Using RMChart with Xbase++

Introduction

Xbase++ 1.9 and 2.0 have full support for Active-X and this capability can be used to produce graphs and charts using a FREE Active-X control named RMChart. End user application screens and reports can be given new life and improved visual appeal with only a few hours or a few days of work using this robust graphing control.

This seminar discusses the basics of charting and graphing such as:

* Non Grid-Based Charts

  * Pie Charts
  * Donut Charts
  * Pyramid Charts

* Grid-Based Charts

  * Single Bars
  * Double Bars
  * Stacked Bars
  * Floating Bars
  * Simple Line Graphs
  * Indexed Line Graphs
  * Bars and Lines

Included in this seminar is the RMChart Control and documentation and several complete new classes with source code and sample programs that are ready to be used in your Xbase++ or eXpress++ programs with no other software required. The download of the RMChart control is available at http://rmchart.softpile.com. The author's name is Rainer Morgan and the license is freeware.

There are code samples that are included with RMChart, however none of these are Xbase++ code. The code samples that most closely resemble Xbase++ are those written in VB. Included on the conference disk is a sample program named RMNATIVE.PRG. This is an example of how to use RMChart in native mode, i.e. no special classes are required. Unfortunately, code like this is rather difficult to write, debug and maintain in Xbase++.

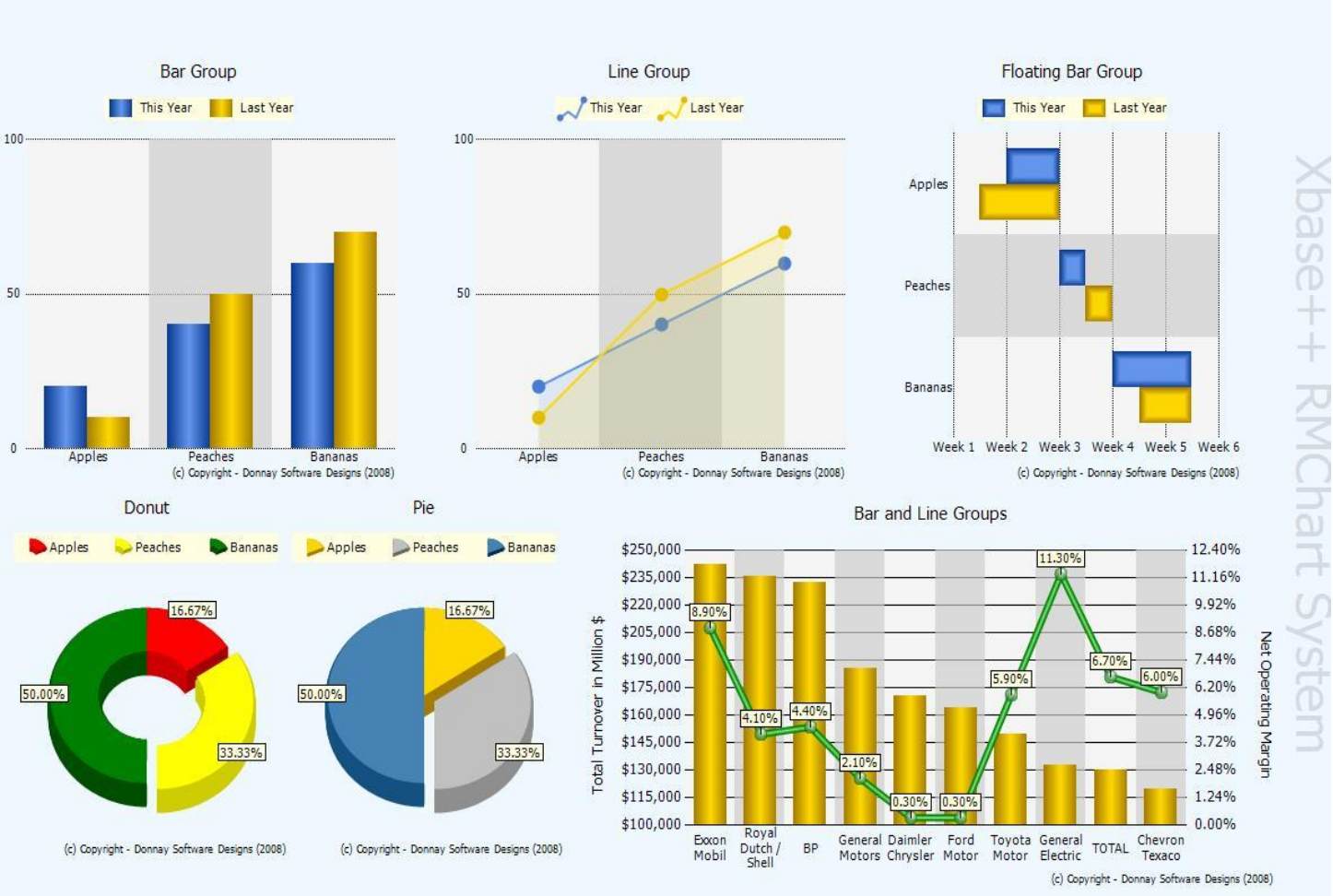
The code that is included is designed to provide a more abstracted and simpler way to create graphs and charts than writing in native mode.

