ReadMe.1st

Welcome to VBMax Electronic Message Display DLL. Now you can add electronic message board controls to your VB 4.0 applications—without the overhead of OCXs. Less filling—works great.

I know you're probably champing at the bit and can't wait to jump right in and run the demonstration program to see what all the fuss is about. Before you can do that, however, there is a little matter of the Windows registry to take care of first.

VBMaxEM.dll is an in-process OLE server and must be installed properly before you can use it. If you don't install it, the demo won't work. For information on how to do this, see Installation Instructions.

Don't worry, if—for some strange reason—you are not super-impressed by this product you can easily uninstall it later.

About VBMax

Visual Basic is no longer just for prototypes. VB has evolved into a rich and powerful language and continues to go from strength to strength with each release. With version 4.0, VB makes it possible to write VB add-ins and both in-process and out-of-process OLE servers without even breaking a sweat.

Whether they realize it or not, this is a major leap forward for VB programmers. My belief is the technology will continue to advance and that using VB to develop software components instead of C, C++ or Pascal is going to be the way of the future. **Microsoft has announced already that the next release of VB will be able to create ActiveX controls.**

VBMax programming components herald the dawn of this new era. Written *by* a VB developer *for* VB developers, VBMax add-ins and DLLs are comprised entirely of VB 4 code without any help from third-party controls. Hence the name VBMax—it means 'Visual Basic to the Max'.

My goal is to provide high-quality Visual Basic products at an affordable price. Also, registered users receive the complete, commented source code so that they can see how it works and can even change it to suit their own preferences if they so desire.

Key Benefits

- Less overhead than third-party controls.
- Objects can be used in servers or incorporated directly into applications.
- VB source-code provided.
- Saves hours of coding.
- Inexpensive.
- No royalties.
- No need to continually buy upgrades with each new release of VB.

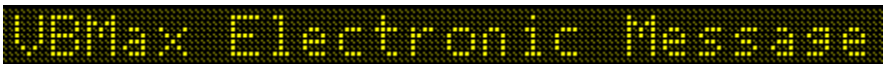
Contact Information

Postal address: 25 Lansdowne Street, Manchester, NH 03103, USA

Electronic mail: 74632.2227@compuserve.com

Web Site: http://ourworld.compuserve.com/homepages/vbmax

Introduction



Overview

Have you seen those electronic billboards that display messages using moving lights? Wouldn't it be cool to add one to your killer VB app? You can't do that in VB though. Right?

Wrong! VBMaxEM.dll handles the task with aplomb.

VBMaxEM.dll is an in-process OLE server for adding electronic message style controls to your VB 4.0 applications. It contains methods and properties for:

- Controlling static or scrolling displays
- Adjusting the speed
- Changing the foreground and background colors
- Showing or hiding the grid
- Handling callbacks
- *Dozens* of special effects

Price: $10

What's New in Version 1.0a

Version 1.0a

Credit Cards

SWReg is great—if you're a CompuServe member. But what if you're not? Registration can be both difficult and expensive for non-CompuServe members who live outside the USA.

To address this issue, you can now register the software using your Visa, MasterCard or American Express credit cards. See Credit Card Registration for more information.

Another day, another problem solved.

Web Site

There's just no getting away from it—you have to have a web site these days. The ubiquitous web has snagged us all whether we like it or not. How did we ever manage before?

Anyway, if you can't beat 'em, join 'em. The VBMax home page is

http://ourworld.compuserve.com/homepages/vbmax

Installation Instructions

Important—Use at Your Own Risk

While every effort has been made to ensure a reliable, high-quality product, you should note that this software is provided 'as is' without any kind of warranty whatsoever, neither explicit nor implied.

In order to use this software, you must agree 100% that I will not be held liable in any way, shape or form (not even a *little* bit) for anything untoward that befalls anyone or anything, either directly or indirectly, as a result of using this software no matter how calamitous, disastrous or inauspicious the occasion may be.

Your use of this software constitutes acceptance of these terms.

Installation

Before you can use the DLL, you need to do two things:

1. Create an entry for it in the Windows registry.
2. Add a reference to it in your VB project.

You register the DLL in Windows using **RegSvr32.exe**. You may have this program installed already in your Windows system directory. If not, you can find it in the Tools\PSS directory on your VB 4.0 CD. All you have to do is run the program, passing it the DLL filename as a command line argument like this:

REGSVR32 VBMAXEM.DLL

To uninstall the DLL, just add */u* to the command line:

REGSVR32 /U VBMAXEM.DLL

To make life a little easier, I created two batch files—Reg.bat and Unreg.bat—for you to run if you prefer. You may need to edit them first so that they have the correct path for RegSvr32.exe.

