VBA coding for Excel

AMAN LOHARUKA


Lunch Time
Continue at 1pm

Today We're Going to Cover

Morning

- Introduction to Macros
- Recording Macros
- Editing Macros
- VBA Interface
- Macro Security
- Debugging
- Customizing Ribbon and Toolbars

Afternoon

- VB Grammar
- Inserting & Formatting Text
- Sorting Data
- Variables & Operators
- Duplicating Data
- Generating Reports
- Various Properties, Objects and Methods

Today We're Going to Cover

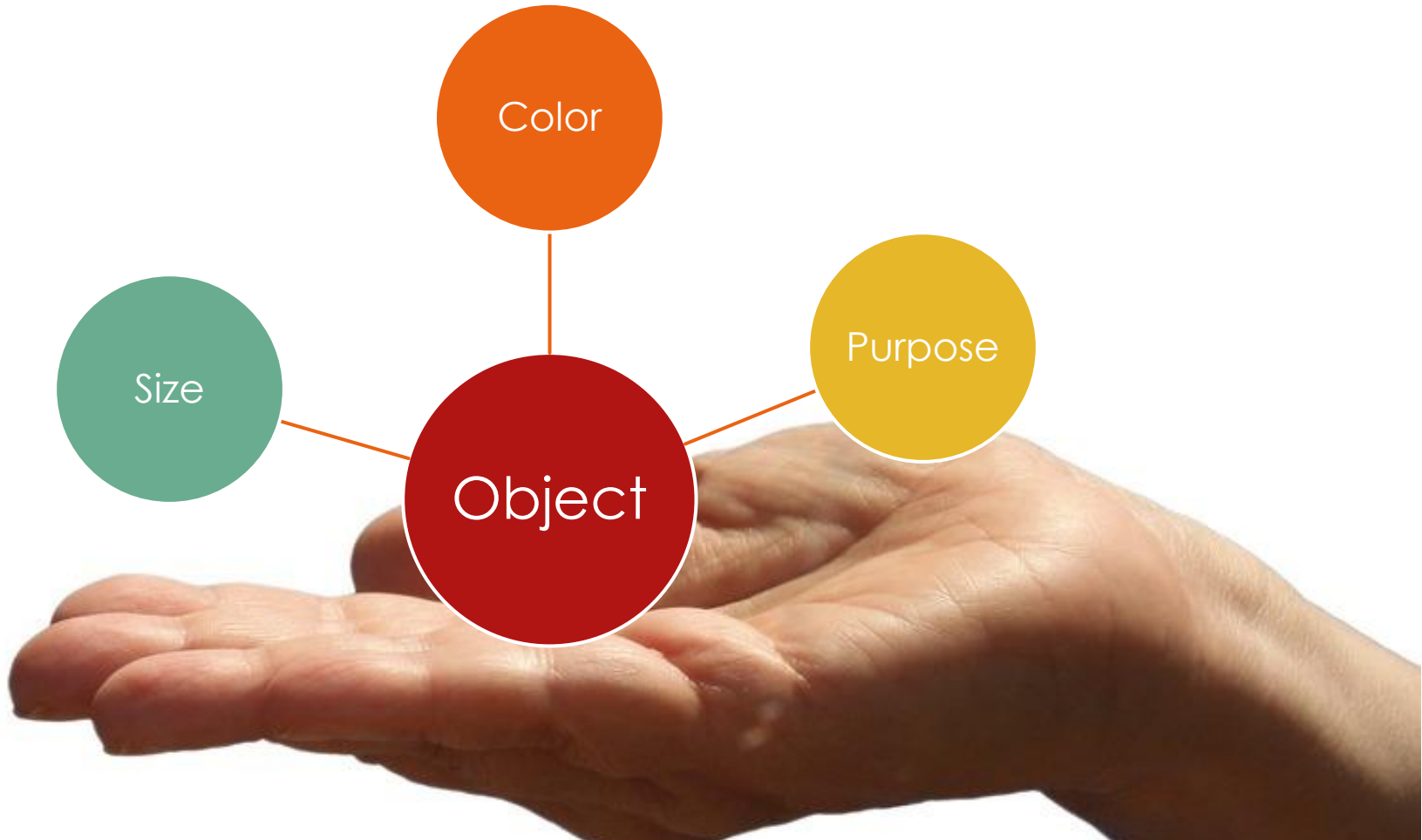
Morning

- Introduction to Macros
- Recording Macros
- Editing Macros
- VBA Interface
- Macro Security
- Debugging
- Customizing Ribbon and Toolbars

Afternoon

- VB Grammar
- Inserting & Formatting Text
- Sorting Data
- Variables & Operators
- Duplicating Data
- Generating Reports
- Various Properties, Objects and Methods

Object-Oriented Programming

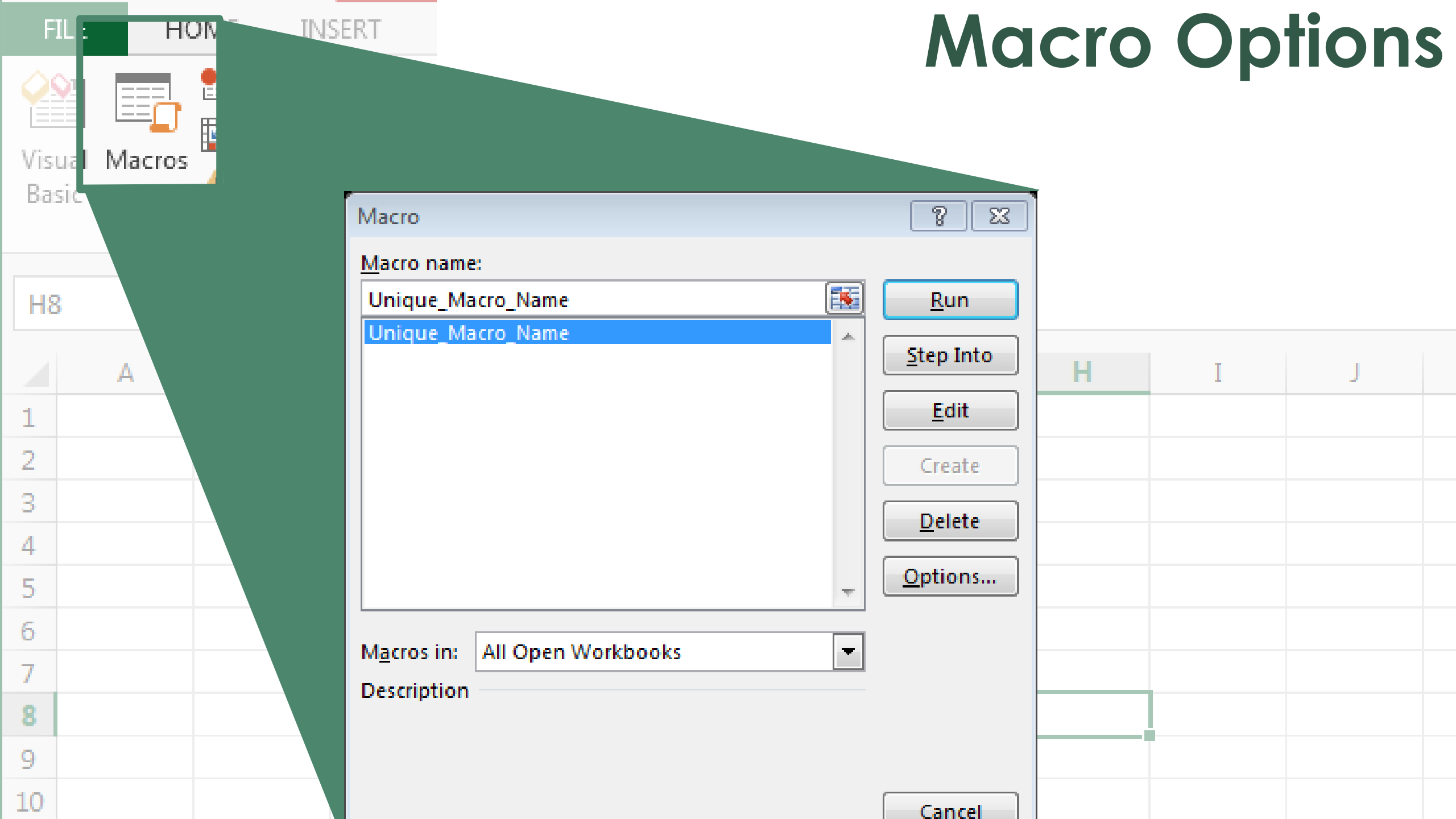


Modelled like real time objects

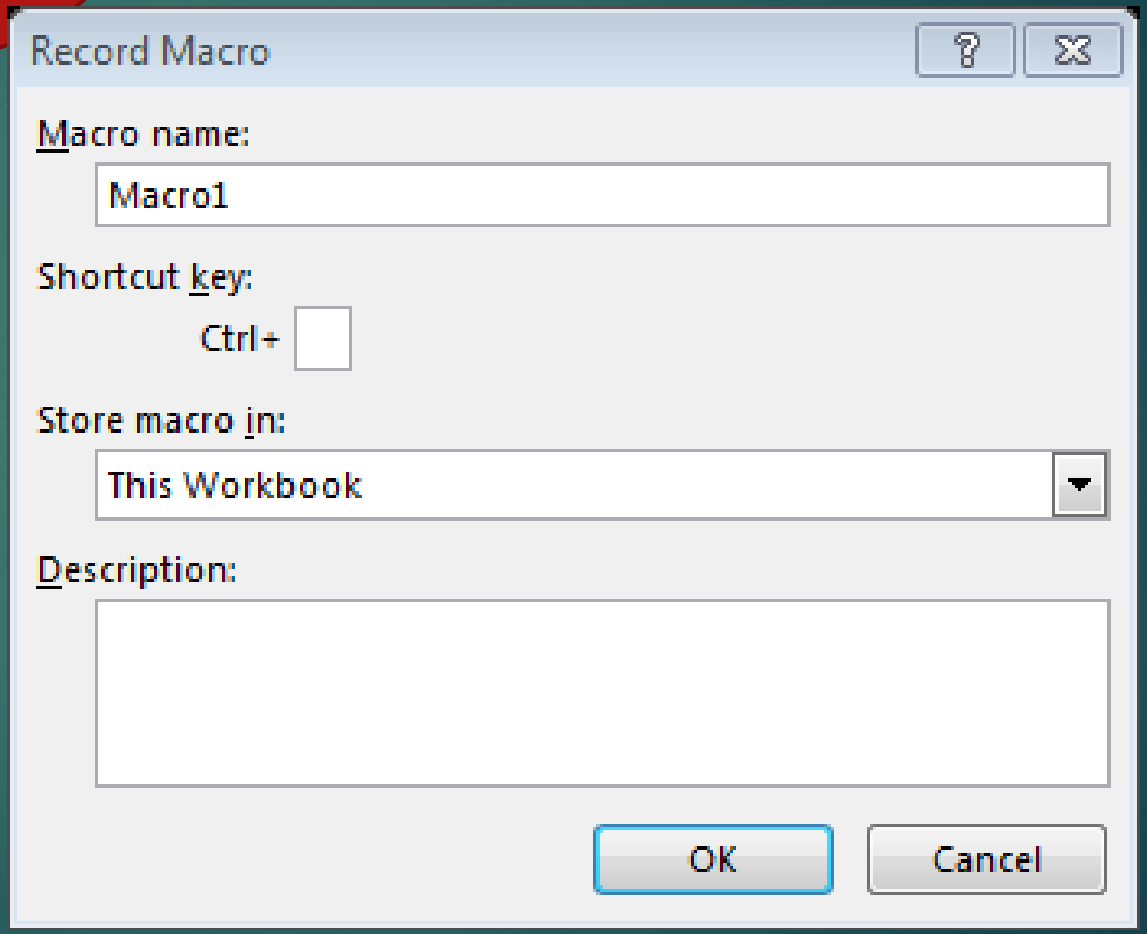
Macros

```
Public Sub Unique_Macro_Name()

Object.Property
Object.Method

'This is a Macro!


End Sub
```

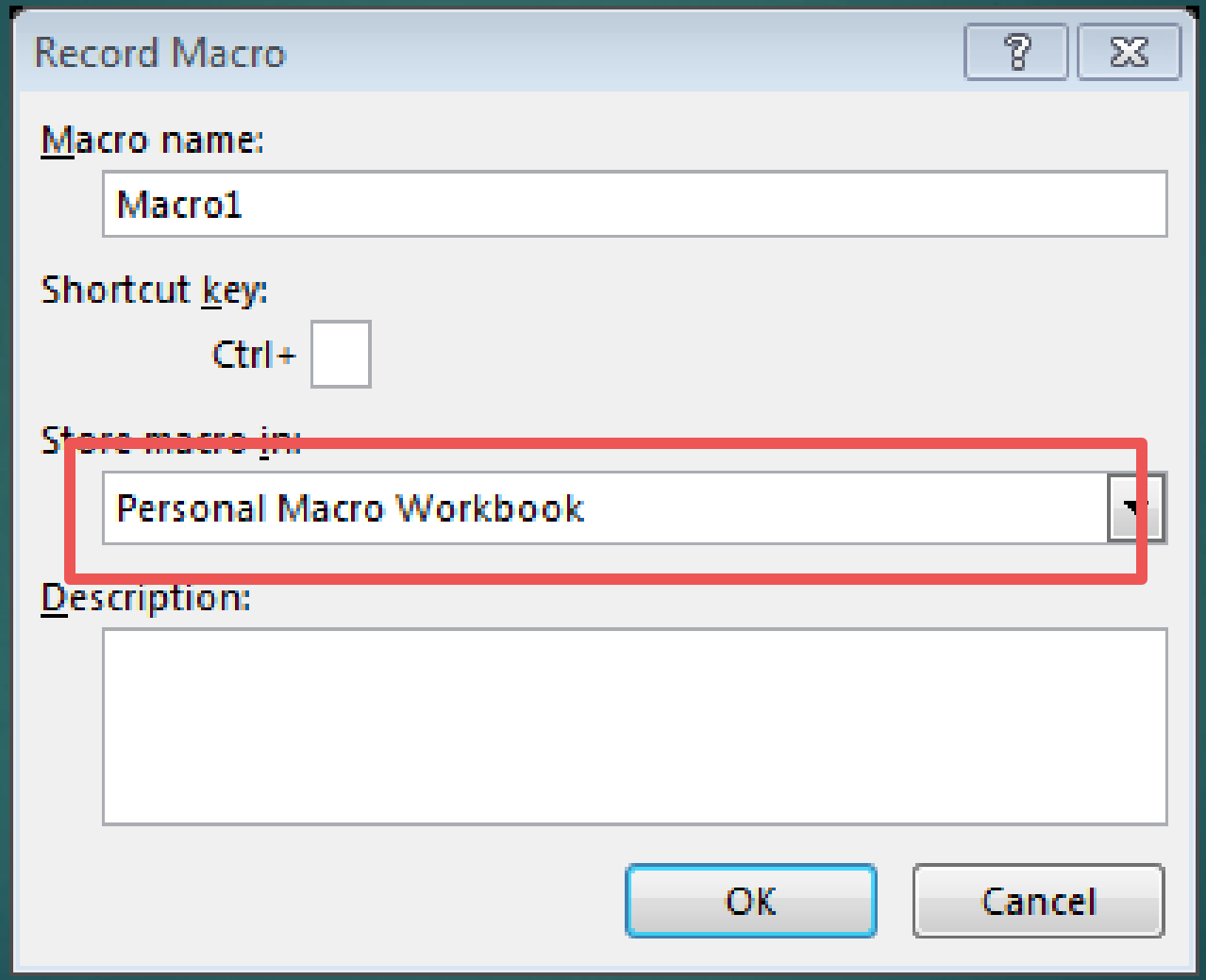
Macro Options



The Macro Recorder



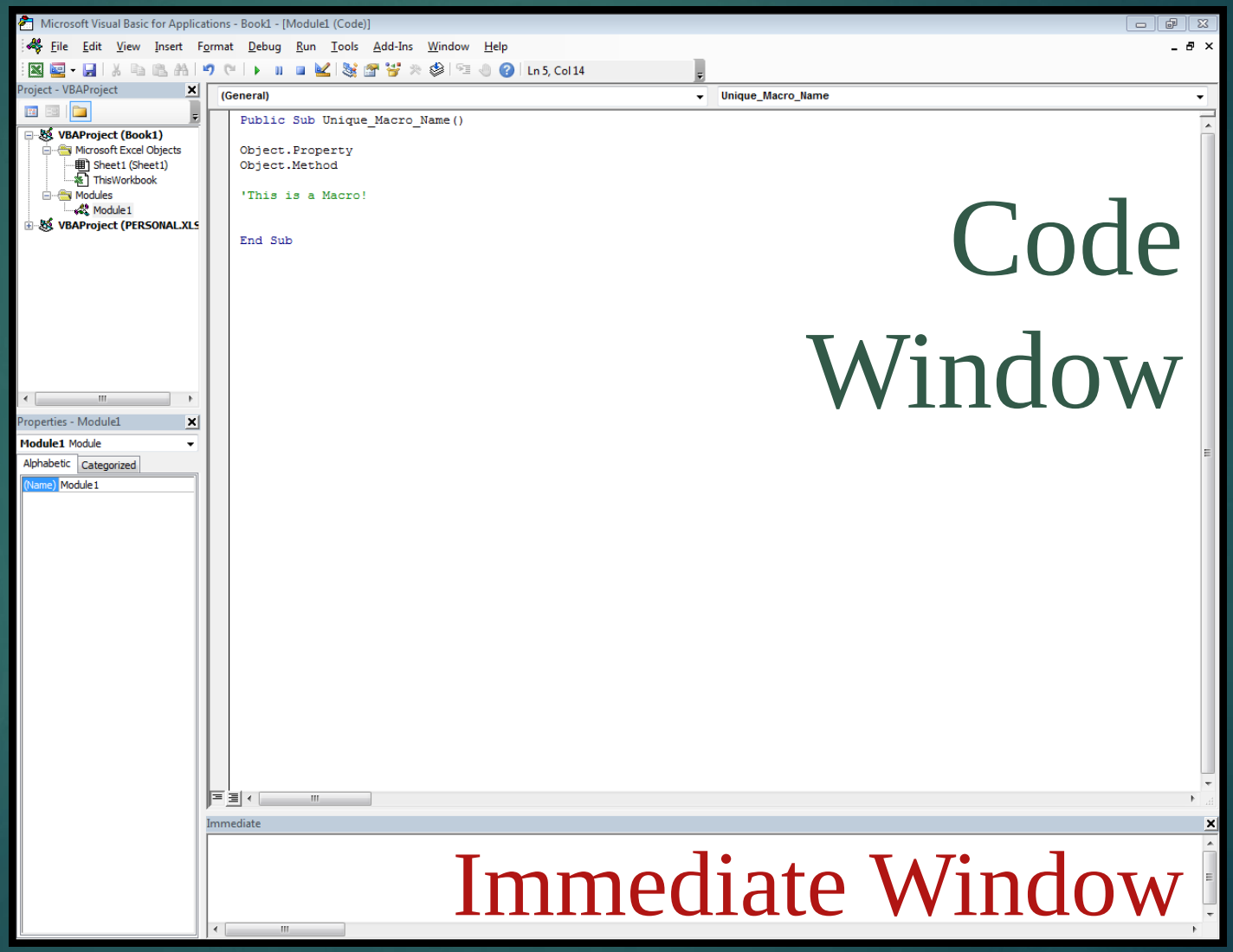
Personal Macro Workbook



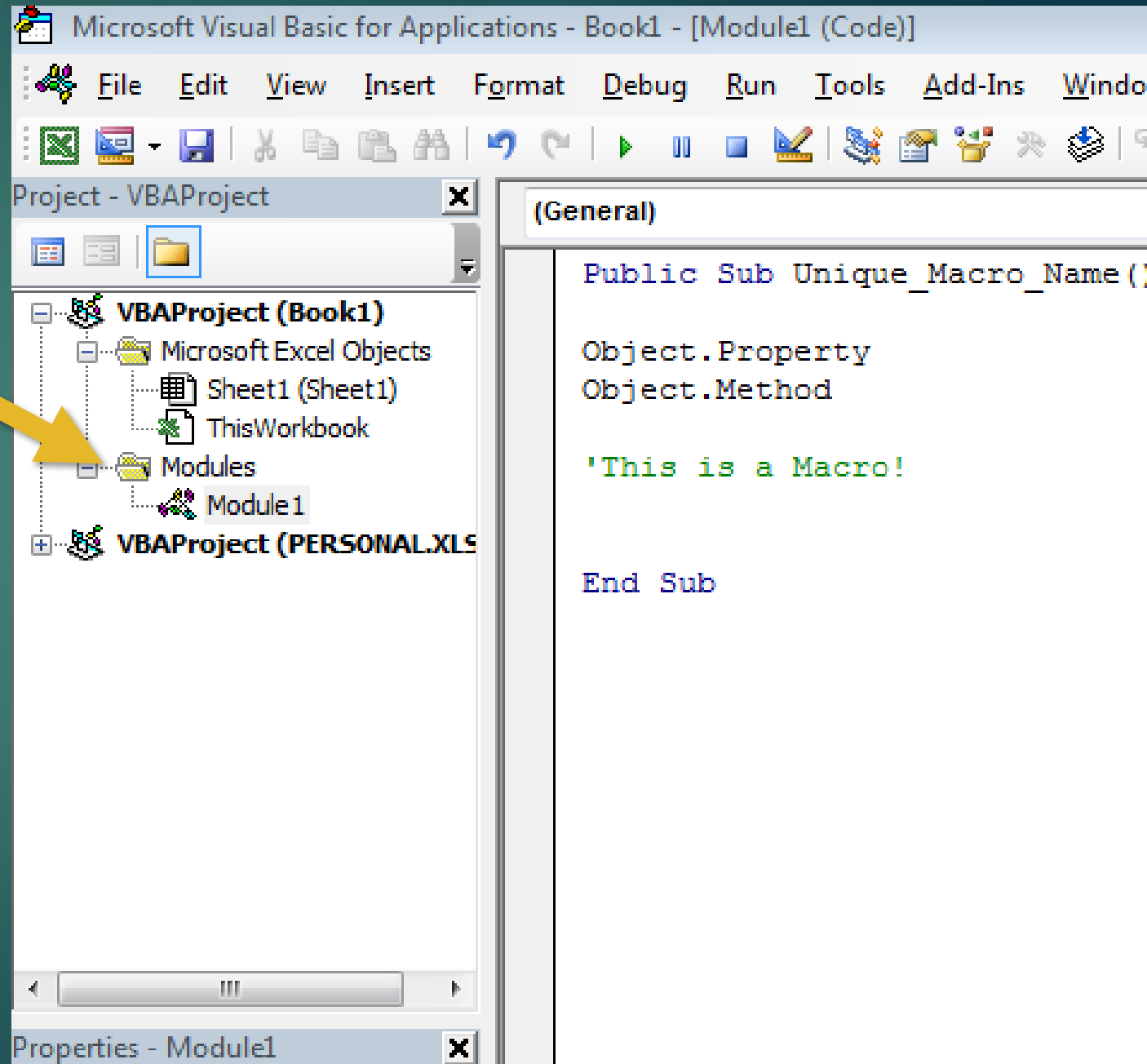
Visual Basic Editor

Project Explorer

Properties Window

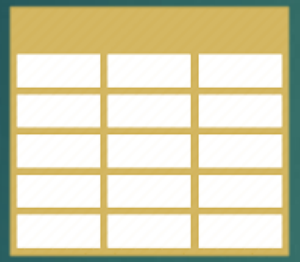


Modules


Microsoft Visual Basic for Applications - Book1 - [Module1 (Code)]
File Edit View Insert Format Debug Run Tools Add-Ins Windo
Project - VBAProject
VBAProject (Book1)
Microsoft Excel Objects
Sheet1 (Sheet1)
ThisWorkbook
Modules
Module1
VBAProject (PERSONAL.XLS
Properties - Module1
(General)
Public Sub Unique_Macro_Name()
Object.Property
Object.Method
'This is a Macro!
End Sub

