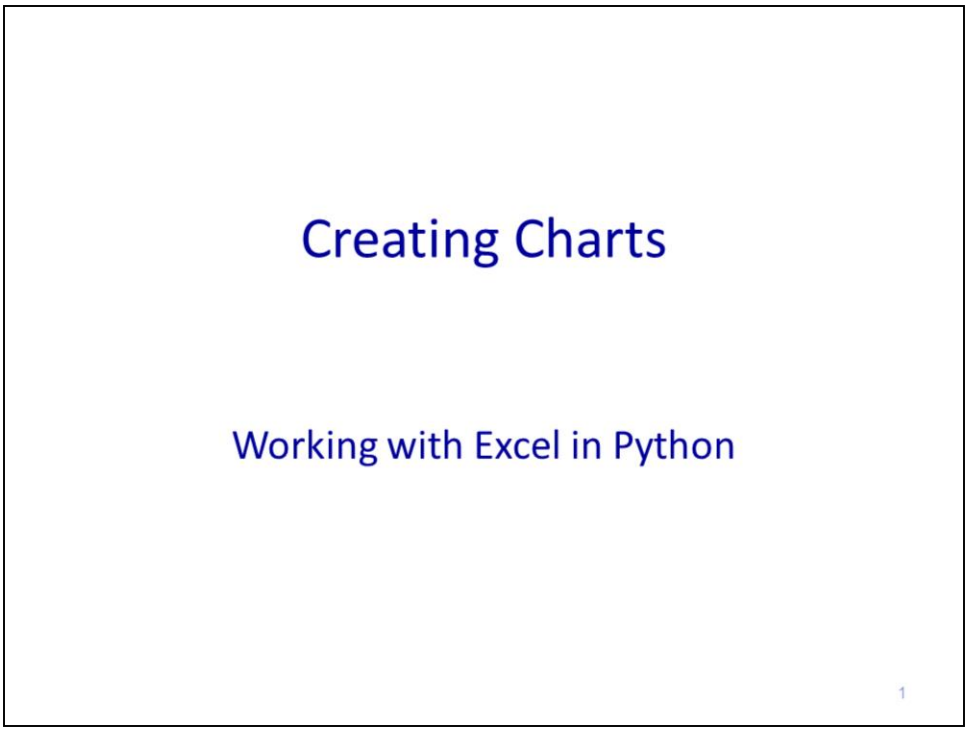
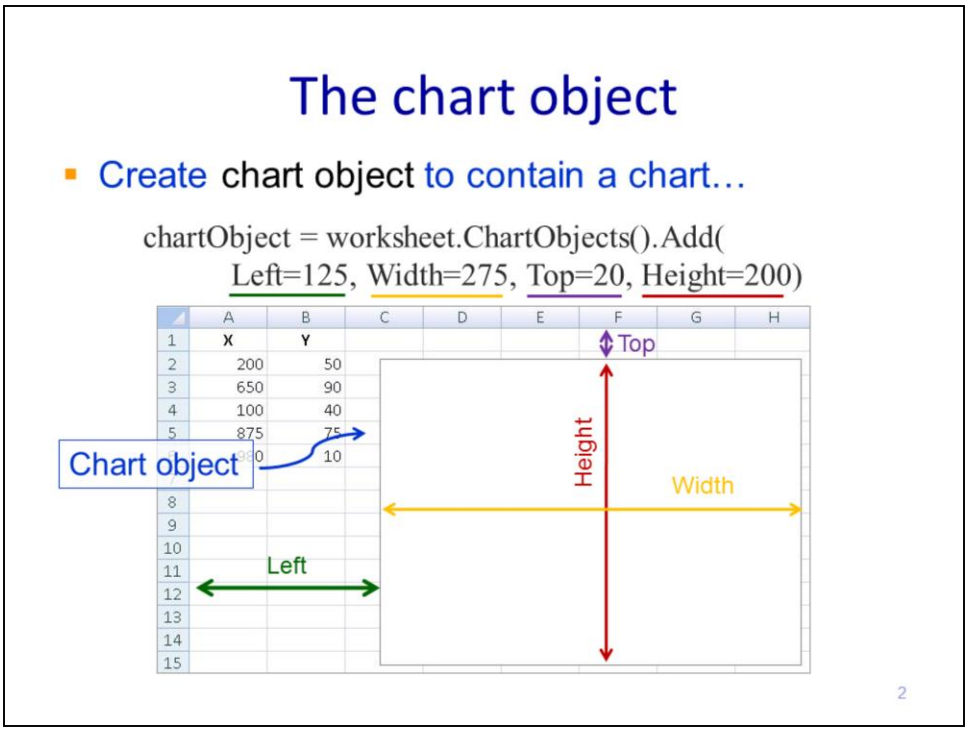