After registering the DLL, the next step is to include a reference to it in your VB project. You need to do this for the demo program and any of your own programs which use the class. If you see the message *"Could not create reference: 'VBMax Electronic Message Display for VB 4.0'.—Continue Loading Project?"* when loading the demonstration program, ignore it and click the 'Yes' button.

You add the reference to your project as follows:

- Select Tools from VB's menu.
- Then select References
- Locate *VBMax Electronic Message Display* in the list and click the check box so that an X appears in it.
- Click OK.

The OLE server is now visible in the Object Browser and you can use the CElectronicDisplay class.

Packing List

VBMaxEM.hlp	This online help file.
VBMaxEM.cnt	Help file contents.
Demo.vbp (and related files)	Source code for the demo program showing the DLL in action.
Reg.bat	Registers the DLL with Windows (you may have to edit the path).
Unreg.bat	Unregisters the DLL with Windows (you may have to edit the path).
VBMaxEM Source Code.grk	Source code for the DLL in gronked format.
Degronker.exe	Utility to extract the source code from the gronked file.

How to Use the DLL

When you include a reference to VBMaxEM.dll in your VB project, as described in Installation Instructions, you will have access to a class called **CElectronicDisplay**. This help file contains detailed information on how to use the properties and methods of the class and includes sample code which you can copy into your program.

The same information is also available through the Object Browser. To access the help file in this way, open up the Object Browser and select ***VBMax Electronic Message Display*** from the Libraries/Projects pull-down list. Click on a method or property and you will see a brief description of it at the bottom of the dialog. If you want more information and example code, click on the {button ?,} button which will display the online help for the selected item.

In your application, you create one or more objects from this class module. You need a separate object for each electronic message control.

Objects are created from the class by using code like this:

```
Dim moEDisp As New CElectronicDisplay
```

You create an electronic message control by drawing a PictureBox control on a form and passing it to the Container property in the object created by CElectronicDisplay. You then set other properties and apply various methods to complete the look and behavior of the control.

The DLL provides a high level of control over the appearance and behavior of the electronic message display. The demo program includes many examples.

Frequently Asked Questions

I try to run your demo program but it keeps telling me that OLE cannot create the object. Why?

You have not registered VBMaxEM.dll with Windows. Please refer to the Installation Instructions.

I have searched my hard-drive but cannot find regsvr32.exe—how can I install VBMaxEM.dll?

Regsvr32.exe is not copied to your hard-drive automatically when you install VB—you can find it in the Tools\Pss folder on your VB CD.

Why don't you just create a regular installation program?

One thing I don't like about VB 4 is the amount of extra baggage that must be included with the installation of even the simplest of programs—this translates to long download times from online services.

Seeing as VBMaxEM.dll is meant for programmers who already have all the necessary software installed on their computers, I decided that manually installing the DLL was a fair tradeoff to racking up your online connect time.

I don't live in the USA and I'm not a CompuServe member. How can I register your software?

You can register using Visa, MasterCard or American Express. See Credit Card Registration for more information.

Technical Support

Technical support is available to both registered and non-registered users via ***e-mail*** only.

Please send questions, comments, criticisms, etc. to Mike Stanley at **74632.2227@compuserve.com**.

Registration

This is a shareware utility which you can register for US $10 per copy.

You can register using any of the following methods:

- CompuServe's Shareware Registration facility. For more information, sign on to CompuServe and **GO SWREG**. The SWReg ID is **12553**.
- Sending payment to:

  Mike Stanley
  25 Lansdowne Street
  Manchester
  NH 03103
  USA
- You can use your Visa, MasterCard or American Express credit cards.

In return, I will give you an encryption key which will unlock the source code from the file 'VBMaxEM Source Code.grk'—***don't forget to include your Internet e-mail address***. Better still, also send me an e-mail to let me know that your registration is on the way.

Any future updates to the software will use the same encryption key, so you can always get them at no additional cost.

If you don't have an e-mail address, you can receive the registration ID and encryption key via snail-mail by sending me a postage-paid, self-addressed envelope.

Gronked Files

These are encrypted files, created using the **VBMax Gronk Meister**, that contain one or more other files. In this case, the file 'VBMaxEM Source Code.grk' contains the complete, commented VB 4.0 source code for VBMaxEM.dll.

The source files may be extracted using the included **VBMax Degronker** utility. Before you can do that, however, you need to know the encryption key. This key will be given to you when you register VBMaxEM.dll.

Credit Card Registration

There are two methods of registering by credit card:

1) On-line registration. Visit my web site **http://ourworld.compuserve.com/homepages/vbmax** and click on the **Registration** link for more information.