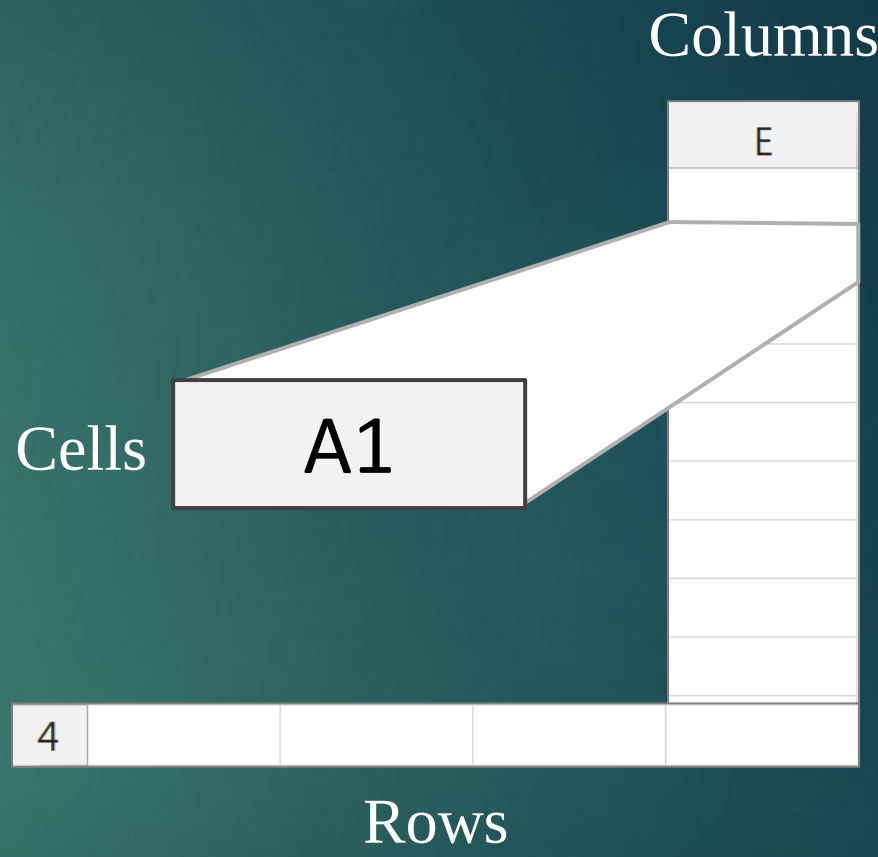
Objects

Sheets

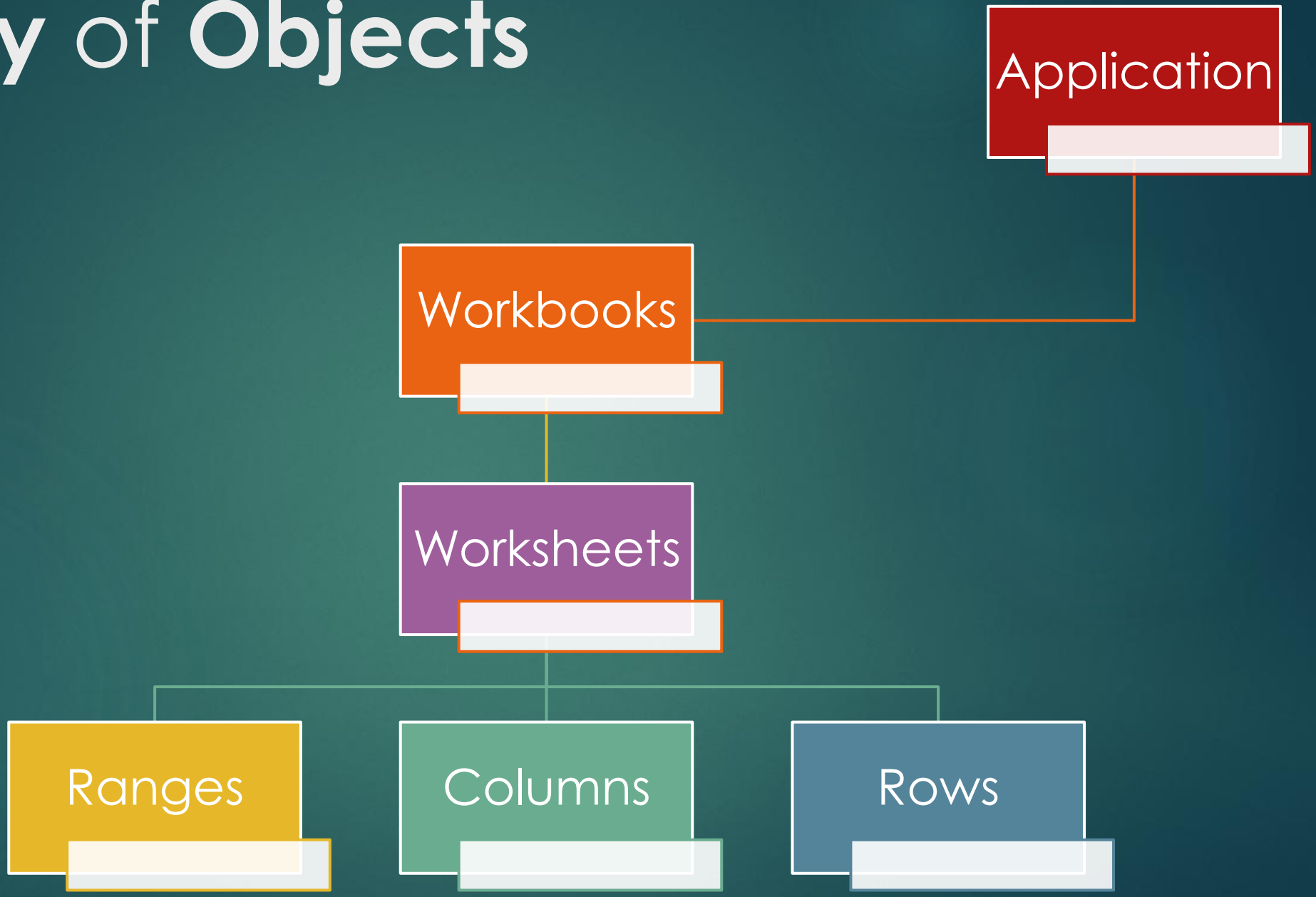


Tables

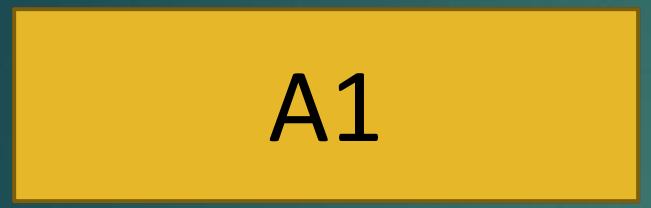
Charts



Hierarchy of Objects



Properties



.Color

.Value

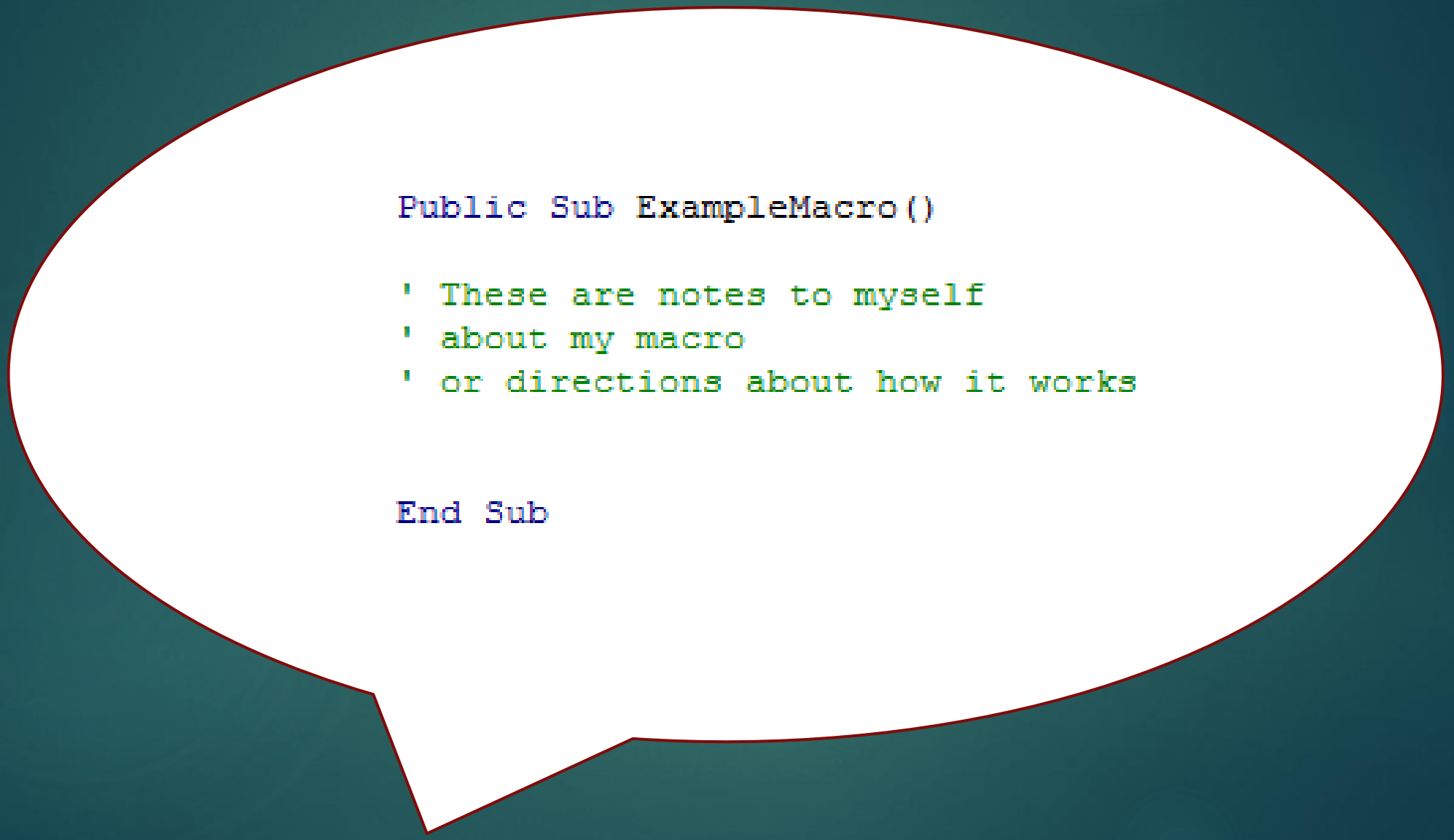
.Font

Methods

Copy

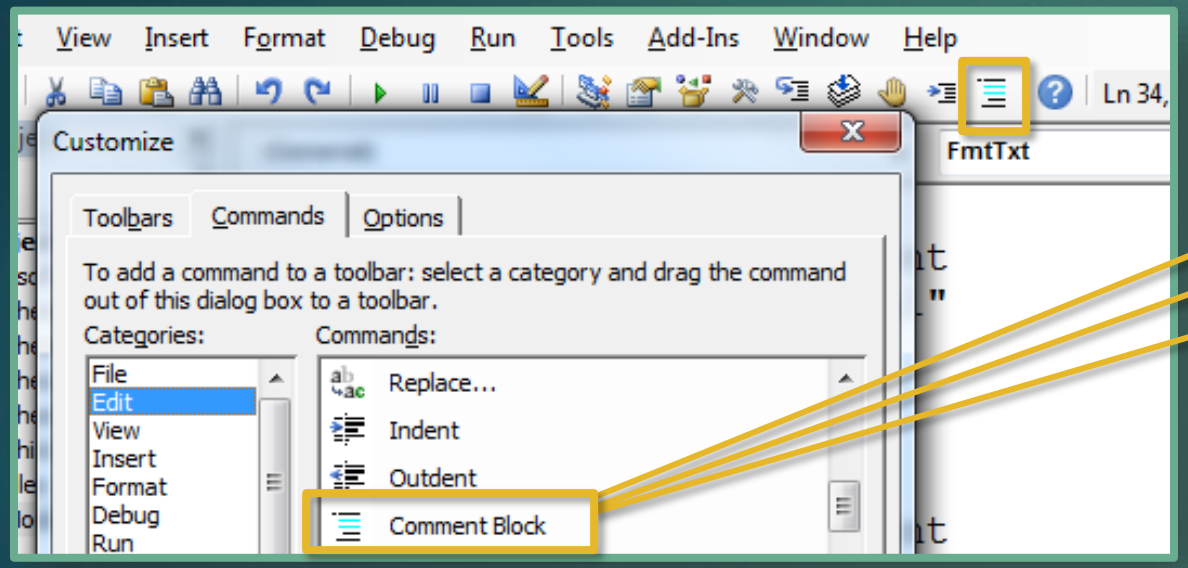
Clear Contents

VBA Comments



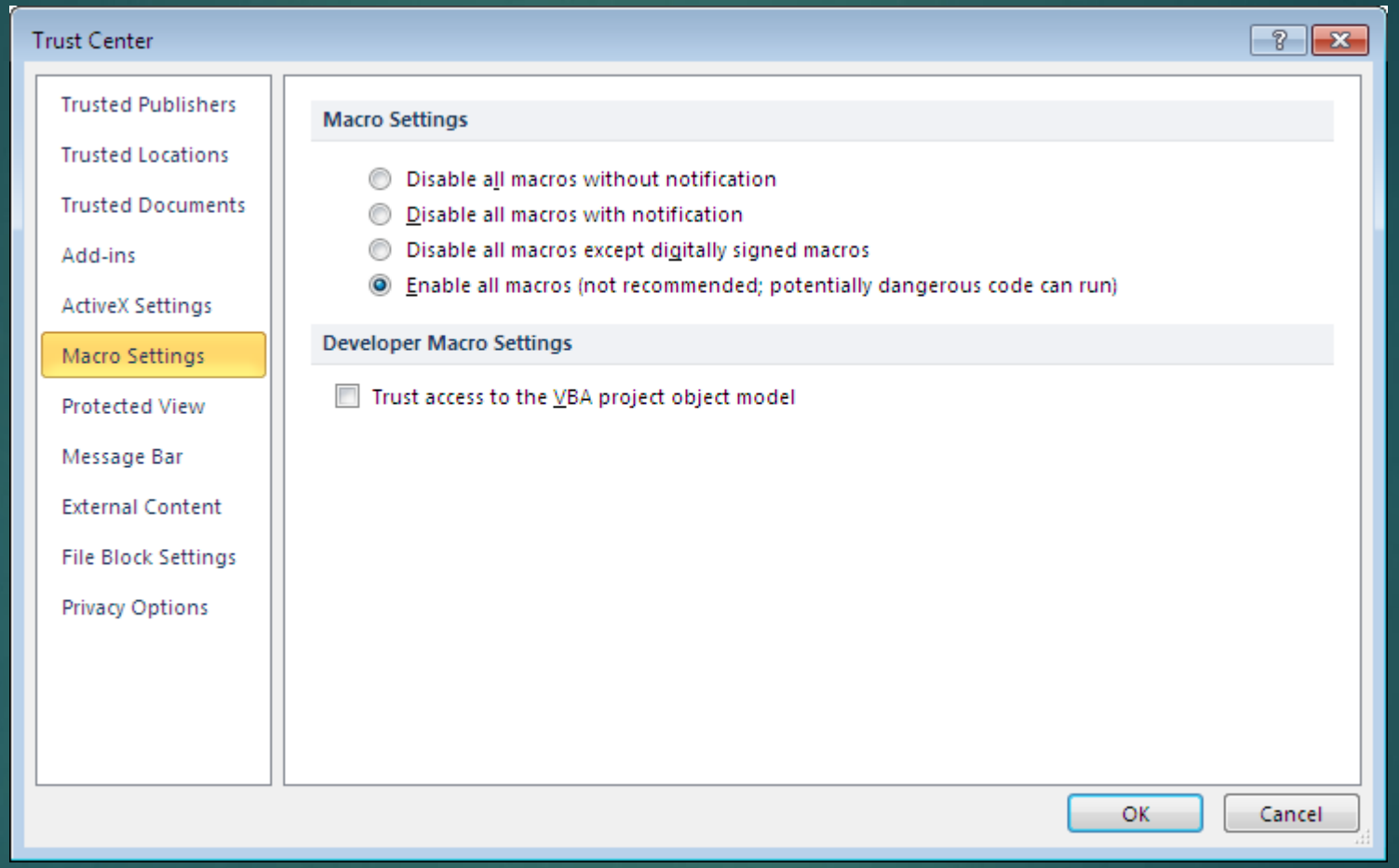
```
Public Sub ExampleMacro()

' These are notes to myself
' about my macro
' or directions about how it works


End Sub
```

“comment out” a block **all at once**

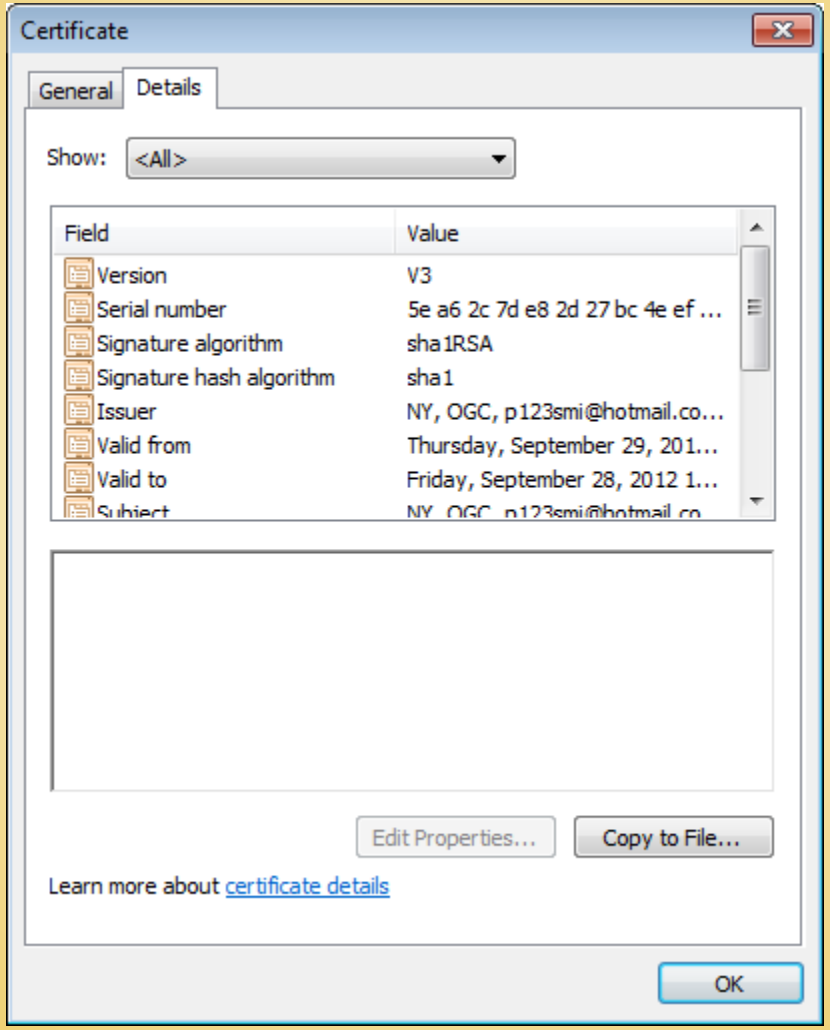



```
Public Sub ExampleMacro()

' These are notes to myself
' about my macro
' or directions about how it works


End Sub
```