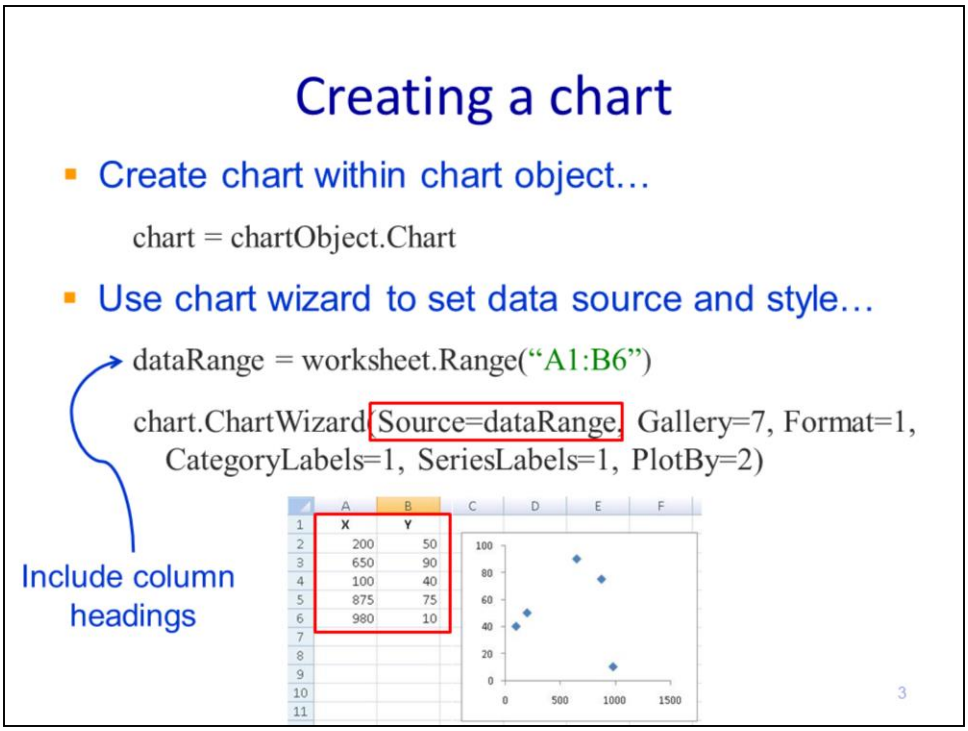
This video will discuss how to create charts in an Excel worksheet.



The first step in creating a chart in an Excel worksheet is to create the **chart object** using the **ChartObjects** method of the worksheet.

The chart object will contain the chart. The locations and dimensions of the chart object should be specified as the parameters of the ChartObjects method.

The reference points for these parameters are illustrated here. Determining the values for the parameters usually requires a trial and error approach.

