(CPrintInfo * pInfo)
- + OnRButtonUp(UINT nFlags, CPoint point)

VENTANA_DE_BIENVENIDAView.cpp

```
(Ámbito global)
CVENTANA_DE_BIENVENIDAView::~CVENTANA_DE_BIENVENIDAView()
{
}

BOOL CVENTANA_DE_BIENVENIDAView::PreCreateWindow(CREATESTRUCT& cs)
{
    // TODO: modificar aquí la clase Window o los estilos cambia
    // CREATESTRUCT cs

    return CView::PreCreateWindow(cs);
}

// Dibujo de CVENTANA_DE_BIENVENIDAView

void CVENTANA_DE_BIENVENIDAView::OnDraw(CDC* /*pDC*/)
{
    CVENTANA_DE_BIENVENIDADoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;

    // TODO: agregar aquí el código de dibujo para datos nativos
}

// Impresión de CVENTANA_DE_BIENVENIDAView
```

Resultados

Mostrar resultados desde:

" "

```
VENTANA_DE_BIENVENIDA (Ejecutando) - Microsoft Visual Studio
Archivo Editar Ver Proyecto Generar Depurar Equipo Datos Herramientas Arquitectura Prueba Analizar Ventana Ayuda
Explorador de soluciones Vista de clases Propiedades Examinador de objetos VENTANA_DE_BIENVENIDAView.cpp X
CVENTANA_DE_BIENVENIDAView OnDraw(CDC * pDC)
}

// Dibujo de CVENTANA_DE_BIENVENIDAView

void CVENTANA_DE_BIENVENIDAView::OnDraw(CDC* pDC)
{
    CVENTANA_DE_BIENVENIDADoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;
    pDC->TextOutW(200,200,_T("Hola Mundo Que Tal os va con Visual C++ 2010? "));

    // TODO: agregar aquí el código de dibujo para datos nativos
}

// Impresión de CVENTANA_DE_BIENVENIDAView
```

Puntos de interrupción

Nuevo | X | Punto de interrupción | Etiquetas | Condición | Número de llamadas | Buscar: En la columna: X

Puntos de interrupción Resultados

Lista | Resultados | Líne 61 Col 5 Car 2 INS E5 20:48

➤ Digitar dentro de la función
OnDraw(CDC *pDC)

➤ pDC-
->TextOutW(200,200,_T("Hola Mundo que tal con Visual C++2010? "));



Sin título - VENTANA_DE_BIENVENIDA



Archivo Edición Ver Ayuda



Vista de archivos



Vista de clases

Hola Mundo Que Tal os va con Visual C++ 2010?