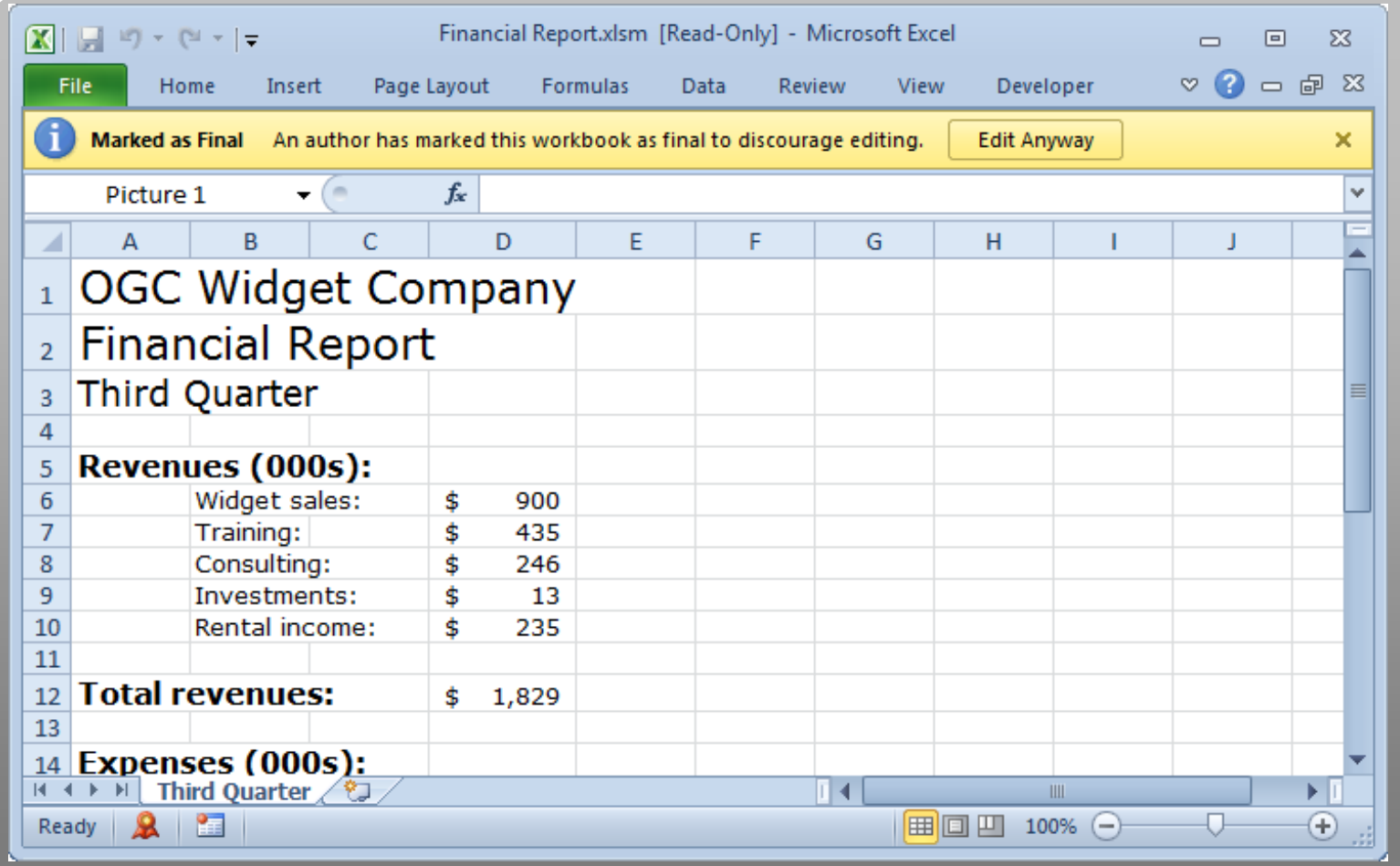
Macro Security Settings



Digital Certificates



Digital Signatures



The Debugging Process

Set up environment

Back up

Run code with error

Determine problem

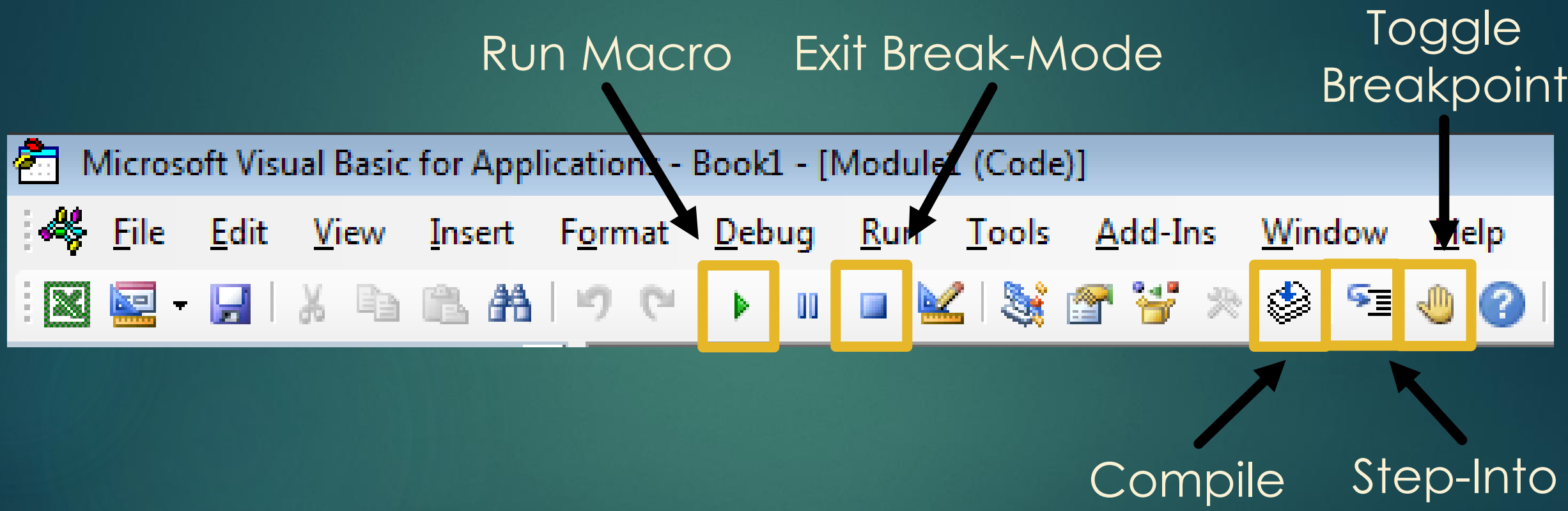
Reset Data

Execute code line by line to find the error

Fix code error

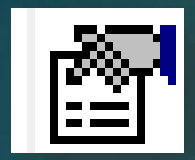
Retest to ensure error is fixed

Debugging Tools

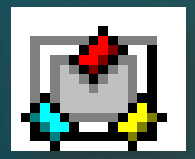


Intellisense Window

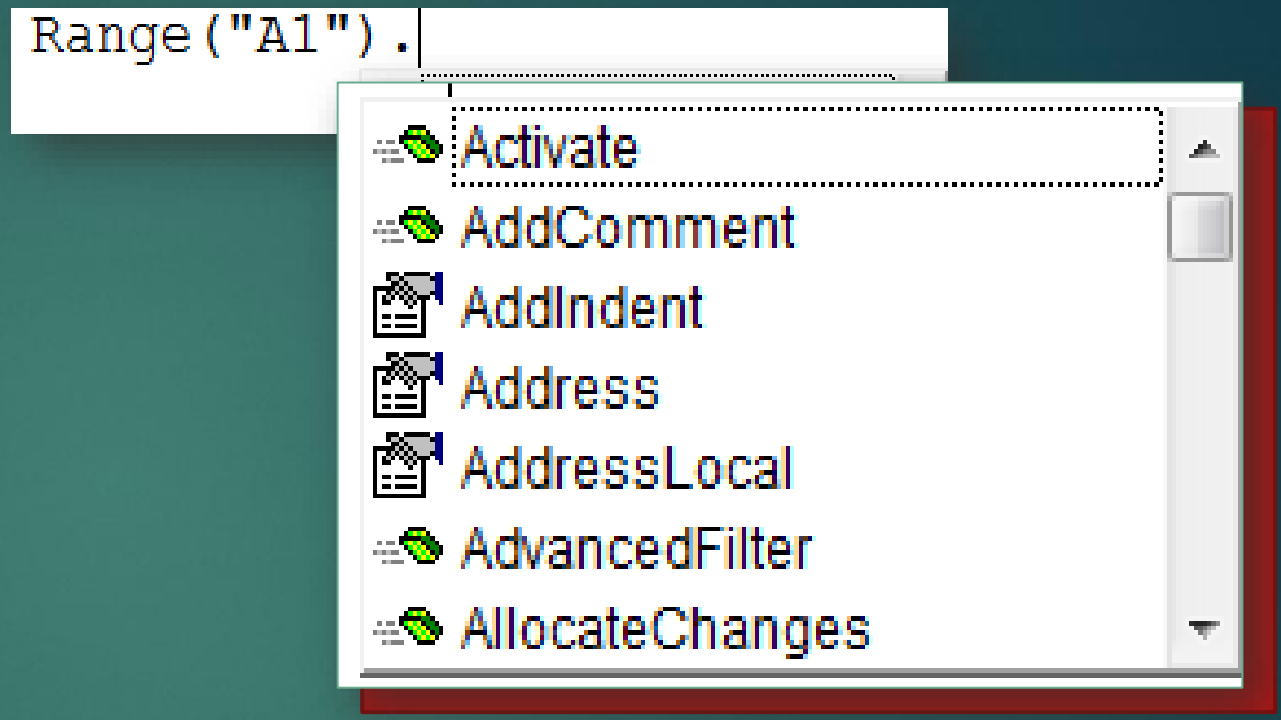
Methods



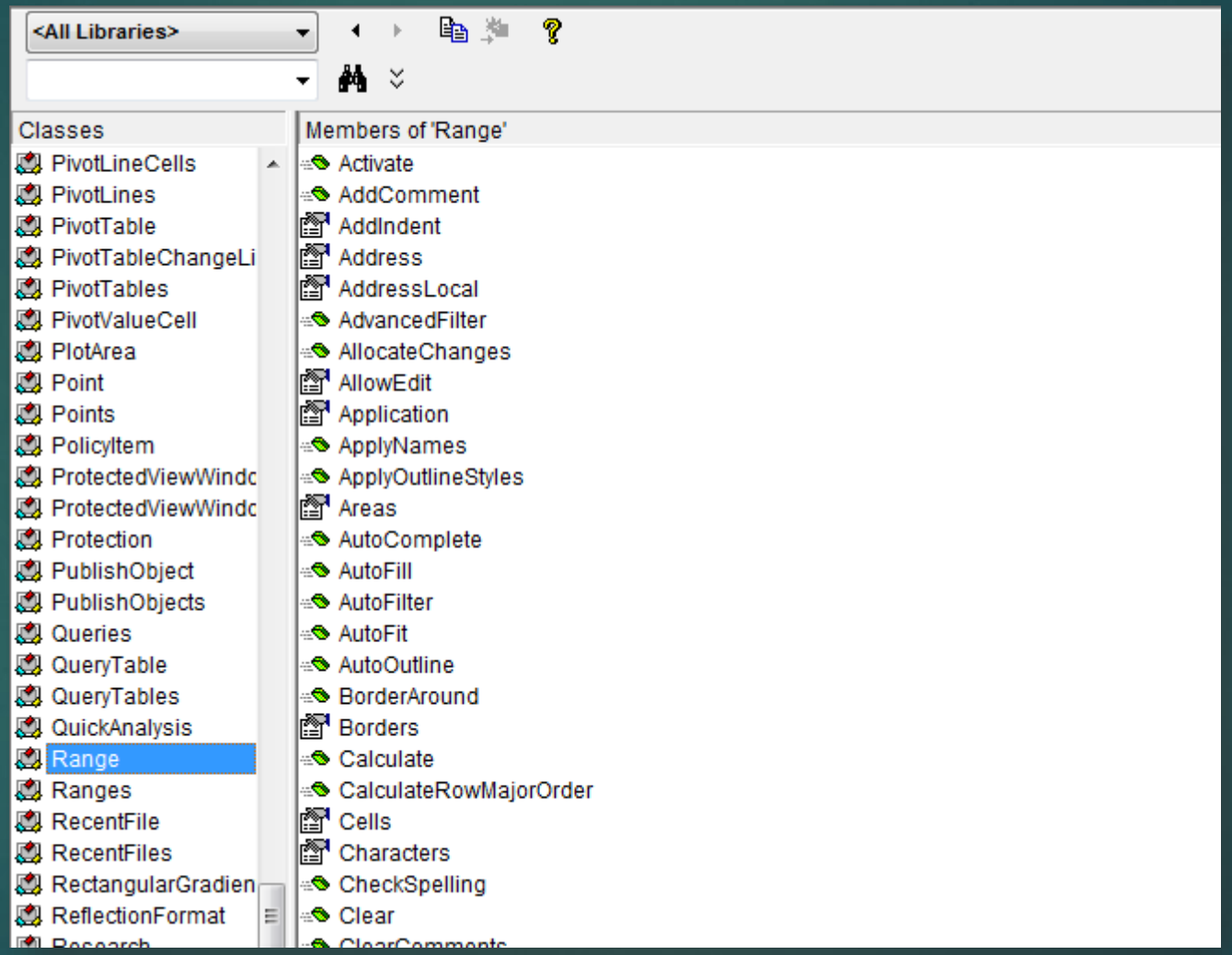
Properties



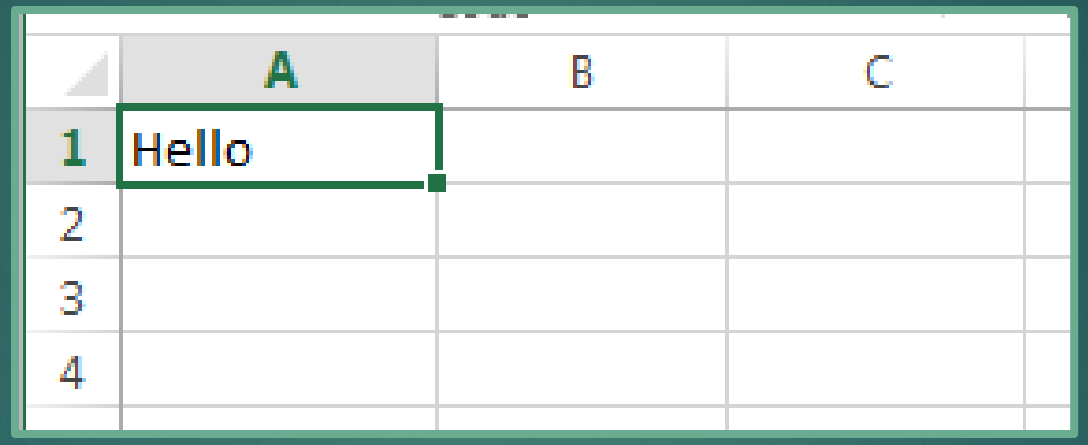
Objects



Object Browser



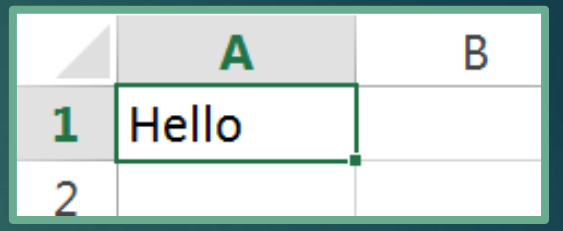
The Range Object



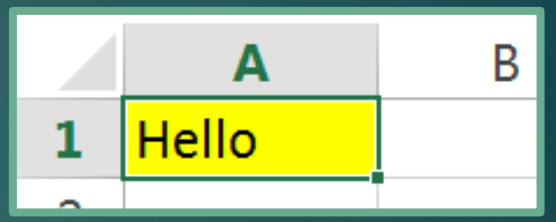
Range (“A1”) . Value = “Hello”

Using the **Range** object

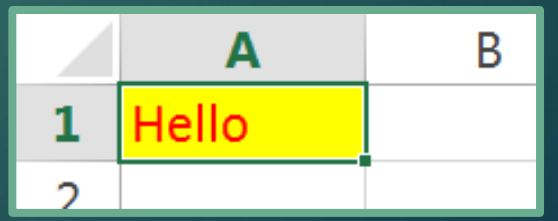
Range (“A1”) . Value = “Hello”



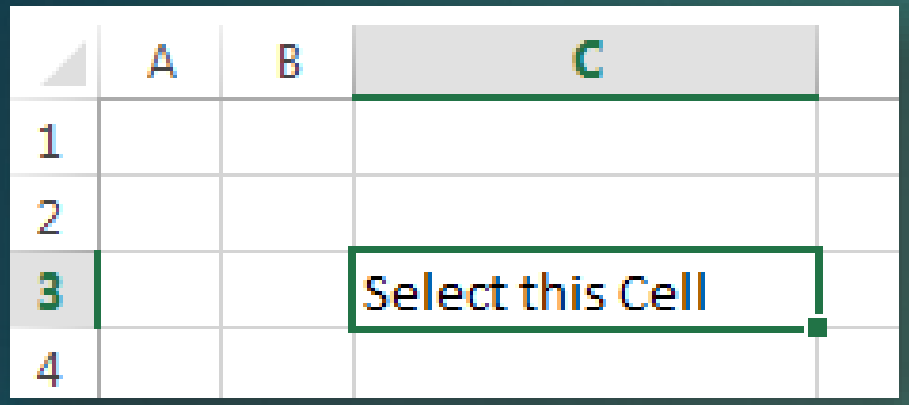
Range (“A1”) . Interior . Color = vbYellow



Range (“A1”) . Font . Color = vbRed



Different ways to express the same Range



Range("C3")

Range("B2").Range("B2")

Cells(3,3)

[C3]

Range("A1").Offset(2,2)

MyRange = "C3"
Range(MyRange)

Color Options in VBA

Theme Colors

xlThemeColorLight1

xlThemeColorDark1

xlThemeColorDark2

xlThemeColorlight2

xlThemeColorAccent1

xlThemeColorAccent2

xlThemeColorAccent3

xlThemeColorAccent4

xlThemeColorAccent5

xlThemeColorAccent6

Range (“A1”) . Font . ThemeColor = xlThemeColorLight1

VB Colors

vbBlack

vbWhite

vbRed

vbLine

vbYellow

vbBlue

vbMagenta

vbCyan

Range (“A1”) . Font . Color = vbRed

Color Options in VBA

RGB Colors

Blue – (0 , 0 , 255)

Yellow – (255 , 255 , 0)

Cyan – (0 , 255 , 255)

Magenta – (255 , 0 , 255)

Range (“A1”) . Font . Color = RGB(1,1,1)

Hexidecimal (Hex)

Blue – 16711680

Yellow – 65535

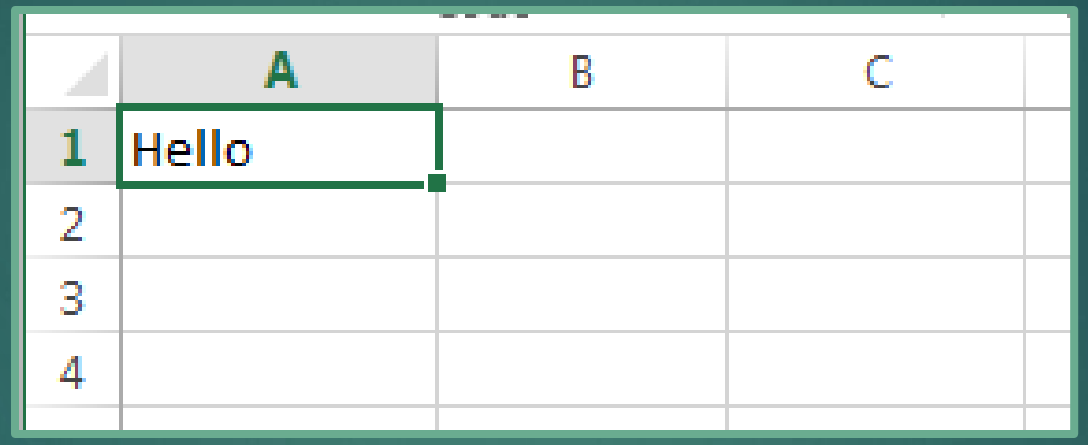
Cyan – 16776960

Magenta – 16711935

(Can’t have the # sign, which people are used to seeing with the hex code)

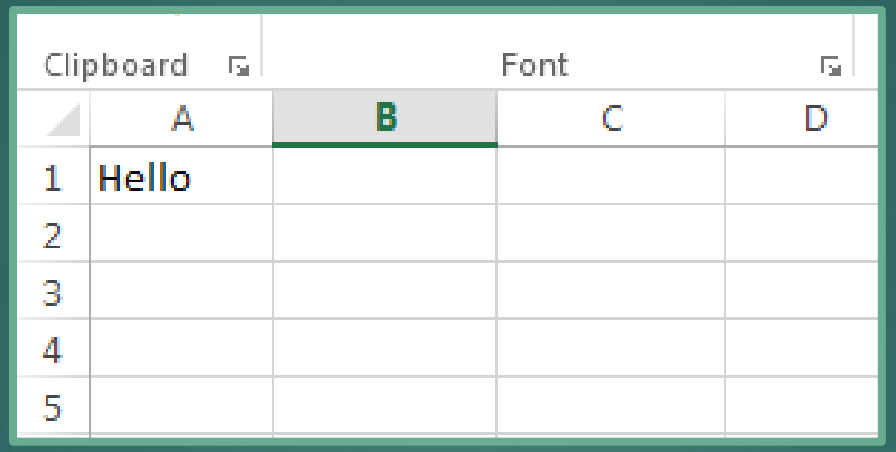
Range (“A1”) . Font . Color = 65535

The Selection Object



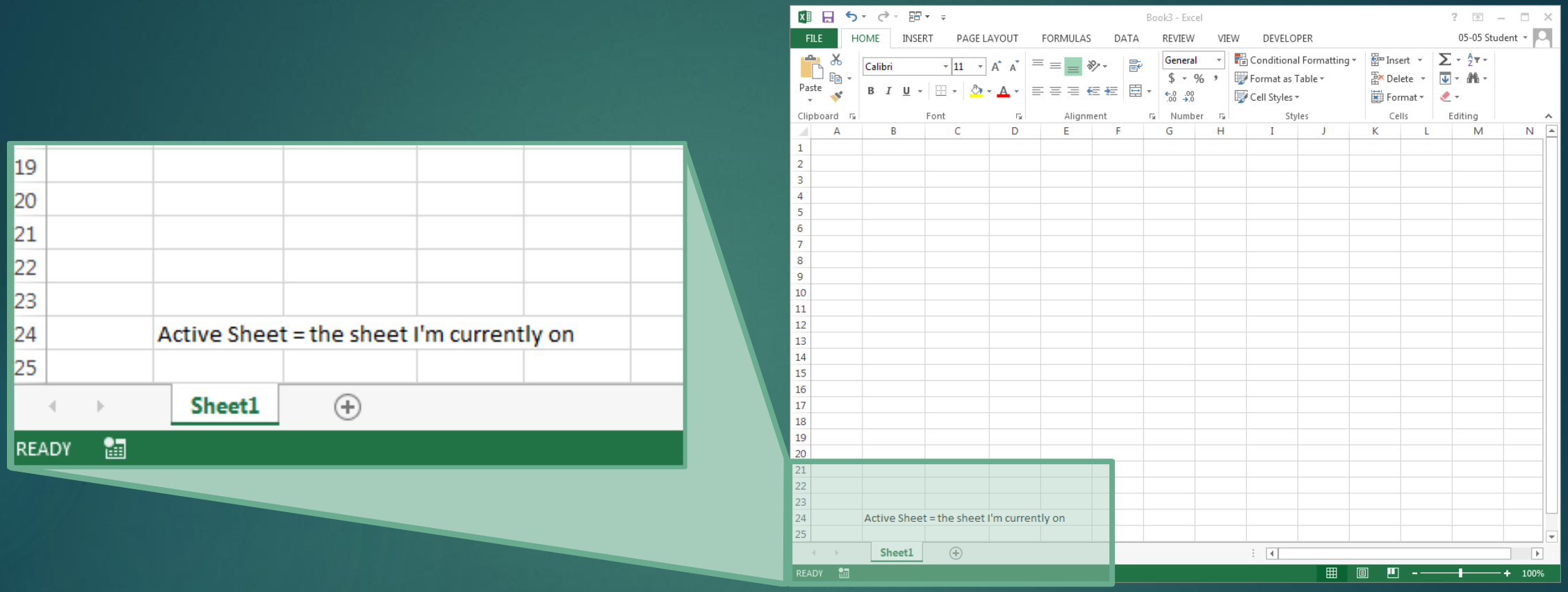
Selection . Value = “Hello”

The Value Property



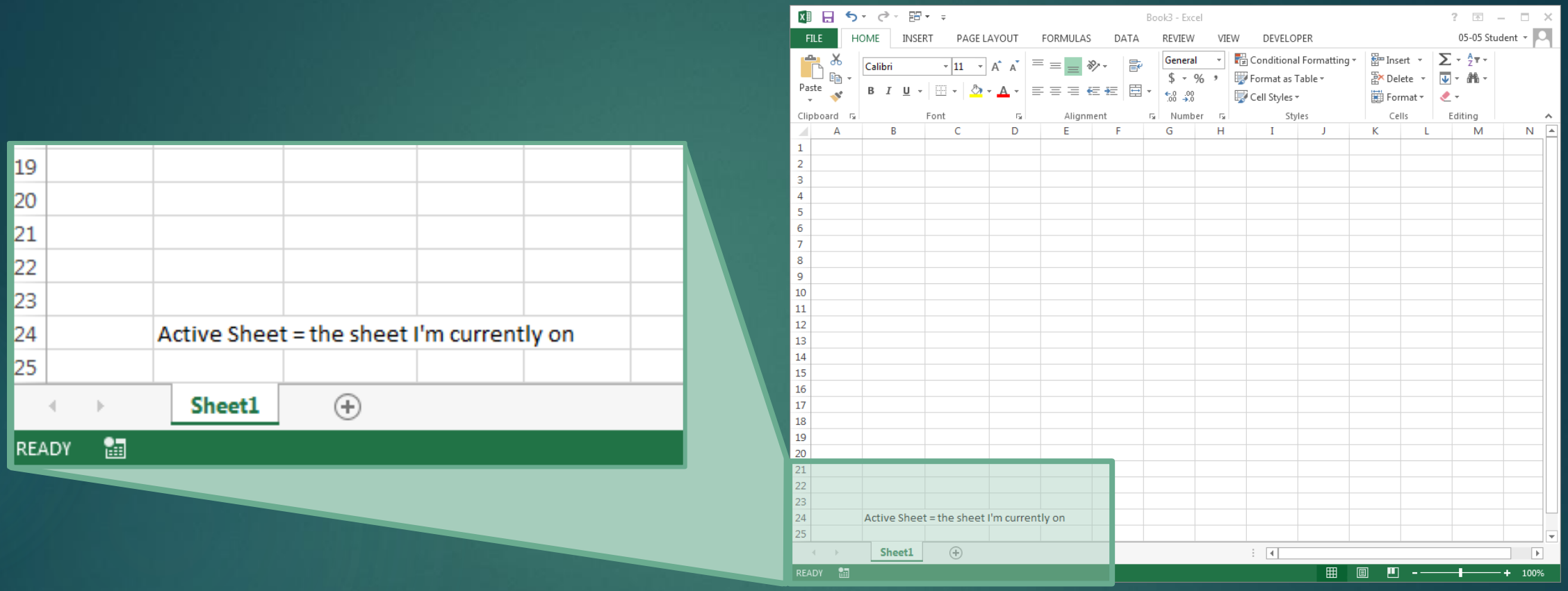
Range("A1") . **Value** = "Hello"

The ActiveSheet Object



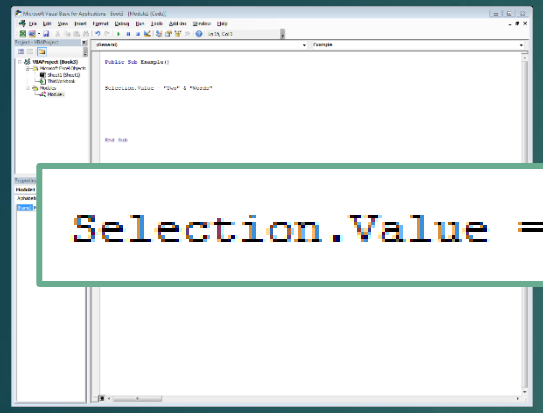
ActiveSheet . Name = “Sheet1”

