Excel Charts

Conestoga College © **v. 2** **Peter Komisar**

How Information Is Conveyed in a Chart

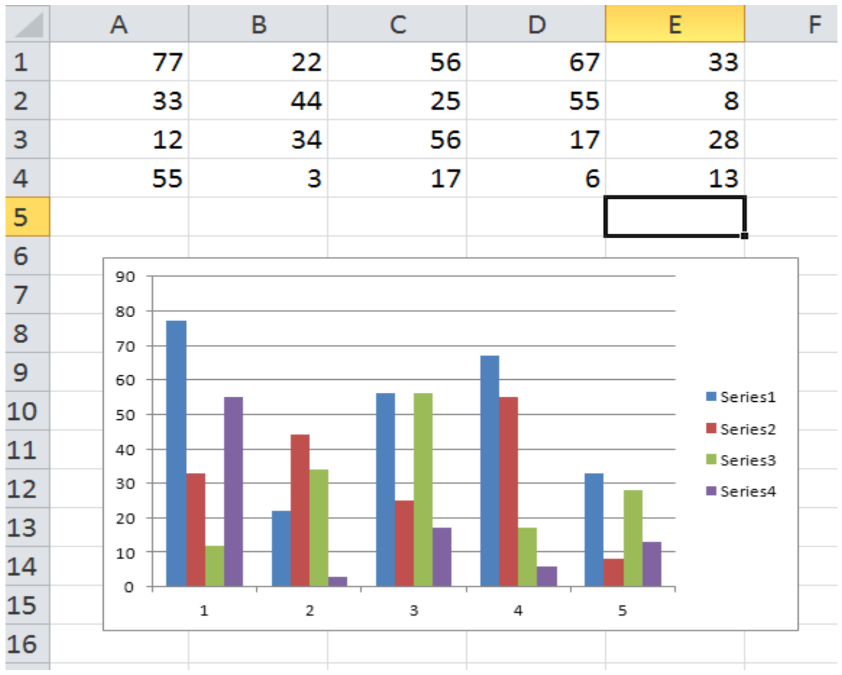
Different sorts of charts convey information in different ways. Excel has offered the same 73 chart types since Excel 97. Consider the following example.

The following chart is made up of 5 groups of data based on columns. In the spreadsheet, each column of data is associated with the set of bars below it. Notice each number in the rows associated with each column has a bar value, with a different color. So all the values in row 4 have a purple bar.

A scenario that can give this chart and data some meaning, would be to think of each these groups as four member families where each bar represents the ages of the people in the family. A large number of such samples would allow defining trends in the group for different population bases, taken across different times, geographic locations or other sorts of demographic criteria.

A chart supplies an image of the data, facilitating an intuitive understanding of the underlying data. "A picture paints a thousand words."

Sample Chart



	A	B	C	D	E	F
1	77	22	56	67	33	
2	33	44	25	55	8	
3	12	34	56	17	28	
4	55	3	17	6	13	

Chart, ChartObject & Shapes

In the 'old days' of Excel, charts were created on their own chart sheets.

In the mid 90s Excel added the capability of integrating charts directly into worksheets.

Each technique uses different objects.

- Chart
  - used for standalone chart sheets

- ChartObject
  - where a chart is contained in a ChartObject embedded in a sheet.

Example of the Standalone Chart Sheet Model *// Excel 2003*
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad.

```
Sheets(“Charts”).ChartArea.Interior.ColorIndex = 4
```

In Excel 2007, Charts became part of the Shapes collection of objects so charts are also referenced via the Shapes collection. The following line shows a Chart being referenced as a Shape object.

Example *// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad.*

```
Worksheets(“Jan”).Shapes(“Chart 1”).Chart.ChartArea.Interior.ColorIndex = 4
```

Load Some Data Into Ranges Being Referenced

Mr. Excel's Chart Macro's need data to work with. You can enter data into a range such as range of cells between A1 to E4 or use the download resource to get started. In later examples other ranges are referenced as well.