OR

2) Cut and paste the following text into an e-mail message, fill in the blanks and send it to Mike Stanley at **74632.2227@compuserve.com**:

Name:

Company:

Address:

Address:

Town/City:

State/Province:

Postal Code:

Country:

E-mail:

Credit Card Type:

[ ] Visa
[ ] MasterCard
[ ] American Express

Credit Card Number:

Credit Card Name:

Expiration Date (Month/Year):

[ ] VBMax 3D Effects	$10.00
[ ] VBMax Liquid Crystal Display	$10.00
[ ] VBMax Electronic Message Display	$10.00
[ ] VBMax Message Box Wizard	$10.00

Total: $

Constants

Properties Methods

Const gnEFFECT_SCROLL = 0
Const gnEFFECT_STATIC = 1

Const gnEFFECT_SLIDE_FROM_TOP_LEFT = 2
Const gnEFFECT_SLIDE_FROM_TOP_CENTER = 3
Const gnEFFECT_SLIDE_FROM_TOP_RIGHT = 4
Const gnEFFECT_SLIDE_FROM_RIGHT = 5
Const gnEFFECT_SLIDE_FROM_BOTTOM_RIGHT = 6
Const gnEFFECT_SLIDE_FROM_BOTTOM_CENTER = 7
Const gnEFFECT_SLIDE_FROM_BOTTOM_LEFT = 8
Const gnEFFECT_SLIDE_FROM_LEFT = 9

Const gnEFFECT_SLIDE_TO_TOP_LEFT = 10
Const gnEFFECT_SLIDE_TO_TOP_CENTER = 11
Const gnEFFECT_SLIDE_TO_TOP_RIGHT = 12
Const gnEFFECT_SLIDE_TO_RIGHT = 13
Const gnEFFECT_SLIDE_TO_BOTTOM_RIGHT = 14
Const gnEFFECT_SLIDE_TO_BOTTOM_CENTER = 15
Const gnEFFECT_SLIDE_TO_BOTTOM_LEFT = 16
Const gnEFFECT_SLIDE_TO_LEFT = 17

Const gnEFFECT_BUILD_FROM_TOP = 18
Const gnEFFECT_BUILD_FROM_RIGHT = 19
Const gnEFFECT_BUILD_FROM_BOTTOM = 20
Const gnEFFECT_BUILD_FROM_LEFT = 21
Const gnEFFECT_BUILD_FROM_TOP_AND_BOTTOM = 22
Const gnEFFECT_BUILD_FROM_LEFT_AND_RIGHT = 23
Const gnEFFECT_BUILD_TO_TOP_AND_BOTTOM = 24
Const gnEFFECT_BUILD_TO_LEFT_AND_RIGHT = 25

Const gnEFFECT_WIPE_FROM_TOP = 26
Const gnEFFECT_WIPE_FROM_RIGHT = 27
Const gnEFFECT_WIPE_FROM_BOTTOM = 28
Const gnEFFECT_WIPE_FROM_LEFT = 29
Const gnEFFECT_WIPE_FROM_TOP_AND_BOTTOM = 30
Const gnEFFECT_WIPE_FROM_LEFT_AND_RIGHT = 31
Const gnEFFECT_WIPE_TO_TOP_AND_BOTTOM = 32
Const gnEFFECT_WIPE_TO_LEFT_AND_RIGHT = 33

Const gnEFFECT_FLASH = 34

Const gnONE_SEGMENT = 0
Const gnTWO_SEGMENTS = 1
Const gnTHREE_SEGMENTS = 2
Const gnENTIRE_CHARACTER = 3

Methods Summary

Constants Properties

Cls

Pause

Restart

SelectBackColor

SelectForeColor

ShutDown

Cls Method

Example See Also

Summary: Clears the contents of the Electronic Message Display.

Syntax: *object*.**Cls**

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.

Cls Method Example

This example clears the contents of a scrolling electronic message display when you click a button. To see how it works, add a PictureBox and CommandButton to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Caption = "How now, brown cow?     "
    End With

End Sub

Private Sub Command1_Click()
    moEDisp.Cls     'Clears the caption
End Sub
```

See Also

Caption

Pause Method

Example See Also

Summary: Suspends the scrolling of the message.

Syntax: *object*.**Pause** [*interval*]

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
interval	Optional. The number of seconds to suspend scrolling.

Remarks: If the interval parameter is present, scrolling will stop for the specified number of seconds and then restart. If the interval parameter is not present, the scrolling will stop indefinitely. Use the Restart method to resume scrolling.

Pause Method Example

This example creates a scrolling message and pauses the scrolling for five seconds when a button is pressed. To see how it works, add a PictureBox and CommandButton to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Caption = "Click button to pause scrolling 5 seconds.     "
        .Effect = gnEFFECT_SCROLL
    End With

    Command1.Caption = "Pause"

End Sub

Private Sub Command1_Click()
    moEDisp.Pause 5        'Pause scrolling for five seconds
End Sub
```

See Also

Restart

Restart Method

Example See Also

Summary: Restarts the message scrolling after it has been suspended with the Pause method.