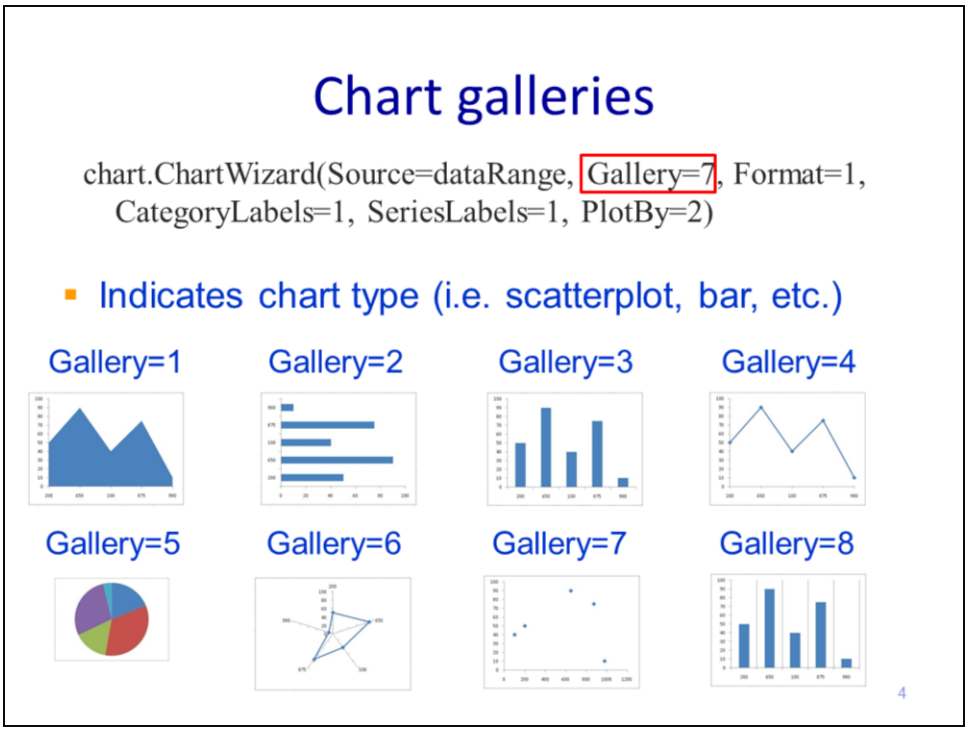


Once a chart object has been created, the object's **Chart** property will give access to the actual chart. At this point, the chart will be blank.

We can use the chart wizard to set the data and style of the chart.

Here, I've assigned a variable to the cell object that corresponds to the data source – this will help keep the chart wizard statement more concise. Note that a row or column object can also be specified for the data range. Also note that the x-axis values should be located in the leftmost column of the data range. Each column to the right of the x-axis will correspond to a separate data series.

The **ChartWizard** method allows the data source and chart style to be specified. There are a number of parameters for this method which we'll discuss in the following slides. Note I'll use keywords to specify the parameters.



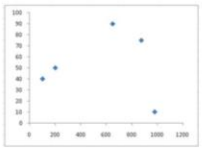
The **Gallery** parameter of the chart wizard specifies the type of chart that will be created.

The available options are shown here with their corresponding gallery code.

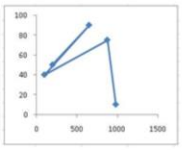
Chart formats

chart.ChartWizard(Source=dataRange, Gallery=7, Format=1, CategoryLabels=1, SeriesLabels=1, PlotBy=2)

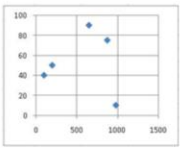
Format=1



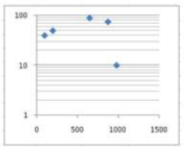
Format=2



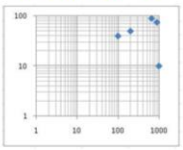
Format=3



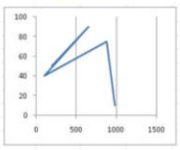
Format=4



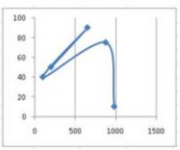
Format=5



Format=6



Format=7



5

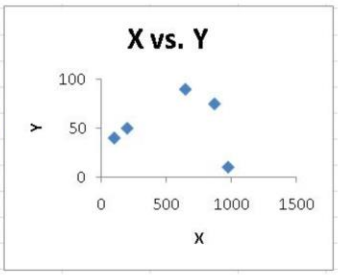
The format parameter determines the style of the chart. The styles depend on the type of the chart.

This slide shows the formats available for the scatterplot chart type.

Chart titles and axis labels

- Enable chart title and edit text…

```
chart.HasTitle = True
chart.ChartTitle.Text = "X vs. Y"
```



- Enable x-axis title and edit text..

```
chart.Axes(1).HasTitle = True
chart.Axes(1).AxisTitle.Text = "X"
```

