Session E-GRAF

Deep Inside MSGraph

Ted Roche
Blackstone Incorporated

Overview

"A picture is worth a thousand words," and your graphs can be so rich that they require a thousand words to describe. This session covers the basics of how graphs can be created, updated and output using Visual FoxPro and MSGraph.

Working with MSGraph in Visual FoxPro

Why use graphs? Graphs provide a better portrayal of information. The old saw that "a picture worth a thousand words" can be true - far richer information can be conveyed with a single graph that paragraphs of data. In some cases, a single column of data can be better expressed in words - you should have to use a thousand words to describe a picture! Use graphs to show trends, patterns, relationships, multiple layers of information and how they correspond.

Manual / Interactive use

The chart above illustrates that MSGraph is capable of far more than simple line and pie charts. It is possible to produce sophisticated charts - those with two-axes, of two different types - bar & line. In addition, it is possible to manually add notations, text boxes and arrows. Your imagination is the only limit. Experiment with the MSGraph interface directly to see the kinds of effects which can be created.

Invoking the Wizard through code

To create the simplest graphs, or to get started on a more sophisticated chart, the Wizard is a good place to start. You don't need to work through the Wizard interface each time, however. The graph Wizard may be invoked through code:

```vfp
DO (_GENGRAPH) WITH <parm1>,<parm2> ...[,<parm9>]
  * parm1 - "AUTOGRAPH" & required
  * parm2 - chart type (number)
  * parm3 - chart subtype (number)
```

http://www.dfpug.de/konf/konf_1996/ole/e_graf/default.htm

10/7/2006
Creating a graph through code

1. Use the APPEND GENERAL ... DATA ... CLASS "MSGraph" command to create a general field containing a graph.
2. Create a rectangular data set: the DATA clause should pass a string containing the values to be graphed. Columns are separated with tabs ,CHR(09), and rows terminated with carriage returns CHR (13). The first row contains the column headings, the first column, the row labels.
3. In order to use OLE Automation on the general field, you must create an object reference to the field. Do this using an OLEBoundControl on a form. The form need never appear - it can be used in the background.
4. Update the data by issuing another APPEND GENERAL - your chart formatting is retained. Issue APPEND GENERAL with no data to blank the graph.
5. Manipulate the OLEBoundControl - using the DoVerb() method and the properties AutoActivate, AutoSize & Sizable, OLELCID, AutoVerbMenu (new to 5.0), Hostname,.
6. Display the field using an OLEBoundControl, @...SAY, or MODIFY GENERAL
7. Print the graph using @...SAY, or (preferred) an OLEBoundControl on reports or labels.
8. Use In-Place Editing to allow the user to manipulate the graph directly: IPE replaces Edit, View, Insert, Format, Tools, Data and help menu pads. Retains VFP File, Program and Window menus.
9. Measurements: fellow MVP George Sexton explains the measurement system used internally for MSGraph objects use twips, rather than pixels or ruler measurements.

Object Model

The object model of MSGraph is not clear in the documentation which is supplied with Visual FoxPro. However, it is a bonus that VFP 5.0 ships with the help file VBA_GRP.HLP, which accurately documents many of the properties and methods of the graph object. What's missing from the help file are the constants declarations, included on your disks as MSGRAPH5.H

You can examine the internal structure of the MSGraph object, as well as many other OLE controls and servers, using tools often called "OLE Snoopers." A simple one is available from Microsoft on their MSDD and Win32 SDK, as well as on their web site:

Ken Lassesen had an excellent article in the October 1995 issue of the Microsoft Developer's Library, " Mapping the MSGraph Object," where he charts out the entire MSGraph object. Thanks to Ken and Microsoft for permission to reproduce that map here!
Attached are my notes on some of the properties in the MSGraph object which I've explored and tested. There's much more left to go, but I hope these notes (and the object map above) will give you a good start. Please let me know what discoveries you make. Happy hacking!

