Tailor Your Displays

by Karl E. Peterson and Phil Weber

Click & Retrieve Source CODE!

Q DISPLAY PROGRESS BAR IN STATUS BAR

I'd like to display a progress bar within a status bar panel, but VB's status bar control just doesn't seem up to it. How can I accomplish this?

A You're right. The status bar supplied by the common controls doesn't act as a control container, so there's no direct way to place a progress bar control upon it. But that doesn't mean it's not possible. Status bar panels do provide a Picture property. If you can obtain a picture representation of a progress bar, you can assign that to your chosen panel.

I've written a class that wraps up all the functionality of the standard progress bar by drawing directly to a Picture control (see Programming Techniques, *VBPJ* March 1996). Several small modifications to this class add the capability of implanting this emulated progress bar within a status panel (see Figure 1). You can download the CProgBarEx class and a demo from the free, Registered Level of The Development Exchange (see the Code Online box at the end of the column for details).

In most respects, CProgBarEx operates identically to the original CProgBar class. I've added a Host property to which the status bar control is assigned, and a PanelKey property that tells the class which panel the progress bar should appear in (see Listing 1). Previously, the class drew the rendered progress bar on a visible picture box, but that's no longer desirable. The class now sets the Client picture box to be invisible with AutoRedraw turned on, so the graphic persists. Whenever the graphic is updated, the class reassigns it to the Picture property of the status bar panel. The class also sizes the picture box to match the space available within the panel (see Figure 2).

If your design calls for a single main form and you'd always like to display progress and status messages on it, expose the Progress and Status properties directly from that form (see Listing 2). This way, any other form, module, or class in the project can display its own status/progress to the user.

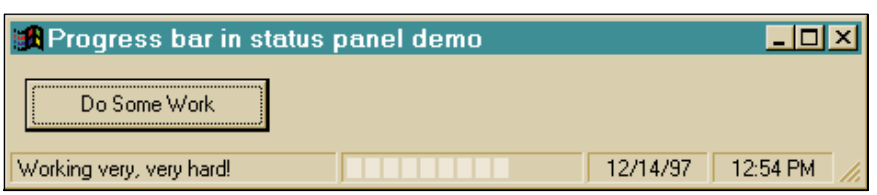

FIGURE 1 ***See Some Progress.*** *Here's the CProgBarEx class in action. The progress bar was drawn into a hidden PictureBox, and the persistent bitmap of that control was assigned to the Picture property of the status bar panel.*

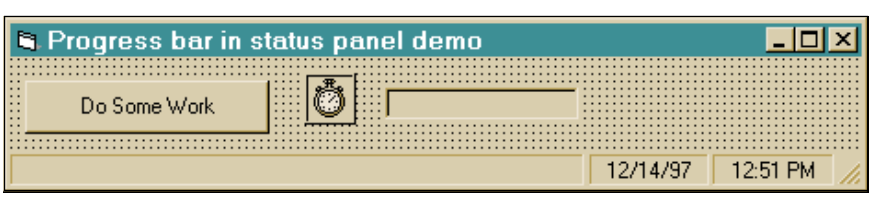

FIGURE 2 ***Client Can Be Any Size.*** *The CProgBarEx class sets all properties required to allow a PictureBox control to act as the canvas upon which a progress bar is rendered. Design-time precision is not required. In this example, the second panel is not visible—it's toggled on and off according to the current value of the progress bar.*

Karl E. Peterson is a GIS analyst with a regional transportation planning agency and serves as a member of the Visual Basic Programmer's Journal *Technical Review and Editorial Advisory Boards. Based in Vancouver, Washington, he's also an independent programming consultant specializing in ActiveX controls and contributes to various journals. Karl coauthored* Visual Basic 4 How-To, *from Waite Group Press. Online, he's a Microsoft MVP, and a section leader in both* VBPJ *online forums. Find more of Karl's VB samples at http://www.mvps.org/vb.*

Phil Weber is an independent consultant specializing in Visual Basic and Web site development. He is a Microsoft Certified Solution Developer and Product Specialist. Find more of Phil's VB tips on his Web site: http://www.teleport.com/~pweber.

