LAMPIRAN A
LISTING PROGRAM

1. Listing Program Utama

```
Private Declare Function SendMessage Lib "user32" Alias "SendMessageA"
(ByVal hWnd As Long, ByVal wMsg As Long, ByVal wParam As Long, lParam
As Any) As Long
Private Declare Function capCreateCaptureWindow Lib "avicap32.dll" Alias
"capCreateCaptureWindowA" (ByVal lpszWindowName As String, ByVal
dwStyle As Long, ByVal X As Long, ByVal Y As Long, ByVal nWidth As Long,
ByVal nHeight As Long, ByVal hwndParent As Long, ByVal nID As Long) As
Long
Private Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)

Private mCapHwnd As Long

Private Const CONNECT As Long = 1034
Private Const DISCONNECT As Long = 1035
Private Const GET_FRAME As Long = 1084
Private Const COPY As Long = 1054
Dim continue As Boolean
Private m_Mouse As CMouseEvent


Dim i As Integer, j As Integer
Dim r As Integer, G As Integer, B As Integer, L As Long
Dim X As Long, Y As Long
Dim hitam9 As Integer, hitam10 As Integer
Dim hitam1 As Integer, hitam2 As Integer, hitam3 As Integer, hitam4 As Integer,
hitam5 As Integer, hitam6 As Integer, hitam7 As Integer, hitam8 As Integer
Dim totalhitam As Long
Option Explicit

Dim fDraw As New FastDrawing
Dim ImageData() As Byte
Dim Rb As Long, Gb As Long, Bb As Long
Dim lebarFot As Long, tinggiFot As Long
Dim QuickX As Long
Dim z As Integer

Sub STARTCAM() 'Memulai kamera untuk mengambil gambar
mCapHwnd = capCreateCaptureWindow("WebcamCapture", 0, 0, 0, 320, 240,
Me.hWnd, 0)
DoEvents
SendMessage mCapHwnd, CONNECT, 0, 0 'menghubungkan kamera
End Sub
```

```
Sub STOPCAM() ' untuk memberhentikan kamera
DoEvents: SendMessage mCapHwnd, DISCONNECT, 0, 0
End Sub
```

TOMBOL STARTCAM

```
Private Sub cmdstart_Click() ' STARTCAM
STARTCAM
Timer2.Enabled = False
End Sub
```

TOMBOL CAPTURE

```
Private Sub cmdcapture_Click() ' CAPTURE
Timer2.Enabled = True
Timer3.Enabled = False
End Sub
```

TOMBOL STOPCAM

```
Private Sub cmdstop_Click() 'STOPCAM
continue = False
Timer2.Enabled = False
Timer3.Enabled = False
STOPCAM
End Sub
```

TOMBOL BUKA MATA PADA SETTING MATA

```
Private Sub Command1_Click()
Picture4.Picture = Picture3.Picture
hitam9 = 0
lebarFot = fDraw.GetImageWidth(Picture4)
tinggiFot = fDraw.GetImageHeight(Picture4)
fDraw.GetImageData2D Picture4, ImageData()
For X = 0 To lebarFot - 1
  QuickX = X * 3
  For Y = 0 To tinggiFot - 1
        r = ImageData(QuickX + 2, Y)
        G = ImageData(QuickX + 1, Y)
        B = ImageData(QuickX, Y)
        L = (r + G + B) \ 3
        If r = 0 And G = 0 And B = 0 Then
           hitam9 = hitam9 + 1
           Label1.Caption = hitam9
           End If
     Next Y
   Next X
End Sub
```

TOMBOL TUTUP MATA PADA SETTING MATA

```
Private Sub Command2_Click()
Picture5.Picture = Picture3.Picture
hitam10 = 0
lebarFot = fDraw.GetImageWidth(Picture5)
tinggiFot = fDraw.GetImageHeight(Picture5)
fDraw.GetImageData2D Picture5, ImageData()
For X = 0 To lebarFot - 1
  QuickX = X * 3
  For Y = 0 To tinggiFot - 1
        r = ImageData(QuickX + 2, Y)
        G = ImageData(QuickX + 1, Y)
        B = ImageData(QuickX, Y)
        L = (r + G + B) \ 3
        If r = 0 And G = 0 And B = 0 Then
           hitam10 = hitam10 + 1
           Label2.Caption = hitam10
           End If
     Next Y
   Next X
End Sub
```