The Elements of a Chart

A chart consists of the main drawing area and up to 5 regions. Each region may contain a single gridless chart or a mix of line groups and bar groups. Each bar group or line group region may have a grid, a caption, a legend, multiple data axes, a label axis and a footing. Each gridless group may have a caption, a legend and a footing. All graph elements have configurable colors. Bar groups and line groups also have configurable presentations such as 3D, gradient, vertical, horizontal, etc. Text elements have configurable colors and fonts.

Examples of RMChart screens

The below screen was written in Xbase++ using the RMChart ActiveX control and the RMChart and RMChartRegion classes which are included in RMCCLASS.PRG. The sample program that uses the new classes is contained in RMCHART.PRG. Just run PBUILD RMCHART and then run RMCHART.EXE to produce the below screen on your computer. This chart include 5 regions on a single ActiveX object.



Installing RMChart

The RMChart system includes an RMCHART.OCX and RMCHART.DLL. For compatibility with Windows 7 and 8 these files must be dated 01/12/2008 or later. The RMChart installation program will install these files in the \Windows\System32 or \Windows\SysWOW64 folder and register the ActiveX control. It will also install the RMChart documentation into a folder of your choosing.

It is not necessary to include the RMChart install program with end-user applications. Instead, it is recommended that the RMCHART.OCX and RMCHART.DLL files be included in the same folder as the application runtime files and that the application perform the registration of the OCX the first time that the application is executed. RMCCLASS.PRG contains the RMChart class. When an object is instantiated from this class, the Init() method automatically registers RMChart.OCX if it has not been previously installed.

Programming with RMChart

It pays to read the RMChart documentation, even if you do not intend to write in native code. This will help you understand the basics of the control and become acquainted with its instance variables, and its methods. Read the section that describes the ActiveX (rmchart.ocx) because this is what is used in all the programs included with this topic.

It also may be beneficial to look at the source code in RMNATIVE.PRG because this is how programming would be accomplished without the benefit of a more productive and more forgiving class. On the other hand, it is much more advantageous to use the RMChart and RMChartRegion classes included in RMCCLASS.PRG and the commands included in RMCHART.CH.

For example, the below Native Code is what would be required if writing a combination Line and Bar chart using the ***native*** method. The drawn chart is equivalent to the **Bar and Line Groups** chart in the above picture. There are 80 lines of very complicated code that are required to render this graph. Moreover, it is very easy to make syntax errors or otherwise fail to set a value correctly thus making debugging more complicated. The code that uses the ***RmChartRegion*** class requires far fewer lines of code and it is much easier to write and to debug because the class insures that everything is properly formatted to work with RmChart. This abstracted code produces the same chart as the native code.

Native Code (from RMNATIVE.PRG)