Syntax: *object*.**Restart**

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.

Restart Method Example

This example creates a scrolling message and pauses the scrolling indefinitely when a button is pressed. Pressing another button restarts the scrolling. To see how it works, add a PictureBox and two CommandButtons to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Effect = gnEFFECT_SCROLL
        .Caption = "Scrollin' scrollin' scrollin', keep them pixels rollin'…     "
    End With

    Command1.Caption = "Pause"
    Command2.Caption = "Restart"
End Sub

Private Sub Command1_Click()
    moEDisp.Pause                   'Pause scrolling
End Sub

Private Sub Command2_Click()
    moEDisp.Restart                 'Restart scrolling
End Sub
```

See Also

Pause

SelectBackColor Method

Example See Also

Summary: Opens up a common dialog color selection window and allows you to select a color for the background of the message.

Syntax: *object*.**SelectBackColor**

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.

SelectBackColor Method Example

This example allows the user to change the background color of an electronic message. To see how it works, add a PictureBox and CommandButton control to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Caption = "Father's car's a Jaguar     "
    End With

End Sub

Private Sub Command1_Click()
    moEDisp.SelectBackColor      'Displays the color selection dialog and changes the background color
End Sub
```

See Also

BackColor
ForeColor
SelectForeColor

SelectForeColor Method

Example See Also

Summary: Opens up a common dialog color selection window and allows you to select a color for the message text.

Syntax: *object*.**SelectForeColor**

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.

SelectForeColor Method Example

This example allows the user to change the foreground color of an electronic message. To see how it works, add a PictureBox and CommandButton to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Caption = "Peas pudding hot     "
    End With

End Sub

Private Sub Command1_Click()
    moEDisp.SelectForeColor        'Displays the color selection dialog and changes the foreground color
End Sub
```

See Also

ForeColor
BackColor
SelectBackColor

ShutDown Method

Example

Summary: Terminates an Electronic Message Display object after performing all the necessary clean-up routines.

Syntax: *object*.**ShutDown**

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.

Remarks: It is important that this method is called to ensure that the object terminates correctly and releases the resources it is using. Simply setting the object's reference to Nothing does not guarantee that the object will be destroyed.

ShutDown Method Example

This example shows the correct method for terminating an Electronic Message object. To see how it works, add a PictureBox and CommandButton to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Caption = "Don't forget to use the ShutDown method     "
    End With

End Sub

Private Sub Form_Unload(Cancel As Integer)
    moEDisp.Shutdown            'These two lines are
    Set moEDisp = Nothing       'very important
    Set Form1 = Nothing         'This is good too
End Sub
```

Properties Summary

Constants Methods

BackColor

Caption

Container

Effect

FlashCount

ForeColor

Interval

NotifyDelay

NotifyWhenFinished

ScrollAmount

ShowGrid

BackColor Property

Example See Also

Summary: Sets or returns the background color of the Electronic Display.

Syntax: *object*.**BackColor** [=*color*]

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
color	Required. A value or constant representing the background color. The default is black.

BackColor Property Example

This example shows how to set the foreground and background colors of an Electronic Display control. To see how it works, add a PictureBox to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .BackColor = vbBlue         'Sets the background color to blue
        .ForeColor = vbYellow       'Sets the message text color to yellow
        .Caption = "Wascally wabbit     "
    End With

End Sub
```

See Also

ForeColor
SelectForeColor
SelectBackColor

Caption Property

Example See Also

Summary: Sets or returns a string value that determines the information displayed by the Electronic Display control.

Syntax: *object*.**Caption** [=*value*]

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
value	Required. The string to display.

Caption Property Example

This example shows how to set the contents of the Electronic Display control. To see how it works, add a PictureBox to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Caption = "No time to lose…     "   'The message to be displayed
    End With

End Sub
```

See Also

Cls

Container Property

Example

Summary: Sets the PictureBox control that is being used for the Electronic Display control.

Syntax: *object*.**Container** = *control*

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
control	Required. The name of the PictureBox control.