Resultados

Listo

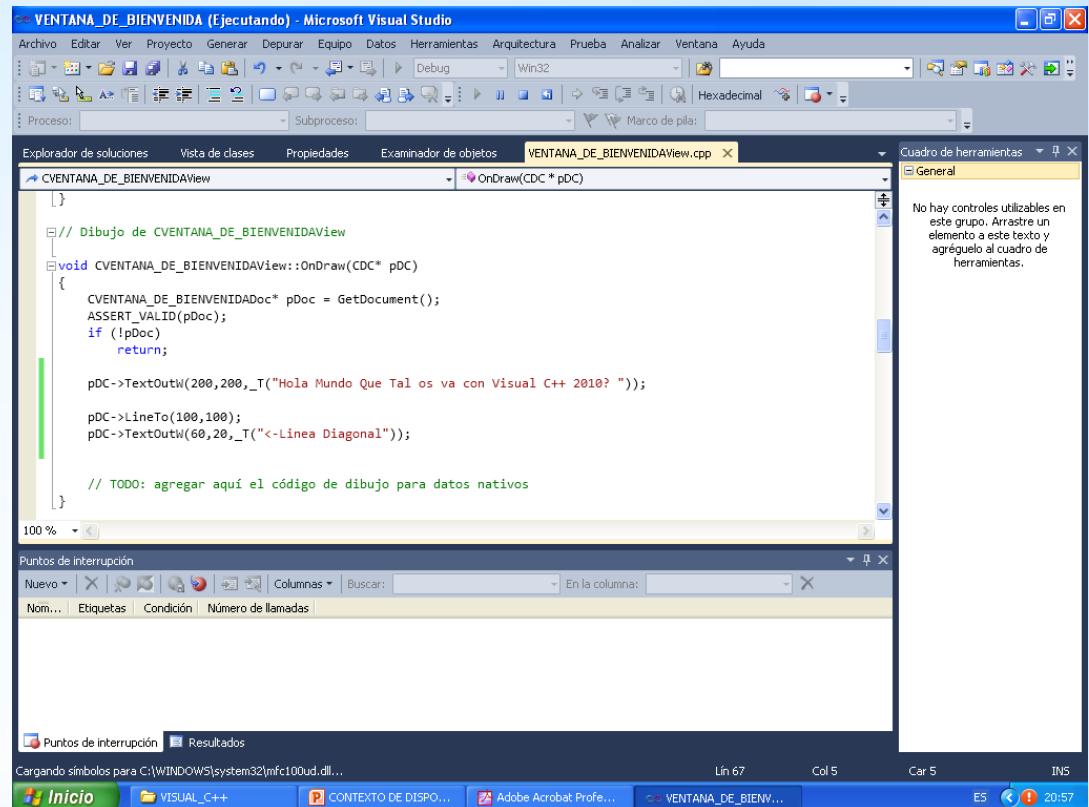
CAP NUM SCRL



Propiedades

GRAFICANDO LINEAS

➤ Para dibujar Líneas haremos uso de la aplicación anterior y agregaremos la siguientes lineas, en la funcion OnDraw(CDC* Pdc) contenido en la clase Ventana_de_BienvenidaView



```
//Dibuje una Linea Diagonal  
pDC->LineTo(100,100);  
pDC->TextOutW(60,20,_T("<-Linea Diagonal"));
```



Archivo Edición Ver Ayuda



Vista de archivos



Vista de clases

<-Linea Diagonal

Hola Mundo Que Tal os va con Visual C++ 2010?



Propiedades

Resultados

Listo

CAP NUM SCRL

GRAFICALINEAS (Ejecutando) - Microsoft Visual Studio

Archivo Editar Ver Proyecto Generar Depurar Equipo Datos Herramientas Arquitectura Prueba Analizar Ventana Ayuda

Debug Win32

Proceso: Subproceso: Marco de pila:

Explorador de soluciones Vista de clases Propiedades Examinador de objetos afxwin.h GRAFICALINEASView.cpp

CGRAFICALINEASView OnDraw(CDC * pDC)

```
// Dibujo de CGRAFICALINEASView
void CGRAFICALINEASView::OnDraw(CDC* pDC)
{
    CGRAFICALINEASDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;
    pDC->LineTo(100,100);
    pDC->TextOutW(60,20,_T("<-Vea la Linea Diagonal "));
}

// TODO: agregar aquí el código de dibujo para datos nativos

// Impresión de CGRAFICALINEASView
```

