Introduction to VBA Macros in MS Excel

Magdy Helal & **Sandra Archer**
Office of University Analysis and Planning Support
University of Central Florida

Feb 07, 2008

FLORIDA ASSOCIATION FOR INSTITUTIONAL RESEARCH
2008 FAIR Conference - February 6 – 8, 2008 – Indialantic, FL

Objectives

- Introduce MS Excel Visual Basic for Applications (VBA)
- Record and runn VBA macros
- Introduce the VB Editor (VBE)
- Edit a VBA macro
- Debug and Run macros in VBE

Visual Basic For Applications

- A macro writing language
- Event-driven programming language

- A version of the Visual Basic programming language, with tools, functions, and methods customized to manipulate the "*object model*" of an application

- Standard in MS Office: *you already have it !*

How Excel VBA Works

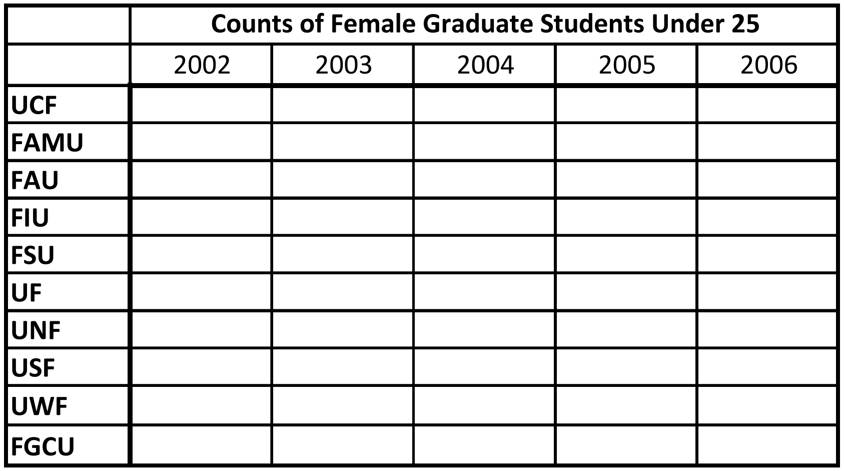
- Record, write, or edit recorded VBA code (macros) to automate a complex or repetitive Excel task
  - Macros are saved in your specific Excel workbook or globally

- Macros can be associated with events; click, double-click, keystroke, etc to fire them
  - You may add Windows objects (buttons, dropdown lists, check boxes, textboxes, dialogue boxes, menus, etc..) then write VBA code for each

- The VB editor (VBE) is a part of Excel to view, edit, and run macros
  - A VB compiler in MS Office executes your code

Demonstration Task

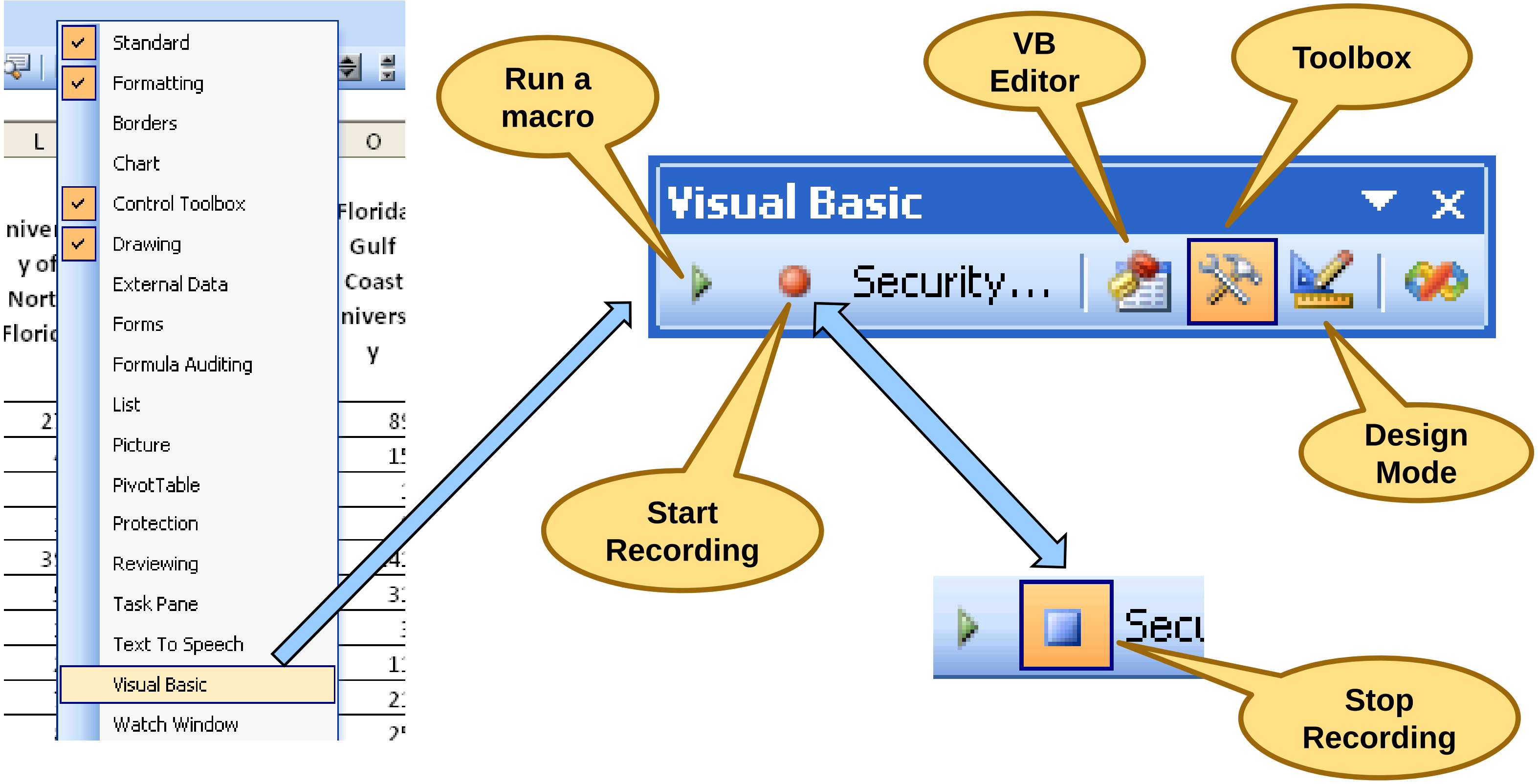
- Goal:
  - Fill in the shown table using the given dataset (in an Excel sheet)
  - Have the table filled automatically using a VBA macro
- Process:
  - Study the data and plan steps to prepare the summary for 2002
  - Record a VBA macro while performing these steps
  - Edit the macro to use it for the rest of the years