- Enable y-axis title and edit text…

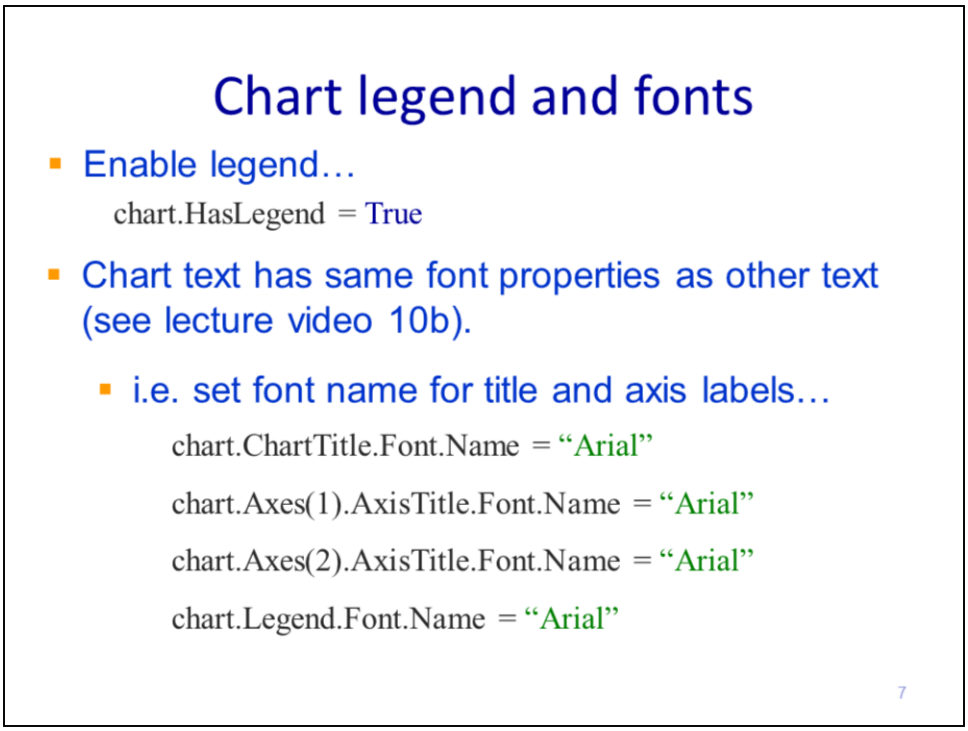
```
chart.Axes(2).HasTitle = True
chart.Axes(2).AxisTitle.Text = "Y"
```

6

This slide will show how to turn on and set the chart title and axis labels.

The chart's **HasTitle** property can be set to display the title. The title can be specified by through the **Text** property of the ChartTitle retrieved from the chart.

The x-axis and y-axis titles can be turned on through the axis object's HasTitle property. The values for the axis titles can be set through the text property of the **AxisTitle**. Note that the x-axis corresponds to Axes(1) and the y-axis corresponds to Axes(2).



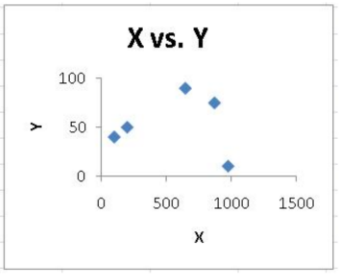
The chart legend can be displayed through the chart's **HasLegend** property.

The text in the chart object can be set through the same properties as any other text in the spreadsheet.

This example sets the font name to "Arial" for the chart title, the x- and y-axis labels, and the legend.

Example: creating a chart

```
chart = chartObject.Chart

dataRange = worksheet.Range("A1:B6")

chart.ChartWizard(Source=dataRange, Gallery=7, Format=1,
   CategoryLabels=1, SeriesLabels=1, PlotBy=2)

chart.HasTitle = True
chart.ChartTitle.Text = "X vs. Y"

chart.Axes(1).HasTitle = True
chart.Axes(1).AxisTitle.Text = "X"

chart.Axes(2).HasTitle = True
chart.Axes(2).AxisTitle.Text = "Y"
```



8

This slide will show an example script that creates a chart in Excel.

The first statement gets the chart from the chart object (which was created previously).

The data range for the source data is cells A1 through B6.

The chart wizard is used to plot the data for the data range specified. The gallery code of 7 corresponds to a scatterplot. The format code of 1 corresponds to linear axis labels with no lines connecting the points. The CategoryLabels and SeriesLabels parameters are set to default settings. The PlotBy parameter value of 2 indicates that the data are arranged in columns.

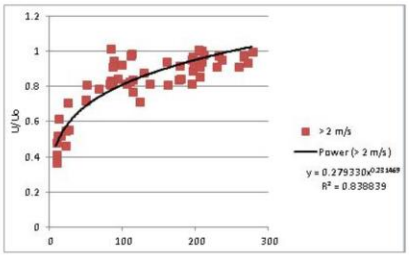
The chart title is set to display “X vs. Y”.