Chart Sheet Code That Works All the Way Back to Excel 2003 and Up to Excel 2013

Following is legacy Excel code that uses the Charts.Add method. If sharing macros with individuals still using Excel 2003 this code should be used.

The Charts collection contains a Chart object for each chart sheet in a workbook. The Chart object is also a member of the Sheets collection. When a chart is the active object, the ActiveChart property is used to refer to it.

Chart Example
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad. download resource

```
Sub OldMacro( )
   Charts.Add
   ActiveChart.SetSourceData Source:=Worksheets("Sheet1").Range("A1:E4")
   ActiveChart.ChartType = xlColumnClustered
   ActiveChart.Location Where:=xlLocationAsObject, Name:="Sheet1"
End Sub
```

Code That Works in Excel 2007 to 2013 *// the AddChart Method*

The newer technique deploys the AddChart method as is shown in the following example. This code shows Chart is also part of the Shapes collection.

Example Works With 2003 Excel As Well
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad.download resource

```
Sub AltF1Macro()  '
   '  works with Excel 2003, too
   '  Create chart on the current sheet
   ActiveSheet.Shapes.AddChart.Select
   ActiveChart.SetSourceData Source:=Range("Sheet1!$A$1:$E$4")
   ActiveChart.ChartType = xlColumnClustered
End Sub
```

Latest Code Introduced With Excel 2013 *// uses the AddChart2 method*

```
Sub CreateChartExcel2013()

Dim WS As Worksheet
Dim ch As Chart
Set WS = ActiveSheet

'Select the data for the chart
Range.(“A1:E4”).Select

'Define the chart, use style 202 for rotated data labels
Set ch = WS.Shapes.AddChart2(_
Style := 202,  _
XlChartType := xlColumnClustered, _
Left := [A6] .Left, _
Top := [A6].Top, _
Width := [A6:C6].Width, _
Height := [A6:A20].Height, _
NewLayout := True).Chart

'Adjust the title
ch.ChartTitle.Text = “2015 Sales By Region”
End Sub
```

// See Figure 15-5 in 2013 book for output

Locating A Chart On a Spreadsheet *// 72 points to the inch*

The numbers used in the code below are point values, 72 pts per inch,
and they are used to locate a chart.

Specifying Location Based on Points
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad,

```
Sub SpecifyLocation( )
     Dim WS As Worksheet
    Set WS = Worksheets("Sheet1")
    WS.Shapes.AddChart(xlColumnClustered, Left:=100, Top:=150, Width:=400,
Height:=300).Select
   ActiveChart.SetSourceData Source:=WS.Range("A1:E4")
End Sub
```

Chart Remover *// from ExcelExperts.com*
http://excelexperts.com/deleting-all-charts-shapes-sheet

Once a chart is on a sheet, a chart added subsequently that overlaps the first
may create error conditions. The following macro clears any and all charts.

Chart Deleting Macro

```
Sub DeleteAllCharts()
'Sheet1.ChartObjects.delete
End Sub
```

// If there are no charts and the Sub is run, an error condition arises and needs to be caught

Exact Locations

The use of points to create sizes is cumbersome. Easier to use are dimensions based on the edges of spreadsheet cells. Mr. Excel shows how to do this in the following example. Note how the (x,y) point and width & height are obtained from cell ranges.

Using Cell Dimensions to Size a Chart
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub SpecifyExactLocation()
    Dim WS As Worksheet
    Set WS = Worksheets("Sheet1")
    WS.Shapes.AddChart(xlColumnClustered, _
        Left:=WS.Range("C11").Left, _
        Top:=WS.Range("C11").Top, _
        Width:=WS.Range("C11:J11").Width, _
        Height:=WS.Range("C11:C30").Height _
        ).Select
    ActiveChart.SetSourceData Source:=WS.Range("A1:E4")
End Sub
```

Moving Charts After They Have Been Created *// note your chart name might be different*

In the above code you are locating the container in which the chart is located. Seems you shouldn't try to move the chart within that container.

To move a chart after the fact, similar code is used, changing the upper left and top values and the width and height parameters. The chart is referenced by name.

Moving a Chart After The Fact Code
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub MoveAfterTheFact()
      Dim WS As Worksheet
    Set WS = Worksheets("Sheet1")
    With WS.ChartObjects("Chart 3")
        .Left = WS.Range("C21").Left
        .Top = WS.Range("C21").Top
        .Width = WS.Range("C1:H1").Width
        .Height = WS.Range("C21:C25").Height
    End With
End Sub
```

