# Read Me.1st

Welcome to VBMax 3D Effects DLL. Now you can add the 3D effects that Microsoft forgot to your VB 4.0 applications—without the overhead of third-party controls. Less filling—works great.

I know, 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 this puppy can do. Before you can do that, however, there is a little matter of the Windows registry to take care of first.

VBMax3D.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 to VBMax 3D Effects

## VBMax 3D Effects DLL



### Overview

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**

# What's New in Version 1.3a

The enhancements listed below are a direct result of the feedback I received from users of the software. This feedback is essential in helping me to deliver a better product and to give you the features you need—so don't stop telling me what you think.

As promised, registered users get free upgrades. Use the same password as before to extract the source code from the gronk file.

## *Version 1.3*

**Containers**

Previous versions of VBMax3D.dll drew the 3D effects only on the form. By far, the most requested enhancement was the ability to draw the 3D effects inside other containers.

Ask and you shall receive. The new Container property allows you specify a PictureBox control as the container to be used instead of the form. This gives you much more flexibility in the use of the 3D effects.

To give you some ideas, I have updated the demo program with an example of how containers may be used.

**Online Help**

"Give us better documentation!", you cried. Tough crowd.

Still, it shows that **real programmers** *do* read the documentation. So here it is—this online help file is the new and improved documentation. Everything you ever wanted to know is now just a few mouse clicks away—information, as they say, at your fingertips.

I also hooked-up the help file to the Object Browser. When you are in the Object Browser, select *VBMax 3D Effects* from the Libraries/Projects pull-down list. When you click on a method or property, 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 bring up the online help for the selected item.

How am I doing?

**FillColor, ForeColor and Font Dialogs**

You were always able to control the FillColor, ForeColor and Font properties through code. Now, with the SelectFillColor, SelectForeColor and SelectFont methods, you can easily extend that capability to your users to allow them to customize their application settings.

**Raised Frames**

The Frame method has been extended to include a *frame style* parameter which allows you to specify either raised or recessed frames. This parameter is optional and defaults to the recessed style so your existing code will continue to work without you having to make any changes.

**C3D**

You *will* have to make a change for this one, however. I changed the class name from cls3D to C3D to be more in keeping with the naming conventions used by others. I apologize for any inconvenience this may cause.

**Read Properties**

All properties can now be read as well as written.

**SWReg**

I received numerous requests from folks wishing to register the DLL via CompuServe's online shareware registration program.

Give the customers what they want, that's what I say. This software now resides in the Shareware Registration Database under the title VBMax 3D Effects for VB 4.0. The **Registration ID** is **10948**.

For more information, sign on to CompuServe and **GO SWREG**.

## *Version 1.3a*

**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 r**egsvr32.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 VBMAX3D.DLL**

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

**REGSVR32 /U VBMAX3D.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 3D Effects 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 3D Effects 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 C3D class.

### Packing List

| | |
|---|---|
| VBMax3D.hlp | This online help file. |
| VBMax3D.cnt | Help file contents. |
| 3Ddemo.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). |
| VBMax3D 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 VBMax3D.dll in VB, as described in Installation Instructions, you will have access to a class called **C3D**. 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 3D Effects*** 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 can create one or more objects from this class module. How many you choose depends on how many effects you want and whether or not they share the same attributes. For example, if you set the style property to gnRAISED, all effects that use this property will use the same style.

If you want to use different styles you have two options:

1. Change the style property each time you use it.
2. Create different objects with different styles.

It's your call. Multiple objects all use the same code but keep their own copy of the variables.

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

**Dim mo3D As New C3D**

You create 3D effects by drawing regular 2D controls on a form or in a picture box and then passing them to methods in the object created by C3D.

These methods then create the desired 3D effect at the location of the 2D control. The 2D controls act as place-holders for the resulting 3D effect and allow you to see what you are doing as you design your form.

You can get a good idea of how the DLL works by examining the source code for the demo program. More detailed information is contained within the DLL source code.

# 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 VBMax3D.dll with Windows. Please refer to the Installation Instructions.