	Counts of Female Graduate Students Under 25				
	2002	2003	2004	2005	2006
UCF					
FAMU					
FAU					
FIU					
FSU					
UF					
UNF					
USF					
UWF					
FGCU					

The VB Tool Bar

- Right-click in the toolbars area in Excel and select the Visual Basic toolbar from the toolbars list



Run a macro

VB Editor

Toolbox

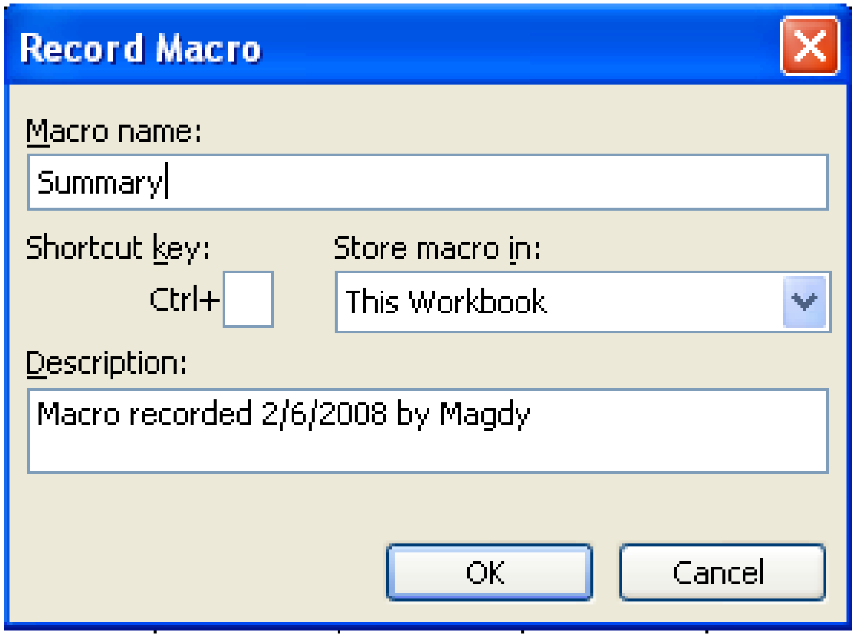
Start Recording

Design Mode

Stop Recording

Recording a Macro

- Click the Record button
- This dialog box appears
- Give a name to the macro
- Give a shortcut, if you want
- Specify where Excel saves it:
  - This Workbook
  - New Workbook
  - Personal Macro Workbook
    - Saved in the STARTUP folder in your Office application directory
    - Becomes available whenever you use Excel