TOMBOL RUN

```
Private Sub cmdrun_Click()
Timer3.Enabled = True
End Sub

Private Sub Form_Load()
Call cmdstart_Click
Set m_Mouse = New CMouseEvent
'z = 0
Text1.BackColor = &H8000000F
End Sub
```

TIMER 1

```
Private Sub Timer1_Timer()
 SendMessage mCapHwnd, GET_FRAME, 0, 0
    SendMessage mCapHwnd, COPY, 0, 0
    Picture1(0).Picture = Clipboard.GetData
    Image1(0).Picture = Picture1(0).Picture
End Sub
```

TIMER 2

```
Private Sub Timer2_Timer()
Picture3.Picture = Picture1(1).Picture
```

```
For i = 1 To 4 Step 1
Dim ImageData() As Byte
Dim Rb As Long, Gb As Long, Bb As Long
Dim lebarFot As Long, tinggiFot As Long
Dim QuickX As Long
```

'Picture diambil nilai RGB nya

```
lebarFot = fDraw.GetImageWidth(Picture1(0))
tinggiFot = fDraw.GetImageHeight(Picture1(0))
fDraw.GetImageData2D Picture1(0), ImageData()
For X = 0 To lebarFot - 1
  QuickX = X * 3
   For Y = 0 To tinggiFot - 1
        r = ImageData(QuickX + 2, Y)
        G = ImageData(QuickX + 1, Y)
        B = ImageData(QuickX, Y)
        L = (r + G + B) \ 3
      Next Y
  Next X
```

'picture menjadi hitam putih (biner)

```
fDraw.SetImageData2D Picture1(0), lebarFot, tinggiFot, ImageData(), False
Image1(0).Picture = Picture1(0).Picture
   Picture1(i).Picture = Picture1(0).Picture
   Picture1(i).Height = shp1(0).Height
   Picture1(i).Width = shp1(0).Width
   Picture1(i).Refresh
   Picture1(i).PaintPicture Picture1(0).Image, 0, 0, shp1(0).Width, shp1(0).Height,
shp1(0).Left, shp1(0).Top, shp1(0).Width, shp1(0).Height, vbSrcCopy
lebarFot = fDraw.GetImageWidth(Picture1(i))
tinggiFot = fDraw.GetImageHeight(Picture1(i))
fDraw.GetImageData2D Picture1(i), ImageData()
For X = 0 To lebarFot - 1
  QuickX = X * 3
   For Y = 0 To tinggiFot - 1
        r = ImageData(QuickX + 2, Y)
        G = ImageData(QuickX + 1, Y)
        B = ImageData(QuickX, Y)
        L = (r + G + B) \ 3
       If L >= 75 Then L = 255 Else L = 0
      ImageData(QuickX + 2, Y) = L
      ImageData(QuickX + 1, Y) = L
      ImageData(QuickX, Y) = L
    Next Y
  Next X
fDraw.SetImageData2D Picture1(i), lebarFot, tinggiFot, ImageData(), False
```

```
'PICTURE 1
hitam1 = 0
lebarFot = fDraw.GetImageWidth(Picture1(1))
tinggiFot = fDraw.GetImageHeight(Picture1(1))
fDraw.GetImageData2D Picture1(1), ImageData()
For X = 0 To lebarFot - 1
  QuickX = X * 3
  For Y = 0 To tinggiFot - 1
        r = ImageData(QuickX + 2, Y)
        G = ImageData(QuickX + 1, Y)
        B = ImageData(QuickX, Y)
        L = (r + G + B) \ 3
        If r = 0 And G = 0 And B = 0 Then
           hitam1 = hitam1 + 1
           Label15.Caption = hitam1
           End If
     Next Y
   Next X


 PICTURE 2
hitam2 = 0
lebarFot = fDraw.GetImageWidth(Picture1(2))
tinggiFot = fDraw.GetImageHeight(Picture1(2))
fDraw.GetImageData2D Picture1(2), ImageData()
For X = 0 To lebarFot - 1
  QuickX = X * 3
  For Y = 0 To tinggiFot - 1
        r = ImageData(QuickX + 2, Y)
        G = ImageData(QuickX + 1, Y)
        B = ImageData(QuickX, Y)
        L = (r + G + B) \ 3
        If r = 0 And G = 0 And B = 0 Then
           hitam2 = hitam2 + 1
           Label26.Caption = hitam2
           End If
     Next Y
   Next X

PICTURE 3
hitam3 = 0
lebarFot = fDraw.GetImageWidth(Picture1(3))
tinggiFot = fDraw.GetImageHeight(Picture1(3))
fDraw.GetImageData2D Picture1(3), ImageData()
For X = 0 To lebarFot - 1
  QuickX = X * 3
  For Y = 0 To tinggiFot - 1
        r = ImageData(QuickX + 2, Y)
        G = ImageData(QuickX + 1, Y)
        B = ImageData(QuickX, Y)
        L = (r + G + B) \ 3
        If r = 0 And G = 0 And B = 0 Then
           hitam3 = hitam3 + 1
           Label31.Caption = hitam3
           End If
     Next Y
  Next X
```