100 %

Puntos de interrupción

Nuevo | Buscar: En la columna: | Resultados

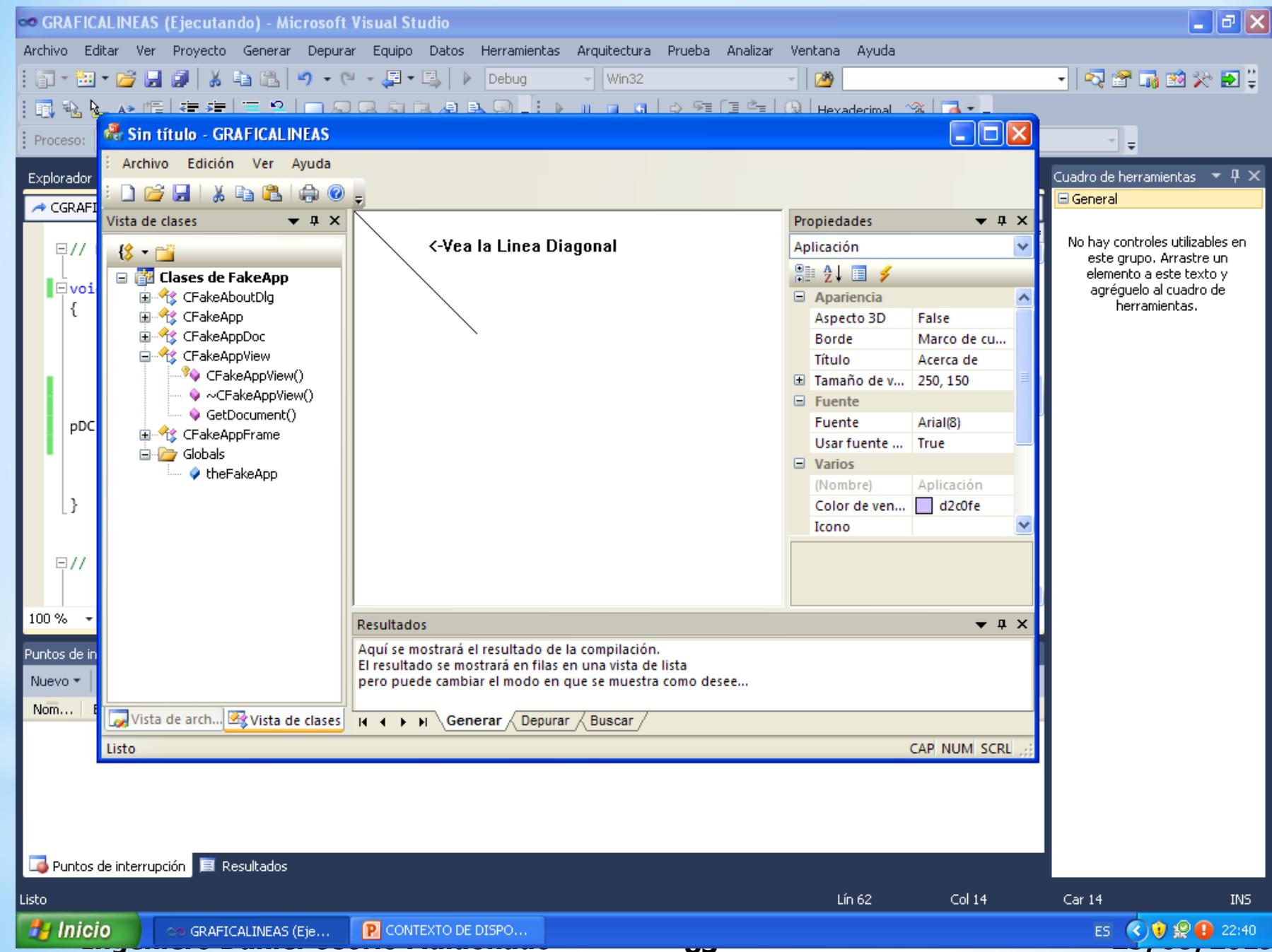
Nom... | Etiquetas | Condición | Número de llamadas |

Puntos de interrupción | Resultados

Lín 64 Col 1 Car 1 INS

No hay controles utilizables en este grupo. Arrastre un elemento a este texto y agréguelo al cuadro de herramientas.

Cuadro de herramientas General



Dibujando una Elipse, una Cuerda, un Sector Circular y un Rectangulo

➤ Una elipse la dibujaremos en la función OnDraw:

```
pDC->Ellipse(220,10,310,70);
```

```
pDC->TextOutW(330,20,_T("<--Elipse-->"));
```

➤ Una Cuerda la dibujamos en la función OnDraw:

```
pDC->Chord(10,100,100,200,20,125,120,175);
```

```
pDC->TextOutW(70,130,_T("<--Una Cuerda-->"));
```

➤ Un Sector Circular :

```
pDC->Pie(200,100,300,200,200,150,250,100);
```

```
pDC->TextOutW(300,130,_T("<--Un Sector-->"));
```

➤ Un Rectangulo

```
pDC->Rectangle(10,250,100,300);
```

```
pDC->TextOutW(110,280,_T("<--Un Rectangulo-->"));
```

GRAFICALINEAS (Ejecutando) - Microsoft Visual Studio

Archivo Editar Ver Proyecto Generar Depurar Equipo Datos Herramientas Arquitectura Prueba Analizar Ventana Ayuda

Debug Win32 Hexadecimal Marco de pila

Explorador de soluciones Vista de clases Propiedades Examinador de objetos afxwin.h GRAFICALINEASView.cpp Cuadro de herramientas