Macro name: Summary

Shortcut key: Ctrl+

Store macro in: This Workbook

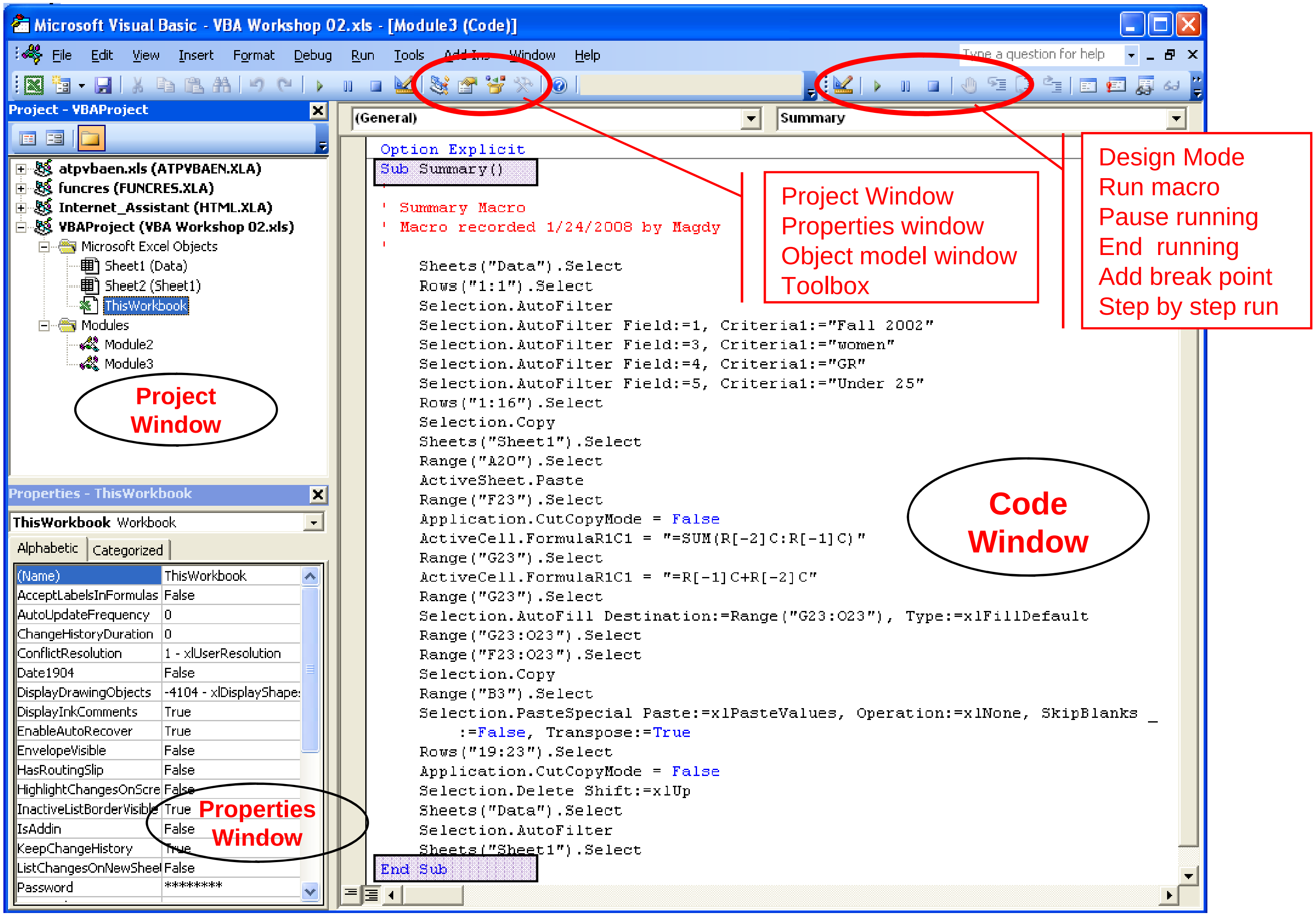
Description: Macro recorded 2/6/2008 by Magdy

Record The Macro

- Based on *my plan*, I will do the following while recording:
  - □ Turn on the auto filter in Excel, using the first row of the data
  - □ Filter to 2002, graduate, women, under 25 years of age
  - □ Copy the filtered records and title row to the table sheet
  - □ Add summation formulas for the filtered records
  - □ Copy and paste-special (values & transpose) the summation cells into the table under the 2002 title
  - □ Go back to Data sheet and remove the filter
- End recording

- Open VBE to view the recorded macro (next slide)

The VB Editor (VBE)


Microsoft Visual Basic - VBA Workshop 02.xls - [Module3 (Code)]
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Type a question for help
Project - VBAProject
atpvbaen.xls (ATPVBAEN.XLA)
funcres (FUNCRES.XLA)
Internet_Assistant (HTML.XLA)
VBAProject (VBA Workshop 02.xls)
Microsoft Excel Objects
Sheet1 (Data)
Sheet2 (Sheet1)
ThisWorkbook
Modules
Module2
Module3
Properties - ThisWorkbook
ThisWorkbook Workbook
Alphabetic
Categorized
(General)
Summary

(Name)	ThisWorkbook
AcceptLabelsInFormulas	False
AutoUpdateFrequency	0
ChangeHistoryDuration	0
ConflictResolution	1 - xlUserResolution
Date1904	False
DisplayDrawingObjects	-4104 - xlDisplayShape
DisplayInkComments	True
EnableAutoRecover	True
EnvelopeVisible	False
HasRoutingSlip	False
HighlightChangesOnScre	False
InactiveListBorderVisible	True
IsAddin	False
KeepChangeHistory	True
ListChangesOnNewSheet	False
Password	********

```
Option Explicit
Sub Summary()

' Summary Macro
' Macro recorded 1/24/2008 by Magdy
'
    Sheets("Data").Select
    Rows("1:1").Select
    Selection.AutoFilter
    Selection.AutoFilter Field:=1, Criteria1:="Fall 2002"
    Selection.AutoFilter Field:=3, Criteria1:="women"
    Selection.AutoFilter Field:=4, Criteria1:="GR"
    Selection.AutoFilter Field:=5, Criteria1:="Under 25"
    Rows("1:16").Select
    Selection.Copy
    Sheets("Sheet1").Select
    Range("A20").Select
    ActiveSheet.Paste
    Range("F23").Select
    Application.CutCopyMode = False
    ActiveCell.FormulaR1C1 = "=SUM(R[-2]C:R[-1]C)"
    Range("G23").Select
    ActiveCell.FormulaR1C1 = "=R[-1]C+R[-2]C"
    Range("G23").Select
    Selection.AutoFill Destination:=Range("G23:O23"), Type:=xlFillDefault
    Range("G23:O23").Select
    Range("F23:O23").Select
    Selection.Copy
    Range("B3").Select
    Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
        :=False, Transpose:=True
    Rows("19:23").Select
    Application.CutCopyMode = False
    Selection.Delete Shift:=xlUp
    Sheets("Data").Select
    Selection.AutoFilter
    Sheets("Sheet1").Select
End Sub
```