The below code was only slightly modified from the Visual Basic sample that is included with RMChart. There are minor changes between VB code and Xbase++ code when using ActiveX iVars and methods.

```
RMChartX1:Reset()
//************** Design the chart **********************
RMChartX1:RMCBackColor := CornflowerBlue
RMChartX1:RMCStyle := RMC_CTRLSTYLEFLATSHADOW
//************** Add Region 1 ****************************
RMChartX1:AddRegion()
```

```
RMChartX1:Region(1):Left := 5
RMChartX1:Region(1):Top := 10
RMChartX1:Region(1):Width := -5
RMChartX1:Region(1):Height := -5
RMChartX1:Region(1):Footer := "data source: F.A.Z"
//************** Add caption to region 1 ******************
RMChartX1:Region(1):AddCaption()
RMChartX1:Region(1):Caption():Titel := "The world's 10 biggest industrial
companies 2003"
RMChartX1:Region(1):Caption():BackColor := Coral
RMChartX1:Region(1):Caption():TextColor := FloralWhite
RMChartX1:Region(1):Caption():FontSize := 12
RMChartX1:Region(1):Caption():Bold := True
//************** Add grid to region 1 ****************************
RMChartX1:Region(1):AddGrid()
RMChartX1:Region(1):Grid():BackColor := Azure
RMChartX1:Region(1):Grid():AsGradient := True
RMChartX1:Region(1):Grid():Left := 0
RMChartX1:Region(1):Grid():Top := 0
RMChartX1:Region(1):Grid():Width := 0
RMChartX1:Region(1):Grid():Height := 0
//************** Add data axis to region 1 ****************************
RMChartX1:Region(1):AddDataAxis()
RMChartX1:Region(1):DataAxis(1):Alignment := RMC_DATAAXISLEFT
RMChartX1:Region(1):DataAxis(1):MinValue := 100000
RMChartX1:Region(1):DataAxis(1):MaxValue := 250000
RMChartX1:Region(1):DataAxis(1):TickCount := 11
RMChartX1:Region(1):DataAxis(1):FontSize := 9
RMChartX1:Region(1):DataAxis(1):TextColor := Cornsilk
RMChartX1:Region(1):DataAxis(1):LineColor := Black
RMChartX1:Region(1):DataAxis(1):LineStyle := RMC_LINESTYLESOLID
RMChartX1:Region(1):DataAxis(1):DecimalDigits := 0
RMChartX1:Region(1):DataAxis(1):AxisUnit := "$ "
RMChartX1:Region(1):DataAxis(1):AxisText := "Total turnover in Mill. Dollar"
//************** Add second data axis to region 1 ****************************
RMChartX1:Region(1):AddDataAxis()
RMChartX1:Region(1):DataAxis(2):Alignment := RMC_DATAAXISRIGHT
RMChartX1:Region(1):DataAxis(2):MinValue := 0
RMChartX1:Region(1):DataAxis(2):MaxValue := 0
RMChartX1:Region(1):DataAxis(2):DecimalDigits := 2
RMChartX1:Region(1):DataAxis(2):AxisUnit := " %"
RMChartX1:Region(1):DataAxis(2):AxisText := "Net operating margin"
//************** Add label axis to region 1 ****************************
RMChartX1:Region(1):AddLabelAxis()
RMChartX1:Region(1):LabelAxis():AxisCount := 1
RMChartX1:Region(1):LabelAxis():TickCount := 10
RMChartX1:Region(1):LabelAxis():Alignment := RMC_LABELAXISBOTTOM
RMChartX1:Region(1):LabelAxis():FontSize := 8
RMChartX1:Region(1):LabelAxis():TextColor := Cornsilk
RMChartX1:Region(1):LabelAxis():TextAlignment := RMC_TEXTCENTER
RMChartX1:Region(1):LabelAxis():LineColor := Black
RMChartX1:Region(1):LabelAxis():LineStyle := RMC_LINESTYLESOLID
sTemp := "Exxon Mobil*Royal Dutch / Shell*BP*General Motors*Daimler Chrysler*Ford
Motor*Toyota Motor*"
sTemp := sTemp + "General Electric*TOTAL*Chevron Texaco"
RMChartX1:Region(1):LabelAxis():LabelString := sTemp
//************** Add Series 1 to region 1 ****************************
```

```
RMChartX1:Region(1):AddBarSeries()
RMChartX1:Region(1):BarSeries(1):SeriesType := RMC_BARSINGLE
RMChartX1:Region(1):BarSeries(1):SeriesStyle := RMC_BAR_FLAT_GRADIENT2
RMChartX1:Region(1):BarSeries(1):Lucent := False
RMChartX1:Region(1):BarSeries(1):Color := Gold
RMChartX1:Region(1):BarSeries(1):Horizontal := False
RMChartX1:Region(1):BarSeries(1):WhichDataAxis := 1         // ' connect this
series to the left data axis
RMChartX1:Region(1):BarSeries(1):ValueLabelOn := 0
RMChartX1:Region(1):BarSeries(1):PointsPerColumn := 1
RMChartX1:Region(1):BarSeries(1):HatchMode := RMC_HATCHBRUSH_OFF
//****** Set data values ******
sTemp := "242365*235598*232571*185524*170457*164196*149321*132797*130067*119703"
RMChartX1:Region(1):BarSeries(1):DataString := sTemp
//************** Add Series 2 to region 1 ******************************
RMChartX1:Region(1):AddLineSeries()
RMChartX1:Region(1):LineSeries(1):SeriesType := RMC_LINE
RMChartX1:Region(1):LineSeries(1):SeriesStyle := RMC_LINE_CABLE
RMChartX1:Region(1):LineSeries(1):LineStyle := RMC_LSTYLE_LINE
RMChartX1:Region(1):LineSeries(1):Lucent := False
RMChartX1:Region(1):LineSeries(1):Color := Green
RMChartX1:Region(1):LineSeries(1):SymbolStyle := RMC_SYMBOL_BULLET
RMChartX1:Region(1):LineSeries(1):WhichDataAxis := 2         // connect this
series to the right data axis
RMChartX1:Region(1):LineSeries(1):ValueLabelOn := 1
RMChartX1:Region(1):LineSeries(1):HatchMode := RMC_HATCHBRUSH_OFF
//****** Set data values ******
sTemp := "8.9*4.1*4.4*2.1*.3*.3*5.9*11.3*6.7*6"
RMChartX1:Region(1):LineSeries(1):DataString := sTemp
```

Code using RmChartRegion Class (From RMCHART.PRG)