OnDraw(CDC* pDC)

```
// Dibujo de CGRAFICALINEASView

void CGRAFICALINEASView::OnDraw(CDC* pDC)
{
    CGRAFICALINEASDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;
    pDC->LineTo(100,100);
    pDC->TextOutW(60,20,_T("<-Vea la Linea Diagonal "));

    pDC->Ellipse(220,10,310,70);
    pDC->TextOutW(330,20,_T("<-Ellipse-->"));

    //Una Cuerda la dibujamos en la funcion OnDraw:
    pDC->Chord(10,100,100,200,20,125,120,175);
    pDC->TextOutW(70,130,_T("<-Una Cuerda-->"));
    //Un Sector Circular :
    pDC->Pie(200,100,300,200,200,150,250,100);
    pDC->TextOutW(300,130,_T("<-Un Sector-->"));

    //Un Rectangulo
    pDC->Rectangle(10,250,100,300);
    pDC->TextOutW(110,280,_T("<-Un Rectangulo-->"));
}
```

100 %

Puntos de interrupción

Nuevo | X | Etiquetas | Condición | Número de llamadas | Buscar: En la columna: | Resultados

Listo Lín 70 Col 23 Car 23 INS

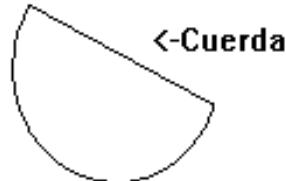
No hay controles utilizables en este grupo. Arrastre un elemento a este texto y agréguelo al cuadro de herramientas.



<-Linea Diagonal



<-Elipse



<-Cuerda

Hola Mundo que tal con Visual C++?

Archivo Edición Ver Ayuda



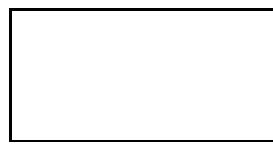
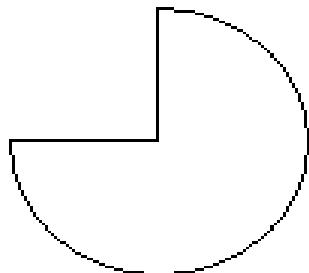
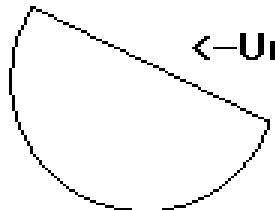
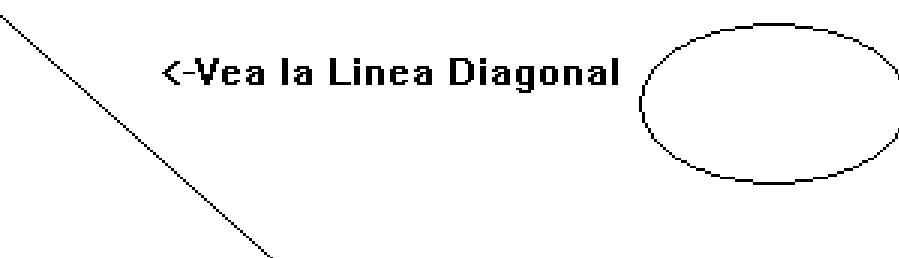
<-Vea la Linea Diagonal