Chart Names *// sequence includes deleted charts*

Charts are by default given names that are sequenced. For example 'Chart 1', 'Chart 2' etc. Notice, that this does not necessarily mean there are as many charts on a page as numbered in the names. This is because deleted charts retain their chart names.

Example

Chart 5 may be one of two charts on a spreadsheet as three others may have been deleted. To complicate matters, the chart name might change the next time the macro runs. A good practice is to store the name that has been given to the chart for reference later in the macro.

Mr. Excel's code shown below builds a chart and then stores it's name.
Later in the macro the name is used to do further actions on the chart.

Remember That Name *// life of the macro*
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad.

```
Sub RememberTheName( )
  Dim WS As Worksheet
   Set WS = Worksheets("Sheet1")
   WS.Shapes.AddChart(xlColumnClustered, _
      Left:=WS.Range("H11").Left, _
      Top:=WS.Range("H11").Top, _
      Width:=WS.Range("H11:M11").Width, _
      Height:=WS.Range("H11:H30").Height _
      ).Select
   ActiveChart.SetSourceData Source:=WS.Range("A1:E4")
   ThisChartObjectName = ActiveChart.Parent.Name

   ' more lines of code...
   ' then later in the macro, you need to re-assign the chart

   With WS.Shapes(ThisChartObjectName)
      .Chart.SetSourceData Source:=WS.Range("L1:P4"), PlotBy:=xlColumns
      .Top = WS.Range("H11").Top
   End With
End Sub
```

What If the Macro Is Closed And You Need The Name of the Chart *// life of the spreadsheet*

In the next example, in the case you need the name after the macro is closed and a new macro is started, Mr. Excel stores the name is an out-of-the-way cell, Z1. Then if another macro is started on the page and needs the name , it can retrieve the chart name from that cell.

Store the Name in an Out of the Way Cell
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub StoreTheName()
   Dim WS As Worksheet
   Set WS = Worksheets("Sheet1")
   WS.Shapes.AddChart(xlColumnClustered, _
      Left:=WS.Range("H11").Left, _
      Top:=WS.Range("H11").Top, _
      Width:=WS.Range("H11:M11").Width, _
      Height:=WS.Range("H11:H30").Height _
      ).Select
   ActiveChart.SetSourceData Source:=WS.Range("A1:E4")
   Range("Z1").Value = ActiveChart.Parent.Name
End Sub
```

Important Note: Students of this course who use Excel everyday 'on the job' say that this is not such a good idea as it artificially modifies the size of the worksheet. They suggest if additional information needs to be stored it might better be put in another sheet, example Sheet2, which can serve to store related information.

// Perhaps the Names directory would be a suitable place to store this information

The next macro picks the name up from it's stored location.

Locating the Chart Name From the Storage Location
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub ChangeTheChartLater()
   ' Page 207
   Dim WS As Worksheet
   Set WS = Worksheets("Sheet1")
   MyName = WS.Range("Z1").Value
   With WS.Shapes(MyName)
      .Chart.SetSourceData Source:=WS.Range("L1:P4"), PlotBy:=xlColumns
      .Top = WS.Range("C26").Top
   End With
End Sub
```

Finding a Chart With Upper Left Corner at a Specific Cell

If there are many charts, a specific one can be located by finding the associated upper left corner cell as is shown in the following code loop.