PICTURE NOMOR 4

```
hitam4 = 0
lebarFot = fDraw.GetImageWidth(Picture1(4))
tinggiFot = fDraw.GetImageHeight(Picture1(4))
fDraw.GetImageData2D Picture1(4), ImageData()
For X = 0 To lebarFot - 1
  QuickX = X * 3
  For Y = 0 To tinggiFot - 1
        r = ImageData(QuickX + 2, Y)
        G = ImageData(QuickX + 1, Y)
        B = ImageData(QuickX, Y)
        L = (r + G + B) \ 3
        If r = 0 And G = 0 And B = 0 Then
           hitam4 = hitam4 + 1
           Label36.Caption = hitam4
           End If
     Next Y
  Next X
  End Sub
```

TIMER 3

```
Private Sub Timer3_Timer()
```

'mengaktifkan fungsi klik pada mouse

```
Text1.BackColor = &HC00000
z = 0
If hitam1 < (hitam9 - 10) And hitam1 > (hitam10) Then z = z + 1 Else z = 0
If hitam2 < (hitam9 - 10) And hitam2 > (hitam10) Then z = z + 1 Else z = 0
If hitam3 < (hitam9 - 10) And hitam3 > (hitam10) Then z = z + 1 Else z = 0
If hitam4 < (hitam9 - 10) And hitam4 > (hitam10) Then z = z + 1 Else z = 0
           Label3.Caption = z
           If z = 4 Then
             m_Mouse.Click
```

```
            hitam1 = 0
             hitam2 = 0
             hitam3 = 0
             hitam4 = 0
            z = 0
            Text1.BackColor = &HFF00&
             Text1.Refresh
          End If

End Sub
```

2. Listing Program Class Module CMouseEvent

```
Option Explicit
Private Declare Sub mouse_event Lib "user32" (ByVal dwFlags As Long, ByVal
dX As Long, ByVal dY As Long, ByVal dwData As Long, ByVal dwExtraInfo
As Long)
Private Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)

' Flags used with mouse_event
Private Const MOUSEEVENTF_ABSOLUTE = &H8000& ' absolute move
Private Const MOUSEEVENTF_LEFTDOWN = &H2     ' left button down
Private Const MOUSEEVENTF_LEFTUP = &H4       ' left button up
Private Const MOUSEEVENTF_MIDDLEDOWN = &H20  ' middle button down
Private Const MOUSEEVENTF_MIDDLEUP = &H40    ' middle button up
Private Const MOUSEEVENTF_MOVE = &H1         ' mouse move
Private Const MOUSEEVENTF_RIGHTDOWN = &H8    ' right button down
Private Const MOUSEEVENTF_RIGHTUP = &H10     ' right button up
Private Const MOUSEEVENTF_WHEEL = &H800      ' wheel button rolled


' A few module level variables...
Private m_ClickDelay As Long
' ***********************************************************
'  Initialize
' ***********************************************************
Private Sub Class_Initialize()
   ' Default duration for mousedown
   m_ClickDelay = 250 'milliseconds
End Sub
' ***********************************************************
'  Public Properties
' ***********************************************************
Public Property Let ClickDelay(ByVal NewVal As Long)
   If NewVal >= 0 Then m_ClickDelay = NewVal
```

```
End Property
Public Property Get ClickDelay() As Long
  ClickDelay = m_ClickDelay
End Property
Public Sub Click()
  ' Click the mouse, with delay to simulate human timing.
  Call mouse_event(MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0)
  If m_ClickDelay Then
    DoEvents ' allow down position to paint
    Call Sleep(m_ClickDelay)
  End If
  Call mouse_event(MOUSEEVENTF_LEFTUP, 0, 0, 0, 0)
End Sub
```