<-Una Cuerda->

<-Un Rectangulo->

<-Elipse->

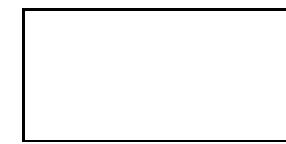
<-Un Sector->



Propiedades

Vista de archivos

Vista de clases



Resultados

Listo

Ingeniero Daniel Osorio Maldonado

87

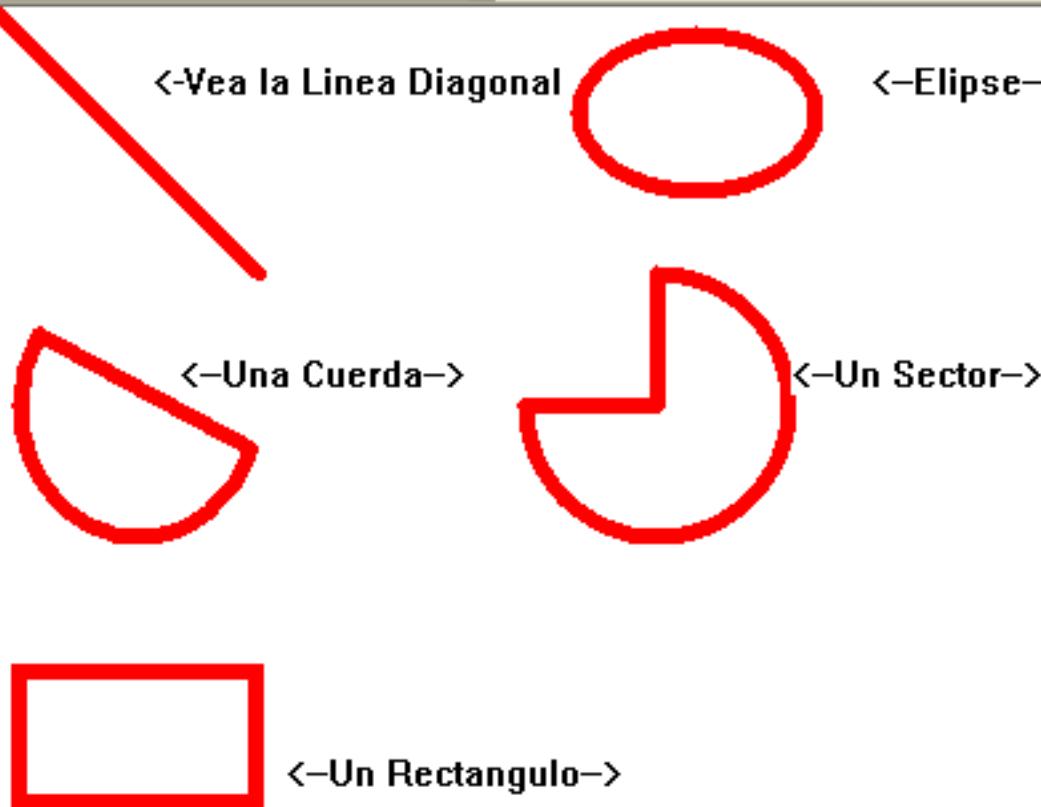
CAP NUM SCRL

25/08/2019

Usando la clase Cpen

- Se trata de utilizar el mismo proyecto para hacer uso de la clase Cpen.
- Seleccione la clase Classview y active la función OnDraw(CDC *pDC).
- Estando en la función OnDraw(...) edítela y posesiónese antes del dibujo de líneas
- Escriba el siguiente código

```
CPen *n_pincel=new CPen; // Crea un objeto puntero del tipo CPen  
n_pincel->CreatePen(PS_SOLID,6,RGB(255,0,0)); //initialization  
pDC->SelectObject(n_pincel);
```



GRAFICALINEAS (Ejecutando) - Microsoft Visual Studio

Archivo Editar Ver Proyecto Generar Depurar Equipo Datos Herramientas Arquitectura Prueba Analizar Ventana Ayuda

Proceso: [2928] GRAFICALINEAS.exe Subproceso: [2932] Subproceso principal Marco de pila: GRAFICALINEAS.exe!CGRAFICALINEASView::

Explorador de soluciones Vista de clases Propiedades Examinador de objetos wingdi.cpp afxwin.h GRAFICALINEASView.cpp

CGRAFICALINEASView OnDraw(CDC * pDC)

```
// Dibujo de CGRAFICALINEASView

void CGRAFICALINEASView::OnDraw(CDC* pDC)
{
    CGRAFICALINEASDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;
    // utilizando un pincel personalizado: sólida, ancho 5, color rojo
    CPen *n_pincel=new CPen;//Crea un objeto puntero del tipo Cpen
    n_pincel->CreatePen(PS_SOLID,6,RGB(255,0,0)); //initialization
    pDC->SelectObject(n_pincel);

    pDC->LineTo(100,100);
    pDC->TextOutW(60,20,_T("<-Vea la Linea Diagonal "));

    pDC->Ellipse(220,10,310,70);
    pDC->TextOutW(330,20,_T("<-Elipse-->"));

    //Una Cuerda la dibujamos en la función OnDraw:
    pDC->Chord(10,100,100,200,20,125,120,175);
    pDC->TextOutW(70,130,_T("<-Una Cuerda-->"));
    //Un Sector Circular :
    pDC->Pie(200,100,300,200,150,250,100);
    pDC->TextOutW(300,130,_T("<-Un Sector-->"));

    //Un Rectángulo
}
```