Container Property Example

This example shows how to assign the PictureBox to be used as the Electronic Display control. To see how it works, add a PictureBox to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1   'The container to be used as a control
        .Caption = "Visual Basic to the Max ~ "
    End With

End Sub
```

Effect Property

Example See Also

Summary: Specifies the special effect used to display or erase the message in the Electronic Display control.

Syntax: *object*.**Effect** = *value*

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
value	Required. A number from the list below identifying the effect to be used.

Effects:

You can try out each one of these effects using the demonstration program.

Use these constants to choose between a scrolling or static message.

- gnEFFECT_SCROLL
- gnEFFECT_STATIC

These constants display a message by slowly sliding it on to the display area from one of eight possible directions.

- gnEFFECT_SLIDE_FROM_TOP_LEFT
- gnEFFECT_SLIDE_FROM_TOP_CENTER
- gnEFFECT_SLIDE_FROM_TOP_RIGHT
- gnEFFECT_SLIDE_FROM_RIGHT
- gnEFFECT_SLIDE_FROM_BOTTOM_RIGHT
- gnEFFECT_SLIDE_FROM_BOTTOM_CENTER
- gnEFFECT_SLIDE_FROM_BOTTOM_LEFT
- gnEFFECT_SLIDE_FROM_LEFT

These constants remove a message from the display area by sliding it off to one of different directions.

- gnEFFECT_SLIDE_TO_TOP_LEFT
- gnEFFECT_SLIDE_TO_TOP_CENTER
- gnEFFECT_SLIDE_TO_TOP_RIGHT
- gnEFFECT_SLIDE_TO_RIGHT
- gnEFFECT_SLIDE_TO_BOTTOM_RIGHT
- gnEFFECT_SLIDE_TO_BOTTOM_CENTER
- gnEFFECT_SLIDE_TO_BOTTOM_LEFT
- gnEFFECT_SLIDE_TO_LEFT

These constants make the message appear gradually.

- gnEFFECT_BUILD_FROM_TOP
- gnEFFECT_BUILD_FROM_RIGHT
- gnEFFECT_BUILD_FROM_BOTTOM
- gnEFFECT_BUILD_FROM_LEFT
- gnEFFECT_BUILD_FROM_TOP_AND_BOTTOM
- gnEFFECT_BUILD_FROM_LEFT_AND_RIGHT
- gnEFFECT_BUILD_TO_TOP_AND_BOTTOM
- gnEFFECT_BUILD_TO_LEFT_AND_RIGHT

These erase the message gradually.

- gnEFFECT_WIPE_FROM_TOP
- gnEFFECT_WIPE_FROM_RIGHT
- gnEFFECT_WIPE_FROM_BOTTOM
- gnEFFECT_WIPE_FROM_LEFT
- gnEFFECT_WIPE_FROM_TOP_AND_BOTTOM
- gnEFFECT_WIPE_FROM_LEFT_AND_RIGHT
- gnEFFECT_WIPE_TO_TOP_AND_BOTTOM
- gnEFFECT_WIPE_TO_LEFT_AND_RIGHT

This causes the message to flash on and off

- gnEFFECT_FLASH

Effect Property Example

This example shows many of the special effects in action. To see how it works, add a PictureBox and a CommandButton to a form and copy and paste this code into the form's code area.

```
Const mcCAPTION = "VBMax"

Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        .Effect = gnEFFECT_STATIC        'Set the initial effect
        Set .Container = Picture1
    End With

    Command1.Caption = "Effect"
End Sub

Private Sub Form_Unload(Cancel As Integer)
    moEDisp.Shutdown
    Set moEDisp = Nothing
    Set Form1 = Nothing
End Sub

Private Sub Command1_Click()
Static nEffect As Integer
Dim lEndTime As Single

nEffect = nEffect + 1                    'Change the effect

    If nEffect > gnEFFECT_FLASH then nEffect = gnEFFECT_STATIC

With moEDisp
Select Case nEffect
    Case gnEFFECT_SLIDE_TO_TOP_LEFT To gnEFFECT_SLIDE_TO_LEFT,
gnEFFECT_WIPE_FROM_TOP To gnEFFECT_WIPE_TO_LEFT_AND_RIGHT
        .Effect = gnEFFECT_STATIC
        .Caption = mcCAPTION      'Display the message
        lEndTime = Timer() + 0.5
        Do Until Timer() >= lEndTime      'Pause for brain to register it
            DoEvents
        Loop
        .Effect = nEffect       'Remove it using special effect
    Case Else
        .Cls
        .Effect = nEffect
        .Caption = mcCAPTION
End Select
End With

End Sub
```

See Also

Pause
Restart
FlashCount
NotifyWhenFinished
ScrollAmount

FlashCount Property

Example See Also

Summary: Sets or returns a value that determines the number of times a message will flash when using the flashing message special effect.

Syntax: *object*.**FlashCount** [=*value*]

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
value	Required. A number specifying the number of times to flash.

FlashCount Property Example

This example flashes the message five times when you click a button. To see how it works, add a PictureBox and a CommandButton to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        .Effect = gnEFFECT_STATIC
        Set .Container = Picture1
        .Caption = "Visual Basic to the Max"
    End With

    Command1.Caption = "Flash"

End Sub

Private Sub Form_Unload(Cancel As Integer)
    moEDisp.Shutdown
    Set moEDisp = Nothing
    Set Form1 = Nothing
End Sub

Sub Command1_Click()
    With moEDisp
        .FlashCount = 5             'Flash message five times
        .Effect = gnEFFECT_FLASH
        .Caption = "Visual Basic to the Max"
    End With
End Sub
```

See Also

Effect

ForeColor Property

Example See Also

Summary: Sets or returns a value that determines the color of the message text characters.