The below code uses a new Xbase++ class name RmChartRegion and a set of commands included in RMCHART.CH that make it simpler to define chart regions.

```
oRegion6 := RmChartRegion():new( oRmChart, {520,420}, {610,320} )

AddBarGroup TO oRegion6:barGroupArray ;
 DATA { 242365,235598,232571,185524,170457,164196,149321,132797,130067,119703 } ;
 TYPE RMC_BARSINGLE COLOR Gold ;
 STYLE RMC_BAR_FLAT_GRADIENT2 ;
 WHICHDATAAXIS 1

AddLineGroup TO oRegion6:lineGroupArray ;
 DATA { 8.9, 4.1, 4.4, 2.1, .3, .3, 5.9, 11.3, 6.7, 6 } ;
 WHICHDATAAXIS 2 ;
 STYLE RMC_LINE_CABLE ;
 COLOR Green ;
 LINESTYLE RMC_LSTYLE_LINE ;
 SYMBOLSTYLE RMC_SYMBOL_BULLET ;
 VALUELABEL 1

AddDataAxis TO oRegion6:dataAxisArray ;
 AXISTEXT "Total Turnover in Million $"  ;
 ALIGN RMC_DATAAXISLEFT  ;
```

```
 MINVALUE 100000 MAXVALUE 250000 ;
 TICKCOUNT 11 FONTSIZE 9 ;
 TEXTCOLOR Black LINECOLOR Black ;
 LINESTYLE RMC_LINESTYLESOLID ;
 DECDIGITS 0 UNIT '$'

AddDataAxis TO oRegion6:dataAxisArray ;
 AXISTEXT "Net Operating Margin"  ;
 ALIGN RMC_DATAAXISRIGHT ;
 TEXTCOLOR Black LINECOLOR Black ;
 MINVALUE 0 MAXVALUE 0 ;
 DECDIGITS 2 UNIT " %"

aLabelText := { "Exxon Mobil", "Royal Dutch / Shell","BP", "General Motors", ;
   "Daimler Chrysler", "Ford Motor", "Toyota Motor", "General Electric", ;
    "TOTAL","Chevron Texaco" }

AddLabelAxis TO oRegion6:labelAxisArray LABELARRAY aLabelText ;
  ALIGN RMC_LABELAXISBOTTOM

oRegion6:captionArray := {'Bar and Line Groups'}
oRegion6:create()
```

RMChart.CH

The RMCHART.CH file contains all the manifest constants (enumerations). There are defines for colors, alignment options, chart styles, symbol styles, line styles, bar styles, etc. These enumerations are identical to those required by the RMChart ActiveX control.

There are also some special commands for building definition arrays that are passed on to RMChartRegion. This class uses several arrays to define the characteristics of a bar group, line group, pie, etc. Such arrays can be created in a variety of ways in Xbase++ code, but the simplest and most readable is often a command with parameters.

Here is an example of a command definition in RMChart.CH:

```
#command ADDBARGROUP [TO <aBarGroup>]                                    ;
         [COLOR <anColor>] [DATA <aData>] [HATCHMODE <nHatchMode>]         ;
         [<h:HORIZONTAL>] [<l:LUCENT>] [POINTS <nPoints>] [STYLE <nStyle>]  ;
         [TYPE <nType>] [VALUELABEL <nValueLabel>] [WHICHDATAAXIS <nAxis>]  ;
=>;
AAdd(<aBarGroup>, { <aData>, <anColor>, <nHatchMode>, <.h.>, <.l.>,         ;
                    <nPoints>, <nStyle>, <nType>, <nValueLabel>, <nAxis> } )
```

Here is how the command would be used in an application :

```
oRegion5 := RmChartRegion():new( oRmChart, {800,50}, {300,350} )

AddBarGroup TO oRegion5:barGroupArray DATA { 20,20,40,10,60,30 } ;
   TYPE RMC_FLOATINGBARGROUP COLOR CornflowerBlue HORIZONTAL ;
   STYLE RMC_BAR_HOVER
```

Custom Tooltips with RMChart

RMChart has a tooltip system that can be enabled simply as follows:

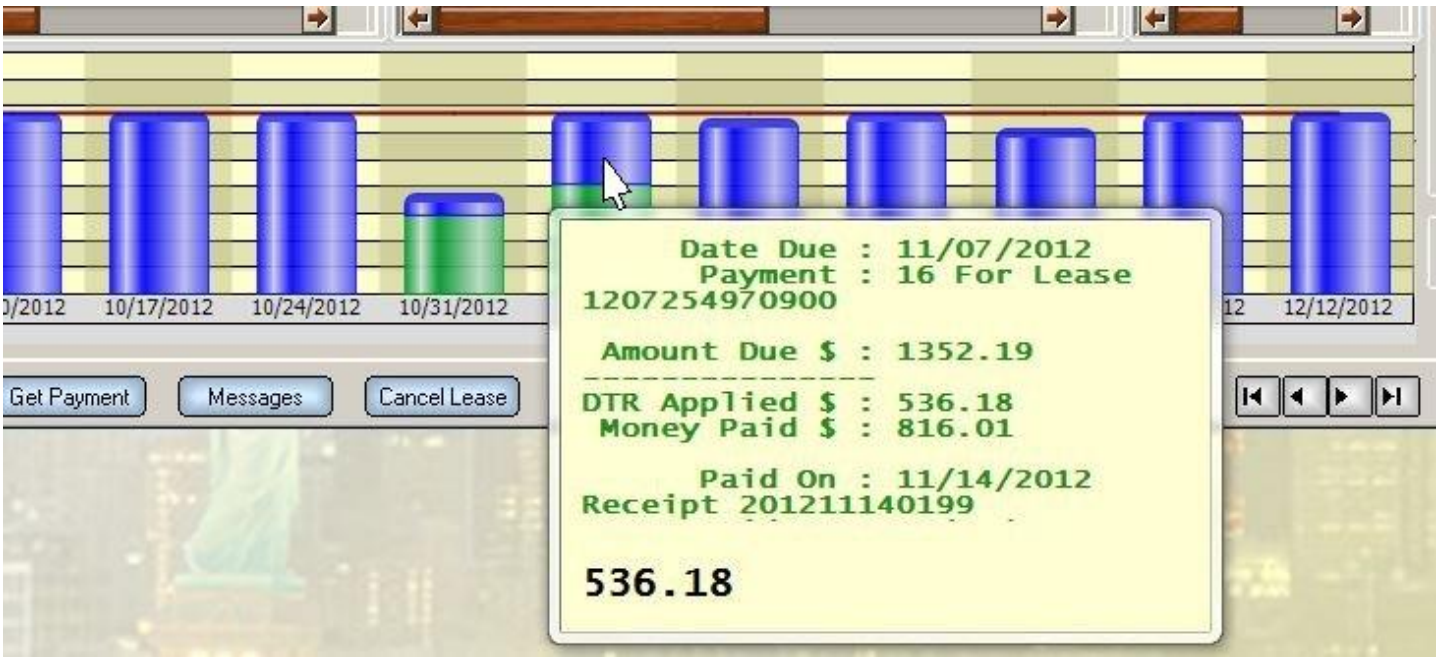
```
oRmChart:RMCToolTipWidth := 100
```

This will cause a tooltip to appear as the mouse is moved over bars or markers. The tooltip simply displays the exact value that is being charted.

A much more robust tooltip system can be created by using the **oRmChart:mouseOver** callback. This allows a code block to be called when the mouse is moved over a chart object. The below code is from the RMCHART.PRG sample program. It calls the **RMChart:showToolTip()** method to display a window that is created in Xbase++ code.

```
oRmChart:mouseMove := ;
{|nMouseButton,b,nX,nY,aData|oRMChart:showToolTip( nMouseButton, nX, nY, aData )}
```

The below picture has been captured from an application program (Medallion) to show a practical usage of this capability.

