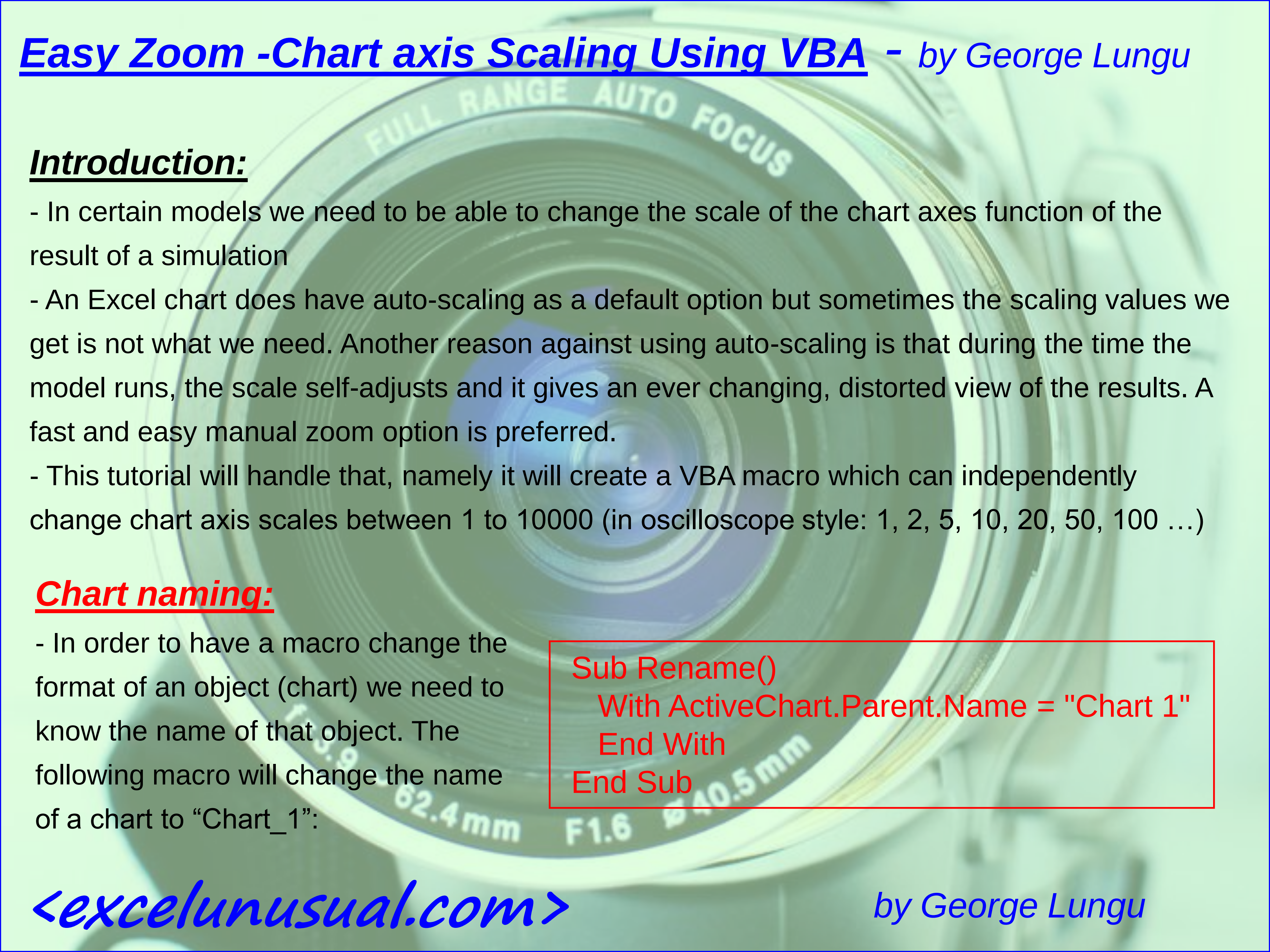


Easy Zoom -Chart axis Scaling Using VBA - by George Lungu

Introduction:

- In certain models we need to be able to change the scale of the chart axes function of the result of a simulation

- An Excel chart does have auto-scaling as a default option but sometimes the scaling values we get is not what we need. Another reason against using auto-scaling is that during the time the model runs, the scale self-adjusts and it gives an ever changing, distorted view of the results. A fast and easy manual zoom option is preferred.