Ask the VB Pro provides you with free advice on programming obstacles, techniques, and ideas. Read more answers from our crack VB pros on the Web at www.inquiry.com/thevbpro. You can submit your questions, tips, or ideas on the site, or access a database of previously answered questions.

VB4 32-bit VB5

```
Public Property Set Client(ByVal NewObj As Object)
  ' Set new PictureBox as Client property.
  If TypeName(NewObj) = "PictureBox" Then
     Set m_Bar = NewObj
     With m_Bar
        .ForeColor = m_fColor
        .BackColor = m_bColor
        .ScaleMode = vbPixels
        .AutoRedraw = True
        .BorderStyle = 0 'None
        .Enabled = False
        .TabStop = False
        .Visible = False
     End With
     ResizeEx
  Else
     ' Raise error #13 --> Type Mismatch
     Err.Raise Number:=13, _
        Source:="CProgressBarEx.Client", _
        Description:="Client property must be " & _
        "of type PictureBox."
  End If
End Property

Public Property Set Host(ByVal NewObj As Object)
  ' Set new StatusBar as Host property.
  If TypeName(NewObj) = "StatusBar" Then
     Set m_Sts = NewObj
     If Not m_Bar Is Nothing Then
        ResizeEx
     End If
  Else
     ' Raise error #13 --> Type Mismatch
     Err.Raise Number:=13, _
        Source:="CProgressBarEx.Host", _
        Description:="Host property must " & _
        "be of type StatusBar."
  End If
End Property

Public Property Let PanelKey(ByVal NewVal As String)
  m_PanKey = NewVal
End Property
```

LISTING 1 ***Provide a Suitable Host.*** *The Host and PanelKey properties, added to the CProgBar class, instruct the class which status bar control, and which panel within that control, to draw a progress bar on. The Client property has been updated to set a few additional properties of the hidden PictureBox control.*

VB4 32-bit VB5

```
Private m_ProgBar As CProgressBarEx
Private m_Progress As Long
Private m_Status As String

' ***************************************************
'  Public Properties
' ***************************************************
Public Property Let Status(ByVal NewVal As String)
  Const StatKey = "pnlStatus"
  m_Status = NewVal
  sbStatusBar.Panels(StatKey).Text = m_Status
End Property

Public Property Get Status() As String
  Status = m_Status
End Property

Public Property Let Progress(ByVal NewVal As Long)
  Const ProgKey = "pnlProgress"

  If m_ProgBar Is Nothing Then
     Set m_ProgBar = New CProgressBarEx
     Set m_ProgBar.Client = picProgress
     Set m_ProgBar.Host = sbStatusBar
     m_ProgBar.PanelKey = ProgKey
     m_ProgBar.ResizeEx
  End If

  With sbStatusBar.Panels(ProgKey)
     If NewVal = 0 Then
        .Visible = False
     ElseIf NewVal > 0 And NewVal <= 100 Then
        If .Visible = False Then .Visible = True
     Else
        Exit Property
     End If
  End With

  m_Progress = NewVal
  m_ProgBar.Value = m_Progress
End Property

Public Property Get Progress() As Long
  Progress = m_Progress
End Property
```

LISTING 2 ***Display Progress and Status from Anywhere in Your App.*** *The main form in your application can expose the Progress and Status properties. Status simply updates the designated panel with whatever status text it receives. Progress handles all aspects of using the CProgBarEx class. If the class has yet to be created, the program instantiates an instance and assigns appropriate values to crucial properties, before passing the desired value along for rendering by CProgBarEx.*

THE TREEVIEW CONTROL SUPPORTS A NEW "CHECK BOX" STYLE.

Q DISPLAY CHECK BOXES IN TREEVIEW

How can I get a TreeView control to display a check box next to each node, like Microsoft Backup does?

A Beginning with version 4.70 of COMCTL32.DLL (the version that shipped with Internet Explorer 3.0), the TreeView control supports a new "check box" style (TVS_CHECKBOXES). You can use the GetWindowLong and SetWindowLong API functions to set a VB TreeView control to this style. Use the SendMessage function to read or change a node's checked state (see Listing 3).