100 %

Puntos de interrupción

Nuevo | X | S | G | Columns | Buscar: | En la columna: | Resultados

Listo Lín 62 Col 28 Car 25 INS

Inicio CONTEXTO DE DISPO... GRAFICALINEAS (Eje... GRAFICOS_SVP Adobe Acrobat Profes... 7:37

Usando la clase Cbrush

- Active la pestaña ClassView y edite la función OnDraw.
- Agregue el siguiente código antes del dibujo de las figuras:

```
CBrush *n_brocha=new CBrush; // Crea un objeto  
n_brocha->CreateHatchBrush(HS_CROSS,RGB(255,0,0));  
  
pDC->SelectObject(n_brocha);  
pDC->LineTo(100,100);  
pDC->TextOutW(60,20,_T("<-Vea la Linea Diagonal "));  
  
pDC->Ellipse(220,10,310,70);  
pDC->TextOutW(330,20,_T("<--Elipse-->"));  
  
//Una Cuerda la dibujamos en la función OnDraw:  
pDC->Chord(10,100,100,200,20,125,120,175);  
pDC->TextOutW(70,130,_T("<--Una Cuerda-->"));  
//Un Sector Circular :  
pDC->Pie(200,100,300,200,200,150,250,100);  
pDC->TextOutW(300,130,_T("<--Un Sector-->"));  
  
//Un Rectangulo  
pDC->Rectangle(10,250,100,300);  
pDC->TextOutW(110,280,_T("<--Un Rectangulo-->"));
```



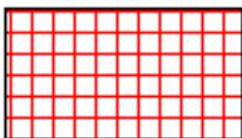
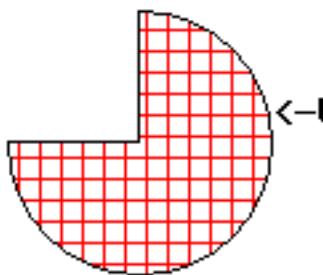
<-Vea la Linea Diagonal

<-Una Cuerda->

<-Un Rectangulo->

<-Elipse->

<-Un Sector->



FIGURASCERRADAS (Ejecutando) - Microsoft Visual Studio

Archivo Editar Ver Proyecto Generar Depurar Equipo Datos Herramientas Arquitectura Prueba Analizar Ventana Ayuda

Debug Win32 Hexadecimal Marco de pila:

Proceso: Subproceso: Marco de pila:

Explorador de solución... FIGURASCERRADASView.cpp

(Ámbito global)

```
return CView::PreCreateWindow(cs);
}

// Dibujo de CFIGURASCERRADASView

void CFIGURASCERRADASView::OnDraw(CDC* pDC)
{
    CFIGURASCERRADASDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;
//POINT vlados[4]={-150,100,150,10};
POINT vlados[10]={150,60,200,200,100,100,200,100,100,200};
//Ahora dibuja las líneas
pDC->Polygon(vlados,5);

    // TODO: agregar aquí el código de dibujo para datos nativos
```

100 %

Automático Pila de llamadas

Nombre	Valor	Tipo
--------	-------	------

Nombre	Leng
--------	------

Explorad... Vista de... Automático Variables l... Subproce... Módulos Inspecció... Pila de llamadas Puntos de interrupción Resultados

Listo Lín 53 Col 34 Car 34 INS



Sin título - FIGURASCERRADAS



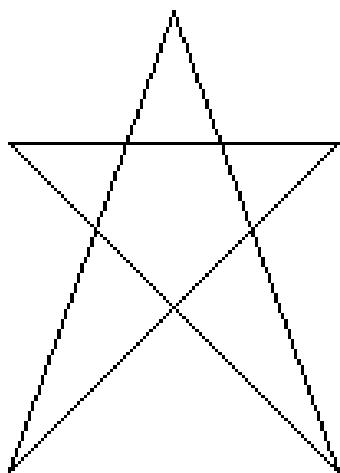
Archivo Edición Ver Ayuda



Vista de archivos



Vista de clases



Propiedades

Resultados

Ingenier

Listo

CAP NUM SCRL

23/08/2019