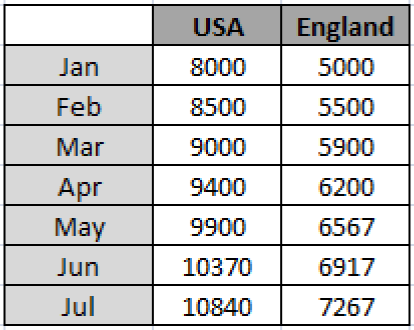
Introduction to VBA for Excel-Tutorial 9

In this tutorial, charting using VBA will be demonstrated. Chart is another object within the VBA environment. Chart has methods such as ***Axes***, ***Copy*** and ***Paste***. It also has properties such as ***ChartTitle***, ***Legend***, and ***Name***. In addition, Chart has also events such as ***Activate*** and ***Select***.

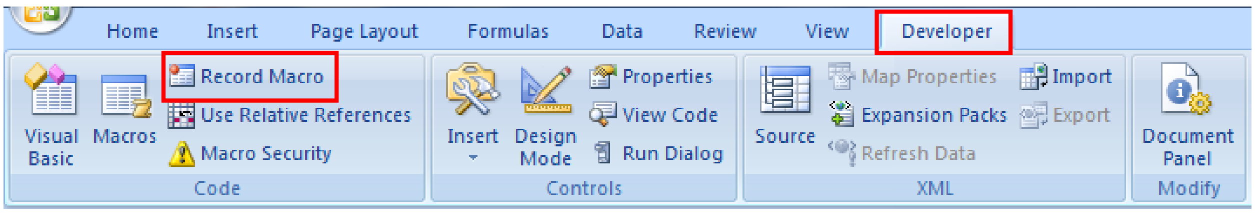
Previously, we discussed Recording Macros as a method to learn the commands syntax needed. This is one of the few instants where this method will be misleading and irreproducible. Let's try to see what the shortfalls of Recording Macro are by following an example.

Example: Let's say you want to create a column plot for the following data and in the process you would like to record a macro for future usage.

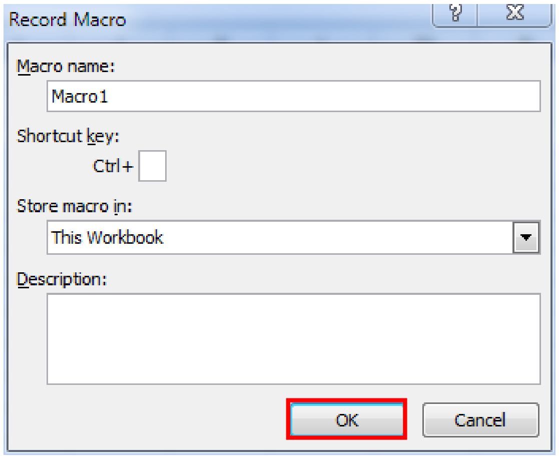


	USA	England
Jan	8000	5000
Feb	8500	5500
Mar	9000	5900
Apr	9400	6200
May	9900	6567
Jun	10370	6917
Jul	10840	7267

1- Click **Developer** Tab on the Ribbon, then click **Record Macro**



2- Excel will prompt you with **Record Macro** dialog box, where you can change the macro name, add a keyboard shortcut to execute the macro and add description of the macro. For this example, we will leave the default values, however feel free to experiment with different values. Click **OK** to start recoding.