Syntax: *object*.**ForeColor** [=*color*]

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
color	Required. A value or constant representing the color of the text. The default is white.

ForeColor Property Example

This example shows how to set the foreground and background colors of an electronic message. To see how it works, add a PictureBox to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .BackColor = vbBlue         'Sets the background color to blue
        .ForeColor = vbYellow       'Sets the text color to yellow
        .Caption = "Less-filling--works great!     "
    End With

End Sub
```

See Also

BackColor
SelectBackColor
SelectForeColor

Interval Property

Example See Also

Summary: Sets or returns a value that determines the number of milliseconds between control updates when using special effects.

Syntax: *object*.**Interval** [=*value*]

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
value	Required. A value specifying the number of milliseconds between control updates.

Interval Property Example

This example creates a scrolling electronic message display. The interval property determines how fast the message scrolls. To see how it works, add a PictureBox to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Effect = gnEFFECT_SCROLL       'Specify scrolling effect
        .Interval = 20                  'Scroll speed
        .Caption = " VBMax: Visual Basic to the Max"
    End With

End Sub

Private Sub Form_Unload(Cancel As Integer)
    moEDisp.Shutdown
    Set moEDisp = Nothing
    Set Form1 = Nothing
End Sub
```

See Also

Effect
ScrollAmount

NotifyDelay Property

Example See Also

Summary: Sets or returns a value specifying the number of seconds to wait before issuing a callback after a special effect has finished.

Syntax: *object*.**NotifyDelay** [=*value*]

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
value	Required. A value specifying the number of seconds to wait before invoking the callback routine.

NotifyDelay Property Example

This example creates a scrolling electronic message and displays a message box when the message reaches the end. The NotifyDelay property specifies how long to wait before displaying the message box. To see how it works, add a PictureBox to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Effect = gnEFFECT_SCROLL
        .NotifyWhenFinished = True          'Activate callbacks
        .NotifyDelay = 5                    'Wait five seconds before issuing callback
        .Caption = "VBMax: Visual Basic to the Max"
    End With

End Sub

Private Sub Form_Unload(Cancel As Integer)
    moEDisp.Shutdown
    Set moEDisp = Nothing
    Set Form1 = Nothing
End Sub

Public Sub VBMaxElectronicDisplay_Finished(roEDisp As CElectronicDisplay)
    MsgBox "Message finished scrolling five seconds ago."
End Sub
```

See Also

NotifyWhenFinished

NotifyWhenFinished Property

Example See Also

Summary: Sets or returns a value that specifies whether or not the callback mechanism is switched on.

Syntax: *object*.**NotifyWhenFinished** [=*boolean*]

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
boolean	Required. An expression that evaluates to True or False.

Remarks: To use callbacks, set NotifyWhenFinished to True and add this routine to your form: *Public Sub VBMaxElectronicDisplay_Finished(roEDisp As CElectronicDisplay)*. When a special effect has finished, VBMaxEM.dll will make a call to that routine. In the routine, you add the code to be executed at the end of the effect.

NotifyWhenFinished Property Example

This example creates a scrolling electronic message and displays a message box when the message reaches the end. To see how it works, add a PictureBox to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Effect = gnEFFECT_SCROLL
        .NotifyWhenFinished = True          'Activate callbacks
        .NotifyDelay = 0
        .Caption = "VBMax: Visual Basic to the Max"
End With

End Sub

Private Sub Form_Unload(Cancel As Integer)
    moEDisp.Shutdown
    Set moEDisp = Nothing
    Set Form1 = Nothing
End Sub

Public Sub VBMaxElectronicDisplay_Finished(roEDisp As CElectronicDisplay)
    MsgBox "Message has finished scrolling."
End Sub
```

See Also

NotifyDelay

ScrollAmount Property

Example See Also

Summary: Sets or returns a value specifying the amount the letters move when scrolling.