**I have searched my hard-drive but cannot find regsvr32.exe—how can I install VBMax3D.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 VBMax3D.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 **10948**.
- 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 'VBMax3D 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 'VBMax3D Source Code.grk' contains the complete, commented VB 4.0 source code for VBMax3D.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 VBMax3D.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**: $

## Methods

Constants Properties

**Border**

**Caption**

**DropShadow**

**FormBorder**

**Frame**

**Line3D**

**Panel**

**PercentMeter**

**SelectFillColor**

**SelectFont**

**SelectForeColor**

**StatusBar**

**VBMax**

Visual Basic to the Max

# Border Method

See also Example

**Summary:** Draws a border around a control.

**Syntax:** *object*.**Border** *control*

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |
| *control* | Required. Name of the control around which the border is to be drawn. |

**Remarks:** The appearance of the border is determined by the Depth, Width and Style properties which should be set before calling this method.

## Border Method Example

This example uses one C3D object to draw the same type of border around two different controls. To see how it works, add a Label and Image control to a form and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Depth = 1              'Sets the depth of the border to one pixel
        .Width = 2              'Sets the width of the border to two pixels
        .Style = gnRAISED       'Constants defined in 3DDemo.bas
        .Border Label1          'Draws the border around the Label control
        .Border Image1          'Draws the border around the Image control
    End With

Set o3D = Nothing

End Sub
```

## See Also

Depth Property
Style Property
Width Property
FormBorder Method

# Caption Method

See also  Example

**Summary:** Draws text using a variety of 3D effects.

**Syntax:** *object*.**Caption** *control, font option*

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |
| *control* | Required. A Label control used as a place-holder and source for the text. |
| *font option* | Optional. Settings: |
| | gnUSE_CONTROL_FONT Use the font attributes of the Label control. |

**Remarks:** The 3D effects are determined by the Style property which should be set before calling this method.

## Caption Method Example

This example uses one C3D object to draw two raised text captions. One of the captions uses the object's font settings while the other uses the Label control's font settings. To see how it works, add two Label controls to a form and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Style = gnRAISED                   'Constants defined in 3DDemo.bas
        .FontItalic = True                  'Sets object's font properties
        .FontBold = True
        .FontSize = 14
        .Caption Label1                     'Uses object's font settings
        .Caption Label2, gnUSE_CONTROL_FONT         'Uses Label control's font settings
    End With

    Set o3D = Nothing

End Sub
```

## See Also

Font Properties
ForeColor Property
Shadow Property
Style Property

# DropShadow Method

See also Example

**Summary:** Draws a drop-shadow behind a control.

**Syntax:** *object*.**DropShadow** *control*

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |
| *control* | Required. The control that will have the shadow drawn behind it. |

**Remarks:** The appearance of the drop-shadow is determined by the Offset and Shadow properties which should be set before calling this method.

## DropShadow Method Example

This example draws drop-shadows behind two controls. To see how it works, add two TextBox controls to a form and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Offset = 5                 'Sets the shadow offset to five pixels
        .Shadow = gnLIGHT_SHADOW    'Uses the lighter colored shadows
        .DropShadow Text1           'Draws a drop-shadow behind the TextBox control
        .DropShadow Text2           'Draws a drop-shadow behind the TextBox control
    End With

    Set o3D = Nothing

End Sub
```

## See Also

Offset Property
Shadow Property

# FormBorder Method

See also Example

**Summary:** Draws a border around the inside edge of a form.

**Syntax:** *object*.**FormBorder** *form*

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |
| *form* | Required. The form on which to draw the border. |

**Remarks:** The appearance of the border is determined by the Style, Width and Depth properties which should be set before calling this method.

# FormBorder Method Example

This example draws a border around a form. To see how it works, create a form, set the BorderStyle property to 0 (None) and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Style = gnRECESSED     'Sets type of border
        .Depth = 2              'Sets the depth of the border to 2 pixels
        .Width = 10             'Sets the width of the border to 10 pixels
        .FormBorder Me          'Draws the border around the form
    End With

    Set o3D = Nothing

End Sub
```

## See Also

Depth Property
Style Property
Width Property
Border Method

# Frame Method

See also  Example

**Summary:** Draws a frame around a control.

**Syntax:** *object*.**Frame** *control, caption, frame style*

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |
| *control* | Required. Control around which to draw the frame. |
| *caption* | Required. The frame's caption. |
| *frame style* | Optional. Settings: |
| | gnRECESSED Draw a recessed style frame. Default. |
| | gnRAISED Draw a raised style frame. |

**Remarks:** The appearance of the frame is determined by the Style, Alignment, Shadow, Font and ForeColor properties which should be set before calling this method.