Example Logic To Find a Chart Based on it's Cellular Address
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
For each Cht in ActiveSheet.ChartObjects
  If Cht.TopLeftCell.Address = “$A$4” then
    Cht.Interior.ColorIndex = 4
end if
Next Cht
```

Chart Types

There are 73 different chart types. Each chart is represented by an 'xlConstant'. These are roughly mapped to a range of numerical values between 51 and 115. A quick subtraction says “ That is not 73 charts. This is because some charts are found having constants in the range 1 to 15 and other are in really 'far in' ranges such as the pie chart at -4102.

You can view the complete set of Excel chart names with associated constants on pages 208 to 210 of the 2010 book and pages 333 of the 2013 book.

Example

xlColumnStacked

The Set of Constant Values

The following macro provides a slide show of the majority of Excel charts types. It misses some at the inner range of 1 to 15 and includes a single chart that has the constant value -4102, one of a small set of charts that have numbers in the negative 4000s. Some error conditions arise, (I speculate from charts not existing for some numbers in the target range) which are neatly swept under the rug with the 'ignore the problem' error handling code. (We could have been more rigorous and found each and every chart but this is the lion's share of charts available.)

Chart Visual Tour

```
Sub ChartTypes()
   ActiveSheet.Shapes.AddChart.Select
   ActiveChart.SetSourceData Source:=Range("Sheet1!$A$1:$E$4")

      'loop through where most of the charts are clustered
   For j = 1 To 62
   ActiveChart.ChartType = j + 50
   Range("F1").Value = j + 50
   Application.Wait Now + TimeValue("00:00:02")

   'error protection protects against null values
   On Error Resume Next
    Next j
   On Error GoTo 0

   'the macro ends with a 3D pie chart with an odd number value -4102
   ActiveChart.ChartType = -4102
   Range("F1").Value = -4102
End Sub
```

Custom Chart Templates

Excel provides for the creation of custom chart templates that may include all the programmer's favorite chart features. If such a macro is to be distributed, the template has to go with it.

Custom Chart Template File Extension, Storage Location & Invocation

Custom chart templates are stored in files with the .crtx extension and are saved to the %appdata%\Microsoft\Templates\Charts\ folder. To use a custom chart in a macro the command is as follows.

Calling a Custom Chart Template

ActiveChart.ApplyChartTemplate “CustomChart1.crtx”

Key Aspects of Using Custom Chart Templates

- extension → .crtx
- location → %appdata%\Microsoft\Templates\Charts\
- invocation → ActiveChart.ApplyChartTemplate “CustomChart1.crtx”

If the chart does not exist an error message will be thrown. The familiar formula that allows a macro to proceed without raising a debug error is used

Example *// ignoring error raised if chart does not exist*

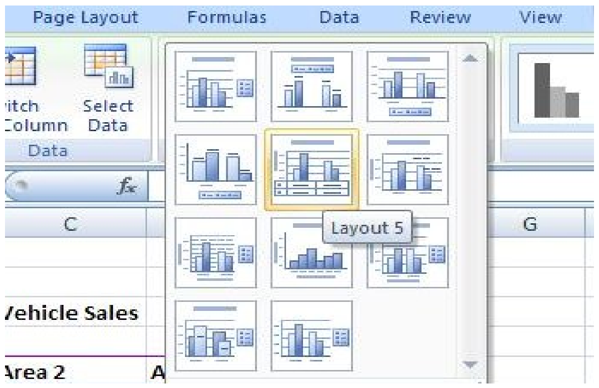
```
On Error Resume Next
ActiveChart.ApplyChartTemplate “CustomChart1.crtx”
On Error GoTo 0
```

Modifying Chart Layout or Style

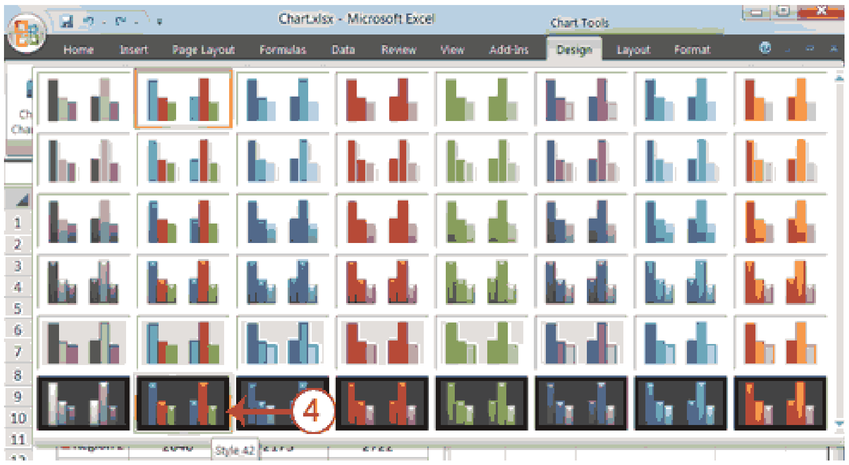
The Design tabs have two 'galleries”.

- Chart Layout gallery
- Styles gallery

Layouts in the Excel User Interface



Styles in the Excel User Interface



The number of layouts available varies between 4 and 12 depending on what type of chart is being created. The layouts appear to the left of the Design Tab under 'Chart Tools'.

To apply a layout the command is as follows.

Example

ActiveChart.Apply Layout 1

Changing Between Chart Types

Combining the effects of Chart Type, Layout and Style results in a staggering number of choices available in chart presentation.

// if you specify a number greater than layouts available a runtime error 5 results

The VBA code for setting chart style is as follows. (The statement is usually followed by a line of code that clears the previous format setting.)

Example