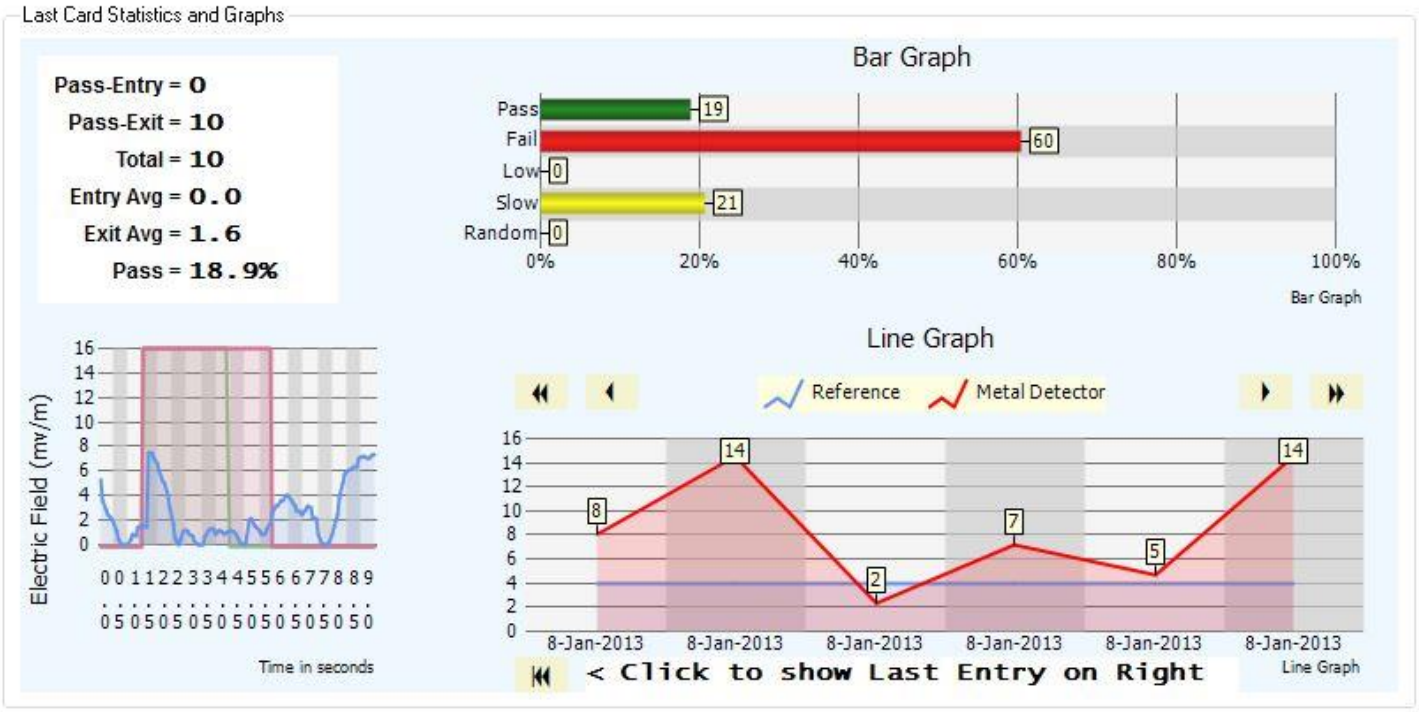


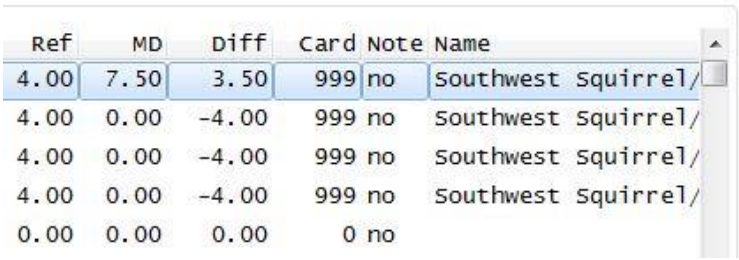
RMChart Applications

Most Xbase++ programs are business applications and so most charts will be used to display information about sales or other financial information as shown in the first picture at the beginning of this text. RMChart is not limited in its capability to business charts and graphs, on the contrary its performance makes it ideal for monitoring activity in real time or for charting many different kinds of data.

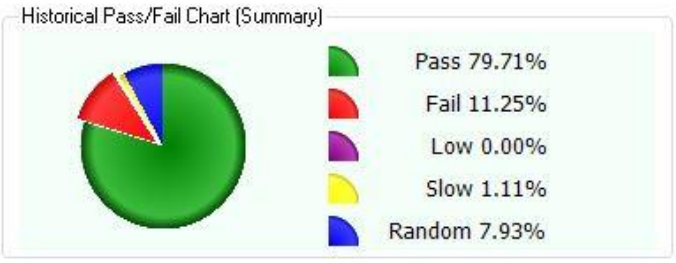
Using Graphs in a Dashboard

The below screen is a snapshot from an application that is used to control a metal detector portal that is used in gold mines, jewelry shops, etc. Employees must pass through the portal to gain access. The graph displays information about the pass/fail history for the employee (as a bar chart), a snapshot of the metal detector electric field as the employee passed through the arch (as a line chart), a scrollable history of readings for said employee (as a line chart), and a historical summary for all employees (as a pie chart).





Ref	MD	Diff	Card	Note	Name
4.00	7.50	3.50	999	no	Southwest Squirrel/
4.00	0.00	-4.00	999	no	Southwest Squirrel/
4.00	0.00	-4.00	999	no	Southwest Squirrel/
4.00	0.00	-4.00	999	no	Southwest Squirrel/
0.00	0.00	0.00	0	no	



Using Graphs for Real Time Monitoring

The metal detector application also has requirements for monitoring the portal electric field in real time and saving each time slice to a database to be reviewed at a later date and time. RMChart and Xbase++ perform very well in this regard. Displaying a line graph that is continuously scrolling requires updating arrays and rendering the graph several times per second. The MONITOR.PRG sample program shows how to do this.

The below pictures show how the graph is used to scroll live data and also to playback that data by synchronizing a browse of the data with a line graph of the data.

Walk through the metal detector to get sample readings
Click [Reset] to reset the Maximum reading