Syntax: *object*.**ScrollAmount** [=*value*]

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
value	Required. One of the following values: gnONE_SEGMENT gnTWO_SEGMENTS gnTHREE_SEGMENTS gnENTIRE_CHARACTER

ScrollAmount Property Example

This example creates a scrolling electronic message. The ScrollAmount property specifies how much of each character is to move at one time. To see how it works, add a PictureBox to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Effect = gnEFFECT_SCROLL
        .ScrollAmount = gnONE_SEGMENT     'Scroll the message one segment's width at a time
        .Caption = "VBMax: Visual Basic to the Max     ~~~     "
    End With
End Sub

Private Sub Form_Unload(Cancel As Integer)
    moEDisp.Shutdown
    Set moEDisp = Nothing
    Set Form1 = Nothing
End Sub
```

See Also

Effect
Interval

ShowGrid Property

Example

Summary: Sets or returns a value that determines whether unlit cells of the letters are displayed.

Syntax: *object*.**ShowGrid** [=*boolean*]

Element	Description
object	Required. An object expression that evaluates to a VBMax CElectronicDisplay object.
boolean	Required. An expression that evaluates to True or False. The default is False.

Remarks: Showing the unlit cells works better with some color combinations than others. You will have to experiment to see which combinations work for you.

ShowGrid Property Example

This example lets you switch the display between visible unlit cells and invisible ones by clicking a button. To see how it works, add a PictureBox and CommandButton to a form and copy and paste this code into the form's code area.

```
Dim moEDisp As New CElectronicDisplay

Private Sub Form_Load()

    With moEDisp
        Set .Container = Picture1
        .Effect = gnEFFECT_STATIC
        .ShowGrid = True            'Show the unlit segments
        .Caption = "VBMax"
    End With