```
ActiveChart.ChartStyle = 1
ActiveChart.ClearToMatchStyle
```

New Chart Features in Excel 2013 *// horizontal rules are there to demarcate 2013 only code*

- new chart styles
- color settings
- filtering
- data labels that act as captions

New Chart Styles *// Valid only for Excel 2013*

See the new Chart styles in the large Chart Style Gallery on the Design Tab. Chart styles is not new. Excel 2007 offered 48 chart styles. The difference is that the new styles Mr. Excel describes as interesting while the old ones were described as 'boring variations on four styles: monochrome, colorful, single-color and dark background.

If you use the new AddChart2 method, you can specify a chart style as the first parameter of that method.

To later specify or change a chart style you use the following code.

Example

```
ch.ClearToMatchStyle
ch.ChartStyle = 339
```

Valid Chart styles are from 201 to 353 (In Excel 2010 the valid styles are 1 to 48)

Mr. Excel says the fastest way to get familiar with chart styles is to turn on the macro-recorder, select a style then stop the recorder and view the code.

Check the 2013 book, pages 343 to 345 for more details on the new Chart styles.

Applying Chart Color in Excel 2013

Excel 2013 introduces the ChartColor property which is used to assign one of 26 color themes to a chart, (no relation to the order they are shown in Excel menus.)

Example

```
ch.ChartColor =13
```

Page 343 in the book gives a breakdown of the color schemes.

Filtering a Chart in Excel 2013

Excel 2013 lets you leave out rows such as 'totals' rows or columns that interfere with presenting a good chart. In Excel 2013 you leave all the unwanted data in place to create the chart and then use the Funnel icon to the right of the chart to remove rows and columns. The effect in VBA is achieved setting the IsFiltered property to true. See code at the bottom of 347 in the 2013 book.

Using Cell Formulas as Data Label Captions in Excel 2013

Excel offers the ability to caption a data point based on the contents
of cell in the datasheet.

See code on page 349 in the 2013 book and the diagram on page
350 Figure 15-14.

These Changes Only Work in Excel 2013

These new features are only compatibile with Excel 2013 so must
be used appropriately.

The next feature, the SetElement method, is backwards compatible to Excel 2007.

The SetElement Method *// used to modify elements of a chart*

The SetElement method is used by Excel to apply the many built-in
settings found in the menus of the Layout Tab. The constants are of
the enumeration of MsoChartElementType.

MsoChartElementType

https://msdn.microsoft.com/en-us/library/microsoft.office.core.msochartelementtype.aspx

Example