Project Window
Properties window
Object model window
Toolbox

Design Mode
Run macro
Pause running
End running
Add break point
Step by step run

Project Window

Properties Window

Code Window

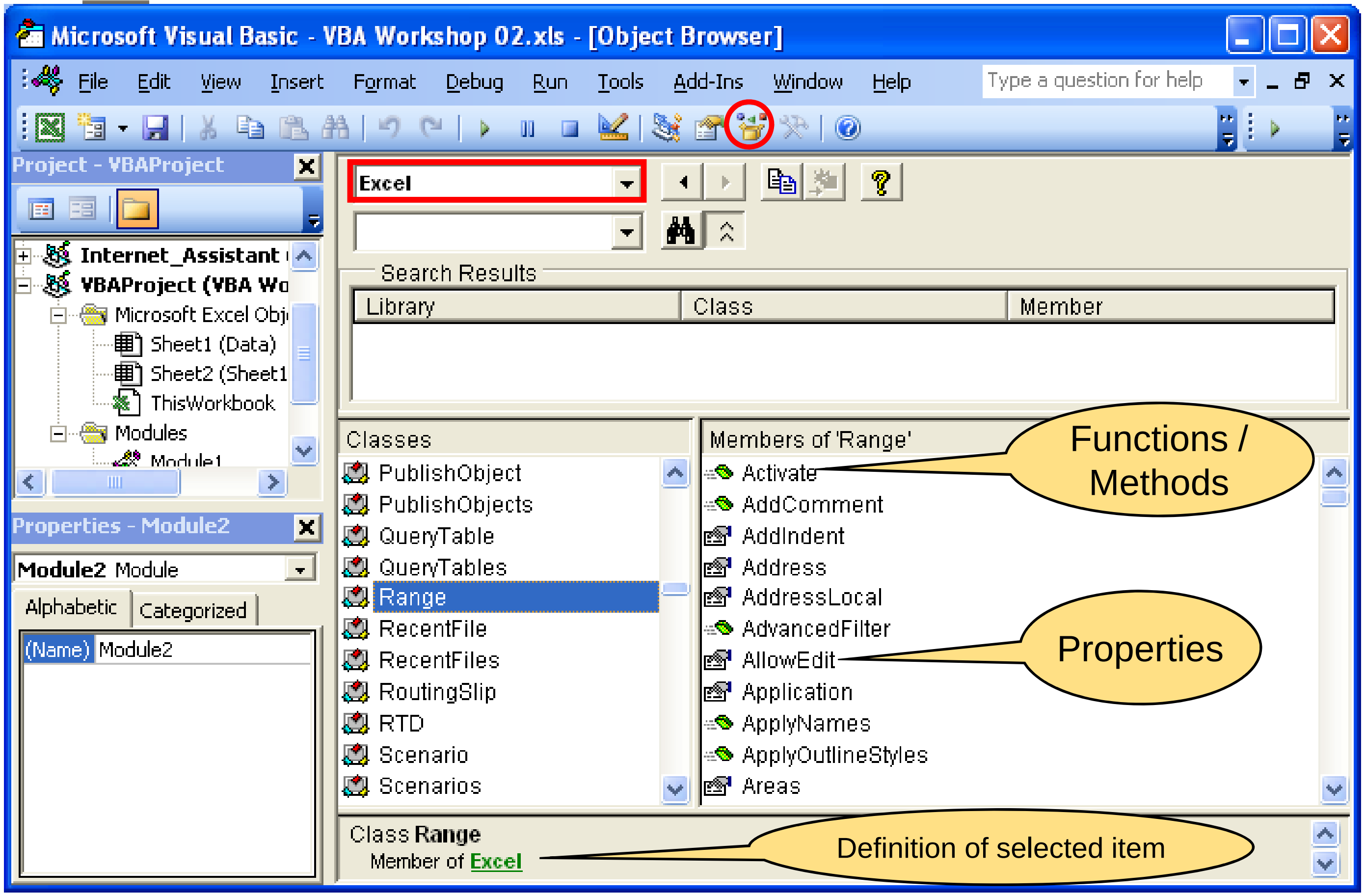
The VBA Subroutine

```
Sub SubName()
      ‘ Comments
      …
      VBA statements
      VBA statements
      …
End Sub
```

- VBA statements deal with the *Excel object model*
  - Object model: collection of workbooks, sheets, columns, rows, ranges, cells, charts, formulas, filters, …etc.
- VBA statement basic uses:
  - Change the properties of an object
  - Apply methods to the object