## Frame Method Example

This example uses one C3D object to draw two different style frames. One frame is recessed and the caption left-aligned while the other is raised with a right-aligned caption. To see how it works, add two PictureBox controls to a form, set their BorderStyle properties to 0 (None) and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Style = gnRAISED                   'Uses raised text for the frame's caption
        .FontItalic = True                  'Sets object's font properties
        .FontBold = True
        .ForeColor = vbBlue                 'Sets caption text color to blue
        .Frame Picture1, "Alignment"                'Draws a recessed style frame
        .Alignment = gnRIGHT_JUSTIFY                'Right-justifies the caption on the next frame
        .Frame Picture2, "Color", gnRAISED          'Draws a raised style frame
    End With

    Set o3D = Nothing

End Sub
```

## See Also

- Alignment Property
- Font Properties
- ForeColor Property
- Shadow Property
- Style Property

# Line3D Method

See also  Example

**Summary:** Draws a 3D line.

**Syntax:** *object*.**Line** *control*

| Element | Description | |
|---|---|---|
| *object* | Required. | An object expression that evaluates to a VBMax C3D object. |
| *control* | Required. | The Line control used as a place-holder. |

**Remarks:** The appearance of the line is determined by the Style property which should be set before calling this method.

## See Also

Style Property

## Line3D Method Example

This example draws a series of 3D lines. To see how it works, add a control array of 5 horizontal or vertical Line controls to a form and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D
Dim i As Integer

    With o3D
        For I = 0 to 4
            .Line3D Line1(i)        'Draws a 3D line at the location of the Line control
        Next i
    End With

    Set o3D = Nothing

End Sub
```

# Panel Method

See also  Example

**Summary:** Replaces a control with a 3D panel.

**Syntax:** *object*.**Panel** *control*

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |
| *control* | Required. Name of the control being used as a place-holder for the panel. |

**Remarks:** The appearance of the panel is determined by the Depth, Width and Style properties which should be set before calling this method.

## See Also

Depth Property
Style Property
Width Property

## Panel Method Example

This example uses one C3D object to create two panels, one raised and one recessed. To see how it works, add two Label control to a form and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Depth = 1              'Sets the depth of the panel to one pixel
        .Style = gnRAISED       'Creates a raised panel
        .Panel Label1           'Replaces the Label control with a panel
        .Style = gnRECESSED     'Creates a recessed panel
        .Panel Label2           'Replaces the Label control with a panel
    End With

    Set o3D = Nothing

End Sub
```

# PercentMeter Method

See also  Example

**Summary:** Draws a percentage meter using either a solid bar or the Windows 95 style progress meter.

**Syntax:** *object*.**PercentMeter** *control, percent*

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |
| *control* | Required. Name of the Label control used as a place-holder for the percentage meter. |
| *percent* | Required. The percentage to display. |

**Remarks:** The appearance of the percent meter is determined by the ShowPercent, SolidMeter, ForeColor, FillColor and Font properties which should be set before calling this method.

## See Also

FillColor Property
Font Properties
ForeColor Property
ShowPercent Property
SolidMeter Property

## PercentMeter Method Example

This example continually updates a percentage meter. To see how it works, add a Timer and Label control to a form and copy and paste this code into the form's code area. Position the Label control where you would like to see the percentage meter, set its Height property to 14 pixels and set the Timer's Interval property to 1.

