

Re: Mike: One last explanation, I hope!

## Re: Mike: One last explanation, I hope!

---

*Source:*

<http://www.tech-archive.net/Archive/VB/microsoft.public.vb.general.discussion/2006-07/msg01050.html>

---

- *From:* Randy Gardner <[RandyGardner@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:RandyGardner@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Thu, 13 Jul 2006 08:20:02 -0700
- 

Hi Mike:

In your code below the `scale.width (35.75)` and `scale.left (-3.75)` values seem logical from a traditional x,y coordinate system:

`Scale.left` sets the  $-x$  value and `scale.width` set the total line length, ( $+x = \text{Left} + \text{Width}$ )

However, with the `scale.height (-89.375)` and `scale.top (80)` make no sense.

Actually it is the `scale.top` that makes no sense.

Can you explain how the Y values (`= : 80` and `0`) work with those settings?

Heres my reasoning: with the `scale.height = -89.375` then: `Line(0, 80)-(0, 0)` would draw a line from the bottom to within 9.375 of the top. I reason that windows adds the line values to the `scale.height` to get the location? If that is true

what is the significants of `scale.top` at `+80`?

```
'  
' set up the  
graphWidth = 30: gridWidth = 5  
leftBorder = graphWidth / 8 ' allow some space at left side  
rightBorder = graphWidth / 15 ' not too close to right edge  
Picture1.ScaleWidth = graphWidth + leftBorder + rightBorder  
graphHeight = 75: gridHeight = 5  
bottomBorder = graphHeight / 8 ' allow some space at the bottom  
topBorder = graphHeight / 15 ' not too close to top edge of window  
Picture1.ScaleHeight = -(graphHeight + bottomBorder + topBorder)  
Picture1.ScaleTop = -Picture1.ScaleHeight - bottomBorder  
Picture1.ScaleLeft = -leftBorder  
'
```

—  
Thank you,

Randy

Re: Mike: One last explanation, I hope!

Re: Mike: One last explanation, I hope!

"Mike Williams" wrote:

"Randy Gardner" <RandyGardner@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message  
[news:969E674F-D52D-4CDB-9B9E-4132E40E4102@xxxxxxxxxxxxxxxxxxxx](mailto:news:969E674F-D52D-4CDB-9B9E-4132E40E4102@xxxxxxxxxxxxxxxxxxxx)

As an example: I could set the display to Scalemode = vbInches and in software scale my data to fit the physical limits of the display when the max value exceeds the limits of the display which is what I have done in the past. From my reading and your code example: ScaleMode = vbUser should allow me to make a scale for any physical display size that has limits that are the worst case values of my data.

It does. That is the purpose of User scalemodes. If you use a fixed units scale, such as inches or centimeters for example, and if you scale your data so that your graph displays properly in a Form (or other object) of a specific size, then as soon as you change the size of the object your graph will either be too small or too large to display the data properly in the object and you will have to change your scaling accordingly. If however you instead use a User scalemode, the changes in the actual size of the object will not present a problem. This is because when you specify a User ScaleWidth (or ScaleHeight) of a specific value you are causing the object to contain the specified number of units whatever its physical size is, and if something happens to change the physical size of the object (increase it, for example) then the number of units will remain the same but the actual size of each unit will increase. In other words, your graph will display correctly whatever its physical size. Also, using a User scale mode it is very easy to have units of one size in a horizontal direction and another completely different size in the vertical direction. For example, suppose you have values which range from -3000 to +3000 in the horizontal direction and from -500 to +500 in the vertical direction, the following code will set up Form to display a typical "four quadrant" graph suitable for those values, with location (0, 0) being at the junction of the four quadrants:

```
Dim xmin As Single, xmax As Single
Dim ymin As Single, ymax As Single
xmin = -3000: xmax = 3000
ymin = -500: ymax = 500
Me.Scale (xmin, ymin)-(xmax, ymax)
```

In such a graph, positive x coordinates will move towards the right and negative coordinates will move towards the left. This is of course what you normally want. However, as far as the y coordinates are concerned the graph will follow the normal VB convention whereby positive values run in a downward direction and negative values in an upward direction. In a standard four quadrant graph of course you usually want the opposite of that. This is a problem that is very easy to fix though. All you need to do is swap the positions of ymin and ymax in the above code so that instead of the

Re: Mike: One last explanation, I hope!

Re: Mike: One last explanation, I hope!

code shown above the Scale code becomes:

```
Me.Scale (xmin, ymax)-(xmax, ymin)
```

The actual values of course can be anything that suits your data, and it is perfectly possible to have a different range of negative vales than positive values, in either or both directions, so that a range of X (-20 to +50 ) and Y ( -30 to + 80) can easily be accommodated simply by using those values in the xmin, xmax, ymin and ymax variables.

As with most things such as this, and example is worth a thousand words. Here's some code you can paste into a VB Form. It sets up a four quadrant graph exactly as described above, with the quadrants junction being offset from the centre in such a way as to accommodate exactly the range of data specified. Just paste the code into a standard VB Form (BorderStyle Sizable and WinmindowState Normal). Run the project and the graph quadrant lines will be displayed together with four differently coloured diagonal lines representing the maximum and minimum data values. Notice that when you drag the Form the graph remains exactly the same, except that it expands and contracts to exactly fill the Form. Is that the sort of thing you are after?

By the way, you should note that the ConvertX and pConvertY stuff in the code I posted previously was needed because that code was designed to print such a graph to various devices (Forms, picture Boxes and Printers) that will normally have been set to a scale mode that suits the user (inches or whatever) so that he can draw other stuff on the same page. The pConvertX and pConvertY functions were needed in order to produce the appropriate conversion from "graph coordinates" to "output device coordinates". That is why the code was more complex than the code shown above, which merely deals with just the graph display itself, which is of course displayed in a User ScaleMode container.

Post again to let me know if this stuff helps you any further, or if you have any other questions.

Mike

Option Explicit

```
Private Sub Form_Load()  
Me.Move 0, 0  
End Sub
```

```
Private Sub Form_Resize()  
Dim xmin As Single, xmax As Single  
Dim ymin As Single, ymax As Single  
xmin = -20: xmax = 50  
ymin = -30: ymax = 80  
Me.Scale (xmin, ymax)-(xmax, ymin)  
Me.BackColor = vbWhite  
Me.Cls
```

Re: Mike: One last explanation, I hope!

```
' draw the quadrant lines
Me.Line (xmin, 0)-(xmax, 0), vbBlack
Me.Line (0, ymin)-(0, ymax), vbBlack
' draw a red line in the top right quadrant
' (x and y both positive)
Me.Line (0, 0)-(50, 80), vbRed
' draw a blue line in the bottom left quadrant
' (x and y both negative)
Me.Line (0, 0)-(-20, -30), vbBlue
' draw a cyan line in the top left quadrant
' (x negative, y positive)
Me.Line (0, 0)-(-20, 80), vbCyan
' draw a green line in the bottom right quadrant
' (x positive, y negative)
Me.Line (0, 0)-(50, -30), vbGreen
End Sub
```