raw
1188

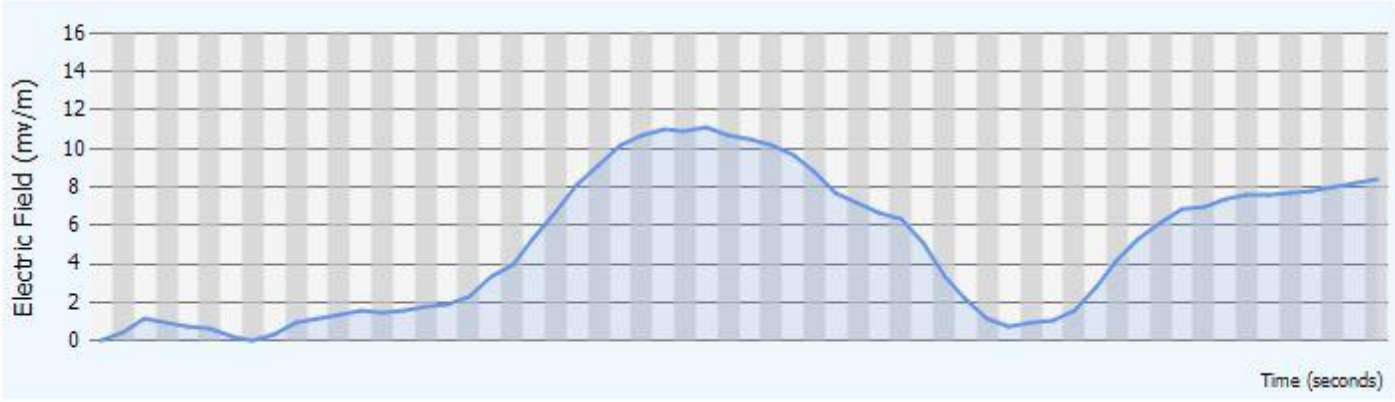
max
2047

Current
8.4

Maximum
14.5

Exit

Reset



MD	Date	Time	Interval	Pir On
5.7	07/30/13	10:08:10	0	N
4.8	07/30/13	10:08:10	0	N
4.0	07/30/13	10:08:10	0	N
3.6	07/30/13	10:08:11	0	N
13.8	07/30/13	10:08:11	0	N
14.5	07/30/13	10:08:11	0	N
14.5	07/30/13	10:08:11	0	N
14.5	07/30/13	10:08:12	0	N
12.6	07/30/13	10:08:12	0	N
10.9	07/30/13	10:08:12	0	N
10.0	07/30/13	10:08:12	0	N
8.2	07/30/13	10:08:13	0	N

Start Date 04/19/13 ...
End Date 07/30/13 ...

Start Time 00:00:00
End Time 23:59:59

Min MD Reading 0.0

Interval (Seconds) 0.0

Set Date / Min MD / Interval

Prev READ (F3) | Next READ (F4)

Show All Records

Prev Min (F5) | Next Min (F6)

Delete Records for Date Range

Prev Hr (F7) | Next Hr (F8)

Delete All Records

Prev Day (F9) | Next Day (F10)

Browse an Archive

Go to Date/Time

Using the RMChart Designer

RMChart includes a handy design tool, the **RMCDesigner**, that can be used to design a new chart or to modify the design of a chart that was created in code or by the designer. The designer reads *.RMC files. Such files can be created in code as follows:

```
oRmChart:writeRMCFile( cFileName )
```

This will save chart information to a file which can later be loaded by the designer and modified.

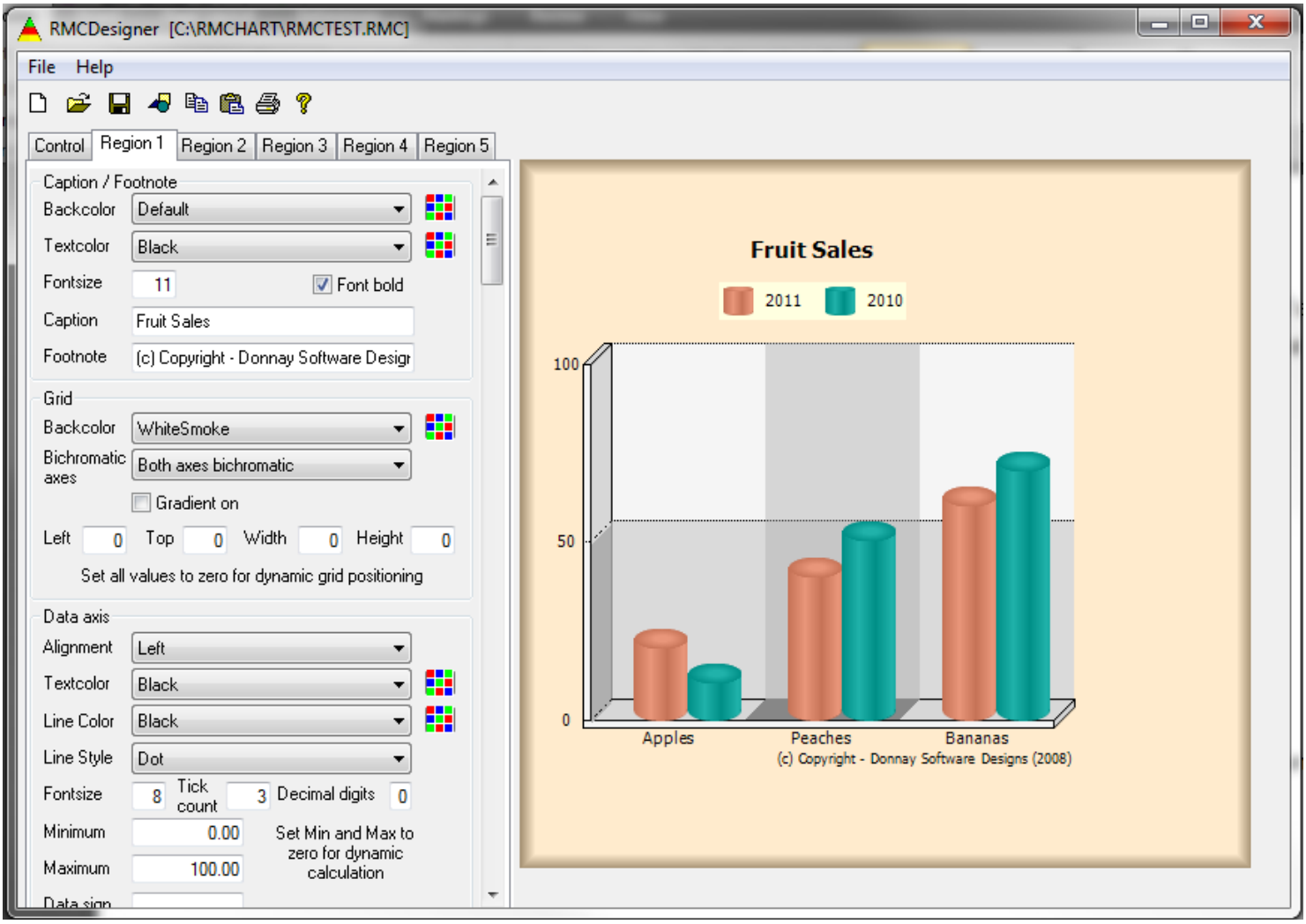
The file can be reloaded by the application like so:

```
oRmChart:rmcFile := cFileName
oRmChart:draw(.f.)
```