```
Dim mo3D As New C3D

Private Sub Form_Load()

    With mo3D
        .SolidMeter = False        'Uses the Windows 95 style progress bar
        .ShowPercent = True        'Specifies that the percentage numbers are to be shown
        .FontBold = True           'Sets the font attribute of the percentage numbers
        .ForeColor = vbWhite       'Sets the color of the percentage numbers
        .FillColor = vbBlue        'Sets the color of the progress bar
    End With

End Sub

Private Sub Timer1_Timer()
Static nPercent As Integer
    mo3D.PercentMeter Label1, nPercent
    nPercent = nPercent + 1
    If nPercent > 100 Then nPercent = 0
End Sub
```

# SelectFillColor Method

See also  Example

**Summary:** Displays a color selection dialog allowing the user to change the color of the percentage meter.

**Syntax:** *object*.**SelectFillColor**

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |

**Remarks:** Use this method to allow the user to customize the percentage meter color. To preset the color, use the FillColor property.

## See Also

FillColor Property
SelectFont Method
SelectForeColor Method

## SelectFillColor Method Example

This example displays a percentage meter and allows the user to change the color of the bar. To see how it works, add a Timer, a Label and a CommandButton control to a form and copy and paste this code into the form's code area. Position the Label control where you would like to see the percentage meter, set its Height property to 14 pixels and set the Timer's Interval property to 1.

```
Dim mo3D As New C3D

Private Sub Form_Load()

    With mo3D
        .SolidMeter = False         'Uses the Windows 95 style progress bar
        .ShowPercent = True         'Specifies that the percentage numbers are to be shown
        .FontBold = True            'Sets the font attribute of the percentage numbers
        .ForeColor = vbWhite        'Sets the color of the percentage numbers
        .FillColor = vbBlue         'Sets the color of the progress bar
    End With

End Sub

Private Sub Timer1_Timer()
Static nPercent As Integer
    mo3D.PercentMeter Label1, nPercent
    nPercent = nPercent + 1
    If nPercent > 100 Then nPercent = 0
End Sub

Private Sub Command1_Click()
    mo3D.SelectFillColor            'Displays the color dialog and changes the color
End Sub
```

# SelectFont Method

See also  Example

**Summary:** Displays a font selection dialog allowing the user to change the font attributes of text controlled by a VBMax 3D object.

**Syntax:** *object*.**SelectFont**

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |

**Remarks:** Use this method to allow the user to customize font attributes. To preset the font attributes, use the Font properties.

## See Also

Font Properties
SelectFillColor Method
SelectForeColor Method

## SelectFont Method Example

This example displays some 3D text and allows the user to change the font attributes. To see how it works, add a Label and CommandButton to a form and copy and paste this code into the form's code area.

```
Dim mo3D As New C3D

Private Sub Form_Load()

    With mo3D
        .Style = gnRAISED       'Uses raised text
        .Caption Label1         'Displays 3D text at the location of Label1
    End With

End Sub

Private Sub Command1_Click()
    With mo3D
        .SelectFont             'Displays the font dialog and changes the font attributes
        Cls                     'Clear previous display
        .Caption Label1         'Redisplays the text
    End With
End Sub
```

# SelectForeColor Method

See also  Example

**Summary:** Displays a color selection dialog allowing the user to change the color of text controlled by a VBMax 3D object.

**Syntax:** *object*.**SelectForeColor**

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |

**Remarks:** Use this method to allow the user to customize text color. To preset the color, use the ForeColor property.

## See Also

ForeColor Property
SelectFillColor Method
SelectFont Method

## SelectForeColor Method Example

This example displays some 3D text and allows the user to change the color of the text. To see how it works, add a Label and CommandButton to a form and copy and paste this code into the form's code area.

