

Re: Rotated Text Help Needed

Source:

<http://www.tech-archive.net/Archive/VB/microsoft.public.vb.general.discussion/2005-10/msg00507.html>

- *From:* "Mike Williams" <Mike@xxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 7 Oct 2005 16:50:24 +0100
-

"Rick Rothstein [MVP – Visual Basic]" <rickNOSPAMnews@xxxxxxxxxxxxxxxxxxxxx> wrote in message news:%23yZG3xxyFHA.2880@xxxxxxxxxxxxxxxxxxxxx

- > Okay, I tried your experiment and got an interesting result.
- > The first print out was as I said originally... only the circles
- > (large and small) and the letters printed out. No shading or
- > boxes anywhere. However, the remaining three print outs
- > . . . had an 8" by 8" almost solid black box (as close as you
- > can get to being solid without actually being solid)

Hi again Rick. The problem with the first printout being different from the others is down to a rather silly mistake of mine. I forgot to set the ScaleMode to vbInches before I did the drawing of the rectangle, so that on the first run it was just a tiny little thing that was in any case covered by the character that was printed in the top left corner of the page. The Scalemode was set to vbInches only in the main character drawing routine. Anyway, as you almost certainly already know, Visual Basic does not fully clear out or reset the printer object on executing the EndDoc, and so many of the previous settings are "remembered". That is why the rectangle was drawn correctly on subsequent printouts. Here is the code as it stands so far, modified according to the code posted by Howard so that the code works the same in WinXP as it does in Win98. The only remaining thing to do is to properly centralise the characters (unless of course we wish to eventually use proportional character spacing or perhaps at least reduce the size of the space character). By "centralise the characters" I mean that we should draw them so that the centre of the character cell is at the desired position (rather than the left edge of it). This is easy to do of course, but I just haven't got round to it yet. To see what I mean make a printout using the following code and look carefully at the "H" character, for example. You'll see that it is not quite sitting where it should on the line, with the "right leg of the H" being a little higher than the left leg, with regards to that portion of the circle. Positioning the characters to the left by an amount equal to half the character cell width will fix this little problem. Check it out and see what you think.

Mike

Option Explicit

Re: Rotated Text Help Needed

```
Private Const LF_FACESIZE = 32
Private Type LOGFONT
IfHeight As Long
IfWidth As Long
IfEscapement As Long
IfOrientation As Long
IfWeight As Long
IfItalic As Byte
IfUnderline As Byte
IfStrikeOut As Byte
IfCharSet As Byte
IfOutPrecision As Byte
IfClipPrecision As Byte
IfQuality As Byte
IfPitchAndFamily As Byte
IfFaceName As String * LF_FACESIZE
End Type
Private Declare Function CreateFontIndirect Lib "gdi32" _
Alias "CreateFontIndirectA" (lpLogFont As LOGFONT) As Long
Private Declare Function SelectObject Lib "gdi32" _
(ByVal hdc As Long, ByVal hObject As Long) As Long
Private Declare Function DeleteObject Lib "gdi32" _
(ByVal hObject As Long) As Long
Private Declare Function TextOut Lib "gdi32" Alias _
"TextOutA" (ByVal hdc As Long, ByVal x As Long, ByVal _
y As Long, ByVal lpString As String, ByVal nCount _
As Long) As Long
Private Declare Function SetBkMode Lib "gdi32" _
(ByVal hdc As Long, ByVal nBkMode As Long) As Long
Private Declare Function GetTextMetrics Lib "gdi32" _
Alias "GetTextMetricsA" (ByVal hdc As Long, _
lpMetrics As TEXTMETRIC) As Long
Private Type TEXTMETRIC
tmHeight As Long
tmAscent As Long
tmDescent As Long
tmInternalLeading As Long
tmExternalLeading As Long
tmAveCharWidth As Long
tmMaxCharWidth As Long
tmWeight As Long
tmOverhang As Long
tmDigitizedAspectX As Long
tmDigitizedAspectY As Long
tmFirstChar As Byte
tmLastChar As Byte
tmDefaultChar As Byte
tmBreakChar As Byte
tmItalic As Byte
tmUnderlined As Byte
tmStruckOut As Byte
```