Record Macro

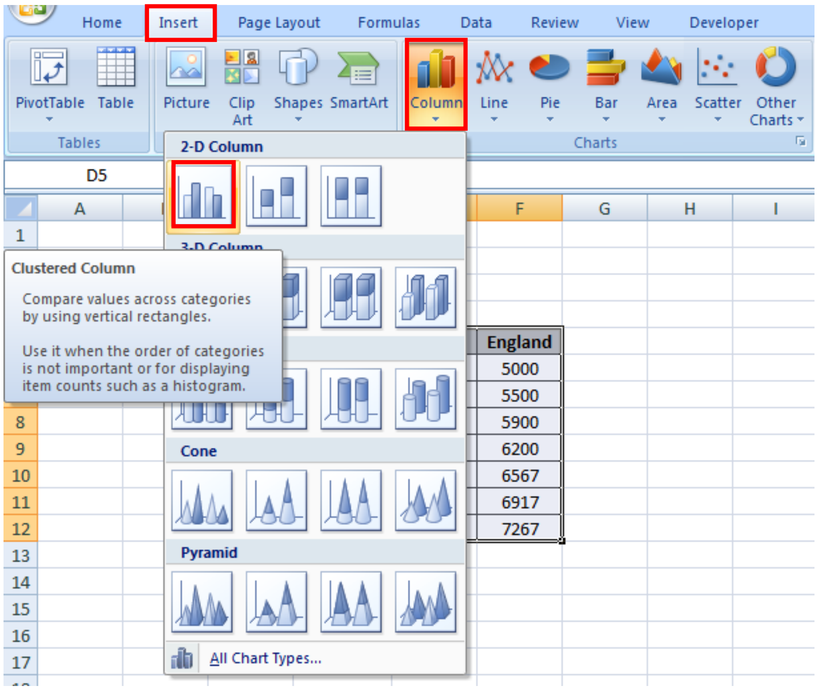
Macro name: Macro1

Shortcut key: Ctrl+

Store macro in: This Workbook

Description:

3- Once you click **OK** on the **Record Macro** dialog box, all subsequent events will be recorded. Go ahead and add Column graph by first selecting all the data in the table then click **Insert/Column Chart/2D Column Chart**



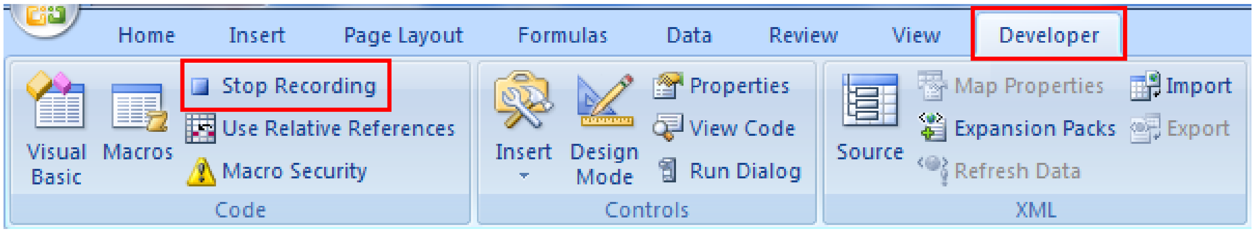
Clustered Column

Compare values across categories by using vertical rectangles.

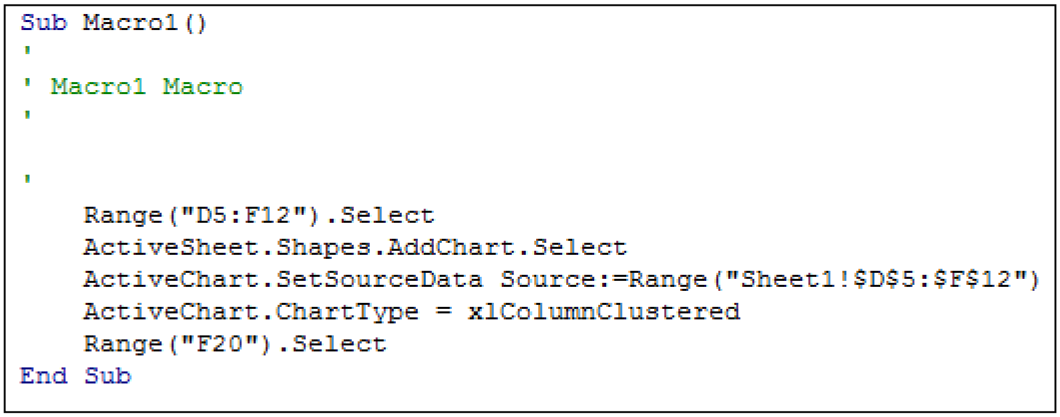
Use it when the order of categories is not important or for displaying item counts such as a histogram.

