

Lesson 1 - VBX controls and MFC

What You Will Learn

By the end of this lesson, you should be able to do the following:

- Know what VBX controls are, and their relationship to MFC.
- Become familiar with the datastructures used to write a VBX control.
- Know how VBX controls are used in MFC applications.
- Understand the basics of writing a VBX control in MFC to be used by applications written in MFC and VB

What is a VBX control and why use it?

- A VBX control is a DLL.
 - This allows the control to have the numerous advantages of a DLL.
 - Applications written in different languages (VB and C/C++ with MFC 2.0) can use the control.
- By writing a VBX control the developer dramatically increases his/her market.
- VBX controls fit well into the object oriented paradigm of encapsulation, as communication between the VBX control and its user is done by passing messages.

What To Do

μ §

μ §

General VBX Control Structures

What is the HCTL?

- The hctl structure contains the VBX specific data.
 - Property tables (explanation soon)
 - Event tables (explanation soon)
 - MODEL structure (explanation soon)
 - Other VB specific data structures
- The handle to the hctl structure is appended as the first parameter to the Control Procedure.
Example:
`ControlProcedure(HCTL,HWND,UINT,WPARAM,LPARAM)`
- Windows does not know anything about the HCTL structure.
- There is a unique hctl structure for every instance of a vbx control.

What is a Property?

- The control property identifies a particular attribute of a control.
- Examples of properties are:
 - Caption, BackgroundColor, Border....etc
- Each control has its own set of unique properties.

----- In MFC -----

- Properties can be set anytime during the execution of a program.
- There are public member functions of class CVBControl that can be used to get and set properties. Refer to the help on Class CVBControl for more information on these functions.
 - Example:
Note:: m_circle is an object of class CVBControl
`m_circle.SetNumProperty("CircleShape", dlg.m_nShape);`
`m_circle.SetStrProperty ("Caption", dlg.m_strCaption);`

Refer to VBCIRCLE sample program under the MFC samples directory.

- You can use AppStudio to set the properties of a VBX control.

What is an Event?

- The control event is a custom message used between the vbx control and the application.
- Examples of events are:
 - ClickIn, ClickOut....etc
- Each control has its own set of unique events.

----- In MFC -----

- It is **very important** to register all events of a control that an MFC application will process. The function used to register events is:

```
UINT AfxRegisterVBEvent( LPCSTR lpszEventName );  
where lpszEventName = The name of the VB event.
```

Example:

```
UINT NEAR VBN_CLICK = AfxRegisterVBEvent("CLICKIN"); // File Scope  
  
BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)  
    ON_VBXEVENT(VBN_CLICK, IDC_CIRCLE1, OnClickCircle1)  
END_MESSAGE_MAP()
```

- You can use ClassWizard to register the events to be used by the application and create member functions for them.
- For an example program on using vbx events refer to the VBCIRCLE mfc sample program.

What is the MODEL Structure ? (optional):

- The MODEL structure is the data structure which defines all attributes of a control class.
- This structure contains the fields that identify the model model flags, property list, event list, pointer to control procedure...etc.
- Knowledge of this structure is only needed when writing a VBX control and not when using the control.
- An example MODEL structure definition:

```
MODEL modelCircle =  
{  
    VB_VERSION, // VB version being used
```

```

0, // MODEL flags
(PCTLPROC)CircleCtlProc, // Control procedure
CS_VREDRAW | CS_HREDRAW, // Class style
WS_BORDER, // Default Windows style
0, // Size of CIRCLE structure
IDBMP_CIRCLE, // Palette bitmap ID
"Circle", // Default control name
"CIRC1", // Visual Basic class name
NULL, // Parent class name
Circle_Properties, // Property information table
Circle_Events, // Event information table
IPROP_CIRCLE_BACKCOLOR, // Default property
IEVENT_CIRCLE_CLICK, // Default event
-1 // Property representing value of ctl
};

```

Summary for Writing an MFC application that uses a VBX control

- **CWinApp::EnableVBX()**

Calling this function is a **must** if your program is to work correctly.
 Call this function from within the InitInstance() of your class that was derived from CWinApp.

- **To create a vbx control in a window, use the Create() for the CVBControl class:**

```

// Create the VB control, referring to its name and .VBX file.
m_circle.Create("circ3.vbx;Circ3;Caption Text",
    //<vb_x_file_name>;<vb_control_name>;<window text>
    WS_CHILD | WS_VISIBLE,
    CRect(10, 30, 210, 180), // initial position of the circle control
    this,
    1); // arbitrary child window i.d., not needed in this application

```

- **Processing Events:**

Register the events using AfxRegisterVBEvent()
 In your MessageMap() use the ON_VBXEVENT() macro
 to define the member function that is to be called.

- **Manipulating Properties:**

Use the public member functions of class CVBControl.
 For more information, refer to the help on CVBControl.

For example:
Refer to VBCIRCLE sample program!!!!

Writing an MFC application that uses a VBX control

- Steps needed:

[1] Use AppWizard to create your basic application with the
“Custom VBX Controls” option checked.

[2] Run AppStudio, load the .rc file.

Select the “File” menu and select “Install controls”

Once you install the the vbx control it will show up in the Tool Box.

Select the control onto your dialog box and then double
click on it to edit the control’s properties if needed.

[3] Run ClassWizard and edit the Dialog class.

You can select the events you want to process.

When this is done, ClassWizard will register the events,
then insert events into the Dialog class’s message map
and insert the member functions needed to process the
events.

[4] You can now do whatever you want to in the member functions that are created.

[5] Compile, link and run the program.

VBX controls written using MFC

- If the VBX control is a _USRDLL, then:

- This allows the vbx to be used by both MFC and VB applications.

Else if the VBX control is a _AFXDLL then:

- Only an MFC application can use it.

- For information on _AFXDLL or _USRDLL refer to Technote 33.

- All Property, Event and Model structure definitions **must** be NEAR.

- Failing to do this will cause random consequences.

Question: Can a _USRDLL support a vbx control?

Answer: No.

Question: Can an _AFXDLL support a vbx control?

Answer: Yes.