Re: Rotated Text Help Needed

```
tmPitchAndFamily As Byte
tmCharSet As Byte
End Type
Private Const TRANSPARENT = 1
Private Const OPAQUE = 2
Private Const RadToTenthDegree As Single = 572.957795
Private Const pi = 3.14159
Private myhDC As Long
Private new_font As Long, old_font As Long

Private Sub RotateFont(outDevice As Object, angle As Single)
Dim myAngle As Long
myhDC = outDevice.hdc
myAngle = angle * RadToTenthDegree ' convert from radians
Dim log_font As LOGFONT
With log_font
.IfEscapement = myAngle
.IfOrientation = myAngle
.IfHeight = outDevice.ScaleY(outDevice.Font.Size * 20, vbTwips, vbPixels)
.IfFaceName = outDevice.Font.Name & vbNullChar
If outDevice.Font.Bold = True Then
.IfWeight = 700
Else
.IfWeight = 400
End If
.IfItalic = outDevice.Font.Italic
.IfUnderline = outDevice.Font.Underline
End With
new_font = CreateFontIndirect(log_font)
old_font = SelectObject(myhDC, new_font)
End Sub

Private Sub CircleText(obj As Object, x1 As Single, _
y1 As Single, r1 As Single, s1 As String)
' add code later to check for valid object type
Dim angle As Single, p As Long, n As Long
Dim xp As Single, yp As Single, position As Single
Dim myhDC As Long, ret As Long
Dim NewFontMetrics As TEXTMETRIC
p = Len(s1)
angle = (2 * pi) / p
position = pi / 2
myhDC = obj.hdc
For n = 0 To p - 1
RotateFont obj, position
GetTextMetrics myhDC, NewFontMetrics

xp = x1 - (r1 * Sin(position))
yp = y1 - (r1 * Cos(position))
xp = obj.ScaleX(xp, vbInches, vbPixels)
yp = obj.ScaleY(yp, vbInches, vbPixels)
```

Re: Rotated Text Help Needed

```
' Adjust for variances in font cell height between individual
' characters by lining up the baselines
xp = xp - NewFontMetrics.tmAscent * Sin(position)
yp = yp - NewFontMetrics.tmAscent * Cos(position)

ret = TextOut(myhDC, xp, yp, Mid$(s1, n + 1, 1), 1)
' change the font back and get rid of the new font
SelectObject myhDC, old_font
position = position - angle
Next n
End Sub

Private Sub Command1_Click()
Me.Cls
Printer.Print
Printer.Font.Name = "Arial"
Printer.Font.Size = 32
' set to FontTransparent = False just for test purposes so
' that we can see the actual top left corner of each
' character block (note we use SetBkMode instead of using
' the Me.FontTransparent property because the
' FontTransparent property has some odd behaviour when
' printing rotated text characters)
SetBkMode Printer.hdc, OPAQUE
Me.ForeColor = vbBlack
' draw a light grey rectangle just for test purposes
Printer.ScaleMode = vbInches
Printer.FillStyle = vbFSSolid
Printer.Line (0, 0)-(8, 8), RGB(180, 180, 180), BF
Printer.FillStyle = vbFSTransparent
Dim s1 As String
s1 = "IS THIS TEXT PRINTED CORRECTLY OR IS IT NOT "
CircleText Printer, 4, 4, 3, s1
' draw a bounding circle just for test purposes
Printer.Circle (4, 4), 3
' draw a regular character just for test purposes
Printer.CurrentX = 0: Printer.CurrentY = 0
Printer.Print "I"
Printer.EndDoc
End Sub

.
```

• **References:**

◆ **Rotated Text Help Needed**

◇ From: Mike Williams

Re: Rotated Text Help Needed

- ◆ **Re: Rotated Text Help Needed**
 - ◇ From: Rick Rothstein [MVP – Visual Basic]
- ◆ **Re: Rotated Text Help Needed**
 - ◇ From: Mike Williams
- ◆ **Re: Rotated Text Help Needed**
 - ◇ From: Rick Rothstein [MVP – Visual Basic]

- Prev by Date: **Font Style!**
- Next by Date: **Re: SetFocus!**
- Previous by thread: **Re: Rotated Text Help Needed**
- Next by thread: **Re: Rotated Text Help Needed**
- Index(es):
 - ◆ **Date**
 - ◆ **Thread**