- This tutorial will handle that, namely it will create a VBA macro which can independently change chart axis scales between 1 to 10000 (in oscilloscope style: 1, 2, 5, 10, 20, 50, 100 …)

Chart naming:

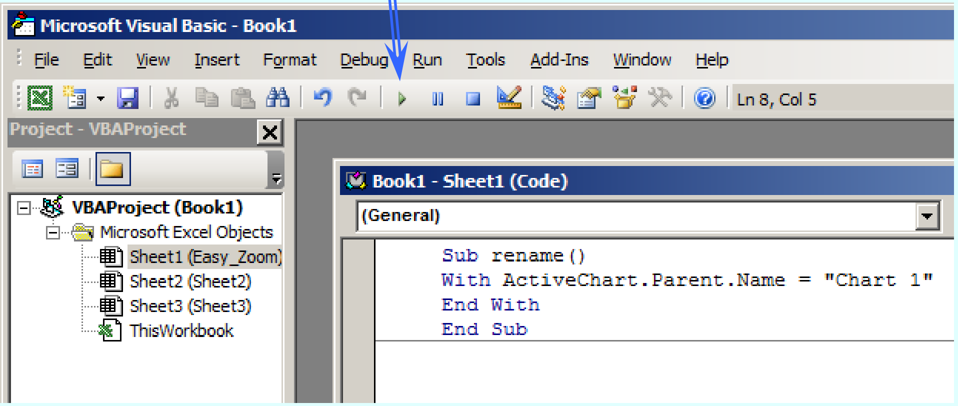
- In order to have a macro change the format of an object (chart) we need to know the name of that object. The following macro will change the name of a chart to “Chart_1”:

```
Sub Rename()
   With ActiveChart.Parent.Name = "Chart 1"
   End With
End Sub
```

How to use the macro:

- Bring up the VBA editor (Alt+F11) and find the “Rename” macro
- Select the chart you need to rename in the worksheet
- Place the cursor inside the macro and click, then click the “Run Sub/User Form” button once
- After this the chart is renamed. If you need to rename another chart make sure you change the name “Chart_1” into something else

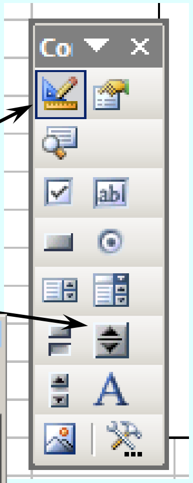
“Run Sub/User Form” button on the VBA editor menu

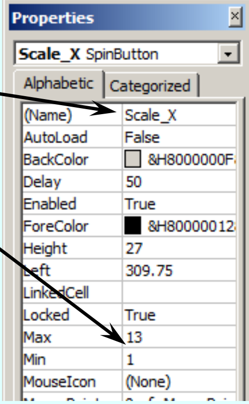


```
Sub rename()
With ActiveChart.Parent.Name = "Chart 1"
End With
End Sub
```

The “Scale_X” macro:

- Bring up the control toolbox to create a button: View => Toolbars => Control Toolbox
- Enter the “design mode” by clicking the upper left icon of the control toolbox
- Drag draw a spin button, then right click it and bring up the “properties menu”
- In the properties box change the (Name) to Scale_X
- In the properties box change the “Min” to 1 and the “Max” to 13
- Exit the design mode” by clicking the upper left icon of the control toolbox if you want to be able to use the macro





(Name)	Scale_X
AutoLoad	False
BackColor	□ &H8000000F
Delay	50
Enabled	True
ForeColor	■ &H80000012
Height	27
Left	309.75
LinkedCell	
Locked	True
Max	13
Min	1
MouseIcon	(None)

