Visual Studio Magazine Online
Classic VB Corner

Using a Standard Terminal Font

ONLINE ONLY

Need to recreate that olde-timey Terminal look? There's a stock object built into every version of Windows that'll serve that need.

June 2, 2009 · by Karl E. Peterson

A reader recently asked how to recreate the look of an old -- some might say ancient -- terminal app that made heavy use of the box drawing characters. Remember those? He'd tried using the "Terminal" font, but found that it wasn't consistently available on all Windows installations, and feared he'd need to distribute it along with his application. While that certainly wouldn't be the end of the world, assuming the font is even licensed for such redistribution, it's obviously a headache -- and one he wouldn't want to take on unnecessarily.

Windows provides a series of "stock" fonts, easily retrievable with a quick call to the GetStockObject API function. This function returns a handle to the desired object, in this case a font, which you can then select into the desired window using SendMessage and WM_SETFONT as follows:

```
Public Sub SetOemFont(ByVal hWnd As Long)
   Dim hFont As Long
   Const OEM_FIXED_FONT As Long = 10
   Const WM_SETFONT As Long = &H30
   ' Select the stock font into the client window.
   ' Not necesary to call DeleteObject.
   hFont = GetStockObject(OEM_FIXED_FONT)
   Call SendMessage(hWnd, WM_SETFONT, hFont, True)
End Sub
```

This sample routine is about as simple as it gets. Just pass it the desired window handle (e.g., Text1.hWnd) and the rest is taken care of for you. The only real option is the last parameter to WM_SETFONT, which is supposed to instruct the control whether or not to redraw itself. This doesn't seem to have any effect with the intrinsic controls in ClassicVB, as they all redraw using the newly selected font regardless of how that parameter is set.

There are actually seven stock fonts offered by Windows, though I wouldn't recommend using any of the others as there are generally far better options available and some are simply well-known for being unreliable. Indeed, the one that sounds most useful -- DEFAULT_GUI_FONT -- is probably the worst offender. For that, you'll want to use SystemParametersInfo to query non-client metrics. Long ago, I provided a drop-in ready class module which you can download from my Web site that will retrieve all that information for you. Here's the complete list of the available stock font objects:

```
' API stock font identifier constants
Private Const OEM_FIXED_FONT        As Long = 10
Private Const ANSI_FIXED_FONT       As Long = 11
Private Const ANSI_VAR_FONT         As Long = 12
Private Const SYSTEM_FONT           As Long = 13
Private Const DEVICE_DEFAULT_FONT   As Long = 14
Private Const SYSTEM_FIXED_FONT     As Long = 16
Private Const DEFAULT_GUI_FONT      As Long = 17
```

It's an easy step from the above hard-wired function, to one that will select any of the available stock font objects into your control. Just expose an enumeration of the available fonts, to make the coding that much easier, and add an option to force or ignore the redraw option:

```
' Enumerated fonts
Public Enum StockFonts
   sfOEM_FIXED = OEM_FIXED_FONT
   sfANSI_VAR = ANSI_VAR_FONT
   sfANSI_FIXED = ANSI_FIXED_FONT
   sfSYSTEM_VAR = SYSTEM_FONT
   sfSYSTEM_FIXED = SYSTEM_FIXED_FONT
   sfDEVICE_DEFAULT = DEVICE_DEFAULT_FONT
End Enum

Public Sub SetStockFont(ByVal hWnd As Long, _
      ByVal NewFont As StockFonts, _
      Optional Redraw As Boolean = True)
   Dim hFont As Long
   ' Select the stock font into the client window.
   ' Not necesary to call DeleteObject.
   hFont = GetStockObject(NewFont)
   Call SendMessage(hWnd, WM_SETFONT, hFont, Redraw)
End Sub
```

If you'd like to just download an example, you can also find that on my Web site. Running it, you'll see something like this:

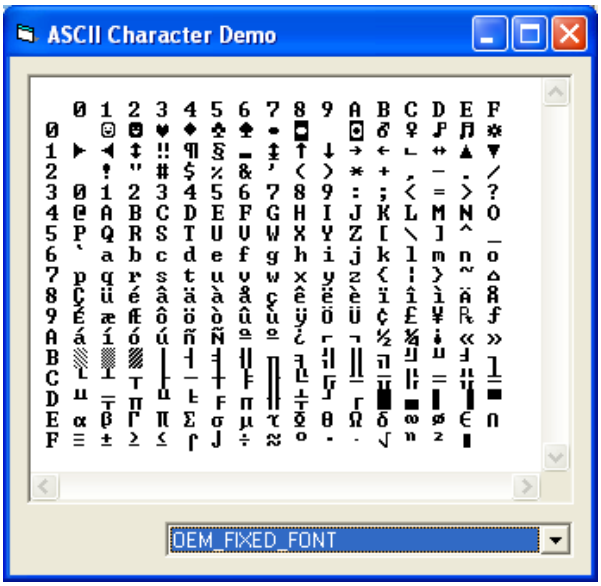

Figure 1. Blast from the past! Remember those old box drawing characters? Yes, you can still use them in your ClassicVB applications. That is, if your clients force you to.

I hope you find this little trick useful -- but likewise hope you don't get stuck recoding too many look-alikes from the '80s.

About the Author

Karl E. Peterson wrote Q&A, Programming Techniques, and various other columns for VBPJ and VSM from 1995 onward, until Classic VB columns were dropped entirely in favor of other languages. Similarly, Karl was a Microsoft BASIC MVP from 1994 through 2005, until such community contributions were no longer deemed valuable. He is the author of VisualStudioMagazine.com's new Classic VB Corner column. You can contact him through his Web site if you'd like to suggest future topics for this column.