3. Listing Program ClassModule FastDrawing

```
Option Explicit
'Stripped down bitmap information
Private Type Bitmap
   bmType As Long
   bmWidth As Long
   bmHeight As Long
   bmWidthBytes As Long
   bmPlanes As Integer
   bmBitsPixel As Integer
   bmBits As Long
End Type

'Call to transfer an object's properties into a custom variable
Private Declare Function GetObject Lib "gdi32" Alias "GetObjectA" (ByVal
hObject As Long, ByVal nCount As Long, ByRef lpObject As Any) As Long

'Standard pixel data
Private Type RGBQUAD
        rgbBlue As Byte
        rgbGreen As Byte
        rgbRed As Byte
        rgbAlpha As Byte
End Type

'Full-size bitmap header
Private Type BITMAPINFOHEADER
        bmSize As Long
        bmWidth As Long
        bmHeight As Long
        bmPlanes As Integer
        bmBitCount As Integer
        bmCompression As Long
        bmSizeImage As Long
        bmXPelsPerMeter As Long
        bmYPelsPerMeter As Long
        bmClrUsed As Long
        bmClrImportant As Long
End Type

'Extended header for 8-bit images
Private Type BITMAPINFO
        bmHeader As BITMAPINFOHEADER
        bmColors(0 To 255) As RGBQUAD
End Type

'Used to ensure quality stretching of color images
Private Declare Function SetStretchBltMode Lib "gdi32" (ByVal hdc As Long,
ByVal nStretchMode As Long) As Long

'DIB section interfaces
Private Declare Function GetDIBits Lib "gdi32" (ByVal hdc As Long, ByVal
hBitmap As Long, ByVal nStartScan As Long, ByVal nNumScans As Long,
lpBits As Any, lpBI As BITMAPINFO, ByVal wUsage As Long) As Long
Private Declare Function StretchDIBits Lib "gdi32" (ByVal hdc As Long, ByVal
X As Long, ByVal Y As Long, ByVal dX As Long, ByVal dY As Long, ByVal
SrcX As Long, ByVal SrcY As Long, ByVal SrcWidth As Long, ByVal
SrcHeight As Long, lpBits As Any, lpBitsInfo As BITMAPINFO, ByVal wUsage
As Long, ByVal dwRop As Long) As Long

'Get the image width (via API - always accurate, unlike
PictureBox.ScaleWidth)
Public Function GetImageWidth(SrcPictureBox As PictureBox) As Long
    Dim bm As Bitmap
    GetObject SrcPictureBox.Image, Len(bm), bm
    GetImageWidth = bm.bmWidth
End Function

'Get the image height (via API - always accurate)
Public Function GetImageHeight(SrcPictureBox As PictureBox) As Long
    Dim bm As Bitmap
    GetObject SrcPictureBox.Image, Len(bm), bm
    GetImageHeight = bm.bmHeight
End Function

'Get the stream length of an image (via API - always accurate)
Public Function GetImageStreamLength(SrcPictureBox As PictureBox) As Long
    Dim bm As Bitmap
```

```
   GetObject SrcPictureBox.Image, Len(bm), bm
   GetImageStreamLength = (bm.bmWidth * (bm.bmHeight + 1)) * 3
End Function

'Get an image's pixel information into an array dimensioned (x * 3 + bgr, y),
with the option to get it in its true orientation
Public Sub GetImageData2D(SrcPictureBox As PictureBox, ImageData() As
Byte, Optional ByVal CorrectOrientation As Boolean = False)
   Dim bm As Bitmap
   'Get the picture box information
   GetObject SrcPictureBox.Image, Len(bm), bm
   'Build a correctly sized array
   Erase ImageData()
   'Generate a correctly-dimensioned array (for 2-dimensional access)
   Dim ArrayWidth As Long
   ArrayWidth = (bm.bmWidth * 3) - 1
   ArrayWidth = ArrayWidth + (bm.bmWidth Mod 4)  '4-bit alignment
   ReDim ImageData(0 To ArrayWidth, 0 To bm.bmHeight) As Byte
   ReDim tmpData(0 To ArrayWidth, 0 To bm.bmHeight) As Byte

   'Create a temporary header to pass to the GetDIBits call
   Dim bmi As BITMAPINFO
   bmi.bmHeader.bmWidth = bm.bmWidth
   bmi.bmHeader.bmHeight = bm.bmHeight
   bmi.bmHeader.bmSize = 40              'Size, in bytes, of the header
   bmi.bmHeader.bmPlanes = 1              'Number of planes (always one for this
instance)