The Name Property

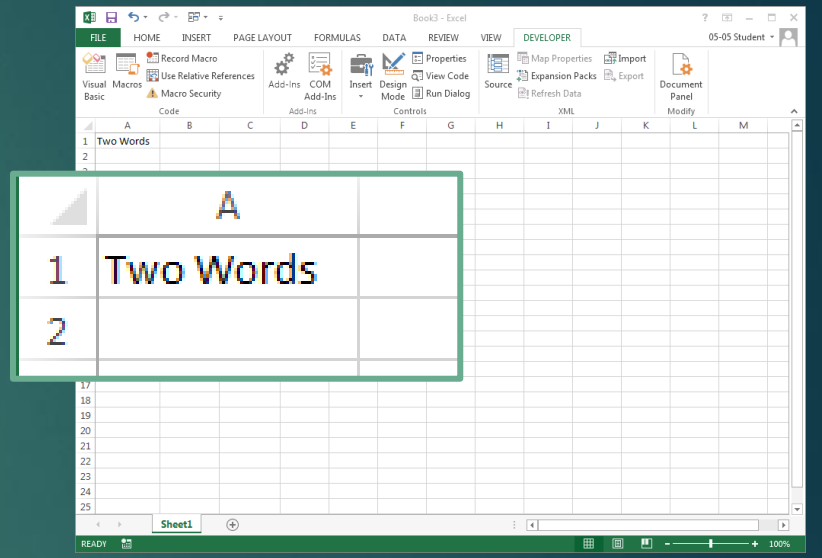


ActiveSheet . **Name** = “Sheet1”

Concatenation

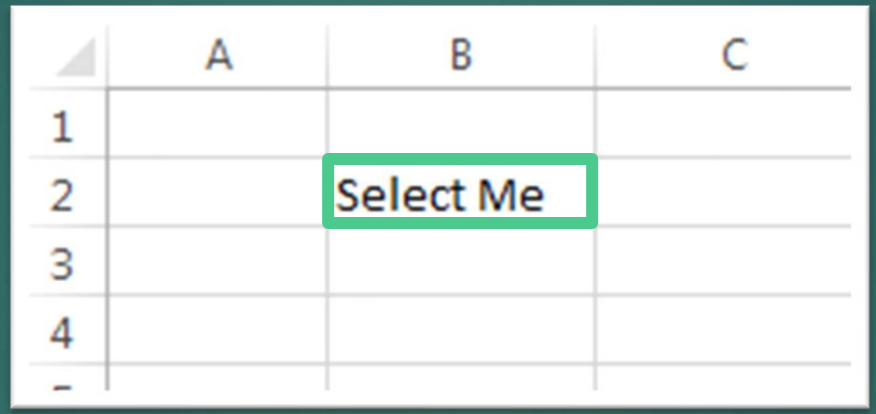


```
Selection.Value = "Two" & "Words"
```



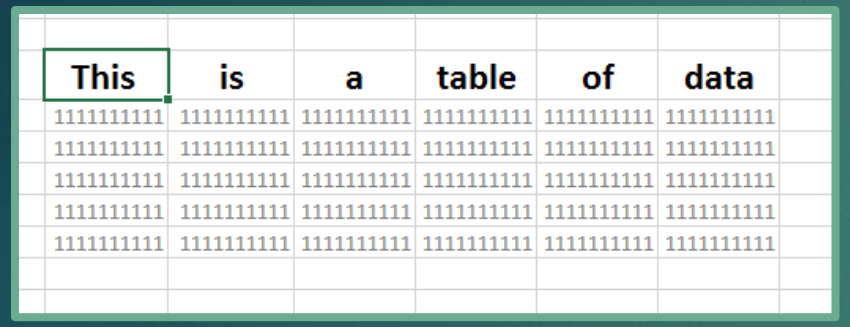
&

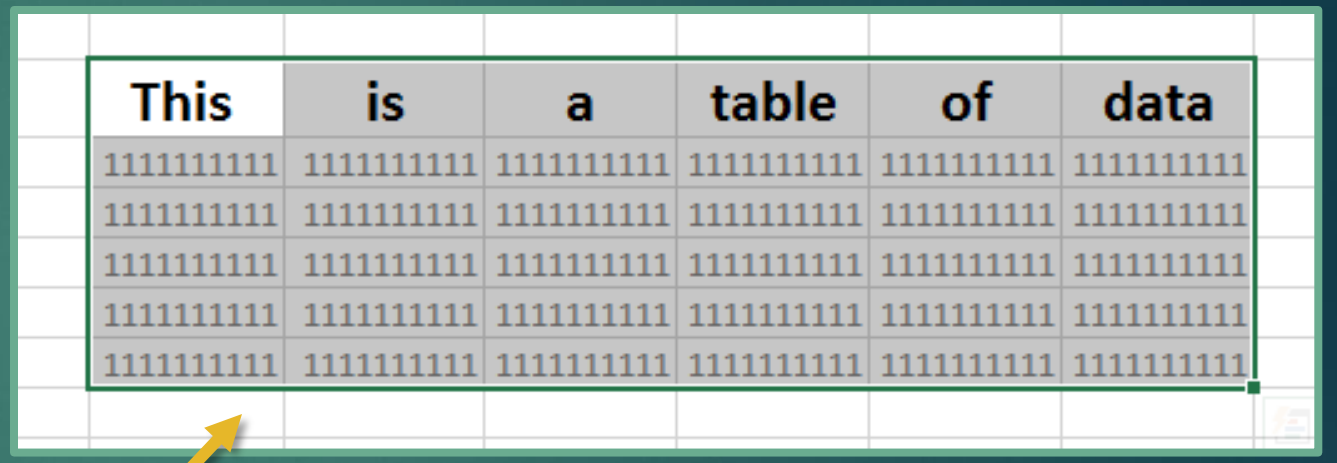
The Select Method



Range("B2") . **Select**

The CurrentRegion Property





Range("A1") . **CurrentRegion** . Select

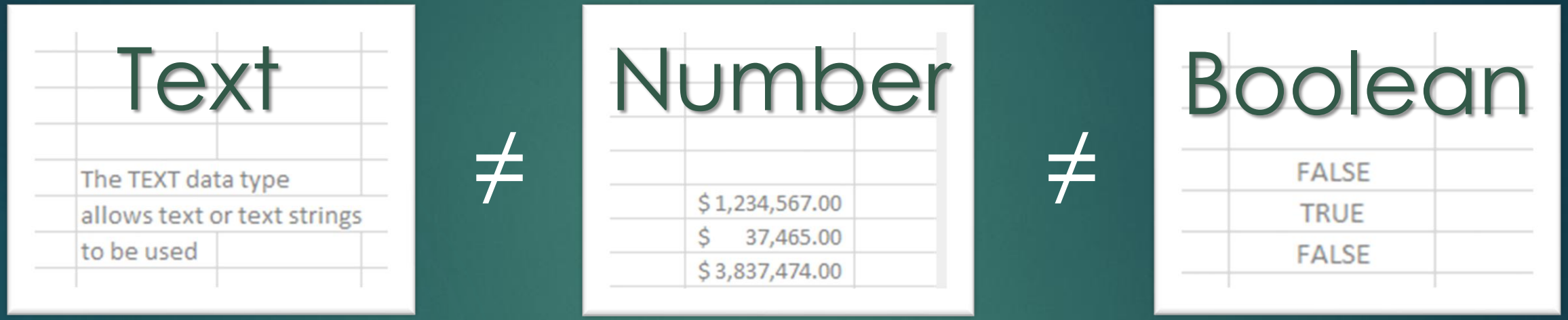
Practice

- Insert Headers
- Format Headers, and Columns
- Format Cell Interior of Lists

Variables

DATA STORAGE

Different Data Types



Common Data Types

Numbers

Decimal
- 28 decimal placeholders
- 14 bytes

Double
- Gazillians
- 8 bytes

Integer
- -2M – 2M
- 2 bytes

Text

Variant
- All
- 16-32 bytes

String
- 2B characters
- Depends

Char
- 0 - 65535
- 2 bytes

Boolean / Byte

Boolean
- True / False
- 2 bytes

Byte
- 0-255
- 1 byte

Misc

Date
- 1900-9999
- 8 bytes

Object
- Any
- 4-32 bytes

Range
- Any
- 4-32 bytes

Variables

Sunshine? Or Clouds?

Sunshine Clouds

Stores a bit of Code

Sunshine

Delivers that code to some kind of process

Stating Variables

```
Dim     NameofVariable     As DataType
```

Variable Declaration:
"Okay VBA, I'm making a variable"

Data Type:
"This is going to be a *number*"
or "*Text*" or "*Boolean*"

Exercise: Create some Variables

- String
  - Name: ExString
  - Data: “This is Dummy Text”
  - Make A1’s value the example text
- Integer
  - Name: ExInteger
  - Data: 5.5
  - MsgBox “Value is” & example value
- Double
  - Name: ExDouble
  - Data: 5.5
  - MsgBox “Value is” & example value
- Boolean
  - Name: ExBoolean
  - ExBoolean = True
  - If ExBoolean = True then MsgBox “Boolean variables are cool!”

(Math) Operators

+ - * /

< > = <>

Loops

TO REPEAT ACTIONS MULTIPLE TIMES

The For Next Loop

Open “For Next Loop” XLSM

To make an action repetitive (a specific number of repetitions)

Scenario:

We want to input the value “100” multiple times into some cells.

Strategy:

We’ll make a variable called “x” and use it to stand in for the numerical value of each iteration of the loop.

The For Next Loop

First: Define your variable

```
Dim x as Integer
```

The For Next Loop

Next: Write the code that we want to repeat

```
Dim x as Integer


Range(x, 1).Value = 100
```

The For Next Loop

Next: Wrap the For-Next Loop around the code to repeat it

```
Dim x as Integer

For

Range(x, 1).Value = 100

Next
```

The For Next Loop

Next: enter the code to establish he number of times this should repeat

```
Dim x as Integer

For x = 1 - 10

Range(x, 1).Value = 100

Next
```

The For Next Loop

Finish: complete the Next statement, tell it to move to the next x

```
Dim x as Integer
For x = 1 - 10
Range(x, 1).Value = 100
Next x
```

The For Next Loop

Challenge! Try out the two other For Loop examples in this workbook.

“For Next Loop” xlsm

Rehearse For Next Loop

- Design a Variable
- For Next Loop (For x = 1 to ……..)
- Inside the Loop
  - Select Worksheet(x)
  - Start with A1
  - Select Current Region
  - Copy Content
  - Select Sheet to paste on
  - Select cell to start with
  - Insert offset function
  - Paste
- End the Loop

For Next Loop Answer

```
Sub DuplData()

Dim x As Integer

For x = 1 To Worksheets.Count - 1
Worksheets(x).Select
Range("A1").Select
Selection.CurrentRegion.Select
Selection.Copy
Worksheets("All Portfolios").Select

Range("A1").Select
ActiveCell.Offset((x - 1) * 10, 0).Select
ActiveSheet.Paste

Next x

End Sub
```

The Do While Loop

Open “Do While Loop” XLSM

To repeat an action, until a specific criteria is met

Scenario:

We want to input the value “100” multiple time, until we reach Row 10.

Strategy:

We’ll make a variable called “x” and use it to stand in for the numerical value of each iteration of the loop.

The Do While Loop

First: Define your variable

```
Dim x as Integer
```

The Do While Loop

Then: Define your starting value

```
Dim x as Integer

x = 1
```

The Do While Loop

Then: Write the code that you want to repeat

```
Dim x as Integer

x = 1


Cells(x, 1).Value = 100
```

The Do While Loop

Then: Surround the repeated code with a Do While Loop

```
Dim x as Integer

x = 1


Do While

Cells(x, 1).Value = 100


Loop
```

The Do While Loop

Then: code the criteria for stopping the loop

```
Dim x as Integer

x = 1


Do While x < 10

Cells(x, 1).Value = 100


Loop
```

The Do While Loop

Then: code the statement to count up after you've executed the repeatable code

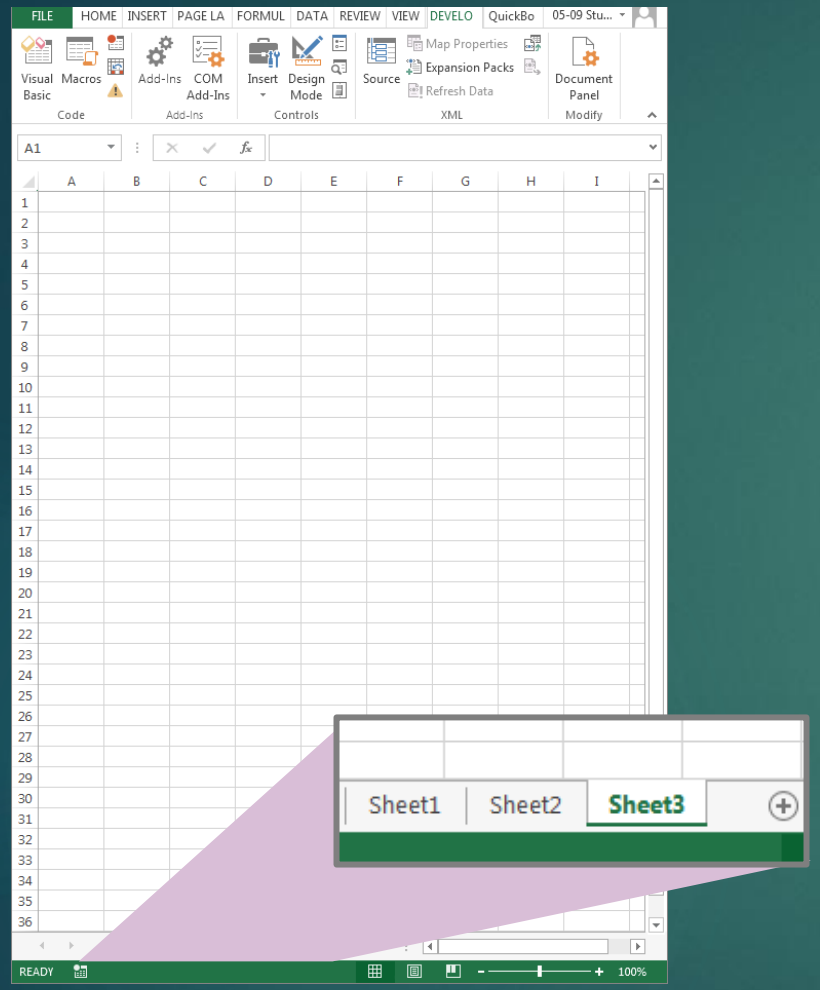
```
Dim x as Integer

x = 1


Do While x < 10

Cells(x, 1).Value = 100

x = x + 1


Loop
```

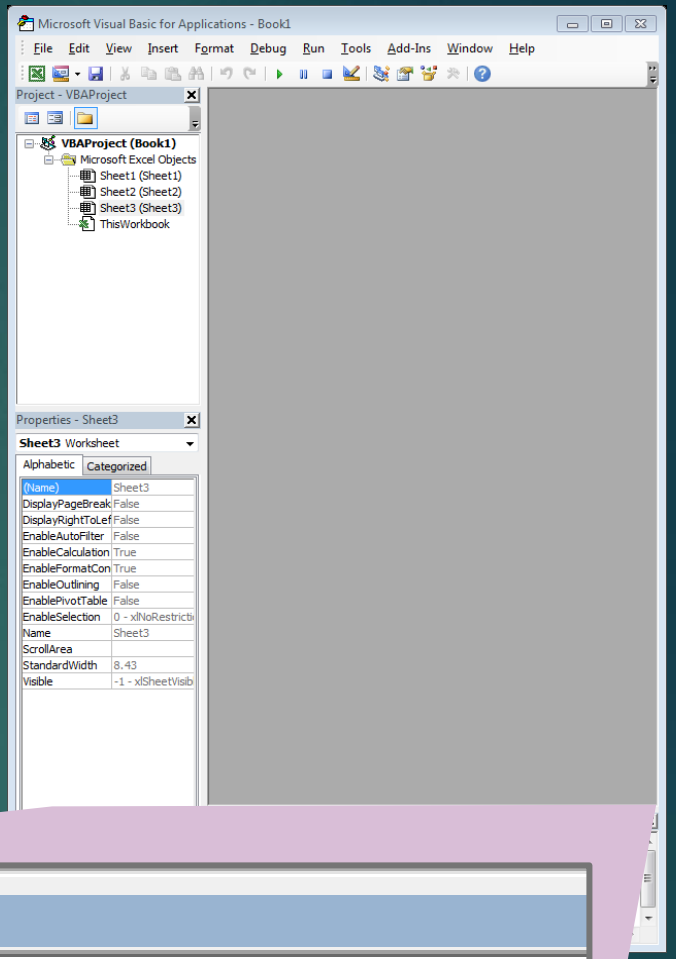
The Do While Loop

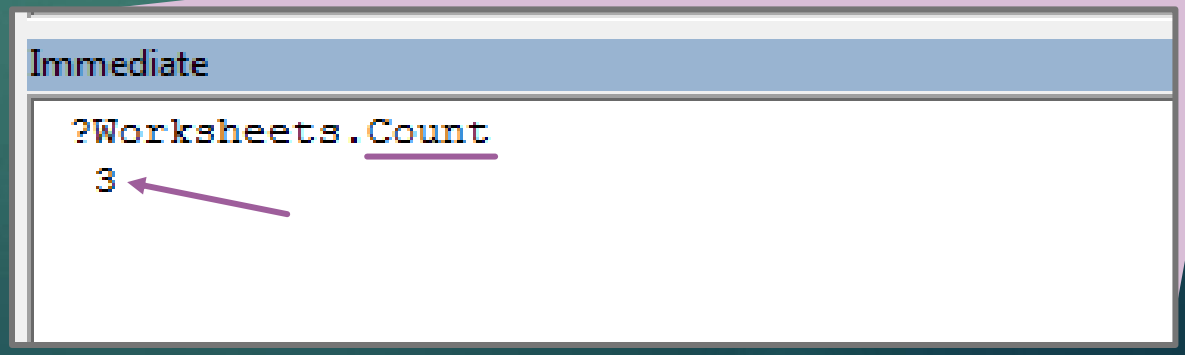
Challenge! Try out the two other For Loop examples in this workbook.

“Do While Loop” xlsm

The Count Property







The Offset Property

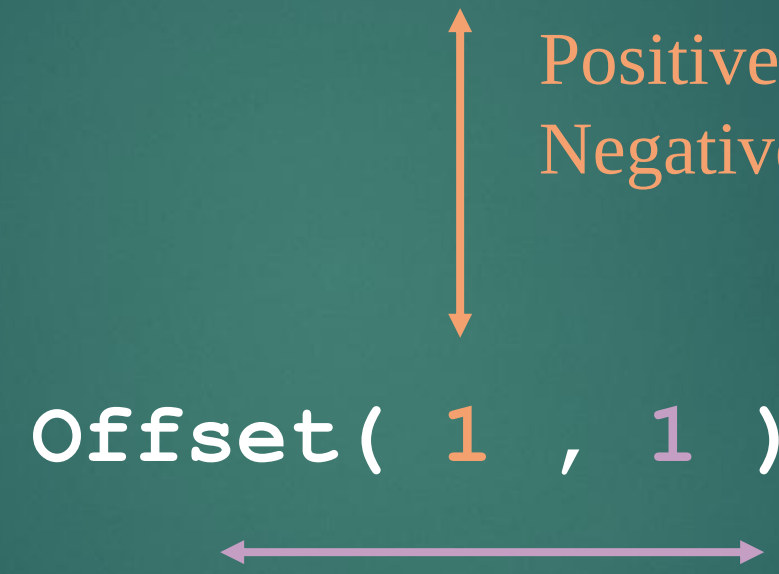
Moves our cell selection using relative cell relationships