<www.excelunusual.com>

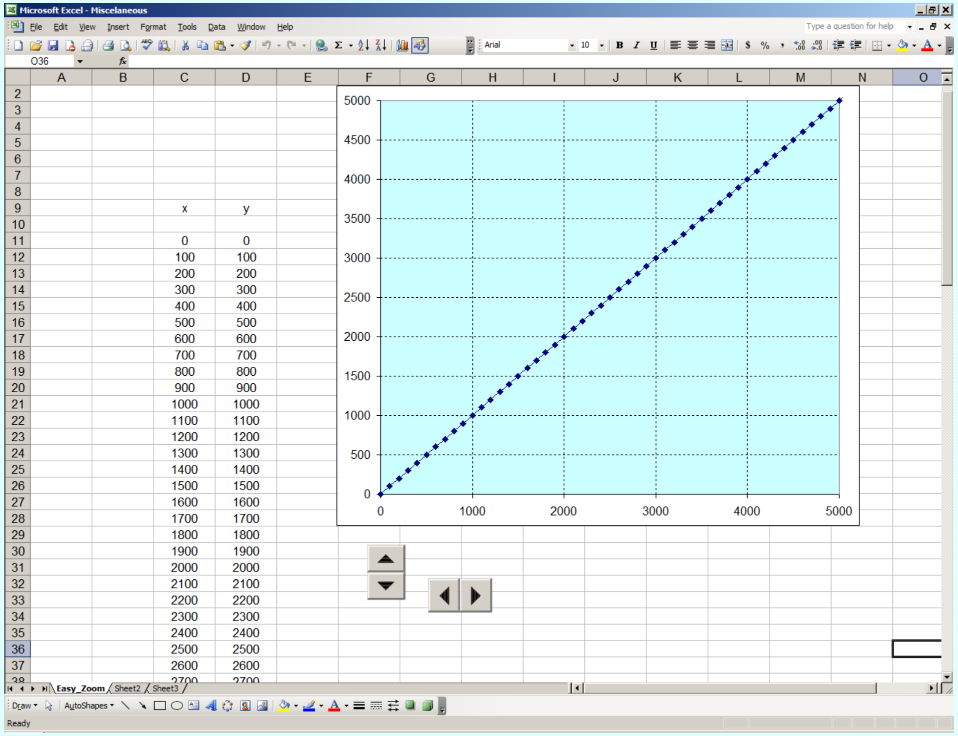
- Create a new button with the same properties as the previous one and name it Scale_Y
- After creating the buttons write the following two macros in the VBA editor under the “Rename” macro (make sure to place the “aX” and “aY” variable declarations on top of the page above all the macros otherwise you get an error when you try to run the macro):

```
Dim aX As Integer
--------------------------------------------------------------
Private Sub Scale_X_Change()
    If Scale_X.Value = 1 Then aX = 1
    If Scale_X.Value = 2 Then aX = 2
    If Scale_X.Value = 3 Then aX = 5
    If Scale_X.Value = 4 Then aX = 10
    If Scale_X.Value = 5 Then aX = 20
    If Scale_X.Value = 6 Then aX = 50
    If Scale_X.Value = 7 Then aX = 100
    If Scale_X.Value = 8 Then aX = 200
    If Scale_X.Value = 9 Then aX = 500
    If Scale_X.Value = 10 Then aX = 1000
    If Scale_X.Value = 11 Then aX = 2000
    If Scale_X.Value = 12 Then aX = 5000
    If Scale_X.Value = 13 Then aX = 10000
With ActiveSheet.ChartObjects("Chart 1").Chart
    .Axes(xlCategory).MinimumScale = 0
    .Axes(xlCategory).MaximumScale = aX
End With
End Sub
```

```
Dim aY As Integer
--------------------------------------------------------------
Private Sub Scale_Y_Change()
    If Scale_Y.Value = 1 Then aY = 1
    If Scale_Y.Value = 2 Then aY = 2
    If Scale_Y.Value = 3 Then aY = 5
    If Scale_Y.Value = 4 Then aY = 10
    If Scale_Y.Value = 5 Then aY = 20
    If Scale_Y.Value = 6 Then aY = 50
    If Scale_Y.Value = 7 Then aY = 100
    If Scale_Y.Value = 8 Then aY = 200
    If Scale_Y.Value = 9 Then aY = 500
    If Scale_Y.Value = 10 Then aY = 1000
    If Scale_Y.Value = 11 Then aY = 2000
    If Scale_Y.Value = 12 Then aY = 5000
    If Scale_X.Value = 13 Then aY = 10000
With ActiveSheet.ChartObjects("Chart 1").Chart
    .Axes(xlValue).MinimumScale = 0
    .Axes(xlValue).MaximumScale = aY
End With
End Sub
```

Excel screenshots:

- The worksheet is named “Easy_Zoom”
- The function on the chart is a plain f(x)= x function plotted in increments of 100 over 100 points within the C11:D110 range
- Before clicking the two buttons make sure you are out of design mode
- Verify the functionality by adjusting the buttons at different levels



x	y
0	0
100	100
200	200
300	300
400	400
500	500
600	600
700	700
800	800
900	900
1000	1000
1100	1100
1200	1200
1300	1300
1400	1400
1500	1500
1600	1600
1700	1700
1800	1800
1900	1900
2000	2000
2100	2100
2200	2200
2300	2300
2400	2400
2500	2500
2600	2600
2700	2700

Better Macros by the Use of Arrays:

Jean-Marc, a visitor to my blog suggested the following macros in a comment:

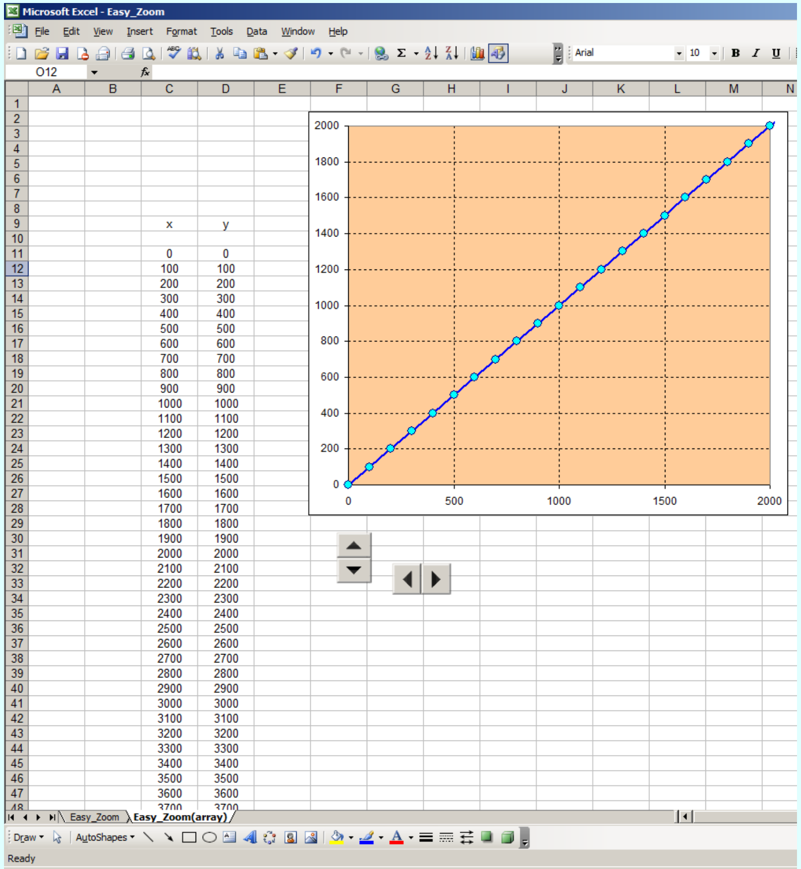
```
 Dim aX As Integer
 Dim aY As Integer
 Dim arrScale As Variant

Private Sub Scale_X_Change()
arrScale = Array(1, 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10000)
aX = arrScale(Scale_X.Value - 1)
With ActiveSheet.ChartObjects("Chart 1").Chart
.Axes(xlCategory).MinimumScale = 0
.Axes(xlCategory).MaximumScale = aX
End With
End Sub
```

- Here “arrScale” is an array set to a fixed value within the macro
- The array has 13 elements and any of them can be retrieved to be assigned to variable “aX”
- In our case we use the value of “Scale_X” as the array argument and since the button sets the value of “Scale_X” between 1 and 13 we can retrieve any of the array’s values between 1 and 10000
- Arrays have integer arguments starting with 0 (zero) and that’s why you see the -1 in the argument: **aX = arrScale(Scale_X.Value - 1)**

The second macro (the one scaling the Y axis looks similar):

```
Private Sub Scale_Y_Change()
arrScale = Array(1, 2, 5, 10, 20, 50, 100, 200, 500, 1000,
2000, 5000, 10000)
aY = arrScale(Scale_Y.Value - 1)
With ActiveSheet.ChartObjects("Chart 1").Chart
.Axes(xlValue).MinimumScale = 0
.Axes(xlValue).MaximumScale = aY
End With
End Sub
```

- A new worksheet called Easy_Zoom(array) has been created, which is identical to the old one but it’s using the new array based VBA code.



x	y
0	0
100	100
200	200
300	300
400	400
500	500
600	600
700	700
800	800
900	900
1000	1000
1100	1100
1200	1200
1300	1300
1400	1400
1500	1500
1600	1600
1700	1700
1800	1800
1900	1900
2000	2000
2100	2100
2200	2200
2300	2300
2400	2400
2500	2500
2600	2600
2700	2700
2800	2800
2900	2900
3000	3000
3100	3100
3200	3200
3300	3300
3400	3400
3500	3500
3600	3600
3700	3700

I used the Select-Case statement before to accomplish the same thing but the macro code is even more bloated than the one with several “IF” statements. Out of the three the array option is the best.

Thanks, Jean-Marc!