   bmi.bmHeader.bmBitCount = 24              'Bits per pixel (always 24 for this
instance)
   bmi.bmHeader.bmCompression = 0              'Compression :standard/none or
RLE

   'Get the image data into our array
   If CorrectOrientation = False Then
      GetDIBits SrcPictureBox.hdc, SrcPictureBox.Image, 0, bm.bmHeight,
ImageData(0, 0), bmi, 0
   Else
      GetDIBits SrcPictureBox.hdc, SrcPictureBox.Image, 0, bm.bmHeight,
tmpData(0, 0), bmi, 0
   End If

   'This code is to orient the image data correctly in the array (i.e. (0,0) as
top-left, (max,max) as bottom right)
   ' (if this option is enabled, we must set the DIB height to negative in the
SetImageData routine below)
   If CorrectOrientation = True Then
```

```
        Dim X As Long, Y As Long, z As Long
        Dim QuickVal As Long
        For X = 0 To bm.bmWidth - 1
           QuickVal = X * 3
         For Y = 0 To bm.bmHeight - 1
          For z = 0 To 2
           ImageData(QuickVal + z, Y) = tmpData(QuickVal + z, bm.bmHeight - Y)
          Next z
         Next Y
        Next X

    End If

    'Save memory...?
    Erase tmpData

End Sub

'Set an image's pixel information from an array dimensioned (x * 3 + bgr, y)
Public Sub SetImageData2D(DstPictureBox As PictureBox, OriginalWidth As
Long, OriginalHeight As Long, ImageData() As Byte, Optional ByVal
CorrectOrientation As Boolean = False)
    Dim bm As Bitmap
    'Get the picture box information
    GetObject DstPictureBox.Image, Len(bm), bm
    'Create a temporary header to pass to the StretchDIBits call
    Dim bmi As BITMAPINFO
    bmi.bmHeader.bmWidth = OriginalWidth
    If CorrectOrientation = False Then
        bmi.bmHeader.bmHeight = OriginalHeight
    Else
        bmi.bmHeader.bmHeight = -OriginalHeight
    End If
    bmi.bmHeader.bmSize = 40              'Size, in bytes, of the header
    bmi.bmHeader.bmPlanes = 1             'Number of planes (always one for this
instance)
    bmi.bmHeader.bmBitCount = 24          'Bits per pixel (always 24 for this
instance)
    bmi.bmHeader.bmCompression = 0        'Compression :standard/none or RLE
    'Assume color images and set the corresponding best stretch mode
    SetStretchBltMode DstPictureBox.hdc, 3&
    'Send the array to the picture box and draw it accordingly
    StretchDIBits DstPictureBox.hdc, 0, 0, bm.bmWidth, bm.bmHeight, 0, 0,
OriginalWidth, OriginalHeight, ImageData(0, 0), bmi, 0, vbSrcCopy
```

```
   'Since this doesn't automatically initialize AutoRedraw, we have to do it
manually
   If DstPictureBox.AutoRedraw = True Then
      DstPictureBox.Picture = DstPictureBox.Image
      DstPictureBox.Refresh
   End If
   'Always good to manually halt for external processes after heavy API
usage
   DoEvents
End Sub

'Get an image's pixel information into an array dimensioned (r/g/b, x, y)
Public Sub GetImageData(SrcPictureBox As PictureBox, ImageData() As Byte)
   Dim bm As Bitmap
   'Get the picture box information
   GetObject SrcPictureBox.Image, Len(bm), bm
   'Build a correctly sized array
   Erase ImageData()
   ReDim ImageData(0 To 2, 0 To bm.bmWidth - 1, 0 To bm.bmHeight - 1)
   'Create a temporary header to pass to the GetDIBits call
   Dim bmi As BITMAPINFO
   bmi.bmHeader.bmWidth = bm.bmWidth
   bmi.bmHeader.bmHeight = bm.bmHeight
   bmi.bmHeader.bmSize = 40    'Size, in bytes, of the header
   bmi.bmHeader.bmPlanes = 1 'Number of planes (always one for this
instance)
   bmi.bmHeader.bmBitCount = 24   'Bits per pixel (always 24 for this instance)
   bmi.bmHeader.bmCompression = 0  'Compression :standard/none or RLE

   'Get the image data into our array
   GetDIBits SrcPictureBox.hdc, SrcPictureBox.Image, 0, bm.bmHeight,
ImageData(0, 0, 0), bmi, 0
End Sub

'Set an image's pixel information from an array dimensioned (r/g/b, x, y)