Navigates up or down Rows

```
Offset( 1 , 1 )
```

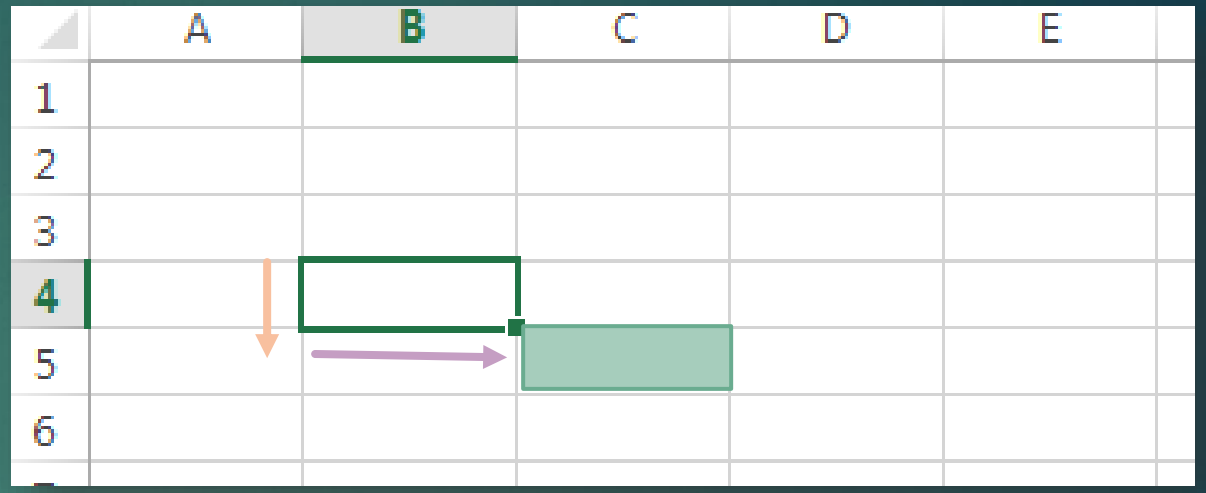
Navigates left or right across columns

The Offset Property

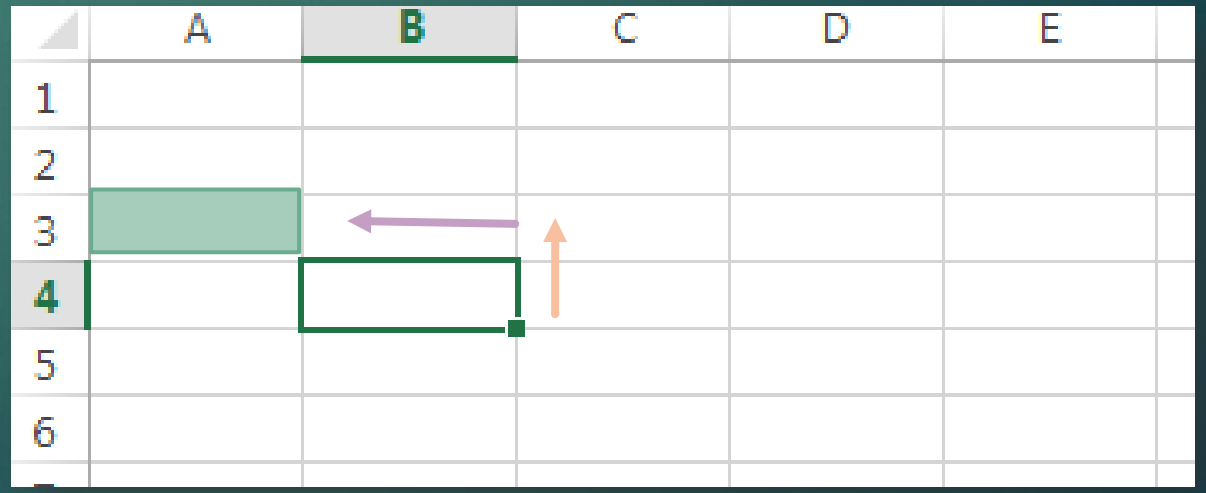


The Offset Property

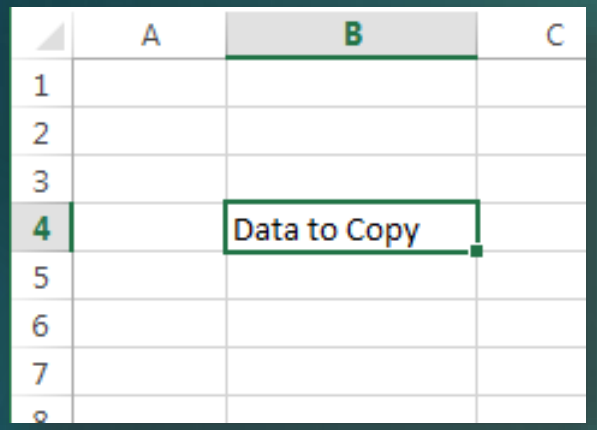
```
Offset(1, 1)
```



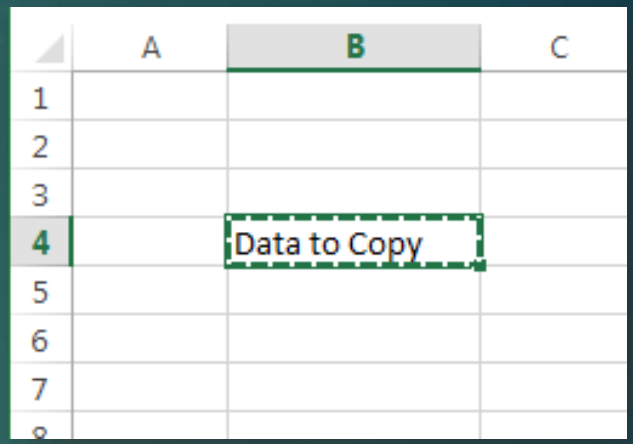

```
Offset(-1, -1)
```



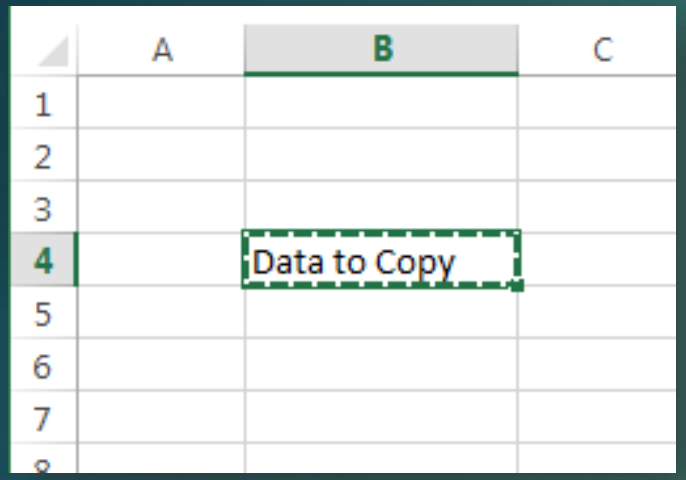
The Copy Method



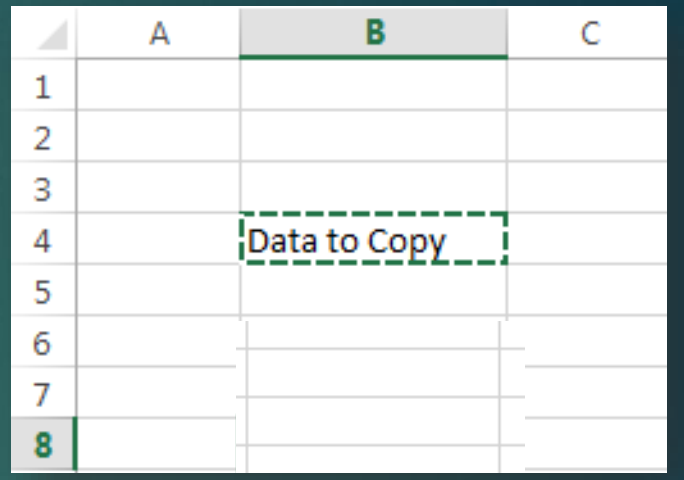

```
Selection.Copy
```



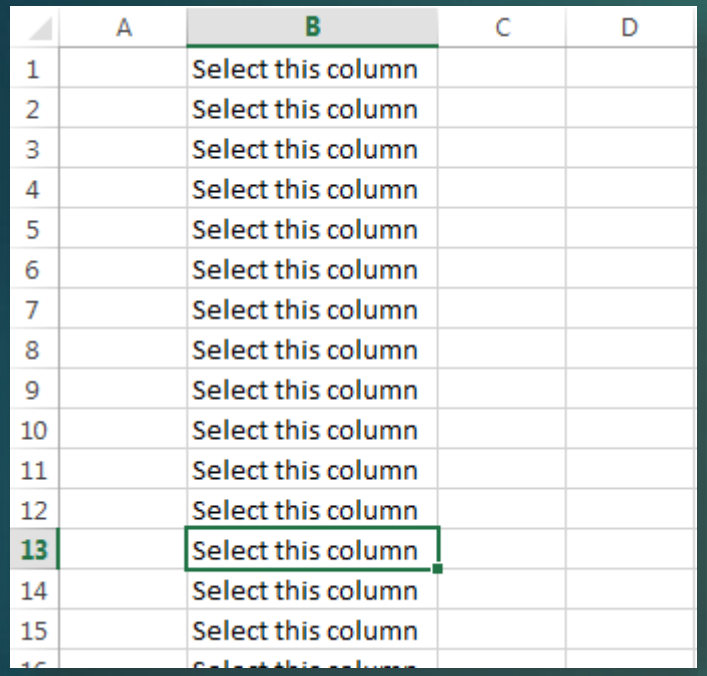
The Paste Method



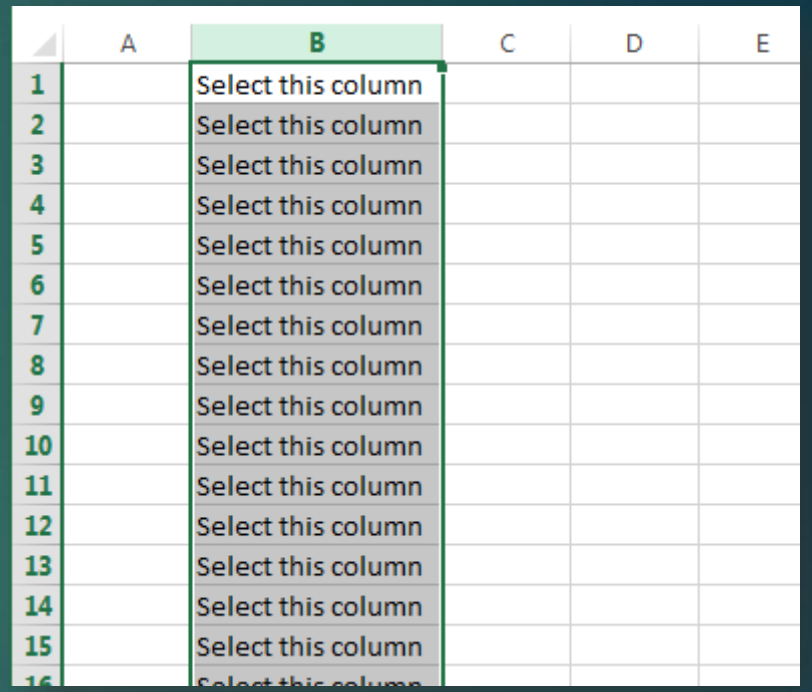

```
ActiveSheet.Paste
```



The Columns Property

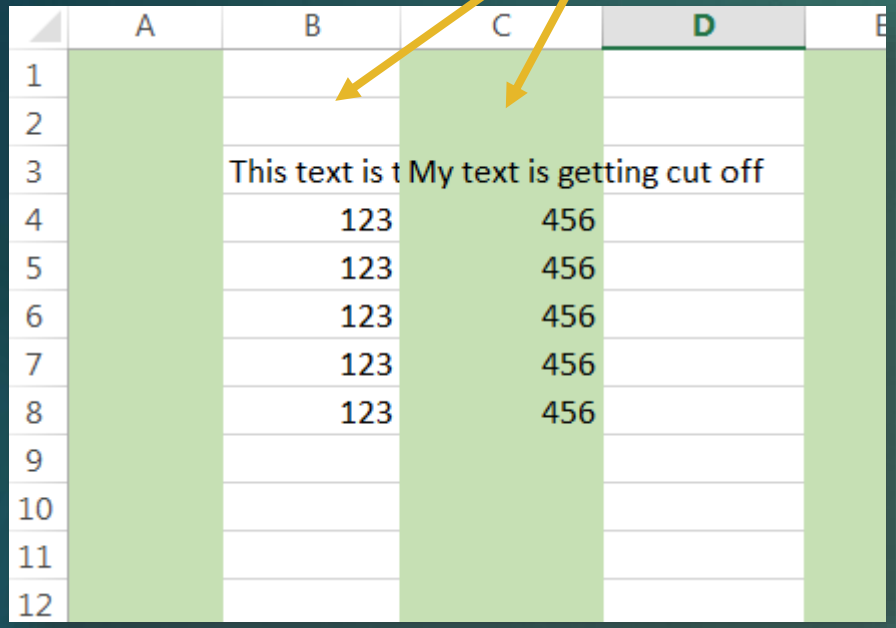



```
Columns("B:B").Select
```



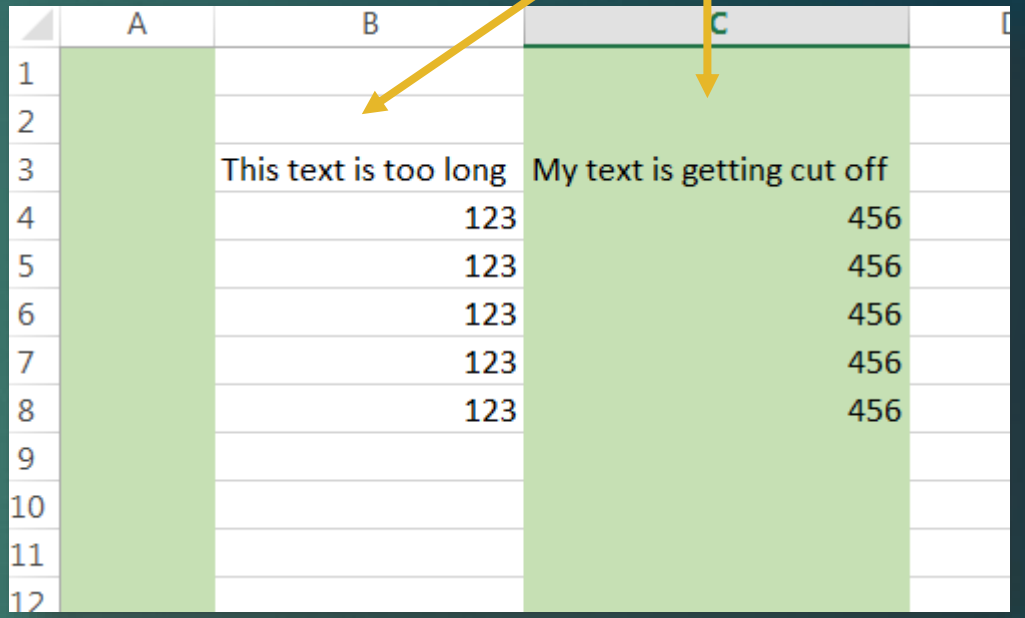
The AutoFit Method

Doesn't fit!



	A	B	C	D
1				
2				
3		This text is t	My text is getting cut off	
4		123	456	
5		123	456	
6		123	456	
7		123	456	
8		123	456	
9				
10				
11				
12				

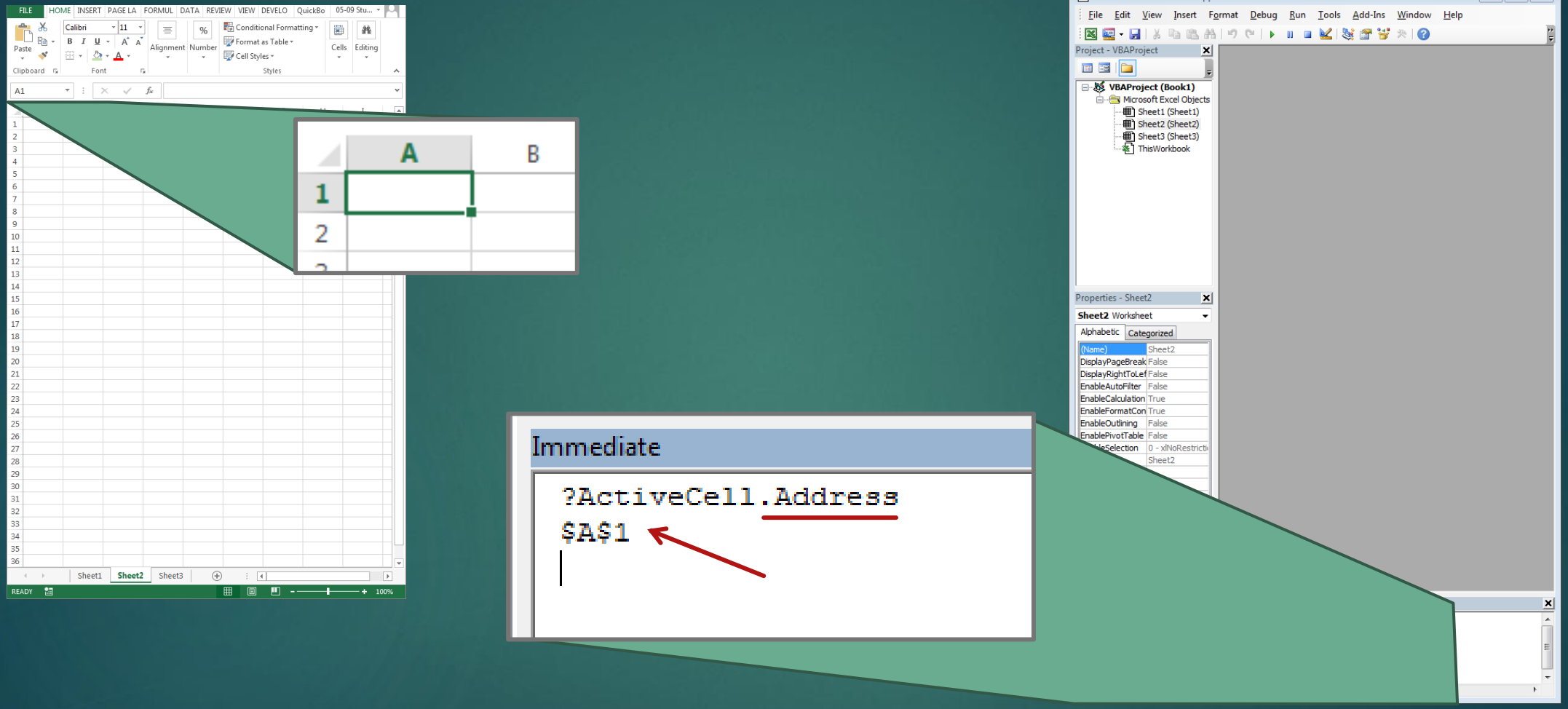
Now it fits!



	A	B	C
1			
2			
3		This text is too long	My text is getting cut off
4		123	456
5		123	456
6		123	456
7		123	456
8		123	456
9			
10			
11			
12			

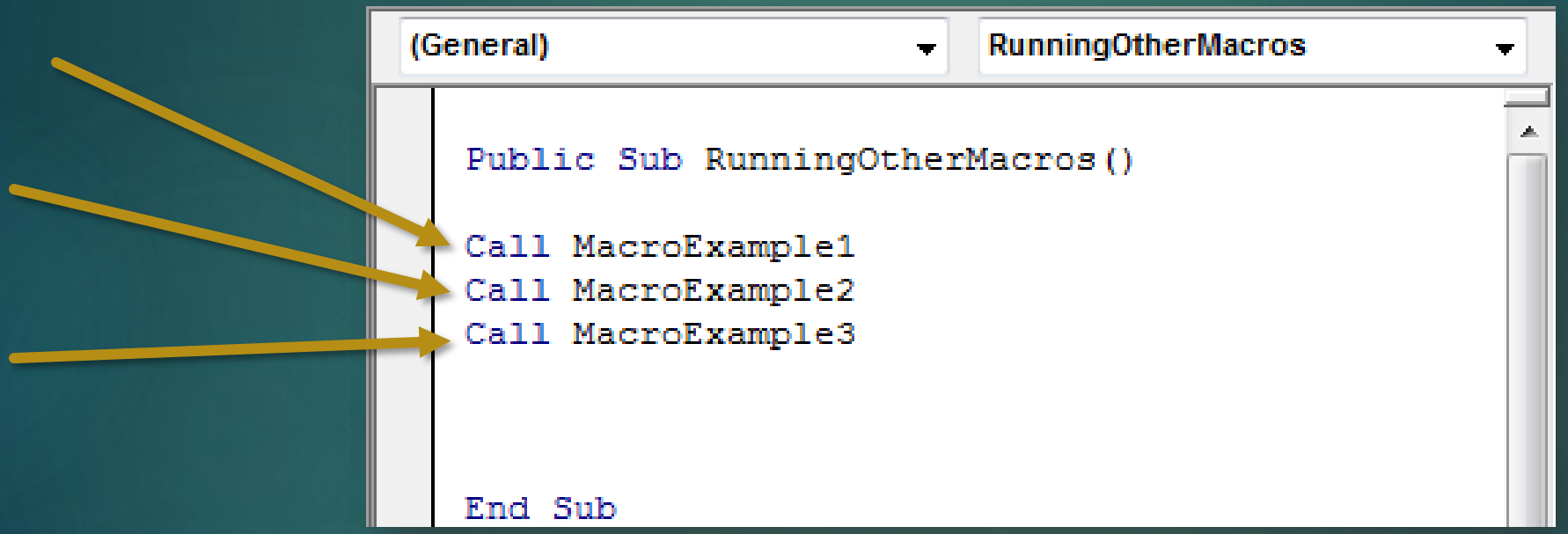
```
Columns("B:C").AutoFit
```

The Address Property

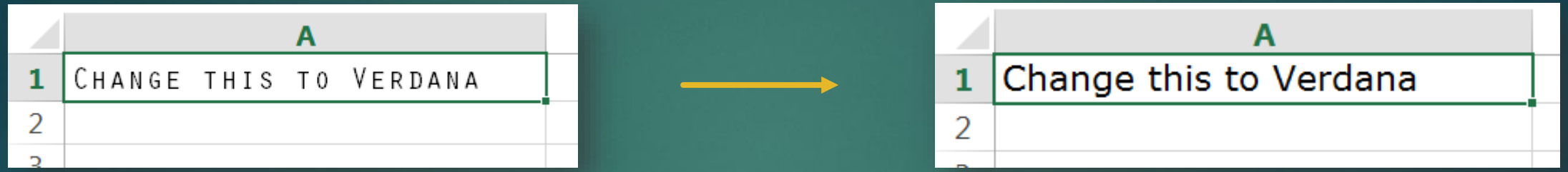


The Call Statement

We can have a Procedure run other macros using the “Call” statement

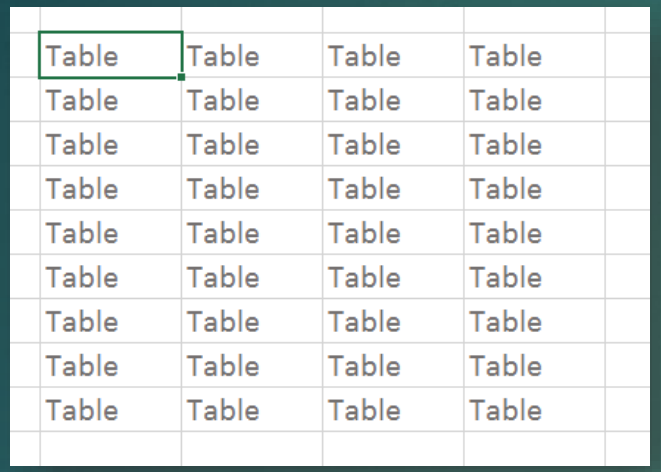


The Font Property

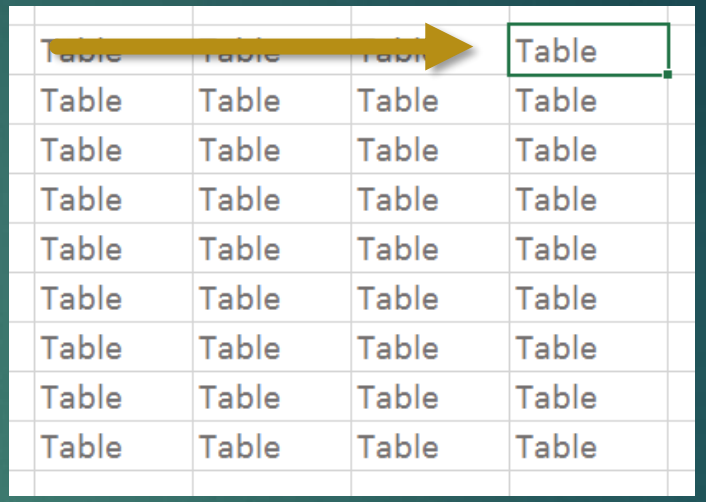



```
ActiveCell.Font.Name = "Verdana"
```

