



by Barry Seymour

The Following Program is brought to you in Living Color...

When I was a kid I used to pester my school buddy, John Zawada. He was the first kid in my class whose family had a color TV! "What color is the Enterprise?" I'd ask -- "What color is the Jupiter Two?"

Nowadays we take color for granted, in our TVs and our video screens. This week we have a method for controlling Windows' color scheme in Visual Basic. I'm not sure if there are many practical applications for this trick, but it sure was fun to come up with it!

If you'll look in your WIN.INI file in the [Colors] section, you'll see the RGB values for your system color set. Windows uses three numbers to specify the amount of Red, Green and Blue in a color, hence the term "RGB." Other programming languages (especially variants of BASIC) use this scheme as well.

In Windows/Visual Basic, you specify color values by long integer. You can get a color value with VB's RGB function; pass it your red, green and blue proportions and RGB returns the long integer which represents it.

Once armed with this method, you can use the API functions GetSysColor and SetSysColors to play around with your color scheme. The names of these two functions are fairly self-explanatory, so I'll just commence with the declarations. Note the Line Join symbol, **<+>**, your clue that what looks like two lines should really be one...



```
Declare Function GetSysColor Lib "User" (ByVal nIndex As Integer) As Long
Declare Sub SetSysColors Lib "User" (ByVal nChanges <+>
    As Integer, sColor As Integer, lpColorValues As Any)
```



GetSysColor does not read WIN.INI; it gets the values for the twenty system colors directly from Windows as defined by the user. nIndex refers to which of the twenty colors you're getting -- zero through nineteen. Which number relates to which color is in Visual Basic's CONSTANT.TXT -- you'll see them in the code declarations for this week as well.

SetSysColors can be a bit tricky. nChanges refers to how many colors you're changing; sColor is the number of the color to change and lpColorValues can be a long integer or an array of longs. For this example we're going to loop through all twenty color values, so we'll be setting nChanges to one all the way through.

VB019EX has a simple startup form which obtains your system's color scheme at startup by looping through those twenty values and storing them in an array...



'Get Current system colors, save to Global Array

```
Text1.Text = "This is a list of the system colors found at startup. "
+ Chr$(13) + Chr$(10) + Chr$(13) + Chr$(10)

ReDim OriginalColors(19)

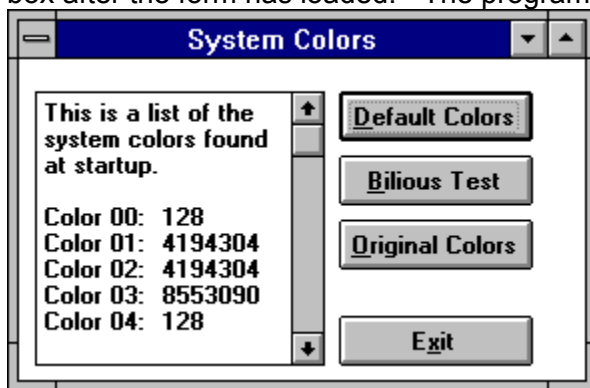
For X% = 0 To 19
    OriginalColors(X%) = GetSysColor(X%)

    Text1.Text = Text1.Text + "Color " + Format$(X%, "00") <+>
        + ": " + Format$(OriginalColors(X%), "0") <+>
        + Chr$(13) + Chr$(10)

Next X%
```



As you can see, the values obtained are stored in a character string which is used to fill a text box after the form has loaded. The program then presents the following form...



This form will at first come up in your normal color scheme, but if you click the "Default Colors" button you'll find your entire system has changed to the Windows Default color scheme. The values that do this were painstakingly culled from WIN.INI after I set those values with the Control Panel. Again, you'll see them in the source code.

Now comes the fun part. Clicking on the "Bilious Test" button will generate a sequence of 60 random integers which are converted into twenty RGB values. A call is then made to SetSysColors to change each color. The results can be quite hilarious and often eye-straining. No matter -- this kind of ridiculous tinkering around is what computers were **made** for.



'generate, set and display random color schemes!

```
ReDim BiliousColors(19) As Long
UpperBound% = 255
LowerBound% = 0

For Bilious% = 1 To 1 'Generate a random color set
```

```

For X% = 0 To 19
    ColorVal1% = Int((UpperBound% - LowerBound% + 1) * <+>
        Rnd + LowerBound%)
    ColorVal2% = Int((UpperBound% - LowerBound% + 1) * <+>
        Rnd + LowerBound%)
    ColorVal3% = Int((UpperBound% - LowerBound% + 1) * <+>
        Rnd + LowerBound%)
    BiliousColors(X%) = RGB(ColorVal1%, ColorVal2%, <+>
        ColorVal3%)
Next X%

'Set the random color set
For X% = 0 To 19
    SetSysColors 1, X%, BiliousColors(X%)
Next X%
D% = DoEvents()
Next Bilious%

```



This is in a loop of 1 to 1, so it only executes once, but if you want a truly sickening color display, set the loop to 10 or 20 and watch the thing run. The results can be unsettling for those of you with an aversion to plaid pants, but are harmless fun otherwise.

Clicking on the "Restore Colors" button restores the original user color scheme as saved in memory. The Exit button also pushes this button prior to ending the program to ensure you're not left with fuschia text on a chartreuse background.

This trick came in handy for me in writing a demonstration version of a program; in a snap I was able to set the system colors to a scheme requested by the client for the demo, and then back to the user color palette when the demo was done. Other uses may include a screen shot application which requires a certain default color set prior to taking 'pictures' off the screen. I hope it comes in useful for you as well!

To try this program, create a form named VB019EX with the following controls...