The x-axis title is set to display “X”.

The y-axis title is set to display “Y”.

Steps to add a trendline to a graph

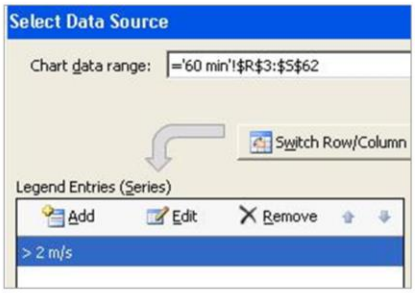
- [x] Get a chart
2. Get a data series for chart
3. Get trendlines set for data series
4. Add new trendline
5. Set trendline name
6. Display equation
7. Display r-squared



9

The remainder of this video will discuss how to add a trendline to a chart. This slide shows a checklist of the steps required to add a trendline. The first step, which we've already discussed, is to create and plot the chart.

Chart data series



- To get a data series from chart…

dataSeries = chart.SeriesCollection(1)

series number

10

The second step in adding a trendline is to get the data series from the chart that will be used to fit the trendline. The SeriesCollection number corresponds to the order of the data series. Assuming the data are arranged in columns, then the series number corresponds to the order of the columns - the first column after the x-axis column is series 1, the second column is series 2, and so on.

Adding a trendline for a data series

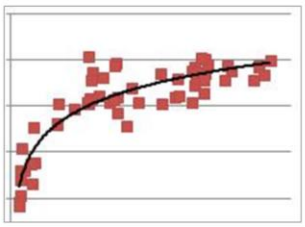
- Get set of trendlines…

```
trendlines = dataSeries.Trendlines()
```

- Add trendline to set…

```
trendline = trendlines.Add (Type = -4133)
```

see next slide for options



11

The Trendlines method of the data series is used to get the trendlines object.

This object's Add method will create a new trendline within the object.

The trendline type options will be discussed on the following slide

The trendline will be displayed in the chart after it has been added to the trendlines object.

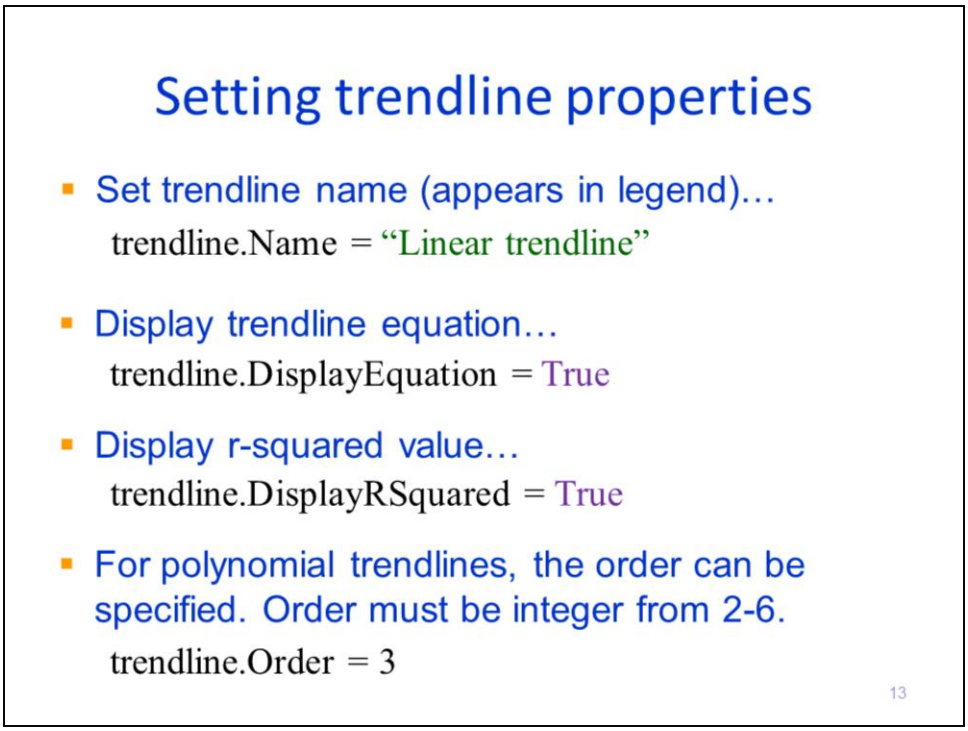
Trendline types

trendline = trendlines.Add (Type = -4133)