The code in Listing 3 suffers from several limitations:

• For the code to work, you and your users must have version 4.70 or higher of COMCTL32.DLL. If you don't have this version, or if you plan to distribute this code to others, you can download an updated and redistributable version of the DLL for Windows 95 from http://premium.microsoft.com/support/downloads/dp2722.asp. Windows NT 4.0 ships with the IE3 version of COMCTL32.DLL, so users on this platform shouldn't have any trouble.

• To read or change a node's checked state, you must get an HTREEITEM handle for that node. The only way I could find to obtain this handle is to select the node and then call the API to get the handle of the currently selected node. You might wish to add code to save the selected node before calling GetCheckedState or SetCheckedState and restore it afterward. You might also want to call LockWindowUpdate to prevent the TreeView from repainting while the code loops through the

VB4 32-bit VB5

```
Public Sub SetCheckBoxStyle(tvw As TreeView)
  Dim hWnd As Long
  Dim lStyle As Long
  hWnd = tvw.hWnd
  lStyle = GetWindowLong(hWnd, GWL_STYLE)
  lStyle = lStyle Or TVS_CHECKBOXES
  Call SetWindowLong(hWnd, GWL_STYLE, lStyle)
End Sub

Public Function GetCheckedState(tvw As TreeView, _
  ByVal lItem As Long) As Boolean
  Dim hWnd As Long
  Dim tvi As TVITEM
  With tvw
     hWnd = .hWnd
     ' Select node so we can get its handle below
     .Nodes(lItem).Selected = True
  End With
  With tvi
     .mask = TVIF_HANDLE Or TVIF_STATE
     ' SendMessage returns handle of currently-selected node
     .hItem = SendMessage(hWnd, TVM_GETNEXTITEM, _
        TVGN_CARET, ByVal 0&)
     .stateMask = TVIS_STATEIMAGEMASK
  End With
  ' Node's checked state returned in upper bits of
  ' tvi.state
  Call SendMessage(hWnd, TVM_GETITEMA, 0, tvi)
  GetCheckedState = CBool((tvi.state \ &H1000) - 1)
End Function
Public Sub SetCheckedState(tvw As TreeView, _
  ByVal lItem As Long, ByVal bState As Boolean)
  Dim hWnd As Long
  Dim tvi As TVITEM
  With tvw
     hWnd = .hWnd
     .Nodes(lItem).Selected = True
  End With
  With tvi
     .mask = TVIF_HANDLE Or TVIF_STATE
     .hItem = SendMessage(hWnd, TVM_GETNEXTITEM, _
        TVGN_CARET, ByVal 0&)
     .stateMask = TVIS_STATEIMAGEMASK
  End With
  Call SendMessage(hWnd, TVM_GETITEMA, 0, tvi)
  tvi.state = (tvi.state And (Not _
     TVIS_STATEIMAGEMASK)) _
     Or (-CLng(bState) + 1) * &H1000
  Call SendMessage(hWnd, TVM_SETITEMA, 0, tvi)
End Sub
```

LISTING 3 ***Make Check Boxes Grow on Trees.*** *By setting the TVS_CHECKBOXES style of a TreeView control, you can make a check box appear adjacent to each node. Due to space limitations, this listing omits the API function declarations and constant definitions.*

nodes to determine which are checked:

```
' Print text of checked items to
' Debug window
With tvwCheck
   Call LockWindowUpdate(.hWnd)
   For I = 1 To .Nodes.Count
      If GetCheckedState(tvwCheck, _
         I) Then
         Debug.Print .Nodes(I).Text
      End If
   Next
   Call LockWindowUpdate(0)
End With
```

• The TreeView must be visible before you can set a node's checked state. ☒

Code Online

You can find all the code published in this issue of VBPJ on The Development Exchange (DevX) at http://www.windx.com. For details, please see "Get Extra Code in DevX's Premier Club" in Letters to the Editor.

Tailor Your Displays

Locator+ Codes

Listings for the entire issue, supplying the complete PageSetup function, plus the class CProgBarEx for progress bar functionality (free Registered Level): VBPJ0398
Listings for this article only, plus the files described above (subscriber Premier Level): AP0398