Excel Object Model

- Click (press F2) to start the Object Browser in VBE



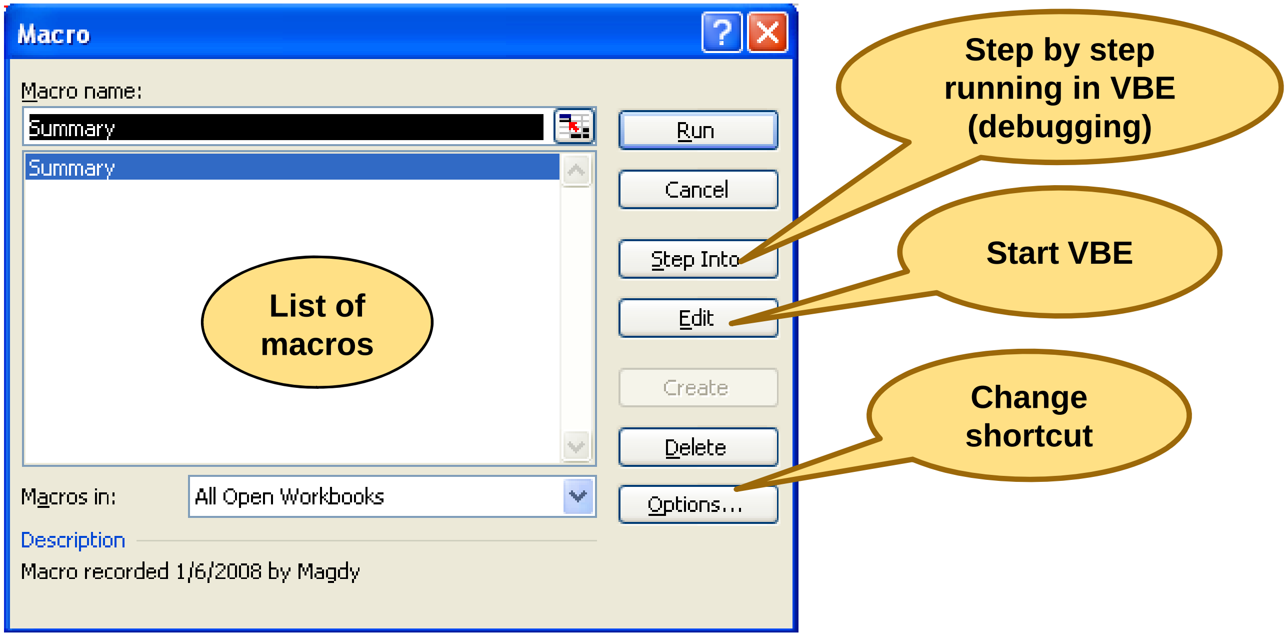
Functions / Methods

Properties

Definition of selected item

Running the Macro

- Click the Run button in the VBA menu bar (Or press Alt + F8)
- From the Macro dialogue box choose a macro to run then click Run



Macro

Macro name: Summary

Summary

Macros in: All Open Workbooks

Description

Macro recorded 1/6/2008 by Magdy

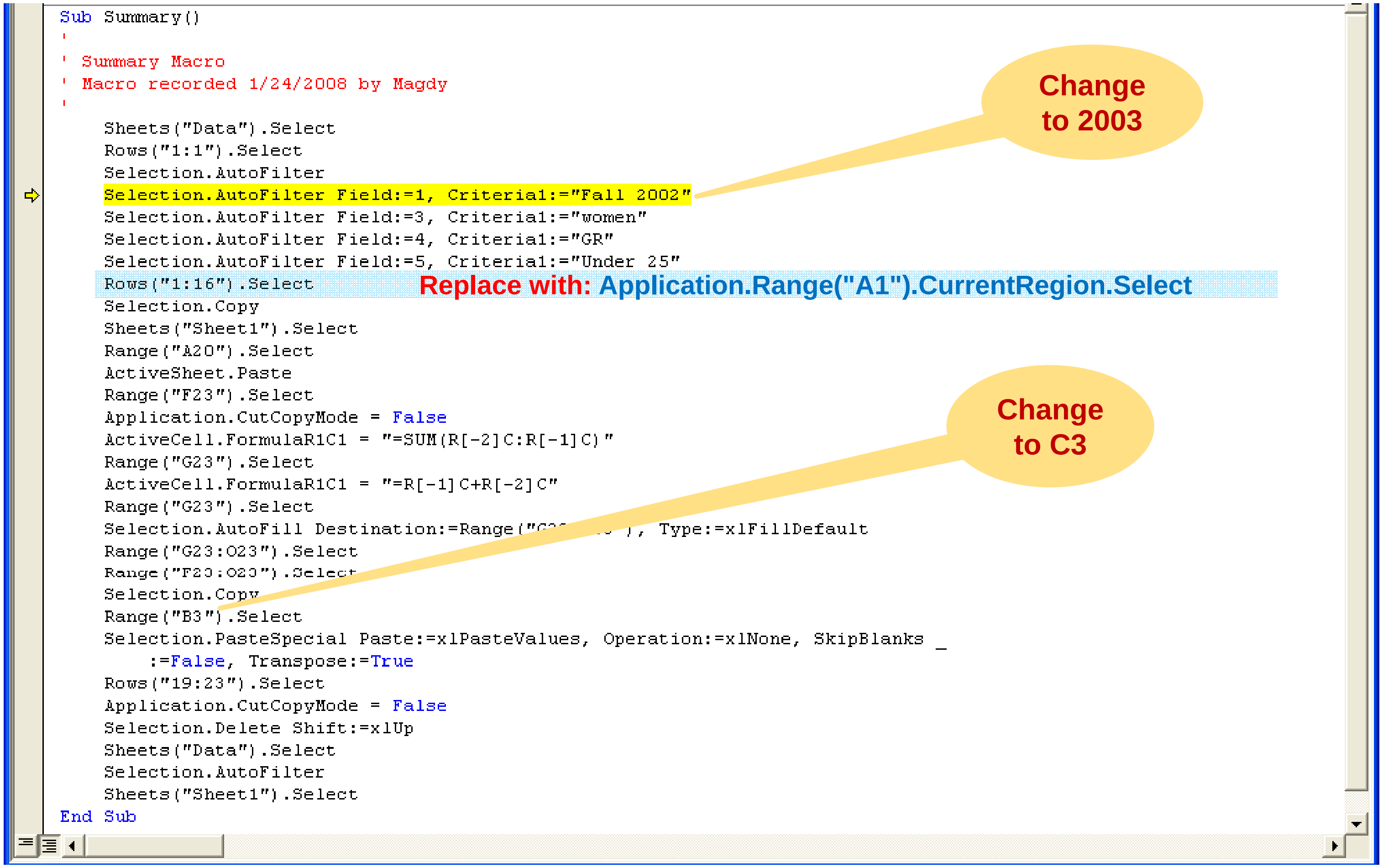
- You may run the macro using the short cut