England
5000
5500
5900
6200
6567
6917
7267

4- After Excel displays the chart, click **Developer** Tab and then stop recording by clicking **Stop Recording**.



5- Access the VBE environment and Click **Module 1** on the project tree. The recorded macro would look like



```
Sub Macro1()
'
' Macro1 Macro
'

'
    Range("D5:F12").Select
    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.SetSourceData Source:=Range("Sheet1!$D$5:$F$12")
    ActiveChart.ChartType = xlColumnClustered
    Range("F20").Select
End Sub
```

By closing examining the code above, you should notice:

A. The data source range is hard coded that means it is hard to make it dynamic.
B. It did not generate the code for formatting the chart, i.e. adding the axes titles and chart title. Well even if it did, all of these would be hard coded, again hard to make it dynamic.
C. Lastly, however the legend is displayed; no commands are recorded to show us how to change the legend from the code.

Well, I guess you got the idea, however recording macros can be a lifesaver in lots of situation, charting is definitely is not one of them. In what follow, we will learn how to compile a code to make a chart in VBA.

1- Declaring needed variables. The first variable we will use is the chart, remember since chart is an object one can declare a variable as a chart type. The second would be the data range of which the data to be plotted.

```
Dim MyChart As Chart
Dim DataRange As Range
```

2- Selecting and setting the data to *DataRange* variable and adding a new chart

```
Set DataRange = ActiveSheet.Range(Cells(5, 4), Cells(12, 6))
Set MyChart = ActiveSheet.ChartObjects.Add(10, 10, 345, 210).Chart
MyChart.SetSourceData Source:=DataRange
MyChart.ChartType = xl3DColumnClustered
```

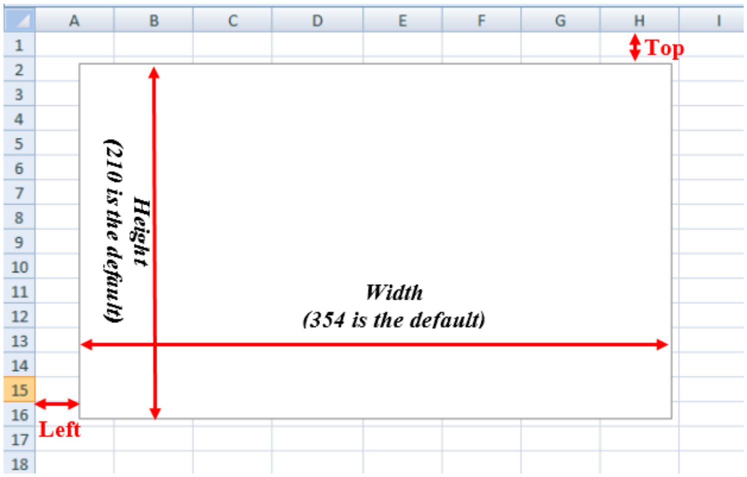
<u>In the first statement</u>, sets the range D5:F12 to be the data source. Note that Cells command is used with Range to allow dynamic sizing.

<u>The second statement</u>, it creates a new chart where the arguments of Add command are:

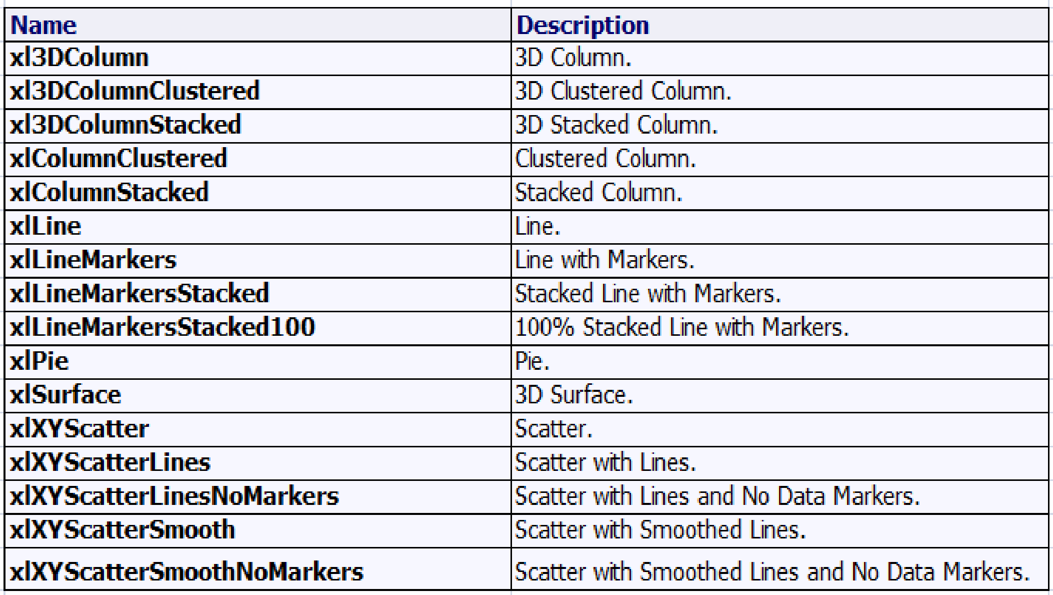
a- Left Position, left position of the chart, if omitted Excel centers the chart horizontally.
b- Top Position, top position of the chart, if omitted Excel centers the chart vertically.
c- Width, width of the chart, if omitted Excel uses 345
d- Height, height of the chart, if omitted Excel uses 210

<u>In third statement</u>, it assigns the selected *DataRange* as the data source of the newly created chart *MyChart*.

<u>The last statement</u>, it selects the chart type of the newly created chart *MyChart*.

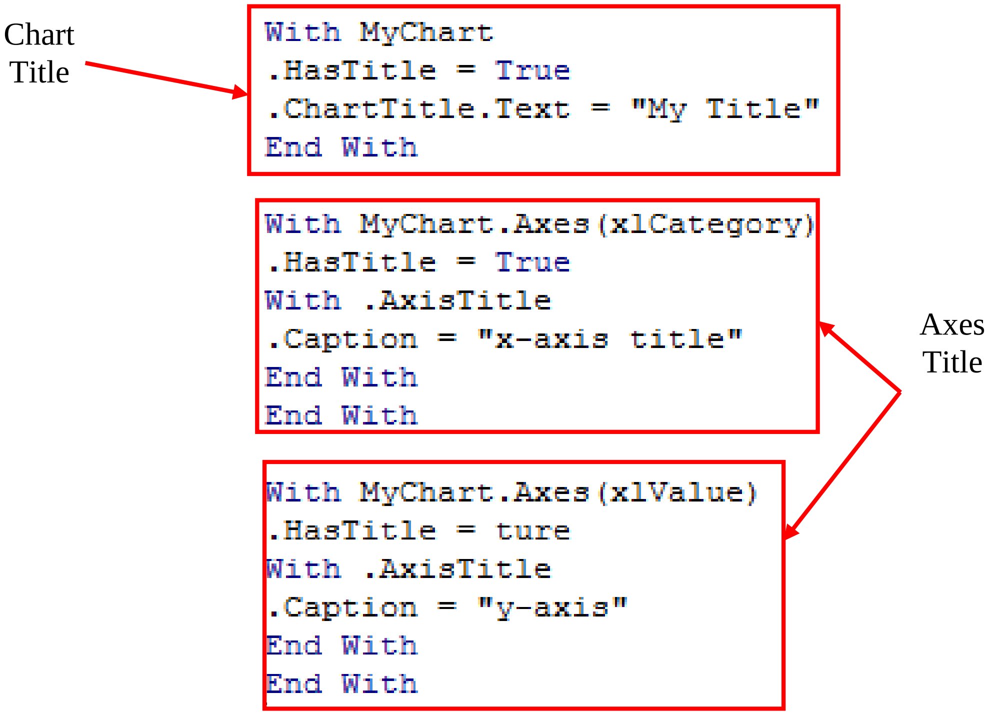


Here is a list of most common *ChartType*, the rest of this list can be found by searching VBA help files.