**Common Properties - used by many object**

Creator - returns a (to me) meaningless number, probably some internal OLE tag.
Parent - points to parent object
Application - points to MSGraph
ColorIndex Range 1 to 56 - points to internal color palette, displayed in many places as a 7 x 8 grid. Other options include -4105, xlAutomatic, which allows MSGraph to select the appropriate color. Those colors can be customized through the interface, but I haven't found access programmatically.

**Application Object**

- Contained objects
  - Chart

- Methods
  - Quit - watch out! Closes VFP!
  - SaveAsOldFileFormat([Major],[Minor]) OLE Error "Unknown name"
  - ChartWizardDisplay?
  - Creator returns numeric 12973....0000 ?
  - HasLinks - does not appear to work? OLE Error "Unknown name"
  - Parent - reference to VFP
  - Visible - logical
  - Property Name - returns "Microsoft Visual FoxPro" for OLEBoundControl and "Microsoft Graph" for OLEBoundControl.Object

**Font Object**

Application - pointer to MSGraph
Background - Numeric,
Bold - Logical
Color - RGB Value
FontStyle - text, "Bold," "Bold Italic," etc.
Italic - logical
Name - fontname, "Arial"
OutlineFont - unsure - does this say whether font is outline-able, or allow an outline effect?
Shadow - logical, creates a shadow effect on font
Size - font point size
Strikethrough - logical
Subscript - logical
Superscript - logical
Underline - logical

**Border Object**

Application - pointer to MSGraph
Color - RGB Value
ColorIndex - see ColorIndex comments
Creator - see Creator, above
LineStyle - solid, dashed, or varied shades of grey
Weight - line thickness

**Interior Object**

Application - pointer to MSGraph
Color - RGB Value
ColorIndex
Creator - see Creator, above
InvertIfNegative - logical
Parent - see above
Pattern - a built-in set of diagonal and hashed patterns
PatternColor - RGB Value
PatternColorIndex - see ColorIndex comments

**Corners Object**
This one is not a true object, but rather an internal object which is inadvertently exposed. Avoid it.

**SeriesLines Object**

Application - pointer to MSGraph
Delete()
Border - points to a Border Object, above
Creator - see Creator, above
Name
Parent - see above

**HiLoLines Object**

Application - pointer to MSGraph
Delete()
Border - points to a Border Object, above
Creator - see Creator, above
Name
Parent - see above

**GridLines Object**

Application - pointer to MSGraph
Delete()
Border - points to a Border Object, above
Creator - see Creator, above
Name
Parent - see above

**DropLines Object**

Application - pointer to MSGraph
Delete()
Border - points to a Border Object, above
Creator - see Creator, above
Name
Parent - see above