```
Dim mo3D As New C3D

Private Sub Form_Load()

    With mo3D
        .Style = gnRAISED       'Uses raised text
        .Caption Label1         'Displays 3D text at the location of Label1
    End With

End Sub

Private Sub Command1_Click()
    With mo3D
        .SelectForeColor        'Displays the color dialog and changes the text color
        .Caption Label1         'Redisplays the text
    End With
End Sub
```

# StatusBar Method

See also  Example

**Summary:** Displays a message in a status bar.

**Syntax:** *object*.**StatusBar** *control, status*

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |
| *control* | Required. Name of the control used as a place-holder for the status bar. |
| *status* | Required. The message to be displayed. |

**Remarks:** The appearance of the text in the status bar is determined by Style, Shadow, ForeColor and Font properties which should be set before calling this method.

## See Also

Font Properties
ForeColor Property
Shadow Property
Style Property

## StatusBar Method Example

This example displays 3D text in a status bar. To see how it works, add a Label control to a form and copy and paste this code into the Form_Load event. Position and size the Label control where you would like to see the status bar.

```
Dim mo3D As New C3D

Private Sub Form_Load()

    With mo3D
        .ForeColor = vbBlue                         'Sets the color of the status bar text
        . StatusBar Label1, "Connecting to network . . ."       'Displays the message in the status bar
    End With

End Sub
```

## Properties

Constants Methods

**Alignment**

**Container**

**Depth**

**FillColor**

**FontBold**

**FontItalic**

**FontName**

**FontSize**

**FontStrikethru**

**FontUnderline**

**ForeColor**

**Offset**

**Shadow**

**ShowPercent**

**SolidMeter**

**Style**

**Width**

# Alignment Property

Example  Applies to

**Summary:** Returns or sets a value that determines the justification of the caption in a frame.

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

| Element | Description | |
|---|---|---|
| *object* | An object expression that evaluates to a VBMax C3D object. | |
| *value* | A numeric expression specifying the alignment of a frame's caption. | |
| | Settings: | |
| | gnLEFT_JUSTIFY Default. | Align the caption to the left of the frame. |
| | gnRIGHT_JUSTIFY | Align the caption to the right of the frame. |
| | gnCENTER | Center the caption in the frame. |

### Alignment Property Applies To

Frame Method

## Alignment Property Example

This example uses one C3D object to draw two different style frames. One frame is recessed and the caption left-aligned while the other is raised with a right-aligned caption. To see how it works, add two PictureBox controls to a form, set their BorderStyle properties to 0 (None) and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Style = gnRAISED                   'Uses raised text for the frame's caption
        .FontItalic = True                  'Sets object's font properties
        .FontBold = True
        .ForeColor = vbBlue                 'Sets caption text color to blue
        .Frame Picture1, "Alignment"                'Draws a recessed style frame
        .Alignment = gnRIGHT_JUSTIFY                'Right-justifies the caption on the next frame
        .Frame Picture2, "Color", gnRAISED          'Draws a raised style frame
    End With

    Set o3D = Nothing

End Sub
```

# Container Property

Example Applies To

**Summary:** Sets or returns a control that is used for drawing the 3D effects. If the Container property is not set, the default behavior of C3D objects is to draw the 3D effects on the form.

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

| Element | Description |
|---|---|
| *object* | An object expression that evaluates to a VBMax C3D object. |
| *control* | Name of the control to be used as a container. |

**Remarks:** Currently, only the PictureBox control can be used as a container.

### Container Property Applies To

Entire object

## Container Property Example

This example draws 3D captions inside a PictureBox container. To see how it works, add a PictureBox control to a form and add two Label controls to the PictureBox. Then copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        Set .Container = Picture1          'Draws the 3D effects on the PictureBox
        .Style = gnRAISED                  'Specifies raised text
        .Caption Label1, gnUSE_CONTROL_FONT        'Draws 3D text at the position of the Label control
        .Caption Label2, gnUSE_CONTROL_FONT        'Draws 3D text at the position of the Label control
    End With

    Set o3D = Nothing

End Sub
```

# Depth Property

See also Example Applies To

**Summary:** Sets or returns a value that determines the depth of panels and borders.

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

| Element | Description |
|---|---|
| *object* | An object expression that evaluates to a VBMax C3D object. |
| *value* | A numeric expression specifying the depth of a panel or border in pixels. |

## See Also

Width Property

### Depth Property Applies To

- Border Method
- FormBorder Method
- Panel Method
- StatusBar Method

## Depth Property Example

This example draws a border around a control. To see how it works, add a Label control to a form and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Depth = 1              'Sets the depth of the border to one pixel
        .Width = 2              'Sets the width of the border to two pixels
        .Style = gnRAISED       'Constants defined in 3DDemo.bas
        .Border Label1          'Draws the border around the Label control
    End With

    Set o3D = Nothing

End Sub
```