End Sub

Private Sub Command1_Click()
    with moEDisp
        . ShowGrid = Not . ShowGrid         'Toggle the unlit segments on and off
        .Caption = "VBMax"
    End With
End Sub
```

Notes on Color Selection

There are no built-in restrictions about which colors you can use for the message text or background.

Realistically, however, you have to exercise a little caution because of potential differences between the color settings of your system and those of the computers on which your application may run.

If your system is running with a color mode greater than 256 colors, the colors you choose may look a lot different on systems running in 256 colors. You could end up with dithered colors which really don't look that great with electronic message displays.

A safe approach may be to select solid colors only while you have your system running in 256 color mode.

VBMax Message Box Wizard

With all those options and new constants, do you ever forget the exact syntax for creating a message box? Me too. There are plenty of utilities on the market to help you create message boxes, both commercial and shareware. But have you tried any of them?

I did, and was so disappointed by what I found that I wrote my own. I called it **VBMax Message Box Wizard**. It is a VB 4.0 add-in that greatly simplifies the process of coding message boxes.

When you reach a point in your code where you want to display a message box, select the VBMax Message Box Wizard from the VB Add-Ins menu and type in your message. Click a control or two and you're done—the generated code is written into your source file.

There are several configuration options and shortcuts you can use to tailor the utility to your own style of working. If they are not enough, you also get the fully-commented source code which you can tweak any way you want. The code is up to date with new VB 4.0 features such as class modules, predefined constants and use of the registry.

As with all VBMax products, the Message Box Wizard is written entirely in VB and uses no third-party controls. With that in mind, check out the icon selection bar in Step 1 and the spin button on the Options dialog—they work with the keyboard too and adapt themselves to changes in the Windows color scheme.

Price: $10

VBMax 3D Effects

VBMax 3D Effects DLL

Windows 95 gave us a snazzy, new and improved three-dimensional user interface. VB 4.0 adds the 3D look to our forms automatically. That's a great feature—as far as it goes. Unfortunately, some 3D effects are still difficult to achieve without using a third-party control.

Have no fear—VBMax is here. VBMax3D.dll is an in-process OLE server that lets you add the 3D effects that Microsoft forgot to your VB 4 applications—without the overhead of third-party controls.

VBMax3D.dll contains methods and properties for creating:

- Panels
- Borders
- Frames
- Lines
- Drop shadows
- Status bars
- Progress meters
- A variety of text effects

Wait, there's more! With the release of Windows 95, 3D doesn't just mean shades of gray anymore. VBMax3D.dll understands this and uses the Windows 95 color scheme settings to create the 3D effects.

VBMax3D.dll is shareware. You are required to register the software if you use it in any of your applications. When you register, you will receive the complete, commented source code which is written entirely in VB 4.0.

Price: $10

VBMax Liquid Crystal Display

Want to add LCD/LED style controls to your applications? Now you can with VBMaxLCD.dll.

VBMaxLCD.dll is an in-process OLE server for adding LCD/LED style controls to your VB 4.0 applications. There are tons of uses for these babies: clocks, timers, meters, calculators, dialers—the list goes on.

VBMaxLCD.dll contains methods and properties for:

- Setting the digit and background colors
- Autosizing the display area
- Blinking colons
- Flashing digits
- Showing or hiding unlit segments
- Aligning digits left, right or centered

Price: $10

Murphy 96

It was Murphy who first observed that if anything can possibly go wrong, it will go wrong.

Deceptive in its simplicity, this profound insight marked a turning point in our understanding of why things happen the way they do. Indelibly etching itself into the human psyche, this revelation ensured that never again would we look at the world in quite the same way.

Not content to rest on his laurels, Murphy went on to expand on his theory and formulate the now famous laws that bear his name. Truly one of the great thinkers of our time, Murphy somehow managed to unravel the very fabric of the cosmos itself and lay bare the fundamental perversity with which it is woven.

“Mother Nature is a bitch.”, he said.

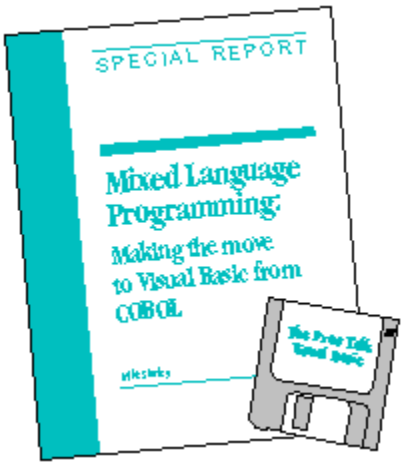
It was a defining moment in history and Murphy’s accomplishments provided the foundation for a host of others who would follow in his giant footsteps. There will only ever be one Murphy but his successors have, nonetheless, made significant contributions to his work.

Murphy 96 is the embodiment of the collective consciousness of these intellects—a compilation of hundreds of laws, corollaries, axioms, rules, maxims and other truisms. Place Murphy 96 in your StartUp folder and it will present you with a different pearl of wisdom from Murphy and his cohorts every time you start Windows.

Murphy 96 is offered free, gratis and for nothing to all those who seek true enlightenment about that which we call reality.

Making the Move to Visual Basic 4 from COBOL

Message to COBOL Programmers



From Pinnacle Publishing, Inc.

This isn't just any old Visual Basic book. This one is written especially for you, the COBOL programmer, as you emerge into the strange, new world of personal computers, or PCs, about which you may know little. Whether you like it or not, the programming world as you know it is changing—and changing fast.

This book introduces you to programming Windows using the Visual Basic programming language, and, at the same time, explains why making the transition to VB is a good move and how to overcome the inevitable culture shock. Using COBOL as a point of reference, I'll show you a different way of doing things, highlighting the differences and similarities between the two languages. As a COBOL programmer, you already know how to program, you just need to become familiar with a new environment and new tools. Wherever possible, I have tried to convey Windows and Visual Basic programming concepts in terms familiar to your COBOL experience.

This book wasn't written by an academic perched in a remote ivory tower, totally divorced from the real world with no concept about what it takes to program computers for a living. I am a working programmer with many years of COBOL programming experience. Now, as an independent consultant specializing in Visual Basic, I not only continue to find gainful employment but am enjoying the change enormously.

Although learning Visual Basic was a little strange at first, it was not hard work. On the contrary, it was fun. Visual Basic programming *is* fun, and if you can get paid for doing it, so much the better. If you enjoy programming, you'll love Visual Basic. It offers many more possibilities and opportunities than you will ever find with COBOL.

If I can make the transition, so can you.

Services

Software Development

I provide Visual Basic software development services in the Southern New Hampshire/Northern Massachusetts area and also areas further afield for those willing to let me telecommute.

The VBMax Electronic Message Display OLE server which this help file accompanies is an example of my work. For more examples, download some of the other software described in this help file. See VBMax 3D Effects, VBMax Liquid Crystal Display and VBMax Message Box Wizard.

Help File Authoring

Most software developers won't touch help files, or any other kind of documentation for that matter, with a ten foot barge-pole. I don't mind at all—it's all software to me. I charge the same rate for help file development as I do for software development.

This help file is an example of my work.

About the Mike Stanley

Although now specializing in VB software development, I am no newbie to the business. I have extensive experience working on mainframes, minis and PCs. I have a strong database background and have designed and developed many business applications using hierarchical (IMS), network (IDMS) and relational (DB2, SQL Server) databases.

I have been programming in Visual Basic ever since version 1.0 was released and have written articles for *Visual Basic Programmer's Journal* and *VB Tech Journal*. I also authored the book Making the Move to Visual Basic 4 from COBOL from Pinnacle Publishing, Inc.

My client-server experience is with VB and MS SQL Server, currently using Remote Data Objects (RDO). I also speak COBOL, CICS and DB2.

E-mail **Mike Stanley** at **74632.2227@compuserve.com**.

VBMax

Visual Basic to the Max