Name	Value	Description
xlExponential	5	Uses an equation to calculate the least squares fit through points, for example, y=ab^x .
xlLinear	-4132	Uses the linear equation y = mx + b to calculate the least squares fit through points.
xlLogarithmic	-4133	Uses the equation y = c ln x + b to calculate the least squares fit through points.
xlMovingAvg	6	Uses a sequence of averages computed from parts of the data series. The number of points equals the total number of points in the series less the number specified for the period.
xlPolynomial	3	Uses an equation to calculate the least squares fit through points, for example, y = ax^6 + bx^5 + cx^4 + dx^3 + ex^2 + fx + g.
xlPower	4	Uses an equation to calculate the least squares fit through points, for example, y = ax^b.

12

The chart on this slide shows the types of trendlines that are available along with their type code. This code is specified when adding a new trendline to the trendline object. The example statement fits a logarithmic trendline to the data series.



The properties for an individual trendline can be set. Note that the trendline object is used here, not the trendlines object. The name of the trendline can be set as it will appear in the legend.

The trendline equation can be displayed.

The r-squared can be displayed.

For polynomial trendline types (type = 3), the polynomial order (i.e. 2-6) can be specified.

Example: adding a trendline to a chart

```
# get set of trendlines for data series 1…
trendlines = chart.SeriesCollection(1).Trendlines()

# add new trendline…
trendline = trendlines.Add(Type = -4132)

# set trendline name…
trendline.Name = "Linear trend"

# display equation…
trendline.DisplayEquation = True

# display r-squared…
trendline.DisplayRSquared = True
```

14

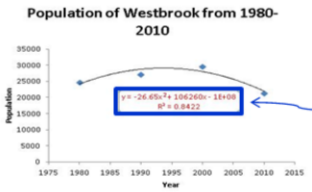
This slide will show an example script that fits a trendline to a data series in a chart. The first statement gets the set of trendlines corresponding to data series 1 in the chart.

The Add method of the trendlines object is used to fit a linear trendline (type = -4132) to the data series.

The trendline name is set to “Linear trend”

The trendline equation and r-squared value are set to display in the chart.

Trendline label



label contains equation and r-squared (if enabled)

- Get trendline label…

  trendLabel = trendline.DataLabel

- Data label font can be changed (see 10b lecture video)…

  i.e. trendLabel.Font.ColorIndex = 3

- Label position can be changed…

  trendLabel.Top = 100 trendLabel.Left = 50

- Number of decimal places can be changed…

  trendLabel.NumbeFormat = "0.000"

15

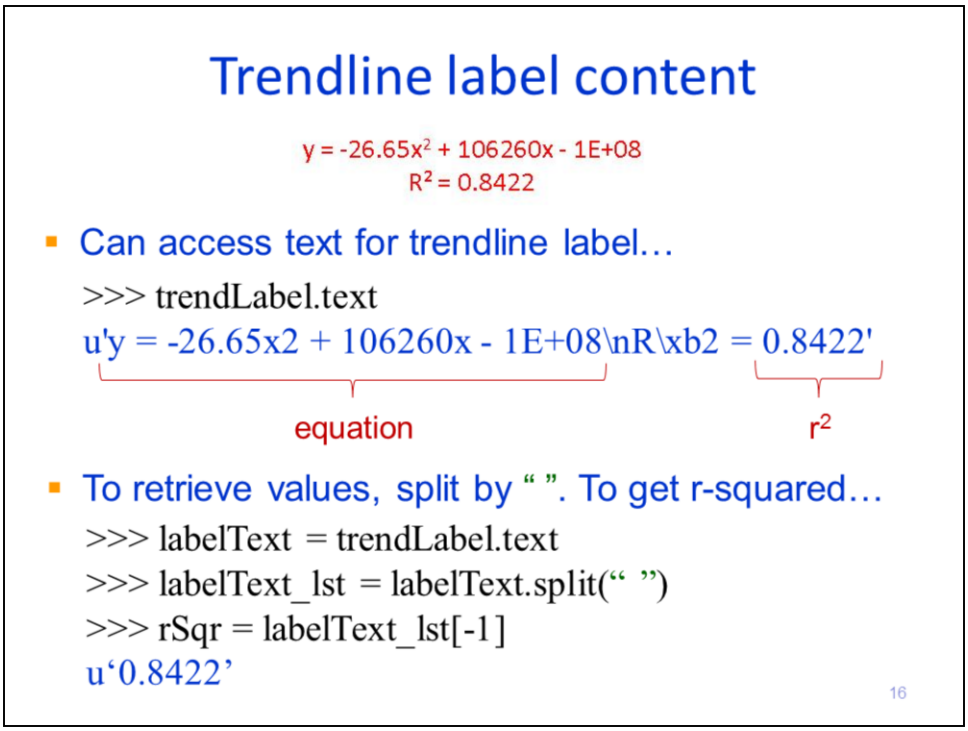
The trendline label contains the equation and r-squared. If either the equation or the r-squared are set to display, then the trendline label can be accessed.