# FillColor Property

See also Example Applies To

**Summary:** Sets or returns the color of the bar in a percentage meter.

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

| Element | Description |
|---|---|
| *object* | An object expression that evaluates to a VBMax C3D object. |
| *value* | A value or constant that determines the fill color. |

## See Also

ForeColor Property

### FillColor Property Applies To

PercentMeter Method

# FillColor Property Example

This example continually updates a percentage meter. To see how it works, add a Timer and Label control to a form and copy and paste this code into the form's code area. Position the Label control where you would like to see the percentage meter, set its Height property to 14 pixels and set the Timer's Interval property to 1.

```
Dim mo3D As New C3D

Private Sub Form_Load()

    With mo3D
        .ForeColor = vbWhite        'Sets the color of the percentage numbers
        .FillColor = vbBlue         'Sets the color of the progress bar
    End With

End Sub

Private Sub Timer1_Timer()
Static nPercent As Integer
    mo3D.PercentMeter Label1, nPercent
    nPercent = nPercent + 1
    If nPercent > 100 Then nPercent = 0
End Sub
```

# FontBold, FontItalic, FontStrikethru, FontUnderline FontSize and FontName Properties

See also Example Applies To

**Summary:** Sets or returns font attributes.

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

*object*.**FontItalic** [=*boolean*]

*object*.**FontStrikethru** [=*boolean*]

*object*.**FontUnderline** [=*boolean*]

*object*.**FontSize** [=*points*]

*object*.**FontName** [=*font*]

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |
| *boolean* | A True or False value indicating whether or not the font attribute is being used. |
| *points* | A numeric expression specifying the font size to use, in points. |
| *font* | A string expression specifying the font name to use. |

## See Also

ForeColor Property

### Font Properties Apply To

Caption Method
Frame Method
PercentMeter Method
StatusBar Method

## FontBold, FontItalic, FontStrikethru, FontUnderline FontSize and FontName Properties Example

This example draws a raised text caption. To see how it works, add a Label control to a form and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Style = gnRAISED          'Draws raised text
        .FontName = "Times New Roman" 'Sets object's font properties
        .FontSize = 14
        .FontItalic = True
        .FontBold = True
        .FontStrikethru = False
        .FontUnderline = False
        .Caption Label1            'Uses object's font settings
    End With

    Set o3D = Nothing

End Sub
```

# ForeColor Property

See also Example Applies To

**Summary:** Sets or returns the color used to display text.

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

| Element | Description |
|---|---|
| *object* | An object expression that evaluates to a VBMax C3D object. |
| *value* | A value or constant that determines the color. |

## See Also

FillColor Property
Font Properties

### ForeColor Property Applies To

- Caption Method
- Frame Method
- PercentMeter Method

## ForeColor Property Example

This example displays 3D text in a status bar. To see how it works, add a Label control to a form and copy and paste this code into the Form_Load event. Position and size the Label control where you would like to see the status bar.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .ForeColor = vbBlue                          'Sets the color of the status bar text
        . StatusBar Label1, "Connecting to network . . ."        'Displays the message in the status bar
    End With

End Sub
```

# Offset Property

Example Applies To

**Summary:** Sets or returns the distance between a control and its drop-shadow.

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

| Element | Description |
|---|---|
| *object* | An object expression that evaluates to a VBMax C3D object. |
| *value* | A numeric expression representing the offset in pixels. |

### Offset Property Applies To

DropShadow Method

## Offset Property Example

This example draws a drop-shadow behind a control. To see how it works, add a TextBox control to a form and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Offset = 5                 'Sets the shadow offset to five pixels
        .Shadow = gnLIGHT_SHADOW    'Uses the lighter colored shadows
        .DropShadow Text1           'Draws a drop-shadow behind the TextBox control
    End With