**MSGRAPH5.H - the header file for MSGraph constants**

* MSGraph5.H
* FoxPro header for MSGraph manipulation
* Generated 13-Sept-96 by OLE2View

```c
#define xl3DArea 4098
#define xl3DBar 4099
#define xl3DColumn 4100
#define xl3DLine 4101
#define xl3DPie 4102
#define xl3DSurface 4103
#define xlArea 1
#define xlAutomatic 4105
#define xlBar 2
#define xlBoth 1
#define xlBottom 4107
#define xlBuiltIn 0
#define xlCap 1
#define xlCategory 1
#define xlCenter 4108
#define xlChecker 9
#define xlCircle 8
#define xlColumn 3
#define xlColumns 2
#define xlCombination 4111
#define xlContinuous 1
#define xlCorner 2
```
#DEFINE xlCrissCross 16
#DEFINE xlCross 4
#DEFINE xlCustom -4114
#DEFINE xlDash -4115
#DEFINE xlDashDot 4
#DEFINE xlDashDotDot 5
#DEFINE xlDefaultAutoFormat -1
#DEFINE xlDiamond 2
#DEFINE xlDistributed -4117
#DEFINE xlDot -4118
#DEFINE xlDouble -4119
#DEFINE xlDoubleAccounting 5
#DEFINE xlDoughnut -4120
#DEFINE xlDown -4121
#DEFINE xlDownward -4170
#DEFINE xlExponential 5
#DEFINE xlFixedValue 1
#DEFINE xlGray16 17
#DEFINE xlGray25 -4124
#DEFINE xlGray50 -4125
#DEFINE xlGray75 -4126
#DEFINE xlGray8 18
#DEFINE xlGrid 15
#DEFINE xlHairline 1
#DEFINE xlHigh -4127
#DEFINE xlHorizontal -4128
#DEFINE xlInside 2
#DEFINE xlInterpolated 3
#DEFINE xlJustify -4130
#DEFINE xlLeft -4131
#DEFINE xlLightDown 13
#DEFINE xlLightHorizontal 11
#DEFINE xlLightUp 14
#DEFINE xlLightVertical 12
#DEFINE xlLine 4
#DEFINE xlLinear -4132
#DEFINE xlLogarithmic -4133
#DEFINE xlLow -4134
#DEFINE xlMaximized -4137
#DEFINE xlMaximum 2
#DEFINE xlMedium -4138
#DEFINE xlMinimum 2
#DEFINE xlMinusValues 3
#DEFINE xlMovingAvg 6
#DEFINE xlNextToAxis 4
#DEFINE xlNoCap 2
#DEFINE xlNone -4142
#DEFINE xlNormal -4143
#DEFINE xlNotPlotted 1
#DEFINE xLOpaque 3
#DEFINE xLOutside 3
#DEFINE xLPercent 2
#DEFINE xLPicture -4147
#DEFINE xlPie 5
#DEFINE xlPlus 9
#DEFINE xlPlusValues 2
#DEFINE xLPolynomial 3
#DEFINE xLPower 4
#DEFINE xLPrimary 1
#DEFINE xLRadar -4151
#DEFINE xLRight -4152
#DEFINE xLRows 1
#DEFINE xLScale 3
#DEFINE xLSecondary 2
#DEFINE xLSemiGray75 10
#DEFINE xLSeries 3
#DEFINE xLSmallGray 4
#DEFINE xLSmallGrayAndPercent 5
#DEFINE xLSmallPercent 3
#DEFINE xLSmallValue 2
#DEFINE xLSingle 2
#DEFINE xLSingleAccounting 4
#DEFINE xlSolid 1
#DEFINE xlSquare 1
#DEFINE xlStDev -4155
#DEFINE xlStError 4
#DEFINE xlStack 2
#DEFINE xlStar 5
#DEFINE xlStretch 1
#DEFINE xlThick 4
#DEFINE xlThin 2
#DEFINE xlTop -4160
#DEFINE xlTransparent 2
#DEFINE xlTriangle 3
#DEFINE xlUp -4162
#DEFINE xlUpward -4171
#DEFINE xlValue 2
#DEFINE xlVertical -4166
#DEFINE xlWizardDisplayAlways 1
#DEFINE xlWizardDisplayDefault 0
#DEFINE xlWizardDisplayNever 2
#DEFINE xlX -4168
#DEFINE xlXYScatter -4169
#DEFINE xlY 1
#DEFINE xlZero 2

Bibliography

Microsoft Developer Library, Randy Brown, "Extending the Visual FoxPro Wizards"

Microsoft Developer Library, Ken Lassesen, "Mapping the MSGraph Object"

Microsoft Developer Library, Ken Lassesen, "Using Microsoft OLE Automation Servers to Develop Solutions"

Microsoft Knowledgebase articles - available via CompuServe (GO MSKB) and on the Microsoft web site (http://www.microsoft.com/kb)

Q129533 - "How to Pass Data to Microsoft Graph Programmatically"

Q131029 - "How to Manipulate Embedded Objects in General Fields"

Q135348 - "How to Update a Graph in a Form"

VBA_GRP.HLP: Help file for using Visual Basic for Applications with MSGraph - should be included with VFP 5.0 (not yet confirmed), installed in <Windows Dir>\MSAPPS\Graph5