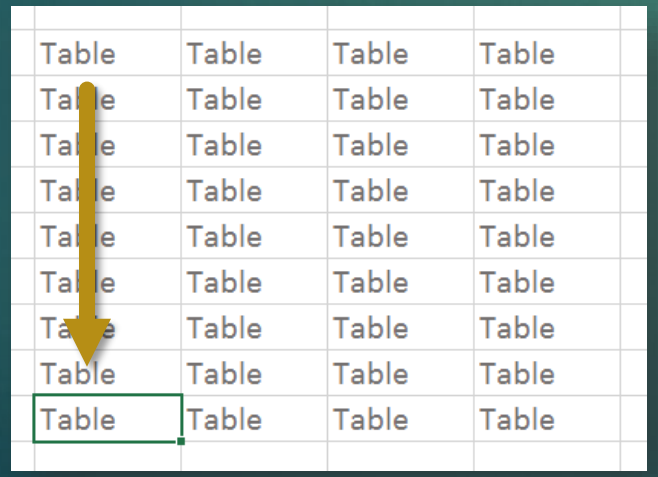
The End Property



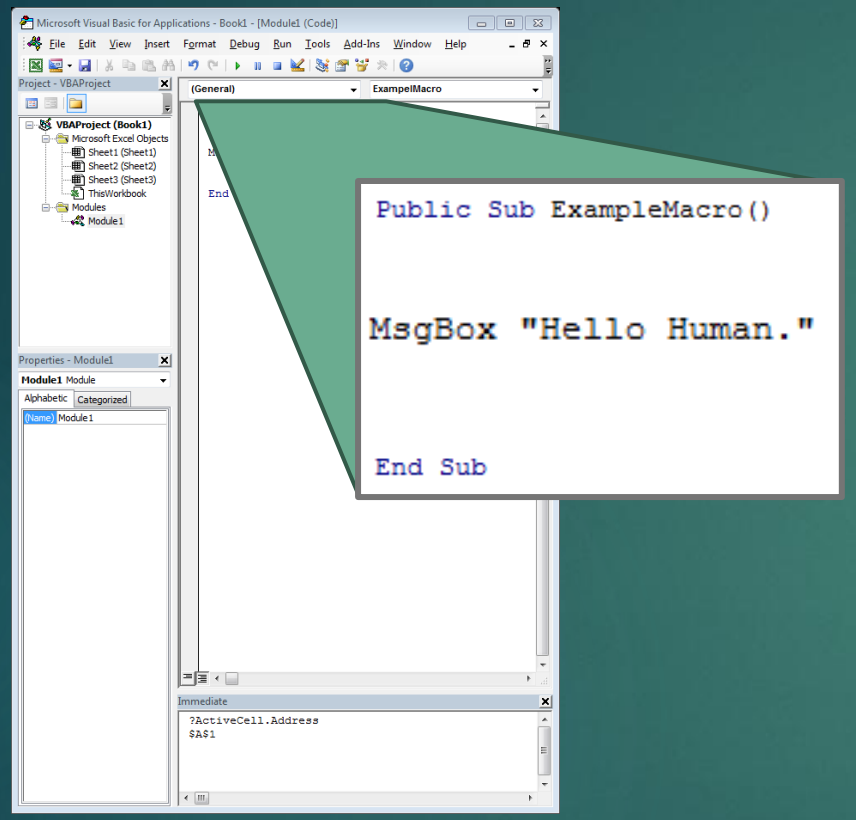

```
ActiveCell.End(xlDown)
```

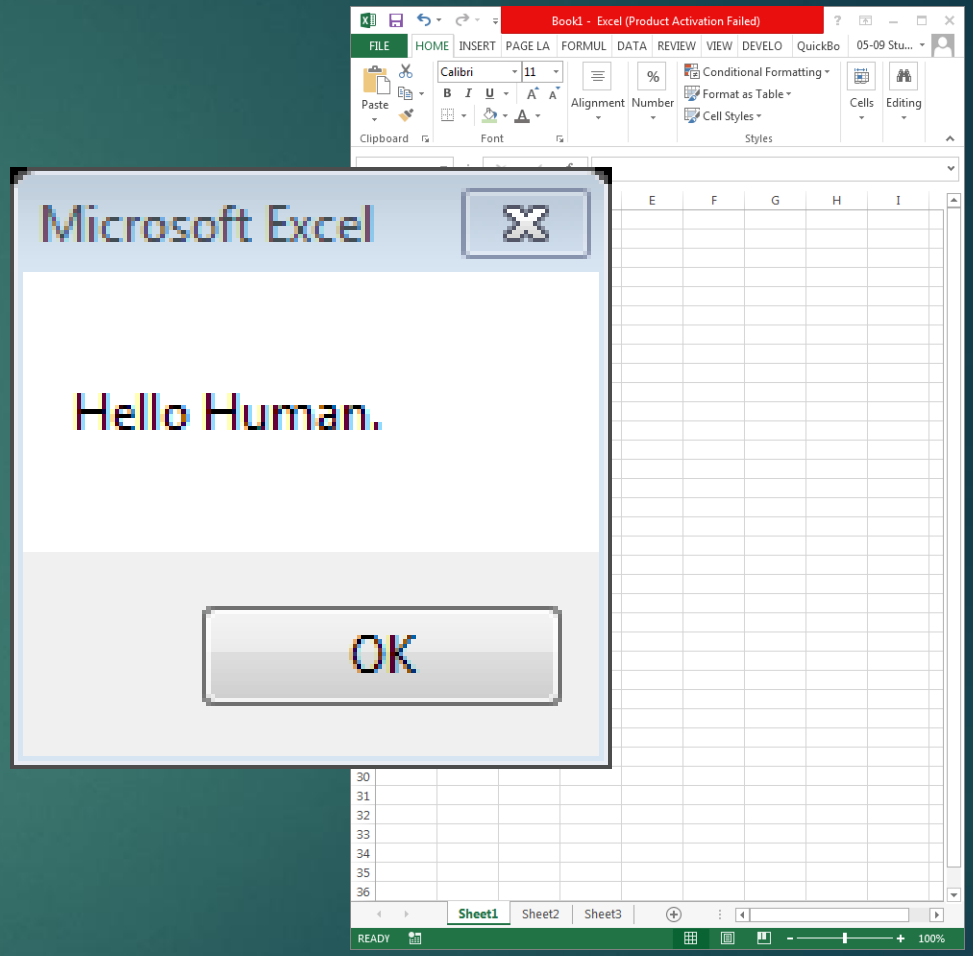



```
ActiveCell.End(xlToRight)
```



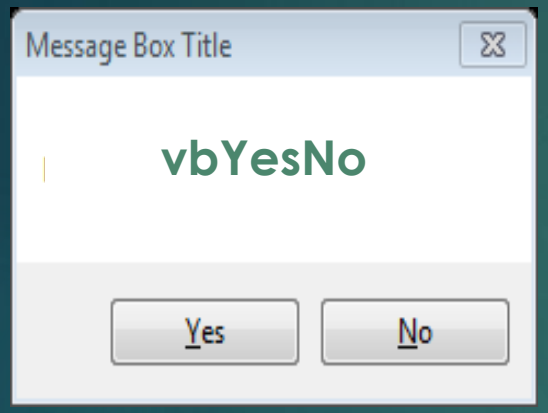
Message Boxes

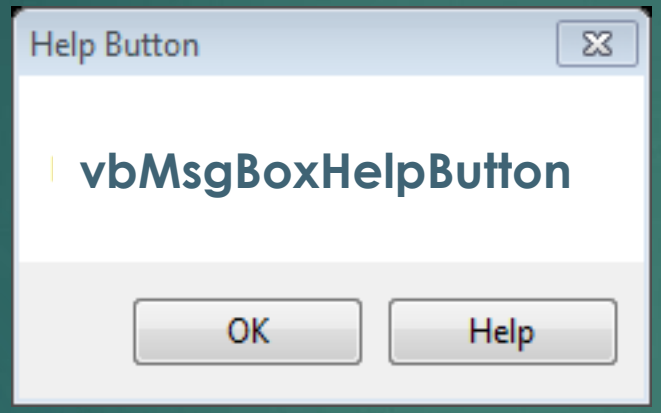


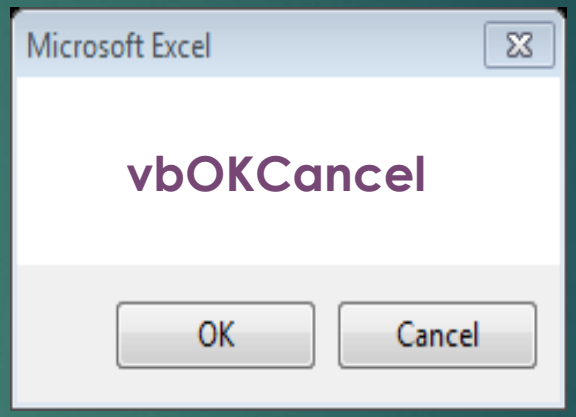


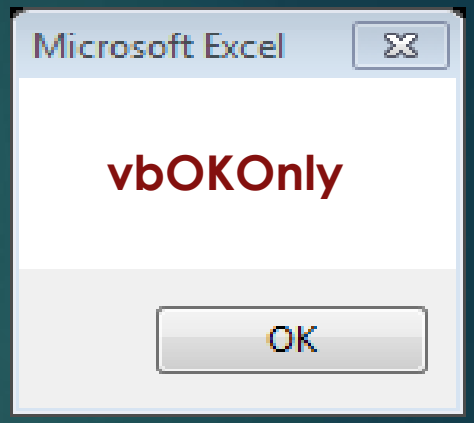
MsgBox options

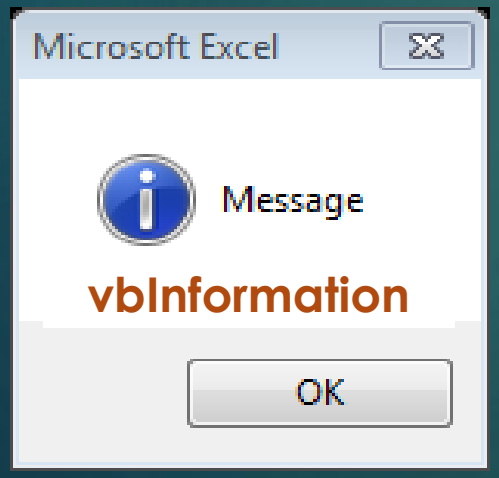
```
MsgBox "Message", vbOption, "Message Box Title"
```

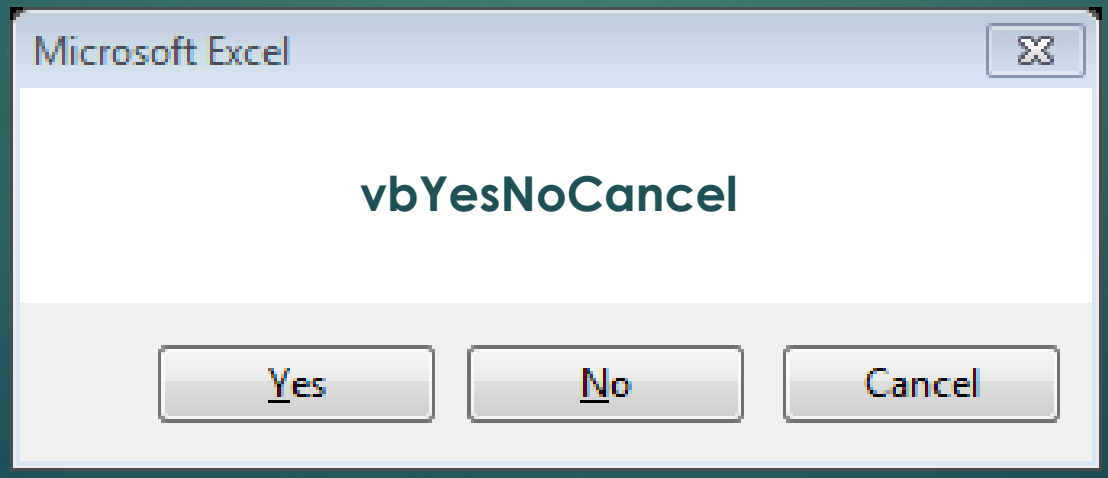


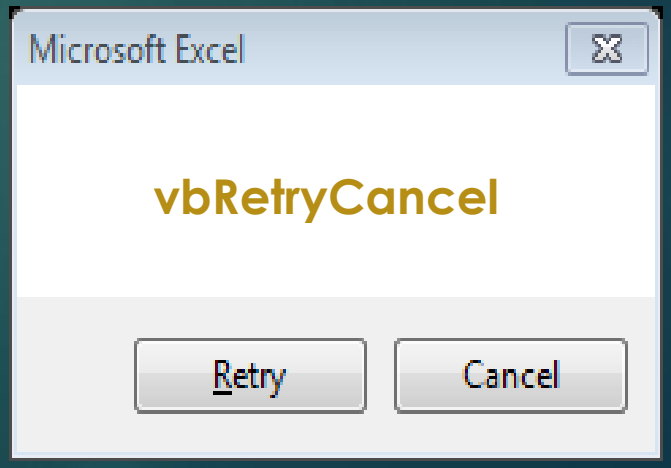




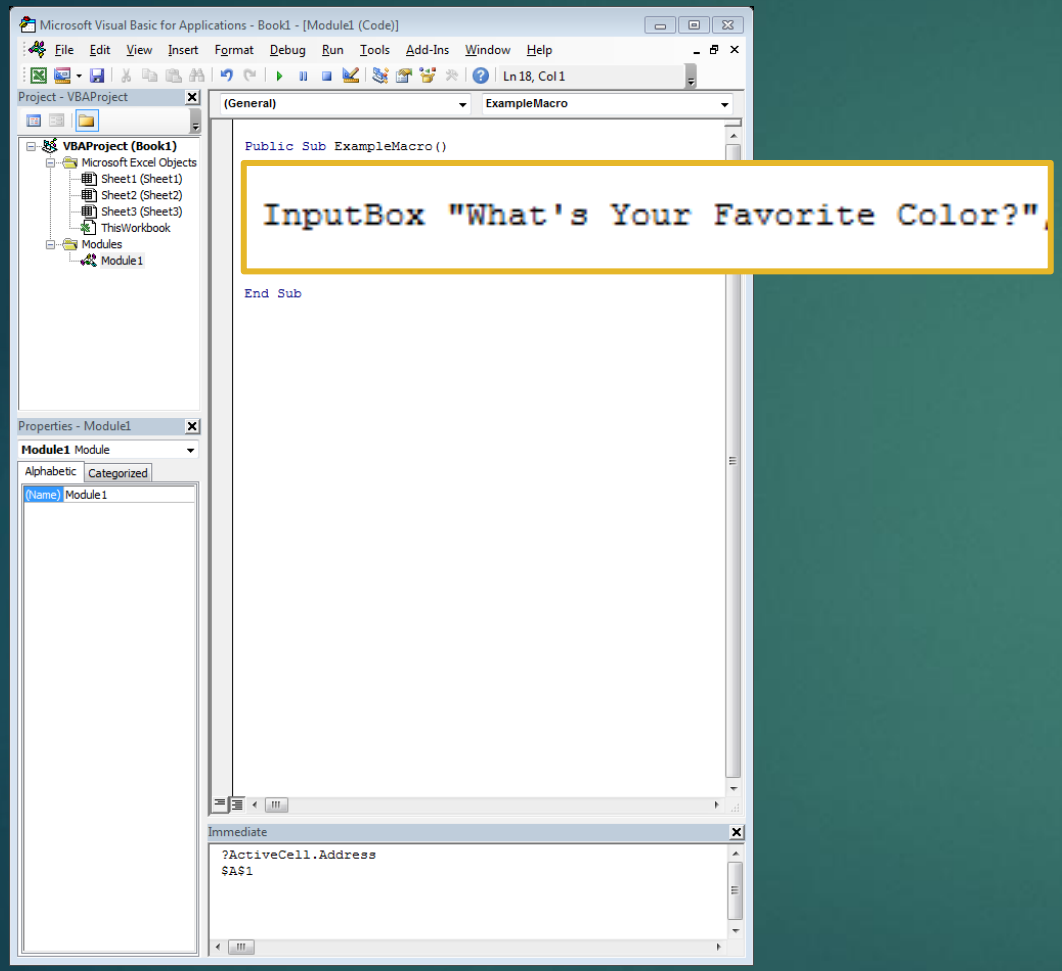


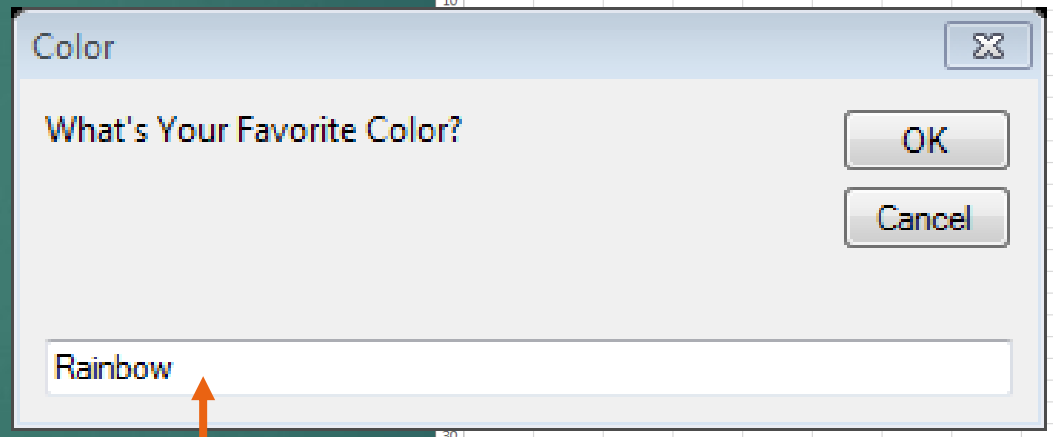






Input Boxes





Type text here.

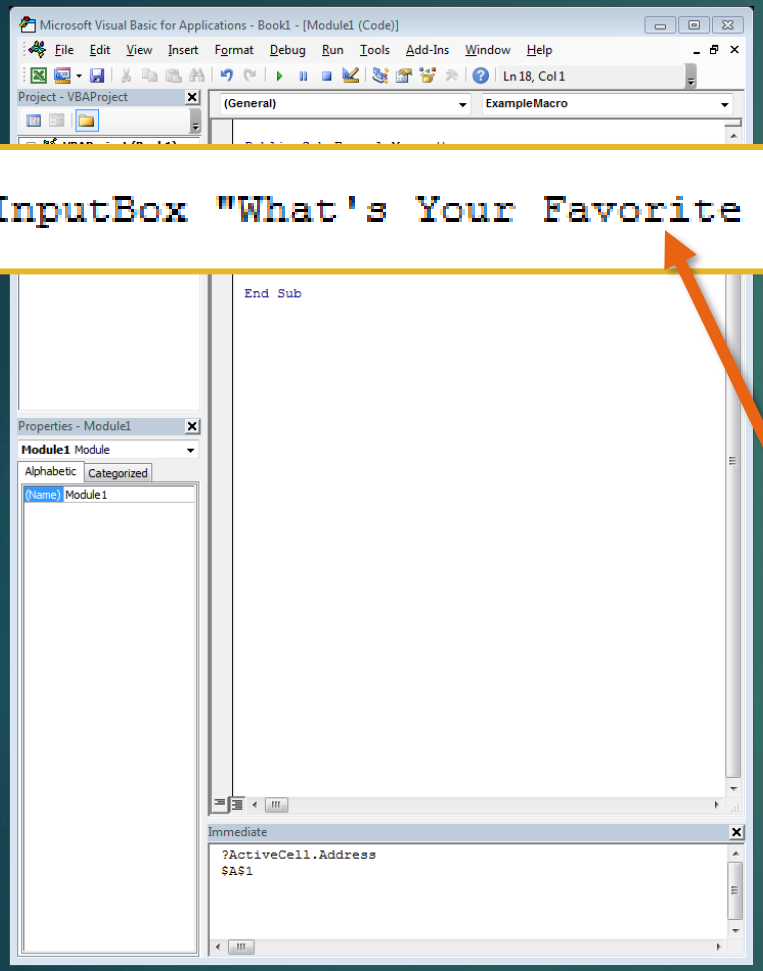
Input Boxes

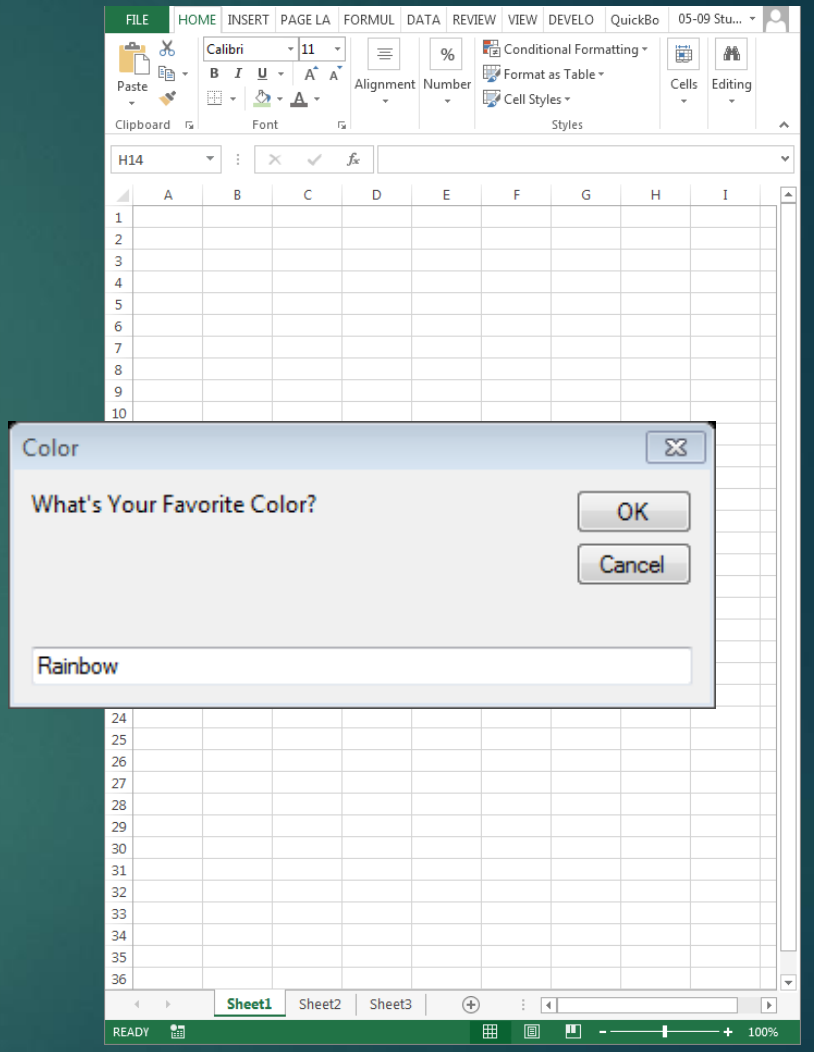
```
InputBox "What's Your Favorite Color?", "Color", "Rainbow"
```

Question

Name of Box

Default Text





Constants

Declaring constants works a lot like declaring variables, EXCEPT the value has to be defined at the same time.

```
Const NameofConstant = "Learnit"
```

Constant Declaration:
"Okay VBA, I'm making a constant"

Constant Value:
"The value of this constant is…"

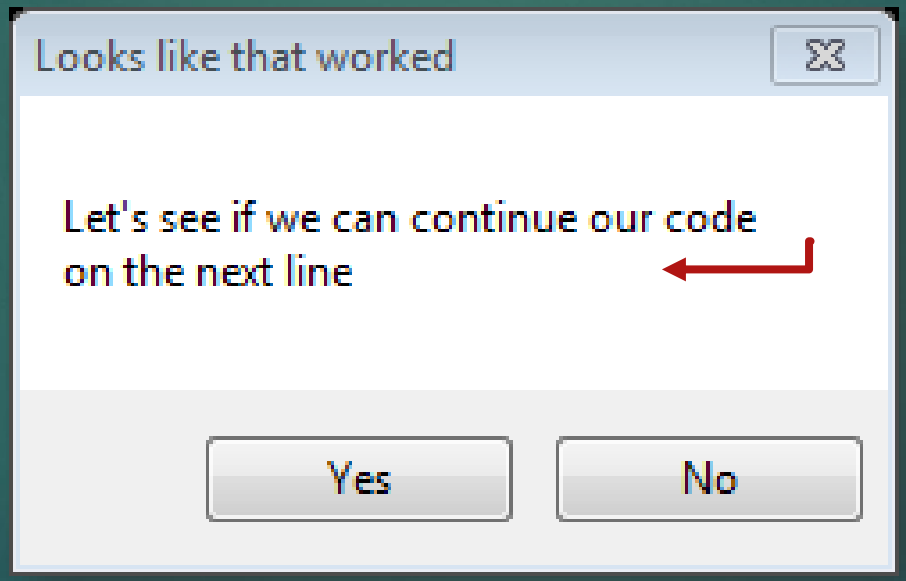
The Code Continuation Character _

```
MsgBox "Let's see if we can continue our code on the next line", _
vbYesNo, "Looks like that worked"
```