```
ActiveChart.SetElement   msoElementChartTitleAboveChart
```

// See text for Tip and figure 11.4 regarding limits of use of setElement in Excel 2010
// on page 213, and Tip on page 350 of Excel 2013 book for similar limitations.

Remember the Macro Recorder

Pages 214 to 217, 2010 book & pages 351 to 354, 2013 book show the constants
available that are used to represent the great number of actions that can be applied
in the Layout Tab of Excel 2010 or the Plus Icon of Excel 2013.

Mr. Excel reminds us, to use the Macro recorder to figure out which constants are used.

The Result   *// worked just as suggested!*

```
Sub ChartStyles( )
ActiveChart.SetElement (msoElementLegendLeft)
End Sub
```

Changing a Charts Title *// same in Excel 2010 & 2013*

In Excel, a charts title can be changed by double clicking the chart title text and typing in a new title. Mr.Excel suggest using the following line of code to set a chart title.

ActiveChart.ChartTitle.Caption = “New Chart Name” *// Text and Caption both work*

The chart whose name is being changed has to be made active. Following is a reduced version macro-recorder code, run on an existing chart.

Chart Change Example

```
Sub ChangingNameInRecorder()
   ActiveSheet.ChartObjects("Chart 23").Activate
   ActiveChart.ChartTitle.Text = "Epiphany"
 End Sub
```

The Format Method

The Format is the key to 'micromanaging' formatting options. The set of chart styling settings, (some call 'chart junk') include the following.

- Fill
- Glow
- Line
- Shadow
- PictureFormat
- SoftEdge
- TextFrame2
- ThreeD

Table 11.3 in the 2010 book & Table 15.3 in the 2013 book shows a partial list of chart elements that can be formatted using the Format method.

Changing a Charts Foreground Color

Using the macro recorder to change Bar Colors yielded something that has been reduced to the following.

(Start with a clean sheet with a single chart so the name is indeed “Chart 1”)

Macro Recorder Reduction Changing Bar Colors

```
Sub ShapeFill()
    With ActiveSheet.Shapes("Chart 1").Fill
        .Visible = msoTrue
        .ForeColor.RGB = RGB(107, 177, 120)
        .Transparency = 0
        .Solid
    End With
End Sub
```

The above code treats the Chart as a shape and doesn't use the Format method. The next two macros from Mr. Excel demonstrate the use of the Fomat method.

Applying a Theme Color to a Chart
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub ApplyThemeColor()
    ' Page 220
    Dim cht As Chart
    Dim ser As Series
    ActiveSheet.ChartObjects(1).Select
    Set cht = ActiveChart
    Set ser = cht.SeriesCollection(1)
    ser.Format.Fill.ForeColor.ObjectThemeColor = msoThemeColorAccent6
End Sub
```

Applying a Texturer to a Chart
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub ApplyTexture()
    ' Page 220
    Dim cht As Chart
    Dim ser As Series
    ActiveSheet.ChartObjects(1).Select
    Set cht = ActiveChart
    Set ser = cht.SeriesCollection(2)
    ser.Format.Fill.PresetTextured (msoTextureGreenMarble)
End Sub
```

We may be pushing the limits of utility here. The following code by Mr.Excel shows how simply an image can be uploaded and used in a chart in VBA., You really need a good use for the image.

Locate an image in the path used in the code below.

Applying an Image to a Chart
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub FormatWithPicture()
   ' Page 221
   Dim cht As Chart
   Dim ser As Series
   ActiveSheet.ChartObjects(1).Select
   Set cht = ActiveChart
   Set ser = cht.SeriesCollection(1)
   MyPic = "C:\PodCastTitle1.jpg"
   ser.Format.Fill.UserPicture (MyPic)
End Sub
```

// Mr. Excel shows how patterns can be applied (page 357, 2013 book & page 221, 2010 book.
// A caution states that this pattern does not work in Excel 2007 so we have omitted this example.