Running the Macro in VBE

- Start VBE from the VBA toolbar (Or press Alt + F11)
- Select the macro module in the Project Window (make sure to click inside the macro you want to run)
- In VBE you may run, edit or debug the macro:
  - □ Running the macro:
    - Use the icons ▶ ⏸ ■ in VBE toolbar or the Run menu
    - Keyboard: F5 to run the current macro and Ctrl + Break to pause the run
  - □ Editing the macro
  - □ Debugging the macro
    - Use the Step Into icon for step by step running
    - Keyboard: press F8 repeatedly

Step by Step Execution and Editing

- Change the following to run the macro for the 2003 year



```
Sub Summary()
'
' Summary Macro
' Macro recorded 1/24/2008 by Magdy
'
    Sheets("Data").Select
    Rows("1:1").Select
    Selection.AutoFilter
    Selection.AutoFilter Field:=1, Criteria1:="Fall 2002"
    Selection.AutoFilter Field:=3, Criteria1:="women"
    Selection.AutoFilter Field:=4, Criteria1:="GR"
    Selection.AutoFilter Field:=5, Criteria1:="Under 25"
    Rows("1:16").Select
    Selection.Copy
    Sheets("Sheet1").Select
    Range("A20").Select
    ActiveSheet.Paste
    Range("F23").Select
    Application.CutCopyMode = False
    ActiveCell.FormulaR1C1 = "=SUM(R[-2]C:R[-1]C)"
    Range("G23").Select
    ActiveCell.FormulaR1C1 = "=R[-1]C+R[-2]C"
    Range("G23").Select
    Selection.AutoFill Destination:=Range("G2[illegible]"), Type:=xlFillDefault
    Range("G23:O23").Select
    Range("F23:O23").Select
    Selection.Copy
    Range("B3").Select
    Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
        :=False, Transpose:=True
    Rows("19:23").Select
    Application.CutCopyMode = False
    Selection.Delete Shift:=xlUp
    Sheets("Data").Select
    Selection.AutoFilter
    Sheets("Sheet1").Select
End Sub
```

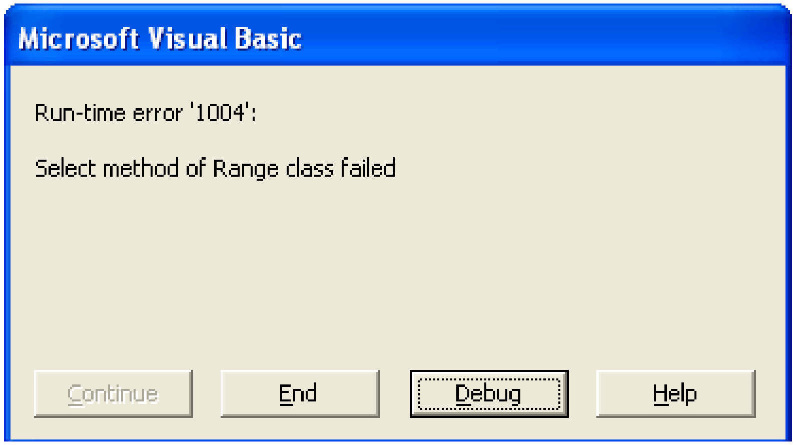
Change to 2003

Replace with: Application.Range("A1").CurrentRegion.Select

Change to C3

Error Messages

- An example of an error message:
  - □ Shows if "Application." is not used in
    - Application.Range("A1").CurrentRegion.Select
    - Sometimes depending on Excel version
  - □ End: Stops executing the macro
  - □ Debug: takes you to the error in code
  - □ Continue: ignore the error if possible



Microsoft Visual Basic

Run-time error '1004':

Select method of Range class failed

- Common errors:
  - □ Misspelling
  - □ Variables not defined
  - □ Missing arguments or options
  - □ Referencing non-existing sheet or workbook (named object)
  - □ Not using **Application.** before Range, Rows, Columns, Sheets, and few others

Some Editing

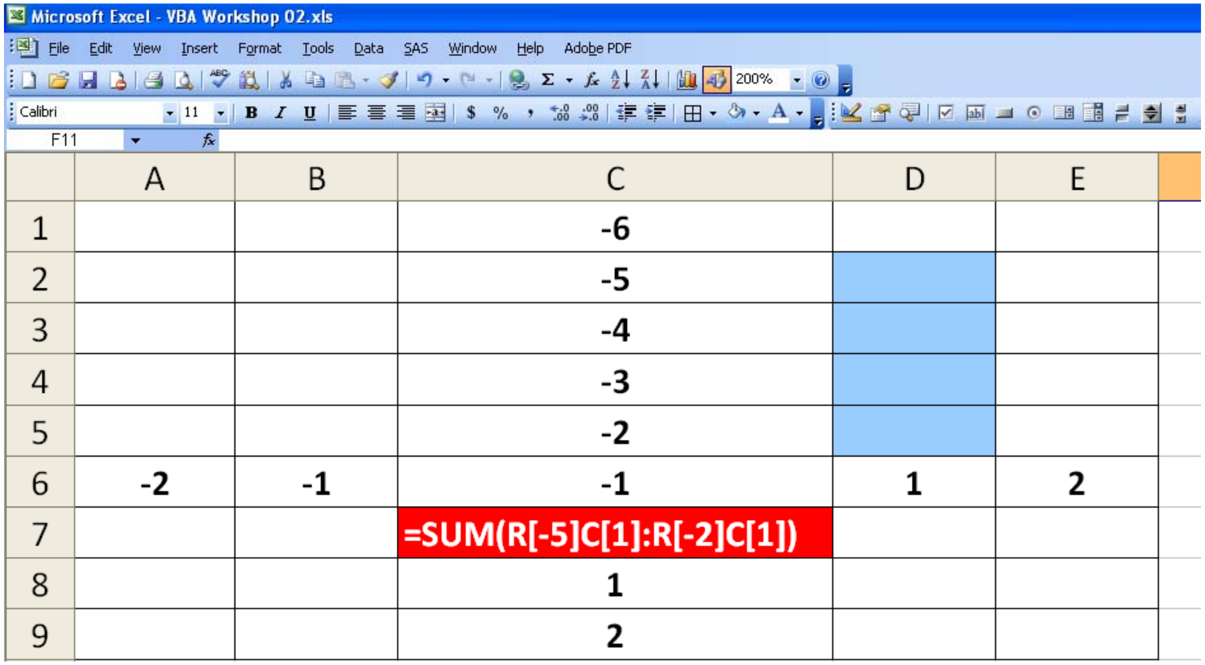
- Referring to a cell: `Cells(Rowindex,Columnindex)`
  - □ `Cell in row 3 and column 21: Cells(3,21).Select`