    Set o3D = Nothing

End Sub
```

# Shadow Property

Example Applies To

**Summary:** Sets or returns a value specifying whether to use light or dark shadows with the various 3D effects.

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

| Element | Description | |
|---|---|---|
| *object* | An object expression that evaluates to a VBMax C3D object. | |
| *value* | A numeric expression specifying the type of shadow. | |
| | Settings: | |
| | gnLIGHT_SHADOW | Use light shadows. |
| | gnDARK_SHADOW | Use dark shadows. |

### Shadow Property Applies To

Entire object

## Shadow Property Example

This example draws a drop-shadow behind a control. To see how it works, add a TextBox control to a form and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Offset = 5                 'Sets the shadow offset to five pixels
        .Shadow = gnLIGHT_SHADOW    'Uses the lighter colored shadows
        .DropShadow Text1           'Draws a drop-shadow behind the TextBox control
    End With

    Set o3D = Nothing

End Sub
```

# ShowPercent Property

See also Example Applies To

**Summary:** Sets or returns a value specifying whether or not to show the numbers on a percentage meter.

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

| Element | Description |
|---|---|
| *object* | Required. An object expression that evaluates to a VBMax C3D object. |
| *boolean* | A True or False value indicating whether or not to show the percentage numbers. |

## See Also

SolidMeter Property

### ShowPercent Property Applies To

PercentMeter Method

## ShowPercent Property Example

This example displays a percentage meter and allows you to toggle the percentage numbers on and off. To see how it works, add a Timer, a Label and a CommandButton control to a form and copy and paste this code into the form's code area. Position the Label control where you would like to see the percentage meter, set its Height property to 14 pixels and set the Timer's Interval property to 1.

```
Dim mo3D As New C3D

Private Sub Form_Load()

    With mo3D
        .FontBold = True          'Sets the font attribute of the percentage numbers
        .ForeColor = vbWhite      'Sets the color of the percentage numbers
        .FillColor = vbBlue       'Sets the color of the progress bar
    End With

End Sub

Private Sub Timer1_Timer()
Static nPercent As Integer
    mo3D.PercentMeter Label1, nPercent
    nPercent = nPercent + 1
    If nPercent > 100 Then nPercent = 0
End Sub

Private Sub Command1_Click()
    With mo3D
        . ShowPercent = Not . ShowPercent'Toggle the numbers on and off
    End With
End Sub
```

# SolidMeter Property

See also Example Applies To

**Summary:** Sets or returns a value specifying whether or not to show the percentage meter as a solid bar.

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

| Element | Description |
|---|---|
| *object* | An object expression that evaluates to a VBMax C3D object. |
| *boolean* | A True or False value indicating whether or not to use a solid bar for the percentage meter. If set to False (default), the Windows 95 progress meter style is used. |

## See Also

ShowPercent Property

### SolidMeter Property Applies To

PercentMeter Method

## SolidMeter Property Example

This example displays a percentage meter and allows you to toggle the Windows 95 style progress meter and a solid bar. To see how it works, add a Timer, a Label and a CommandButton control to a form and copy and paste this code into the form's code area. Position the Label control where you would like to see the percentage meter, set its Height property to 14 pixels and set the Timer's Interval property to 1.

```
Dim mo3D As New C3D

Private Sub Form_Load()

    With mo3D
        .ForeColor = vbWhite        'Sets the color of the percentage numbers
        .FillColor = vbBlue         'Sets the color of the progress bar
    End With

End Sub

Private Sub Timer1_Timer()
Static nPercent As Integer
    mo3D.PercentMeter Label1, nPercent
    nPercent = nPercent + 1
    If nPercent > 100 Then nPercent = 0
End Sub

Private Sub Command1_Click()
    With mo3D
        . SolidMeter = Not . SolidMeter      'Toggle between the bar styles
    End With
End Sub
```