To be safe, always precede it with a space

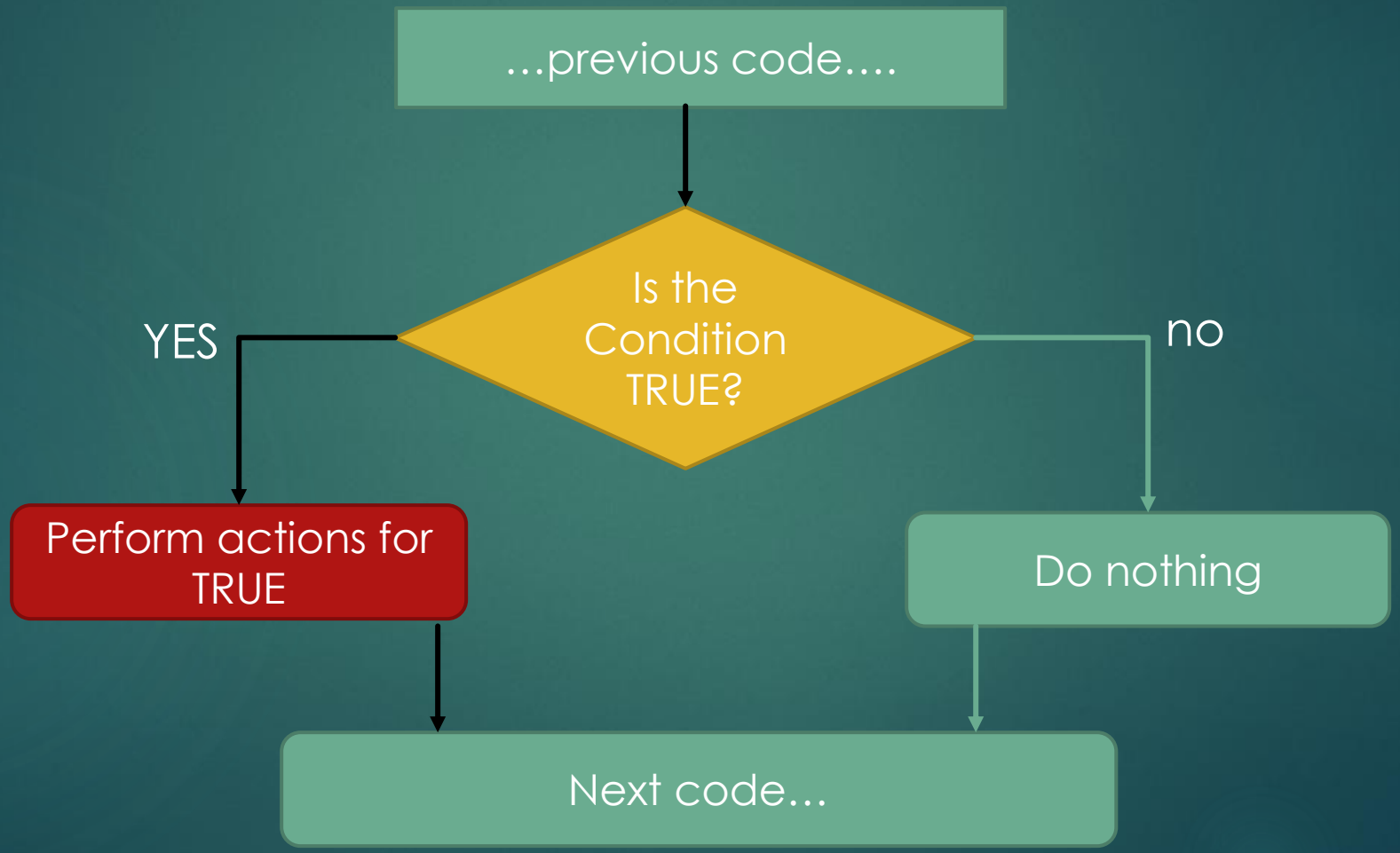
The vbCrLf Constant

```
MsgBox "Let's see if we can continue our code" & vbCrLf & "on the next line", _
vbYesNo, "Looks like that worked"
```



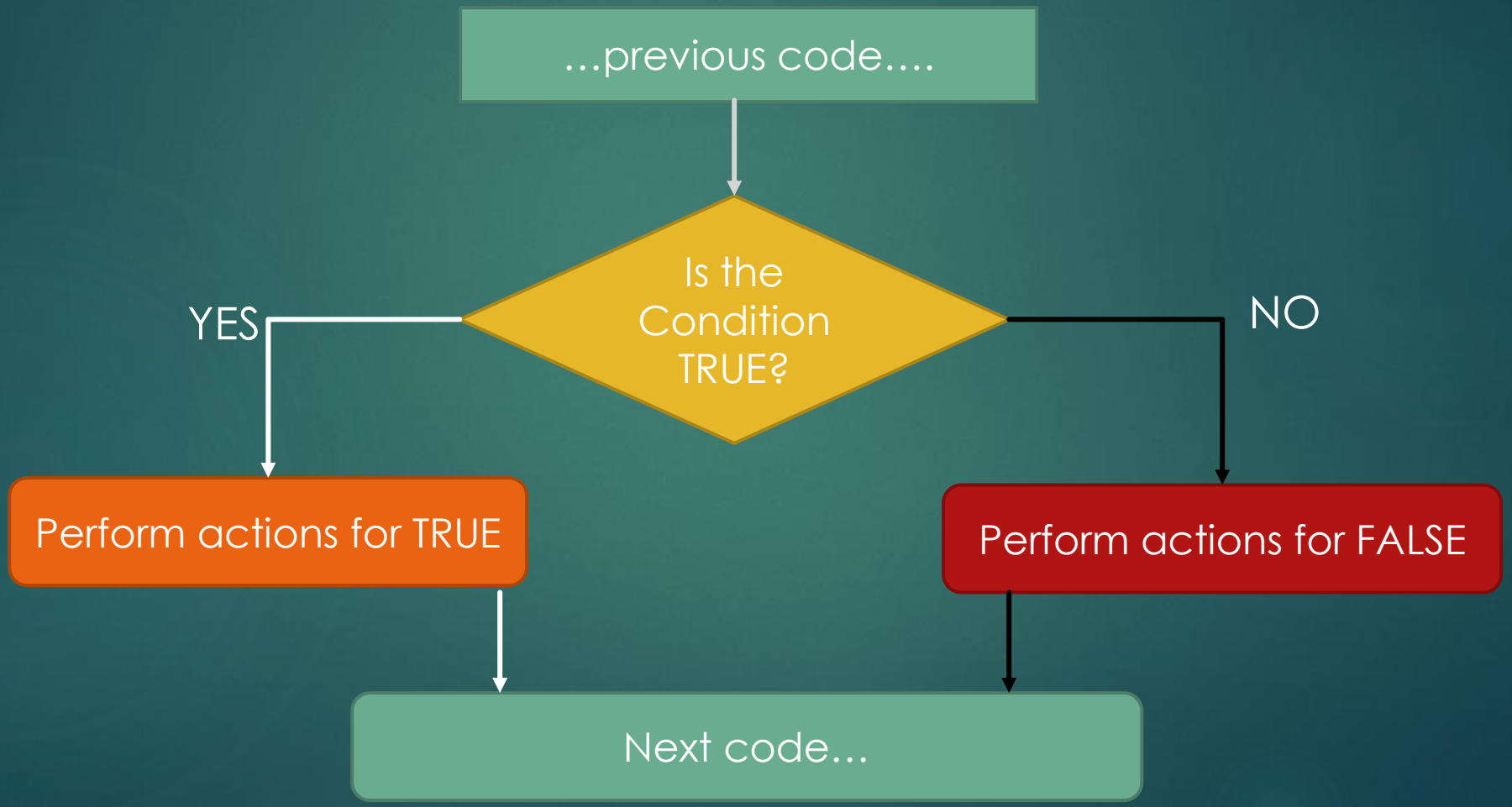
Decision Structures

Code blocks that allow us to: Run a statement if a condition is TRUE



Decision Structures

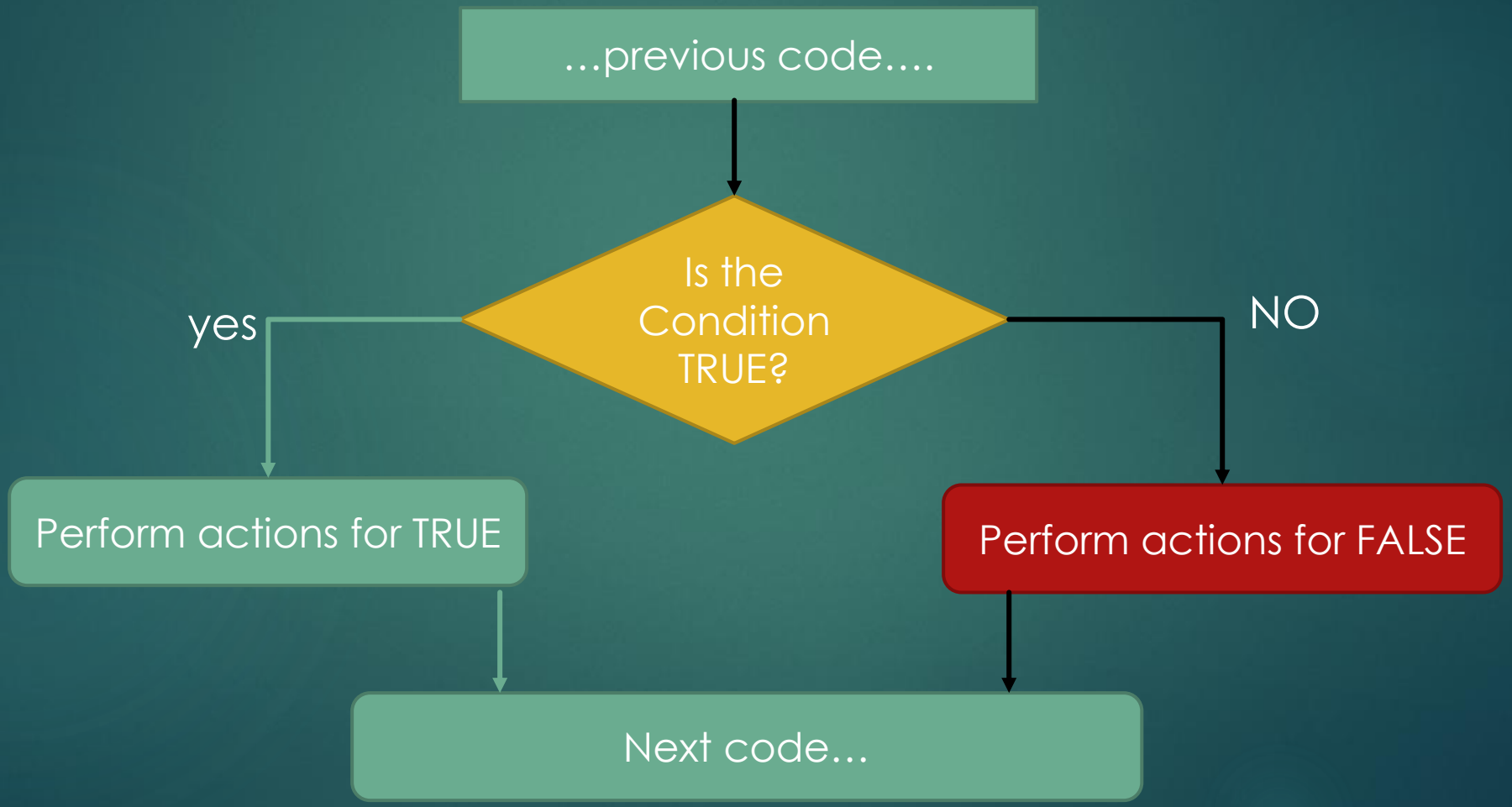
Code blocks that allow us to: Run a statement if a condition is TRUE, and another if FALSE



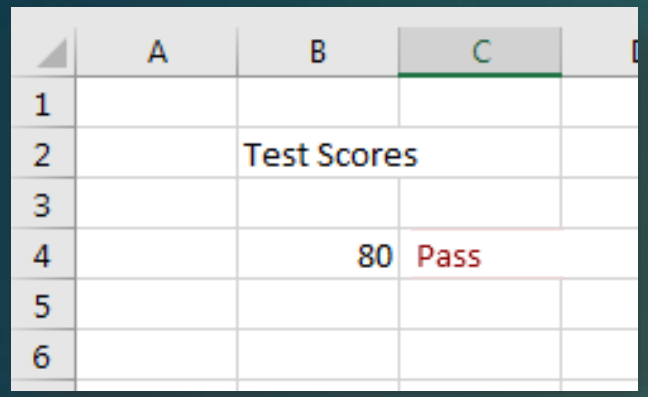
Decision Structures

Code blocks that allow us to:

Run a statement if a condition is FALSE,



The IF THEN Structure



	A	B	C
1			
2		Test Scores	
3			
4		80	Pass
5			
6			

```
If Range("B4") >= 60 Then
    Range("C4").Value = "Pass"

End If
```

If B4 is greater than 60
Enter “Pass” in C4

The IF THEN ELSE Structure

```
If Range("B4") >= 60 Then
    Range("C4").Value = "Pass"
ElseIf Range("B4") <= 60 Then
    Range("C4").Value = "Fail"
End If
```

If B4 is greater than 60
Enter “Pass” in C4

If B4 is LESS than 60
Enter “Fail” in C4

If Then Else Structures can get looong...

```
If Range("B4") >= 90 Then
    Range("C4").Value = "Superb"
ElseIf Range("B4") >= 80 Then
    Range("C4").Value = "Excellent"
ElseIf Range("B4") >= 70 Then
    Range("C4").Value = "Good"
ElseIf Range("B4") >= 60 Then
    Range("C4").Value = "Needs Improvement"
ElseIf Range("B4") >= 50 Then
    Range("C4").Value = "Saturday Training"
ElseIf Range("B4") >= 40 Then
    Range("C4").Value = "Weekend Training"
ElseIf Range("B4") >= 30 Then
    Range("C4").Value = "What can we do to help?"
ElseIf Range("B4") >= 20 Then
    Range("C4").Value = "Well this is just bad"
ElseIf Range("B4") >= 10 Then
    Range("C4").Value = "Circling the Drain"
ElseIf Range("B4") >= 0 Then
    Range("C4").Value = "We need to talk..."

End If
```

Way too much typing.

The Select Case Structure

```
Public Sub IfThenElseTest()

Dim x As Double

x = Range("B4").Value

Select Case x
    Case Is >= 90
        Range("C4").Value = "Superb"
    Case Is >= 80
        Range("C4").Value = "Excellent"
    Case Is >= 70
        Range("C4").Value = "Good"
    Case Else
        Range("C4").Value = "Room for Improvement"

End Select


End Sub
```

```
**Use & define a variable



Let's look as cases for X
In this case
        Do this
In this case
        Do this
In this case
        Do this
In every other case
        Do this
Done
```

Select Case – Case Options

To check if a case is a **Text value**

Case "Word"

To check if a case is a **Number(s)**

Case 1, 2

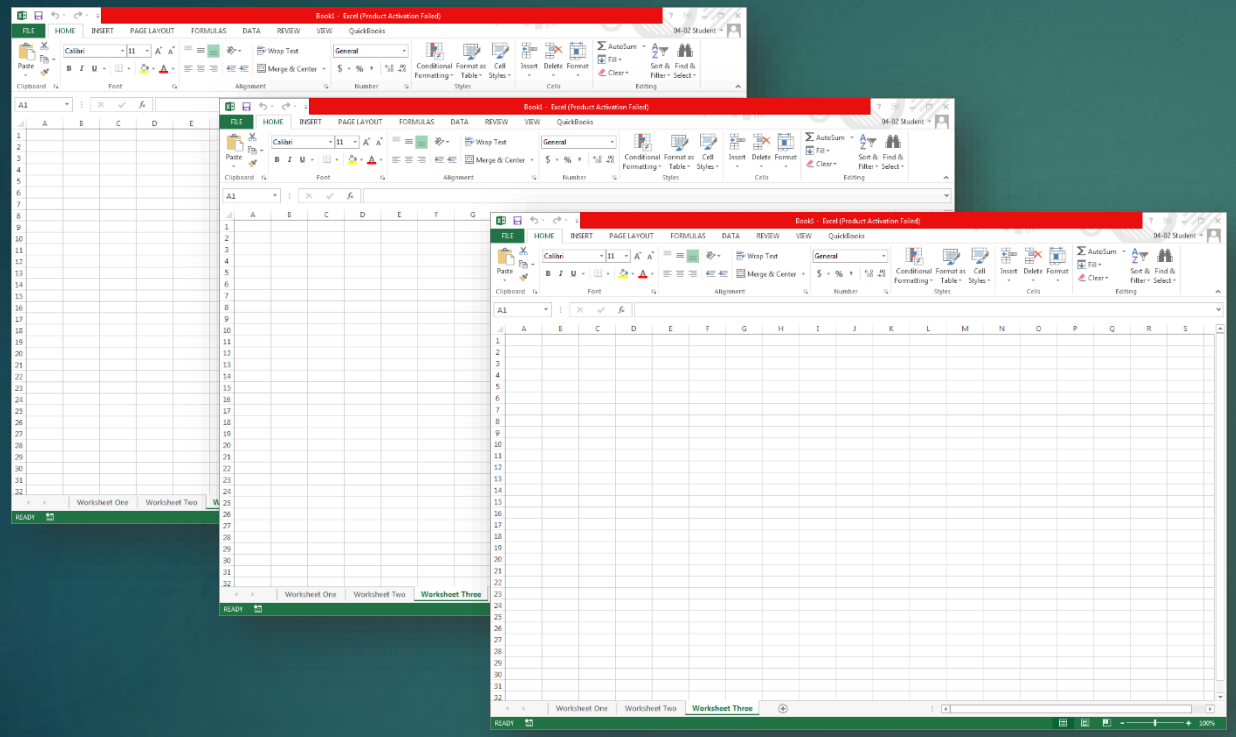
To check if a case is a **number** **Range**

Case 1 To 10

To check if a case **Compares**

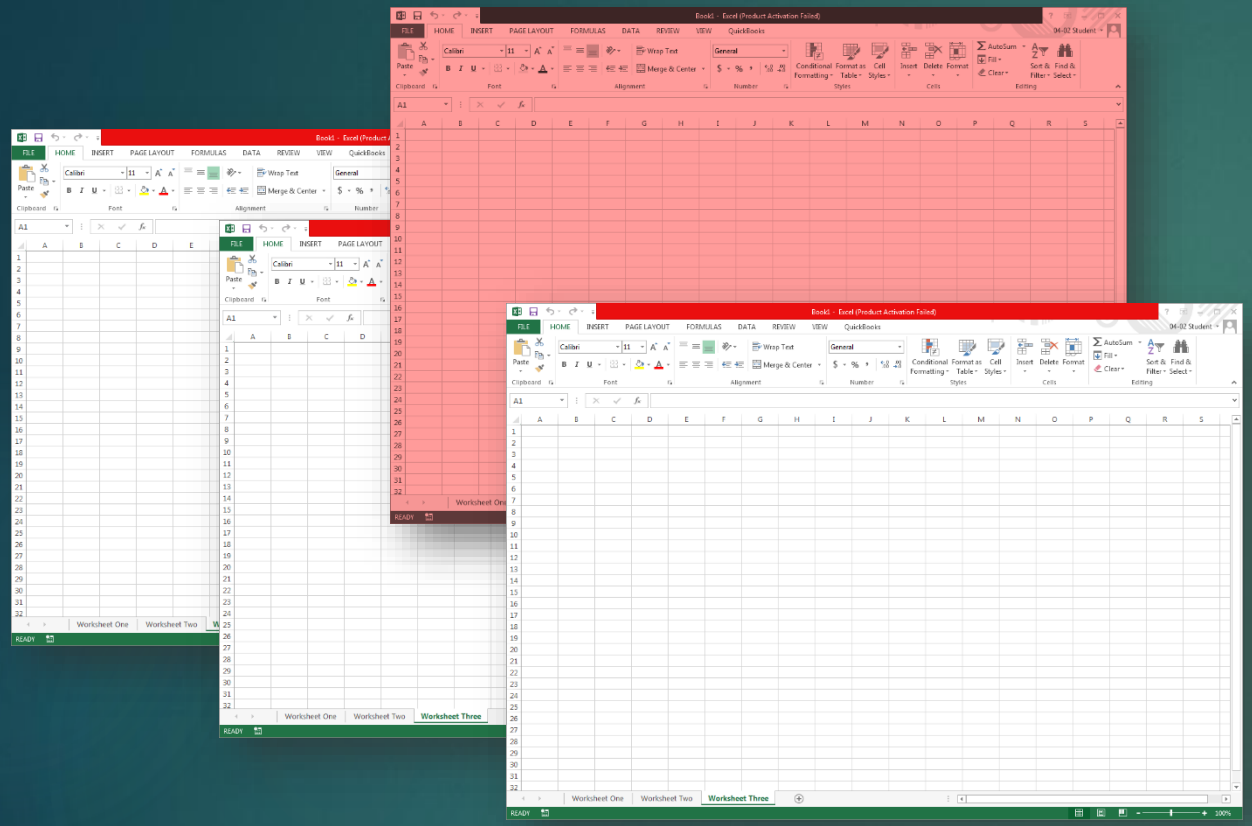
Case Is < 100

The Add Method



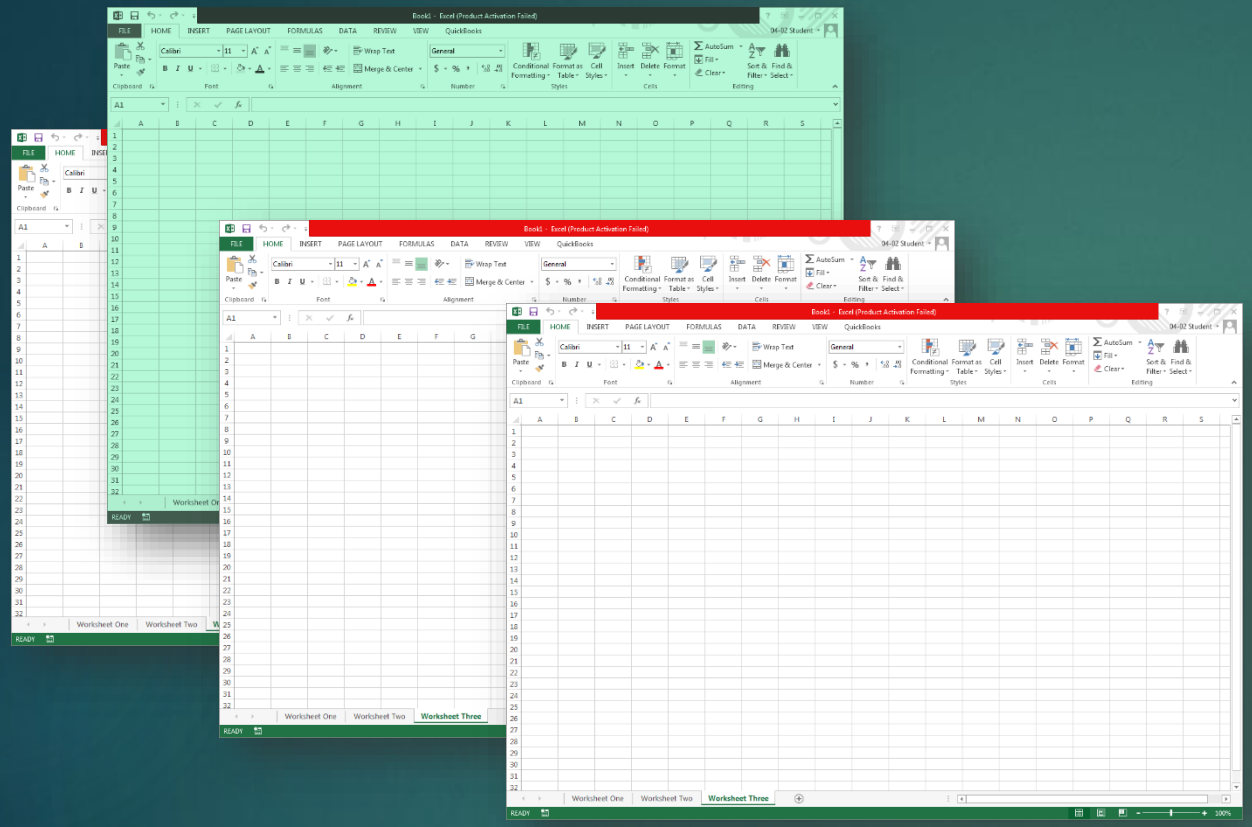
`Worksheets.Add`

The Add Method



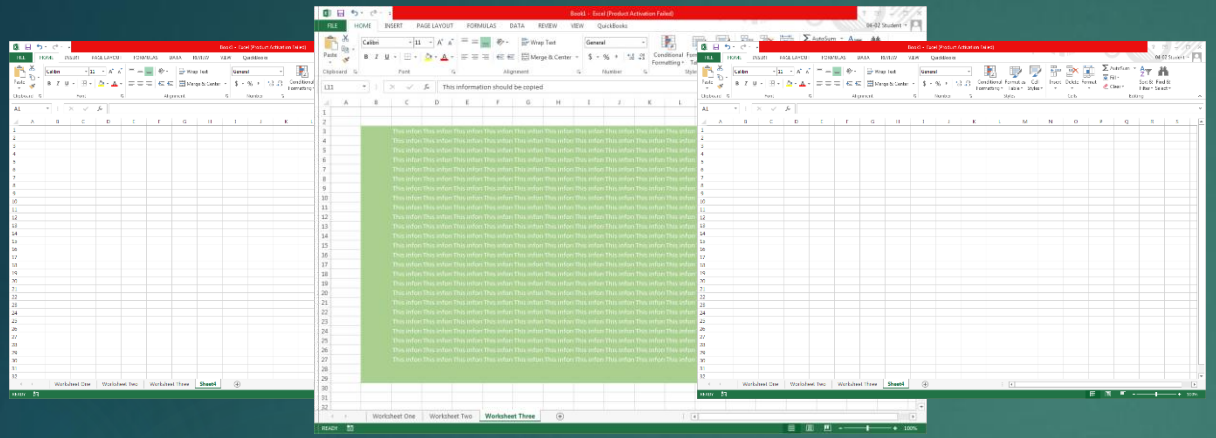
```
Worksheets.Add Before:=Worksheets(3)
```