Control	Properties
Command1	Caption = "Default Colors"
Command2	Caption = "Bilious Test"
Command3	Caption = "Restore User Colors"
Command4	Exit
Text1	Multiline = TRUE ScrollBars = 2 '(Vertical)

Past the code from the [Example](#) into the program and all should fall into place.

This column is available on the Windows Online BBS in Danville, California, phone 1 510 736-8343. This column in Windows HELP format (VB019EX.HLP) is in VB019EX.ZIP, and may be distributed as freeware.

Marquette Computer Consultants

22 Sirard Lane

San Rafael, CA 94901-1066

(415) 459-0835

for Windows Online "the Weekly"





Example Code



```
' Booleans
Const TRUE = -1
Const FALSE = 0

Declare Function GetSysColor Lib "User" (ByVal nIndex As Integer) As
Long
Declare Sub SetSysColors Lib "User" (ByVal nChanges As Integer,
lpSysColor As Integer, lpColorValues As Any)

' Color Types
Const COLOR_SCROLLBAR = 0
Const COLOR_BACKGROUND = 1
Const COLOR_ACTIVECAPTION = 2
Const COLOR_INACTIVECAPTION = 3
Const COLOR_MENU = 4
Const COLOR_WINDOW = 5
Const COLOR_WINDOWFRAME = 6
Const COLOR_MENUTEXT = 7
Const COLOR_WINDOWTEXT = 8
Const COLOR_CAPTIONTEXT = 9
Const COLOR_ACTIVEBORDER = 10
Const COLOR_INACTIVEBORDER = 11
Const COLOR_APPWORKSPACE = 12
Const COLOR_HIGHLIGHT = 13
Const COLOR_HIGHLIGHTTEXT = 14
Const COLOR_BTNFACE = 15
Const COLOR_BTNSHADOW = 16
Const COLOR_GRAYTEXT = 17
Const COLOR_BTNTEXT = 18
Const COLOR_ICONBACKGROUND = 19

Dim OriginalColors() As Long

Sub ColorExit_Click ()
    End
End Sub

Sub Form_Load ()
    'Get Current system colors, save to Global Array

    Text1.Text = "This is a list of the system colors found at
startup. " + Chr$(13) + Chr$(10) + Chr$(13) + Chr$(10)

    ReDim OriginalColors(19)
```

```

    For X% = 0 To 19
        OriginalColors(X%) = GetSysColor(X%)
        Text1.Text = Text1.Text + "Color " + Format$(X%, "00") + ": "
+ Format$(OriginalColors(X%), "0") + Chr$(13) + Chr$(10)
    Next X%
End Sub

```

```

Sub Command1_Click ()
    ReDim NewColors(19) As Long
    NewColors(COLOR_SCROLLBAR) = RGB(192, 192, 192)
    NewColors(COLOR_BACKGROUND) = RGB(192, 192, 192)
    NewColors(COLOR_ACTIVECAPTION) = RGB(0, 0, 128)
    NewColors(COLOR_INACTIVECAPTION) = RGB(255, 255, 255)
    NewColors(COLOR_MENU) = RGB(255, 255, 255)
    NewColors(COLOR_WINDOW) = RGB(255, 255, 255)
    NewColors(COLOR_WINDOWFRAME) = RGB(0, 0, 0)
    NewColors(COLOR_MENUTEXT) = RGB(0, 0, 0)
    NewColors(COLOR_WINDOWTEXT) = RGB(0, 0, 0)
    NewColors(COLOR_CAPTIONTEXT) = RGB(255, 255, 255)
    NewColors(COLOR_ACTIVEBORDER) = RGB(192, 192, 192)
    NewColors(COLOR_INACTIVEBORDER) = RGB(192, 192, 192)
    NewColors(COLOR_APPWORKSPACE) = RGB(255, 255, 255)
    NewColors(COLOR_HIGHLIGHT) = RGB(0, 0, 128)
    NewColors(COLOR_HIGHLIGHTTEXT) = RGB(255, 255, 255)
    NewColors(COLOR_BTNFACE) = RGB(192, 192, 192)
    NewColors(COLOR_BTNSHADOW) = RGB(128, 128, 128)
    NewColors(COLOR_GRAYTEXT) = RGB(192, 192, 192)
    NewColors(COLOR_BTNTEXT) = RGB(0, 0, 0)

    For R% = 0 To 19
        SetSysColors 1, R%, (NewColors(R%))
    Next R%

```

```

End Sub

```

```

Sub Command2_Click ()
    'generate, set and display random color schemes!

    ReDim BiliuousColors(19) As Long
    UpperBound% = 255
    LowerBound% = 0

    For Biliuous% = 1 To 1 'Generate a random color set
        For X% = 0 To 19
            ColorVal1% = Int((UpperBound% - LowerBound% + 1) * Rnd +
LowerBound%)
            ColorVal2% = Int((UpperBound% - LowerBound% + 1) * Rnd +
LowerBound%)
            ColorVal3% = Int((UpperBound% - LowerBound% + 1) * Rnd +
LowerBound%)
            BiliuousColors(X%) = RGB(ColorVal1%, ColorVal2%,
ColorVal3%)

```

```

        Next X%

        'Set the random color set
        For X% = 0 To 19
            SetSysColors 1, X%, BiliousColors(X%)
        Next X%
        D% = DoEvents()
    Next Bilious%
End Sub

Sub Command3_Click ()
    'restore user Colors
    For R% = 0 To 19
        SetSysColors 1, R%, OriginalColors(R%)
    Next R%
End Sub

Sub Command4_Click ()
    Command3_Click
    Unload VB019EX
End Sub

```