- Referring to a cells range: `Range(Cell1, [Cell2]).Select`
  - □ `Cell in row 6 and column 4: Range(“D6”).Select`
  - □ `Range across several rows and columns:`
    - `Range(cells(4,6), Cells(12,8)).Select`
    - `Range(“D4”,”H12”).Select`

- Range of columns: `Columns(FirstColIndex, LastColIndex)`
  - □ `Columns 5,6,and 7`: `Columns(“E”,“G”).Select`
- Range of rows: `Columns(FirstRowIndex, LastRowIndex)`
  - □ `Rows 2 through 7: Rows(“2:7”).Select`

R1C1 Notations in Formulas

- R1C1 uses numbers for columns instead of letters
- Excel uses this format when recording a macro
- Must be inserted as a character in the selected cell range
  - □ Ex: Range("C7").FormulaR1C1 = “=SUM(R[-5]C[1]:R[-2]C[1])“

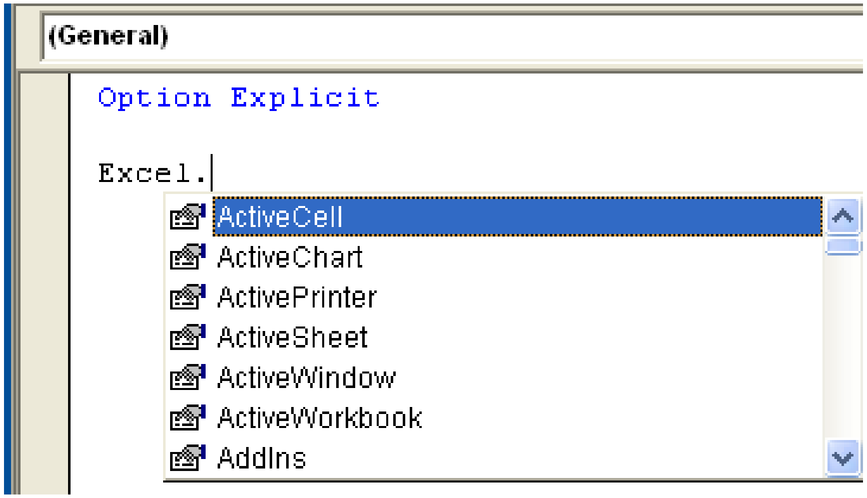


Microsoft Excel - VBA Workshop 02.xls

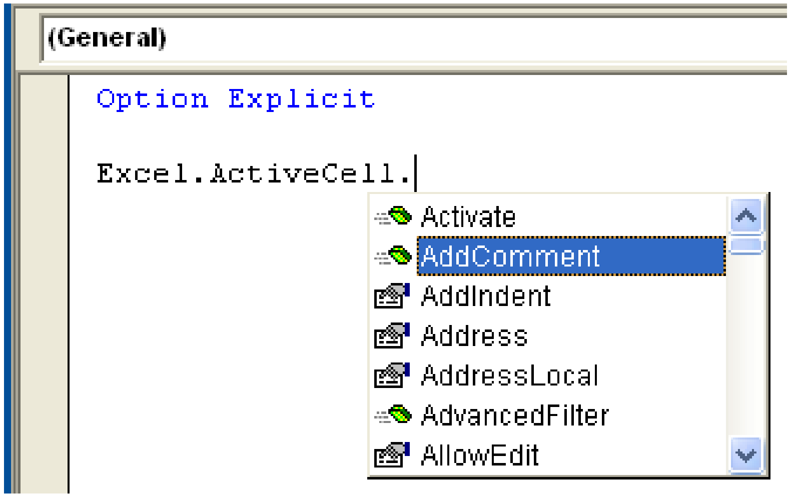
	A	B	C	D	E
1			-6		
2			-5		
3			-4		
4			-3		
5			-2		
6	-2	-1	-1	1	2
7			=SUM(R[-5]C[1]:R[-2]C[1])		
8			1		
9			2		

Context Menus in Editing

- Each member of the Excel object model has properties, methods, functions, and sometimes members of its own
- The "." operator following the object name activates its members list
- Lists all that you may do with the object
- There can be several levels of membership