The Add Method



```
Worksheets.Add After:=Worksheets(1)
```

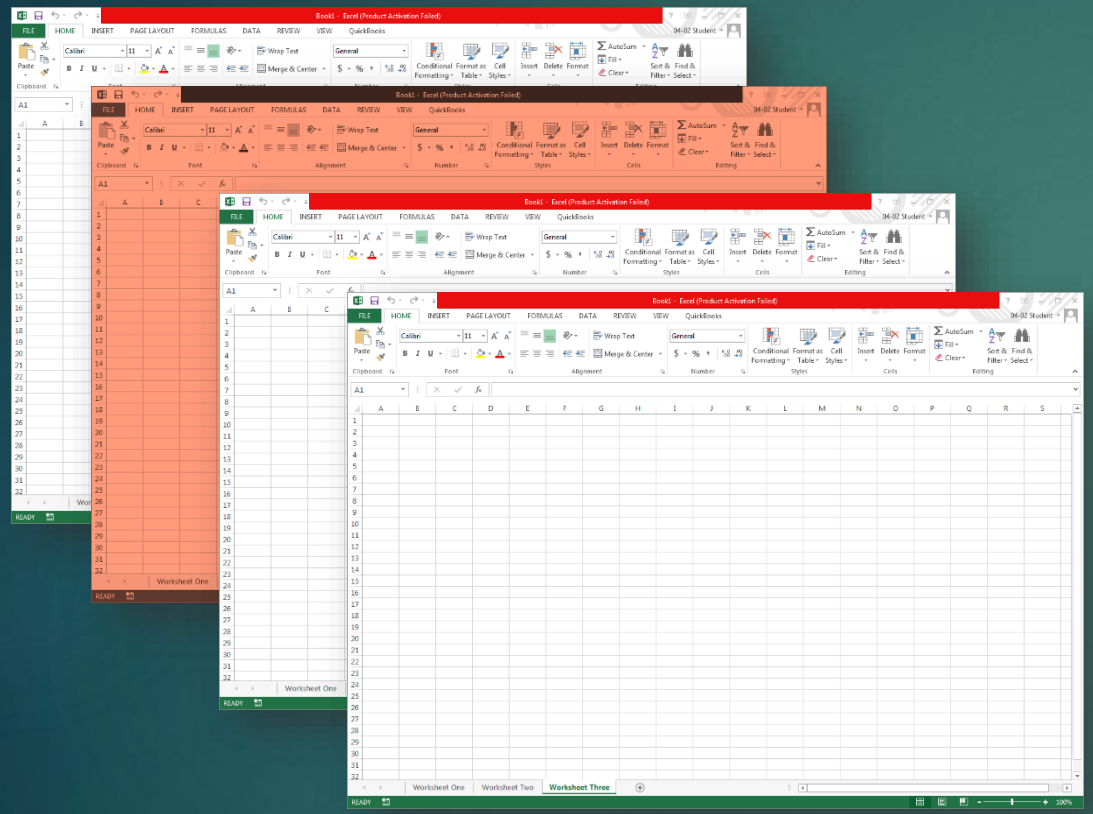
The Copy Method



```
Worksheets(2).Copy Before:=Worksheets(4)
```

```
Worksheets(2).Copy After:=Worksheets(2)
```

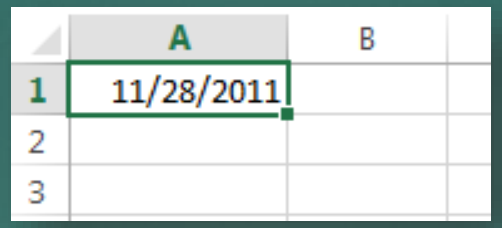
The Delete Method



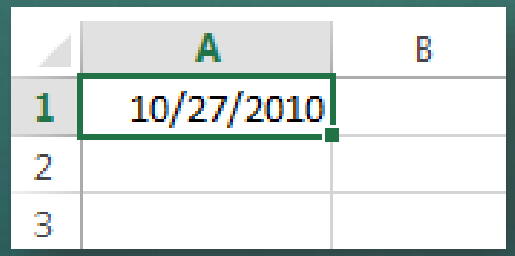
```
Worksheets(2).Delete
```

The DateSerial Function

```
Range("A1").Value = DateSerial(2011, 11, 28)
```

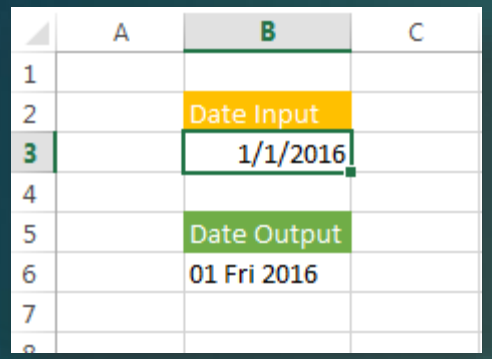



```
Range("A1").Value = DateSerial(2011-1, 11-1, 28-1)
```



The Format Function (for Dates)

Note! This returns a text value, date's serial number is lost

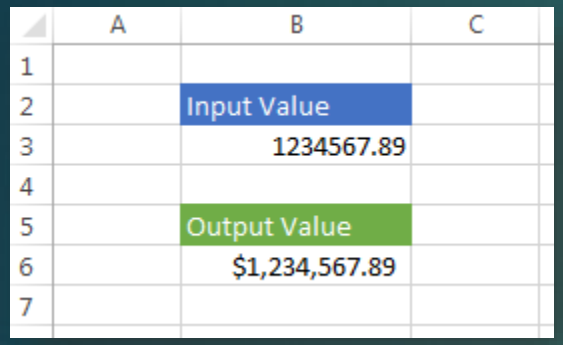


	A	B	C
1			
2		Date Input	
3		1/1/2016	
4			
5		Date Output	
6		01 Fri 2016	
7			

```
Date_Test = Range("B3").Value

Range("B6") = Format(Date_Test, "mm ddd yyyy")
```

Characters	Example	Description
m	2	Month (numerical without zeros)
mm	02	Month (numerical with zeros)
mmm	Feb	Month (abbreviated text)
mmmm	February	Month (full-length text)
d	7	Day (numerical without zeros)
dd	07	Day (numerical with zeros)
ddd	Tue	Day (abbreviated text)
dddd	Tuesday	Days (full-length text)
yy	12	Year (last 2 digits)
yyyy	2012	Year (4 digits)
h	8	Hours without zeros (0-23)
hh	08	Hours with zeros (00-23)
n	3	Minutes without zeros (0-59)
nn	03	Minutes with zeros (00-59)
s	8	Seconds without zeros (0-59)
ss	08	Seconds with zeros (00-59)
AM/PM	AM	Display AM/PM

The Format Function (for Non-Dates, Main Categories)



	A	B	C
1			
2		Input Value	
3		1234567.89	
4			
5		Output Value	
6		$1,234,567.89	
7			

```
Number_Test = Range("B3").Value

Range("B6") = Format(Number_Test, "Currency")
```

General Number

1234567.89

Standard

1,234,567.89

Currency

$1,234,567.89

Percent

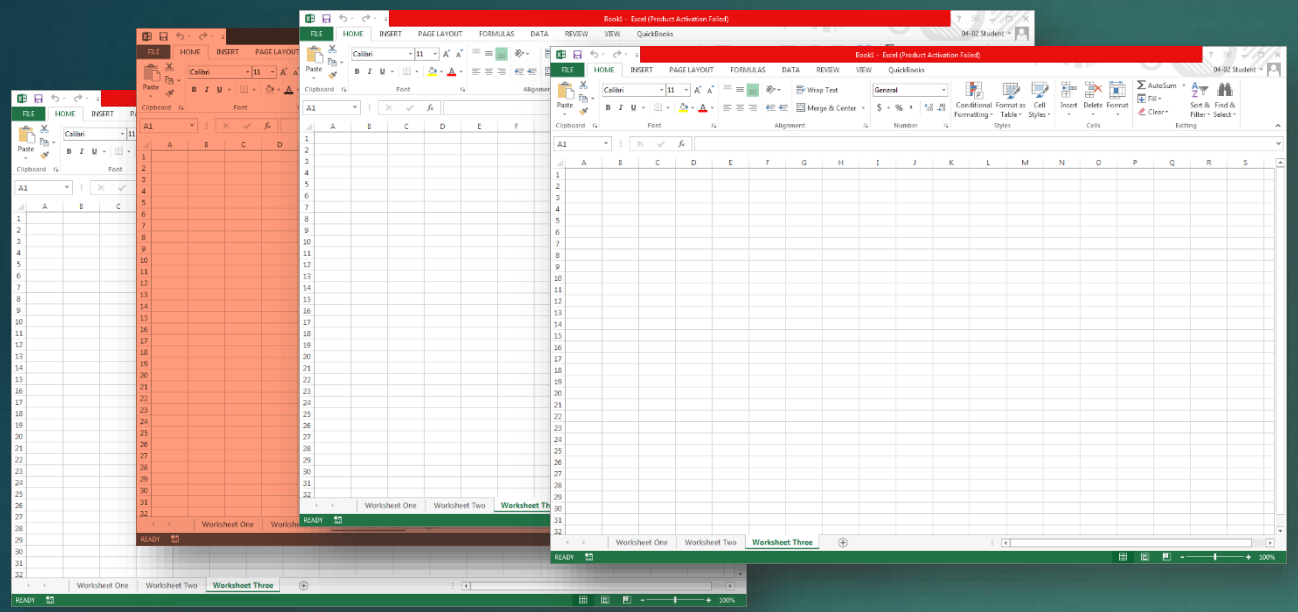
123456789.00%

The Number Format Option

Unlike the Format Function, this just changes the formatting of the number; Maintains the serial number for the date

```
Selection.NumberFormat = "m/d/yyyy"
```

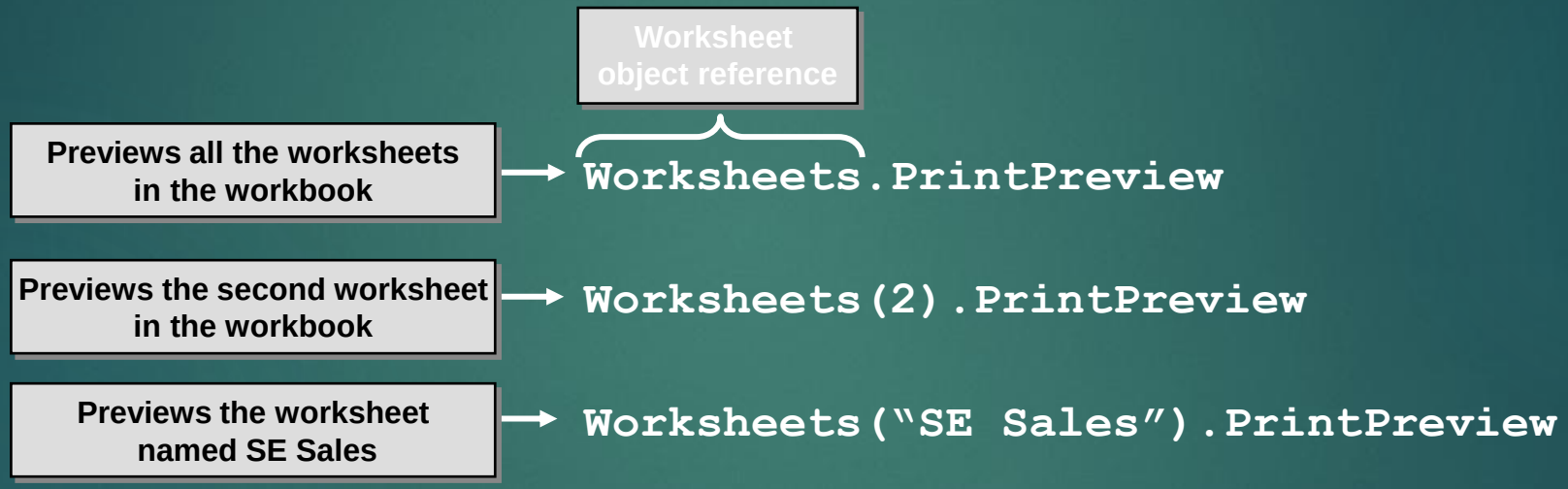
The Move Method



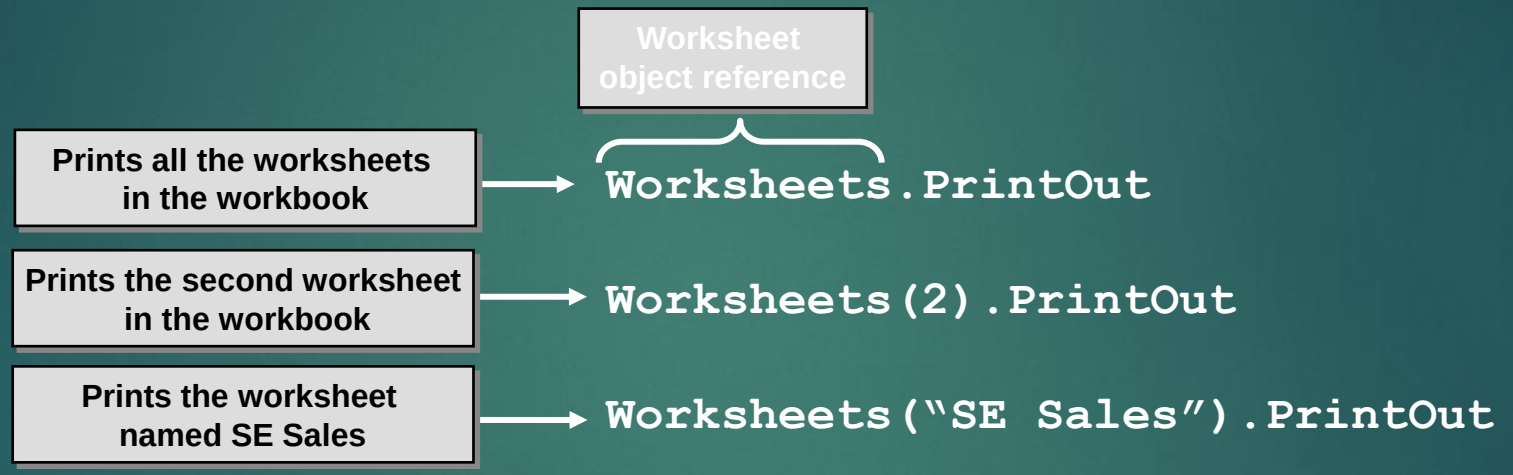
```
Worksheets(2).Move Before:=Worksheets(4)
```

```
Worksheets(2).Move After:=Worksheets(2)
```

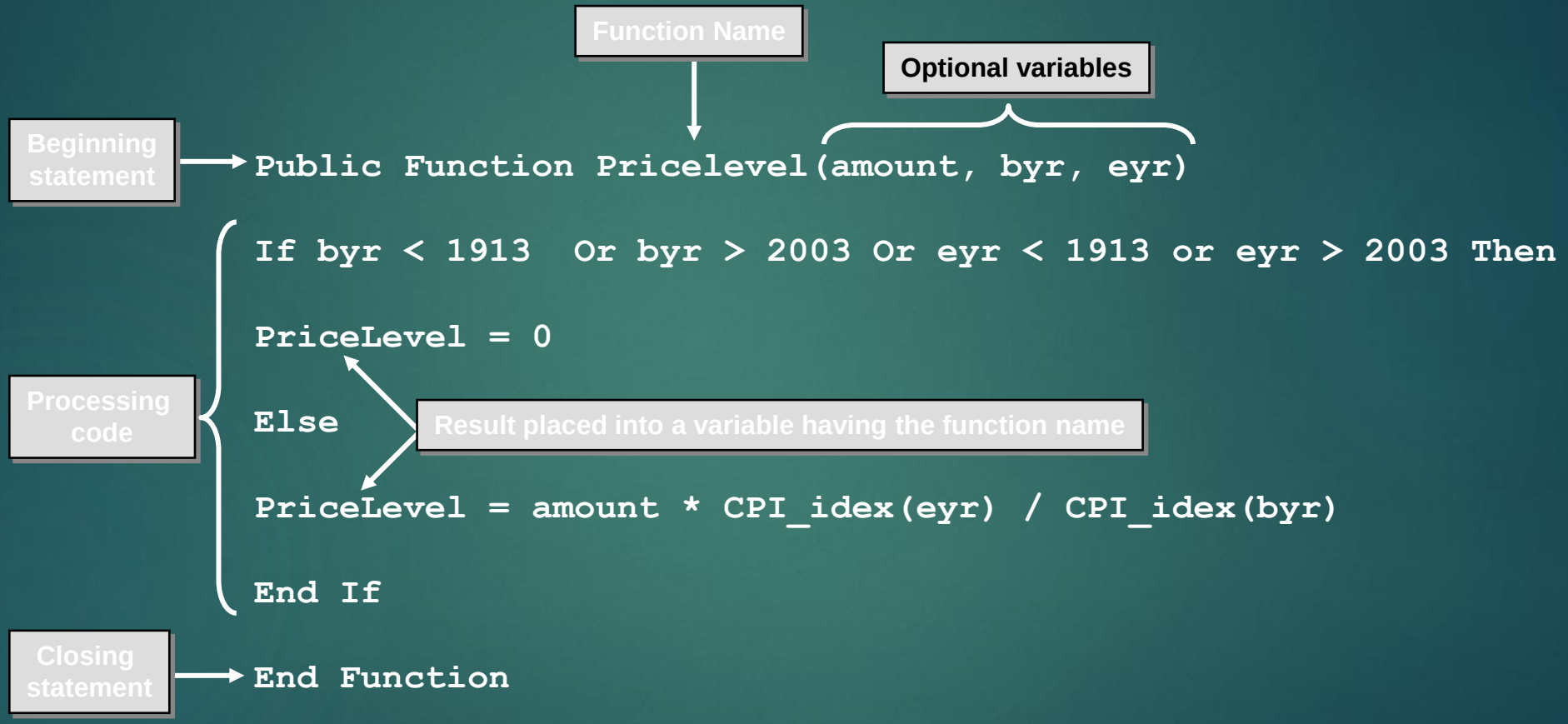
The PrintPreview Method



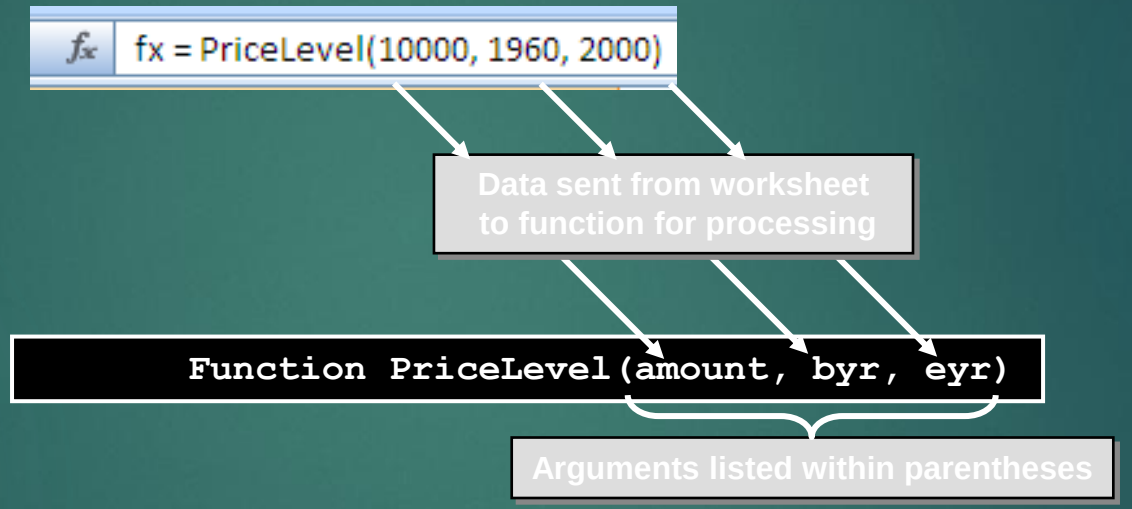
The PrintOut Method



User-Defined Functions



Arguments


fx = PriceLevel(10000, 1960, 2000)
Data sent from worksheet to function for processing
Function PriceLevel(amount, byr, eyr)
Arguments listed within parentheses

Declared Range Objects

```
Dim MyRange As Range
```

The Set Statement

	A	B	C	D	E
1	**Rep:**	**Client:**	**Fund:**	**Amount:**	**Commission:**
2	Adams	Daniels, T	GISH	$ 74,150.62	$ 1,668.39
3	Jones	Shuman, R	WKDH	$ 282,128.53	$ 4,937.25
4	Williams	Girdley, P	FISH	$ 1,392,406.26	$ 17,405.08
5	Gregory	Traymore, E	QKDH	$ 1,139,173.09	$ 14,239.66

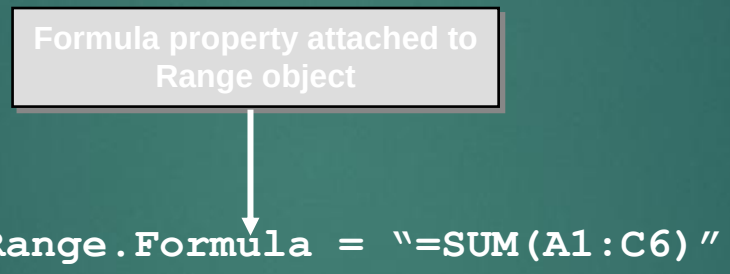
```
Set MyRange = Range("A2:E5")
```

The Rows Property

	A	B	C	D	E
1	**Rep:**	**Client:**	**Fund:**	**Amount:**	**Commission:**
2	Adams	Daniels, T	GISH	$ 74,150.62	$ 1,668.39
3	Jones	Shuman, R	WKDH	$ 282,128.53	$ 4,937.25
4	Williams	Girdley, P	FISH	$ 1,392,406.26	$ 17,405.08
5	Gregory	Traymore, E	QKDH	$ 1,139,173.09	$ 14,239.66

```
Set MyRange = Range("A2:E5")
MyRange.Rows(3)
```

The Formula Property



The Columns Property

	A	B	C	D	E
1	**Rep:**	**Client:**	**Fund:**	**Amount:**	**Commission:**
2	Adams	Daniels, T	GISH	$ 74,150.62	$ 1,668.39
3	Jones	Shuman, R	WKDH	$ 282,128.53	$ 4,937.25
4	Williams	Girdley, P	FISH	$ 1,392,406.26	$ 17,405.08
5	Gregory	Traymore, E	QKDH	$ 1,139,173.09	$ 14,239.66

```
Set MyRange = Range("A2:E5")
MyRange.Columns(3)
```

Format Function

General Number	Displays a number without thousand separators.
Currency	Displays thousand separators as well as two decimal places.
Fixed	Displays at least one digit to the left of the decimal place and two digits to the right of the decimal place.
Standard	Displays the thousand separators, at least one digit to the left of the decimal place, and two digits to the right of the decimal place.
Percent	Displays a percent value - that is, a number multiplied by 100 with a percent sign. Displays two digits to the right of the decimal place.
Scientific	Scientific notation.
Yes/No	Displays No if the number is 0. Displays Yes if the number is not 0.
True/False	Displays False if the number is 0. Displays True if the number is not 0.
On/Off	Displays Off if the number is 0. Displays On is the number is not 0.
General Date	Displays date based on your system settings
Long Date	Displays date based on your system's long date setting
Medium Date	Displays date based on your system's medium date setting
Short Date	Displays date based on your system's short date setting
Long Time	Displays time based on your system's long time setting
Medium Time	Displays time based on your system's medium time setting
Short Time	Displays time based on your system's short time setting

Time for a Break

- We'll resume class at 2:50pm