Name	Description
xl3DColumn	3D Column.
xl3DColumnClustered	3D Clustered Column.
xl3DColumnStacked	3D Stacked Column.
xlColumnClustered	Clustered Column.
xlColumnStacked	Stacked Column.
xlLine	Line.
xlLineMarkers	Line with Markers.
xlLineMarkersStacked	Stacked Line with Markers.
xlLineMarkersStacked100	100% Stacked Line with Markers.
xlPie	Pie.
xlSurface	3D Surface.
xlXYScatter	Scatter.
xlXYScatterLines	Scatter with Lines.
xlXYScatterLinesNoMarkers	Scatter with Lines and No Data Markers.
xlXYScatterSmooth	Scatter with Smoothed Lines.
xlXYScatterSmoothNoMarkers	Scatter with Smoothed Lines and No Data Markers.

3- Adding the chart title and the axis names:



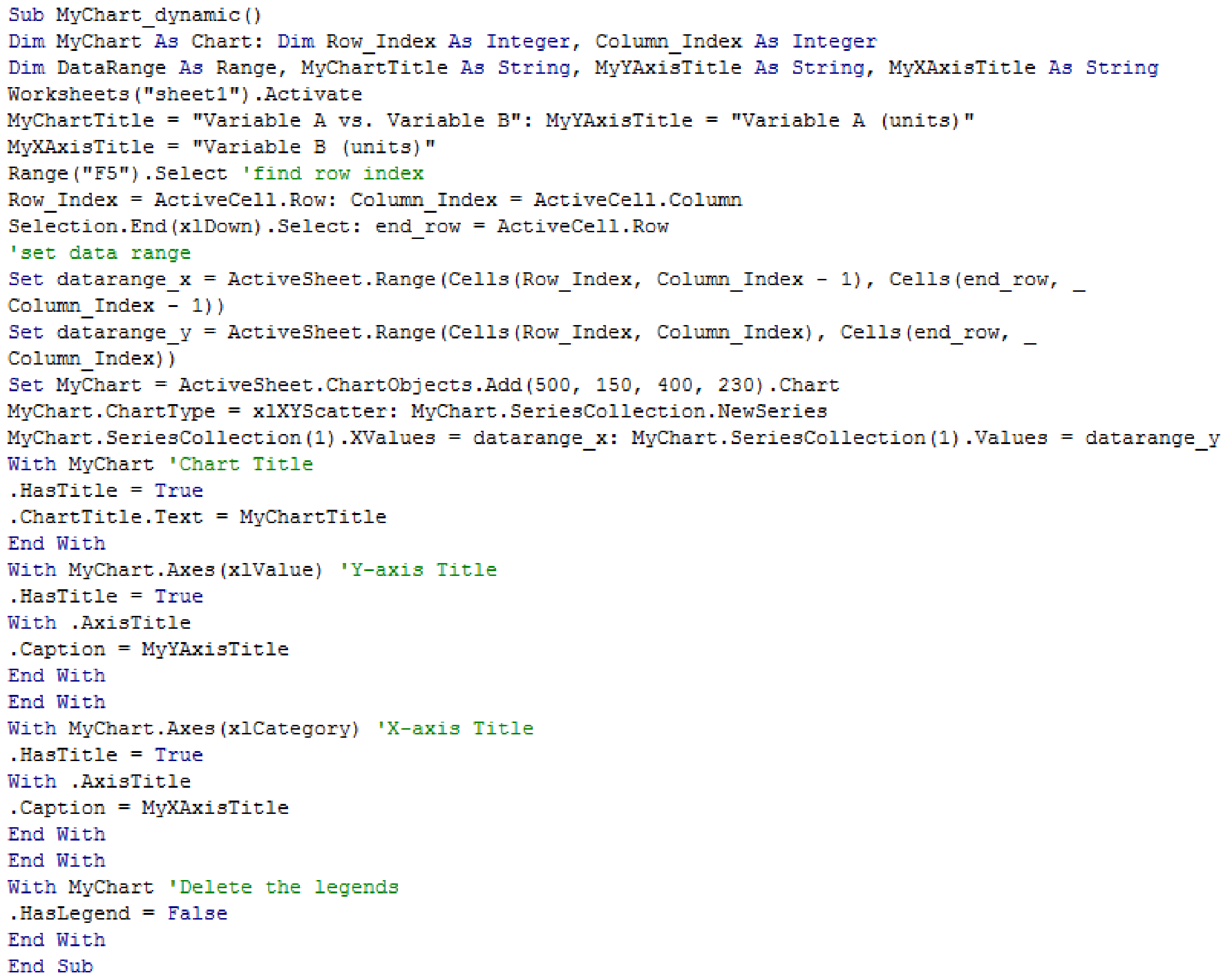
```
With MyChart
.HasTitle = True
.ChartTitle.Text = "My Title"
End With
```

```
With MyChart.Axes(xlCategory)
.HasTitle = True
With .AxisTitle
.Caption = "x-axis title"
End With
End With
```

```
With MyChart.Axes(xlValue)
.HasTitle = ture
With .AxisTitle
.Caption = "y-axis"
End With
End With
```