The following example shows how gradients are applied. There does come a point where a visual builder begins to improve on the efficiency of creating graphical aspects of programs.

Excel 2013 (as does Excel 2010) provides OneColorGradient and TwoColor Gradient methods, the latter which is used in the example below.

// Mr. Excel in the 2010 book states he is indebted to the macro recorder for lot of his code.

Applying a Two Tone Gradient to the Bars of a Chart
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub TwoColorGradient()
   ' Page 221-222
   Dim cht As Chart
   Dim ser As Series
   ActiveSheet.ChartObjects(1).Select
   Set cht = ActiveChart
   Set ser = cht.SeriesCollection(1)
   ser.Format.Fill.TwoColorGradient msoGradientFromCorner, 3
   ser.Format.Fill.ForeColor.ObjectThemeColor = msoThemeColorAccent6
   ser.Format.Fill.BackColor.ObjectThemeColor = msoThemeColorAccent2
End Sub
```

Line And Border Setting

The LineFormat object formats either a line or a border around an object. Many properties of a line can be changed such as colors, arrows and dash style.

This example, (one of two) shows a border being formatted around a chart.

A Macro That Dashes A Charts Border
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub FormatBorder()
   ' Page 222
   Dim cht As Chart
   Set cht = ActiveChart
   ActiveSheet.ChartObjects(1).Select
   With cht.ChartArea.Format.Line
       .DashStyle = msoLineLongDashDotDot
       .ForeColor.RGB = RGB(50, 0, 128)
   End With
End Sub
```

The next macro puts a shadow on the title of a chart. The macro fails if the chart it is applied to does not have a title.

A Macro That Adds Glow To the Title of a Chart
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub AddGlowToTitle()
   ' Page 223
   Dim cht As Chart
   Set cht = ActiveChart
   ActiveSheet.ChartObjects(1).Select
   cht.ChartTitle.Format.Line.ForeColor.RGB = RGB(255, 255, 255)
   cht.ChartTitle.Format.Line.DashStyle = msoLineSolid
   cht.ChartTitle.Format.Glow.Color.ObjectThemeColor = msoThemeColorAccent6
   cht.ChartTitle.Format.Glow.Radius = 8
End Sub
```

The effects that the following macro demonstrate can easily escape observation. This macro changes the shadow on the legend box which may be a very subtle effect if the shadow is diminutive and the legend is small.

A Macro That Puts a Shadow on a Chart Legend
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub FormatShadow()
   ' Page 223
   Dim cht As Chart
   ActiveSheet.ChartObjects(1).Select
   Set cht = ActiveChart
   With cht.Legend.Format.Shadow
       .ForeColor.RGB = RGB(0, 0, 128)
```

```
        .OffsetX = 5
        .OffsetY = -3
        .Transparency = 0.5
        .Visible = True
    End With
End Sub
```

// had problems with any macros that used cht.SeriesCollection(1)

A Good Thing To Do

Build a Chart From Scratch Recording the Code in he Macro Recorder in order to generate code based on Shapes. Discuss what advantages there might be in focusing on using the Shapes object for generating Charts.

3-D Rotation Settings in 2010 Book vs Creating a Combo Chart in 2013

The 2010 book provides a discussion of applying 3D rotations and Bevels to charts and supplies a long list of 'Camera Shot' constants.

In the 2013 book Mr. Excel omits this discussion and instead adds a long case study on creating a combo chart. (See Figure 15-16 on page 360 of the 2013 book.

Both are specialized uses. You now know at least where there is some information on both topics should you need this in the future.

The book goes on to show some complicated Advanced Charts.

Following is Mr. Excels sophisticated Stock Market chart.

Advanced Charts, OHLC Chart *// Open High Low Close*

If you are willing to take the details in hands, sophisticated custom charts can be designed in VBA. Mr. Excel creates a typical stock market chart by 'cheating in' a gif file of a shape that is missing in Excels graphical libraries. He outlines the procedure and the resulting Macro. It is including here for you convenience should you wish to experiment with it. Don't forget to include some data for it.