Public Sub SetImageData(DstPictureBox As PictureBox, OriginalWidth As Long,
OriginalHeight As Long, ImageData() As Byte)
   Dim bm As Bitmap
   'Get the picture box information
   GetObject DstPictureBox.Image, Len(bm), bm
   'Create a temporary header to pass to the StretchDIBits call
   Dim bmi As BITMAPINFO
   bmi.bmHeader.bmWidth = OriginalWidth
   bmi.bmHeader.bmHeight = OriginalHeight
   bmi.bmHeader.bmSize = 40              'Size, in bytes, of the header
   bmi.bmHeader.bmPlanes = 1              'Number of planes (always one for this
instance)
   bmi.bmHeader.bmBitCount = 24             'Bits per pixel (always 24 for this
instance)
   bmi.bmHeader.bmCompression = 0           'Compression :standard/none or
RLE
   'Assume color images and set the corresponding best stretch mode
   SetStretchBltMode DstPictureBox.hdc, 3&
   'Send the array to the picture box and draw it accordingly
   StretchDIBits DstPictureBox.hdc, 0, 0, bm.bmWidth, bm.bmHeight, 0, 0,
OriginalWidth, OriginalHeight, ImageData(0, 0, 0), bmi, 0, vbSrcCopy
   'Since this doesn't automatically initialize AutoRedraw, we have to do it
manually
   If DstPictureBox.AutoRedraw = True Then
      DstPictureBox.Picture = DstPictureBox.Image
      DstPictureBox.Refresh
   End If
   'Always good to manually halt for external processes after heavy API
usage
   DoEvents
End Sub

'Get an image's pixel data into a one-dimesional array (stream)
Public Sub GetImageDataStream(SrcPictureBox As PictureBox, ImageData() As
Byte)
   Dim bm As Bitmap
   'Get the picture box information
   GetObject SrcPictureBox.Image, Len(bm), bm
   'Build a correctly sized array - in this case, designed as a stream
   Erase ImageData()
   ReDim ImageData(0 To GetImageStreamLength(SrcPictureBox))
   'Create a temporary header to pass to the GetDIBits call
   Dim bmi As BITMAPINFO
   bmi.bmHeader.bmWidth = bm.bmWidth
   bmi.bmHeader.bmHeight = bm.bmHeight
   bmi.bmHeader.bmSize = 40               'Size, in bytes, of the header
   bmi.bmHeader.bmPlanes = 1              'Number of planes (always one for this
instance)
   bmi.bmHeader.bmBitCount = 24             'Bits per pixel (always 24 for this
instance)
   bmi.bmHeader.bmCompression = 0           'Compression :standard/none or
RLE
   'Get the image data into our array
   GetDIBits   SrcPictureBox.hdc,   SrcPictureBox.Image,   0,   bm.bmHeight,
ImageData(0), bmi, 0
End Sub
```

```
'Set an image's data from a one-dimensional array (stream)
Public Sub SetImageDataStream(DstPictureBox As PictureBox, OriginalWidth As Long, OriginalHeight As Long, ImageData() As Byte)
   Dim bm As Bitmap
   'Get the picture box information
   GetObject DstPictureBox.Image, Len(bm), bm
   'Create a temporary header to pass to the StretchDIBits call
   Dim bmi As BITMAPINFO
   bmi.bmHeader.bmWidth = OriginalWidth
   bmi.bmHeader.bmHeight = OriginalHeight
   bmi.bmHeader.bmSize = 40              'Size, in bytes, of the header
   bmi.bmHeader.bmPlanes = 1              'Number of planes (always one for this instance)
   bmi.bmHeader.bmBitCount = 24              'Bits per pixel (always 24 for this instance)
   bmi.bmHeader.bmCompression = 0              'Compression :standard/none or RLE
   'Send the array to the picture box and draw it accordingly
   StretchDIBits DstPictureBox.hdc, 0, 0, bm.bmWidth, bm.bmHeight, 0, 0, OriginalWidth, OriginalHeight, ImageData(0), bmi, 0, vbSrcCopy
   'Since this doesn't automatically initialize AutoRedraw, we have to do it manually
   If DstPictureBox.AutoRedraw = True Then
      DstPictureBox.Picture = DstPictureBox.Image
      DstPictureBox.Refresh
   End If
   'Always good to manually halt for external processes after heavy API usage
   DoEvents
End Sub
```