Now if you put the code together and run it, it will generate the same chart. The one difference, it can be customized. As a good student of ME209, you can declare variables for data ranges, the chart title, and the axes titles and then use an **InputBox**, for example, to give flexibility to the user.

Here is another example to create XY Scatter plot. In this example, the dynamic aspects of programming chart object within VBA environment are demonstrated such as the chart title, the x-axis title, the y-axis title, the series location and the chart size.



```
Sub MyChart_dynamic()
Dim MyChart As Chart: Dim Row_Index As Integer, Column_Index As Integer
Dim DataRange As Range, MyChartTitle As String, MyYAxisTitle As String, MyXAxisTitle As String
Worksheets("sheet1").Activate
MyChartTitle = "Variable A vs. Variable B": MyYAxisTitle = "Variable A (units)"
MyXAxisTitle = "Variable B (units)"
Range("F5").Select 'find row index
Row_Index = ActiveCell.Row: Column_Index = ActiveCell.Column
Selection.End(xlDown).Select: end_row = ActiveCell.Row
'set data range
Set datarange_x = ActiveSheet.Range(Cells(Row_Index, Column_Index - 1), Cells(end_row, _
Column_Index - 1))
Set datarange_y = ActiveSheet.Range(Cells(Row_Index, Column_Index), Cells(end_row, _
Column_Index))
Set MyChart = ActiveSheet.ChartObjects.Add(500, 150, 400, 230).Chart
MyChart.ChartType = xlXYScatter: MyChart.SeriesCollection.NewSeries
MyChart.SeriesCollection(1).XValues = datarange_x: MyChart.SeriesCollection(1).Values = datarange_y
With MyChart 'Chart Title
.HasTitle = True
.ChartTitle.Text = MyChartTitle
End With
With MyChart.Axes(xlValue) 'Y-axis Title
.HasTitle = True
With .AxisTitle
.Caption = MyYAxisTitle
End With
End With
With MyChart.Axes(xlCategory) 'X-axis Title
.HasTitle = True
With .AxisTitle
.Caption = MyXAxisTitle
End With
End With
With MyChart 'Delete the legends
.HasLegend = False
End With
End Sub
```

Enjoy!