# Style Property

See also Example Applies To

**Summary:** Sets or returns a value representing the type of 3D effect to be used.

**Syntax:** *object*.**Style** [=*style*]

| Element | Description | |
|---|---|---|
| *object* | An object expression that evaluates to a VBMax C3D object. | |
| *style* | A numeric expression specifying the 3D effect. | |
| | Settings: | |
| | gnNORMAL | No 3D effects. |
| | gnRAISED | Creates a raised 3D look. |
| | gnRECESSED | Gives a recessed 3D look. |
| | gnEMBOSSED_RAISED | Gives text a raised embossed look. |
| | gnEMBOSSED_RECESSED | Gives text a recessed embossed look. |
| | gnFLOATING | Makes text look like it is floating. |
| | gnOUTLINE_LEFT | Shows only the left edges of the text. |
| | gnOUTLINE_RIGHT | Shows only the right edges of the text. |
| | gnPROJECTED | Displays raised text from a different perspective. |

## See Also

Font Properties
ForeColor Property
Shadow Property

### Style Property Applies To

- Caption Method
- Frame Method
- StatusBar Method

## Style Property Example

This example draws a caption on a form and allows you to test each one of the different styles. To see how it works, add a Label control and a CommandButton control to a form and copy and paste this code into the form's code area.

```
Dim mo3D As New C3D

Private Sub Form_Load()

    With mo3D
        .Style = gnNORMAL                   'Start with no 3D effects
        .Caption Label1, gnUSE_CONTROL_FONT         'Uses Label control's font settings
    End With

End Sub

Private Sub Command1_Click()
Dim n As Integer

    With mo3D
        n =.Style + 1                       'Cycle through the effects
        If n > gnPROJECTED Then n = gnRAISED
        .Style = n
        Cls                                 'Clear the previous effect
        .Caption Label1, gnUSE_CONTROL_FONT         'Redisplays the text
    End With
End Sub
```

# Width Property

See also Example Applies To

**Summary:** Sets or returns a value that determines the width of panels and borders.

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

| Element | Description |
|---|---|
| *object* | An object expression that evaluates to a VBMax C3D object. |
| *value* | A numeric expression specifying the width of a panel or border in pixels. |

### See Also

Depth Property

### Width Property Applies To

- Border Method
- FormBorder Method
- Panel Method
- StatusBar Method

## Width Property Example

This example draws a border around a control. To see how it works, add a Label control to a form and copy and paste this code into the Form_Load event procedure.

```
Private Sub Form_Load()
Dim o3D As New C3D

    With o3D
        .Depth = 1              'Sets the depth of the border to one pixel
        .Width = 2              'Sets the width of the border to two pixels
        .Style = gnRAISED       'Constants defined in 3DDemo.bas
        .Border Label1          'Draws the border around the Label control
    End With

    Set o3D = Nothing

End Sub
```

## 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 Liquid Crystal Display

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

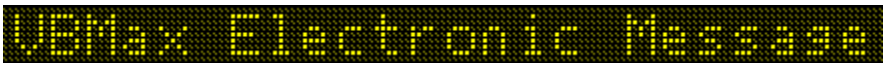
VBMaxLCD.dll is an in-process OLE server for adding LCD/LED style controls to your VB 4.0 applications. There are tons of uses for these babies: clocks, timers, meters, calculators, dialers—the list goes on.

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**

# VBMax Electronic Message Display




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**

# 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.

**Price: Free**

# 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 3D Effects 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 Electronic Display, 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 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**.

# Constants

Properties Methods

```
Const gnNORMAL = 0
Const gnRAISED = 1
Const gnRECESSED = 2
Const gnEMBOSSED_RAISED = 3
Const gnEMBOSSED_RECESSED = 4
Const gnFLOATING = 5
Const gnOUTLINE_RIGHT = 6
Const gnOUTLINE_LEFT = 7
Const gnPROJECTED = 8

Const gnUSE_CONTROL_FONT = 1

Const gnLIGHT_SHADOW = 0
Const gnDARK_SHADOW = 1

Const gnLEFT_JUSTIFY = 0
Const gnRIGHT_JUSTIFY = 1
Const gnCENTER = 2
```