This will override all settings in the oRmChart object and redraw the chart.

Using the designer will save hours of time if the aesthetics of a chart are important to you. The chart can be modified and re-modified simply and quickly until reaching the desired look.

To see this feature in action, run RMCTEST.EXE. The source is RMCTEST.PRG.



The RMChart and RMChartRegion Classes

The RMCCLASS.PRG file contains the source for the sample programs included. These classes are not just wrapper classes, but instead they are used to simplify the creation and manipulation of charts.

RMChart Class

This class is used to create the main RMChart ActiveX control. It inherits from XbpActiveXControl. It includes the following methods:

Init() – Initializes the RMChart ActiveX control

Create() – Creates the RMChart ActiveX control

ApplyData() – Redraw chart with a new array of data for specified region and series

ApplyDataAxis() – Redraw chart with new array of axis text for a specified region and series

ApplyLabelAxis() – Redraw chart with new array of label text for a specified region

ApplyLegend() – Redraw chart with new array of legend text for a specified region

ApplyColors() – Redraw chart with new array of colors for a specified region

ShowToolTip() – Displays an Xbase++ dialog as a tooltip

RMChartRegion Class

This class is used to create each chart region. It sets up the arrays and calls the appropriate RMChart automation methods to build the chart.

Init() – Initializes the RMChart Region ivars and arrays

Create() – Creates a new RMChart Region and continues the initialization process

Configure() – Builds the chart from the arrays and iVars.

ApplyData() – Redraw region chart with a new array of data for this region and series

ApplyDataAxis() – Redraw region chart with new array of axis text for this region and series

ApplyLabelAxis() – Redraw region chart with new array of label text for this region

ApplyLegend() – Redraw region chart with new array of legend text for this region

ApplyColors() – Redraw region chart with new array of colors for this region

RMChart and eXpress++

eXpress++ users will find a set of DC* commands that allow RMChart graphs to be embedded within dialogs that have other eXpress++ commands, therefore it is simple to combine Xbase parts with RMCharts in dialogs, on tabpages, etc. The source file DCCHART.PRG contains the source for charts that that are created with DC* commands.

DCRMCHART – Creates the RMChart ActiveX Control

DCCHARTREGION – Creates a graph region on the RMChart

DCADDBARGROUP – Creates an array of bar group parameters to be used with DCCHARTREGION

DCADDLINEGROUP – Creates an array of line group parameters to be used with DCCHARTREGION

DCADDGRIDLESSGROUP – Creates an array of pie/donut chart parameters to be used with DCCHARTREGION

DCADDDATAAXIS – Creates an array of data axis parameters to be used with DCCHARTREGION

Printing Reports containing Charts

Xbase++ printing merges very well with charts and graphs created by RmChart. The **XbpPrinter()** class and the eXpress++ DCPRINT system both provide for printing of bitmaps anywhere withing a document. The RmChart ActiveX control includes a method called **draw2ClipBoard().** This will place the entire RmChart control onto the clipboard as a bitmap. The clipboard contents can then be extracted as an **XbpBitmap** object and used in a printed report.

The source file PRCHART.PRG contains a sample program that demonstrates this capability.

Conclusion

RmChart and Xbase++ are both well-designed to provide excellent graphing capabilities that can be incorporated in new or legacy applications to give applications a professional and state-of-the-art appearance and functionality.