The trendline **DataLabel** property will retrieve the label object.

The label's fonts can be changed like any other text in Excel. In this example, the font color is changed to red (ColorIndex = 3)

The label position can be changed through the **Top** and **Left** properties of the label object. Positions are specified relative to the edge of the chart object and have units of points.

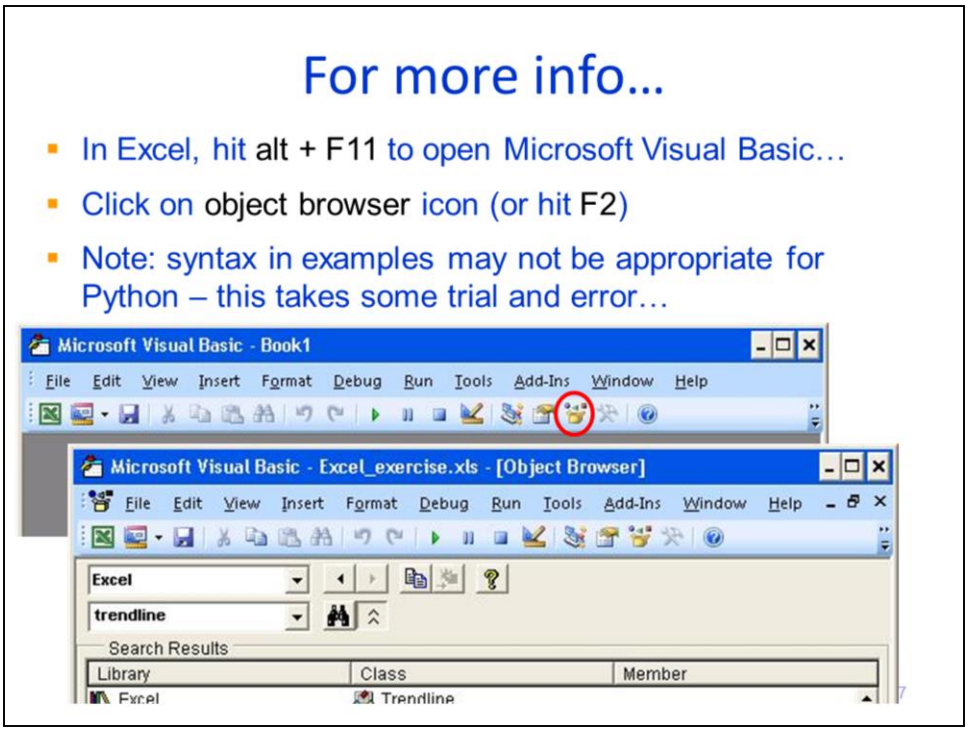
The number of digits after the decimal can be specified through the **NumberFormat** property.



This slide shows example statements that retrieve the text from the trendline label.

The equation and r-squared can be retrieved through the label's **text** property. The text is returned as a string.

The values can be extracted from the text by splitting the string. The example here shows how to extract the r-squared value. The coefficients from the equation can also be extracted but usually require multiple splits, with different characters, depending on the type of trendline.



You can get further information on how to work with Excel in Python by accessing the Microsoft Visual Basic library.

In Excel, the library can be accessed by typing alt + F11 and then F2. Note that the documentation provides example syntax for Visual Basic – the syntax will need to be modified to be suitable for Python.