// Mr. Excel discusses a little format bug in Excel in his VBA Resource for this chart

Mr. Excel's Custom Open High Low Stock Market Chart
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub CreateOHCLChart()
    ' Page 235-236,  Download leftdash.gif from the sample files
    ' for this book and save it in the same folder as this workbook
    Dim Cht As Chart
    Dim Ser As Series
      ' Find path to LeftDash.gif
    DashPath = ThisWorkbook.Path & Application.PathSeparator & "leftdash.gif"
      ActiveSheet.Shapes.AddChart(xlLineMarkers).Select
    Set Cht = ActiveChart
    Cht.SetSourceData Source:=Range("OHLC!$A$1:$E$33")
    ' Format the Open Series
    Set Ser = Cht.SeriesCollection(1)
    With Ser
        .Border.LineStyle = xlNone
        .MarkerStyle = xlMarkerStylePicture
        .Fill.UserPicture (DashPath)
     End With
    ' Format High & Low Series
    With Cht.SeriesCollection(2)
        .MarkerStyle = xlMarkerStyleNone
        .Border.LineStyle = xlNone
    End With
    With Cht.SeriesCollection(3)
        .MarkerStyle = xlMarkerStyleNone
        .Border.LineStyle = xlNone
    End With
    ' Format the Close series
    Set Ser = Cht.SeriesCollection(4)
    With Ser
        .MarkerBackgroundColorIndex = 1
        .MarkerForegroundColorIndex = 1
        .MarkerStyle = xlDot
        .MarkerSize = 9
        .Border.LineStyle = xlNone
    End With
    ' Add High-Low Lines
    Cht.SetElement (msoElementLineHiLoLine)
    Cht.SetElement (msoElementLegendNone)
    ' Try formatting the line away again
    Cht.SeriesCollection(1).Border.LineStyle = xlNone
    ActiveChart.SeriesCollection(1).Select
    Selection.Format.Line.Visible = msoFalse
    ActiveChart.ChartArea.Select
End Sub
```

You are are on your own for

- Frequency Charts Pages 236 to 239
- Stacked Area Charts Pages 239 to 243
- Pivot Charts Pages 246 to 248

We cover one last topic in this chapter.

Exporting a Chart as a Graphic

Although we can screen shot a chart, drop it into a photo editor and export a cropped version as a gif or jpg file, Excel offers a better direct way of generating a graphic from an Excel Chart which could be useful for general reporting.

Macro That Exports a Chart into a Image File
// from 'VBA & Macros' Microsoft Excel 2010, Bill Jelen & Tracy Syrstad., download resource

```
Sub ExportChart()
   ' Bonus
   Dim cht As Chart
   ActiveSheet.ChartObjects("Chart 1").Activate
   Set cht = ActiveChart
   cht.Export Filename:="C:\Chart.gif", Filtername:="GIF"
End Sub
```

Assignment

Just to touch base with Charts. Create a spreadsheet, representing the total product output for a company over a three year period. Each year would be represented by a row and each row would have a number of units produced that month. (This is just a suggestion. You can create any scenario you like that supplies you with the data you need to develop the charts.)

In one or more macros, create three views (chart types) of the data for each year. Consider the very first chart shown at the head of this note, as an example.
If you prefer you could use production values on a quarterly basis.

Year	Jan	Feb	Mar	Apr	May	June	July	Aug	Sept	Oct	Nov	Dec
2010	211	333	301	141	277	348	195	320	422	257	400	460
2011	etc.											
2012												

A Note on the Final

Good luck next week on the test. (You can hand this assignment as late as Sunday after the test if you do not want to be distracted from studying.) The test is multiple choice true or false with the rare fill-in-the-blank. Usually students are finished under an hour. If your term work is up-to-date you are already 'home free' with your mark so do not 'sweat' the test, instead regarding it as another learning experience.