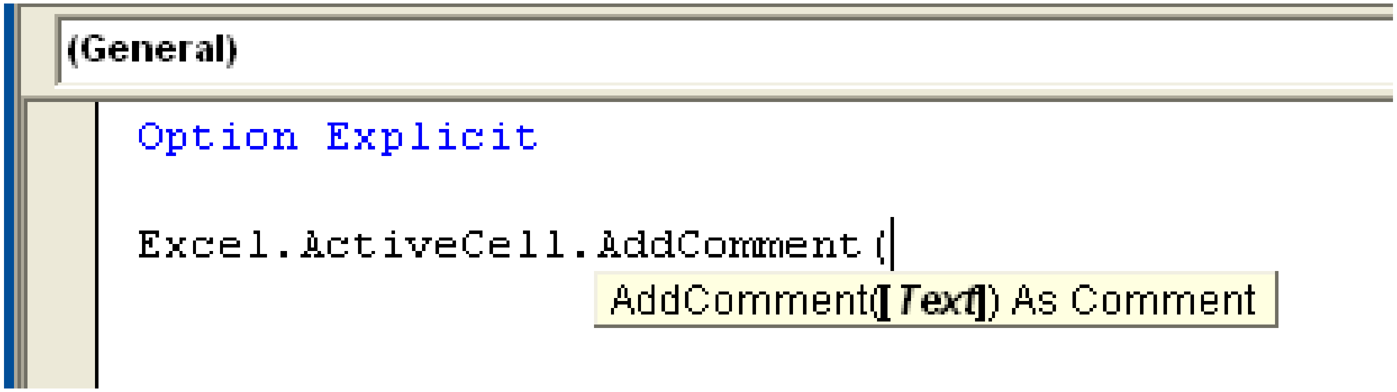
```
(General)
Option Explicit

Excel.
```



```
(General)
Option Explicit

Excel.ActiveCell.
```

Context Menus in Editing

- Many members require arguments and may have options
  - □ When adding a comment in a cell, type **“(“** after selecting AddComment:



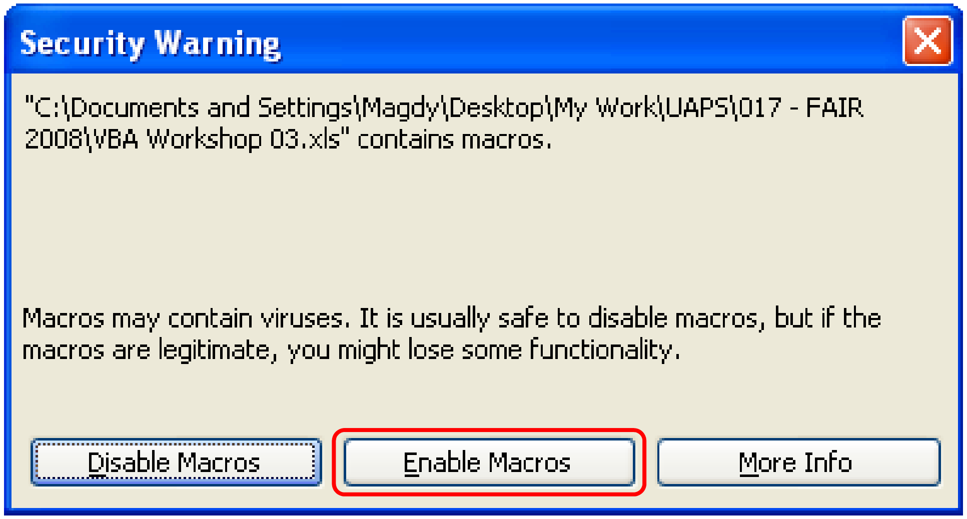
```
Option Explicit

Excel.ActiveCell.AddComment (
                    AddComment([Text]) As Comment
```

- Add the comment text, e.g. “This is my chosen cell”
- Type **“)“** to end the statement

- Options do not have to be between parentheses:
  - □ Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _ :=False, Transpose:=True

Security Settings

- Upon opening an Excel file with a macro the following warning appears
- To use the macro click "Enable Macros"
- If "Disable Macros" clicked, Excel will not load the VBA modules



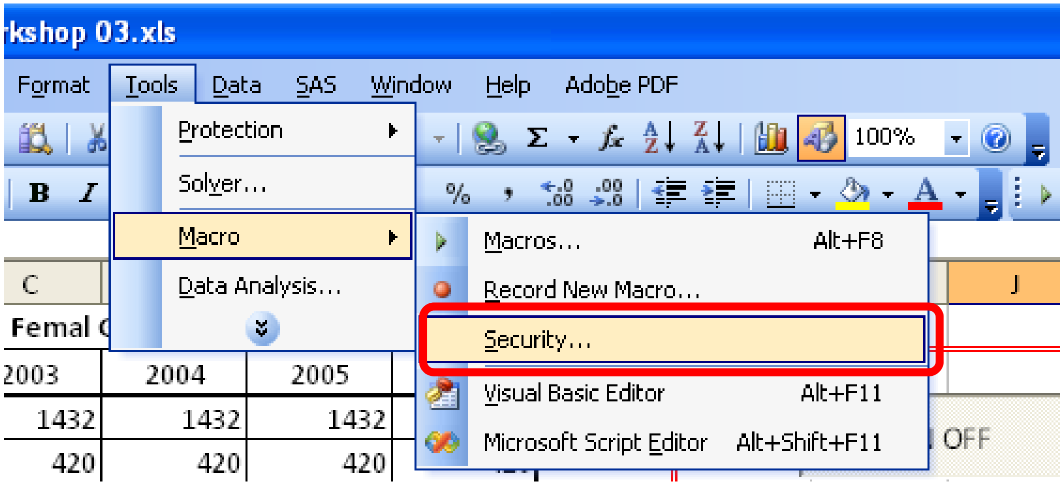
"C:\Documents and Settings\Magdy\Desktop\My Work\UAPS\017 - FAIR 2008\VBA Workshop 03.xls" contains macros.

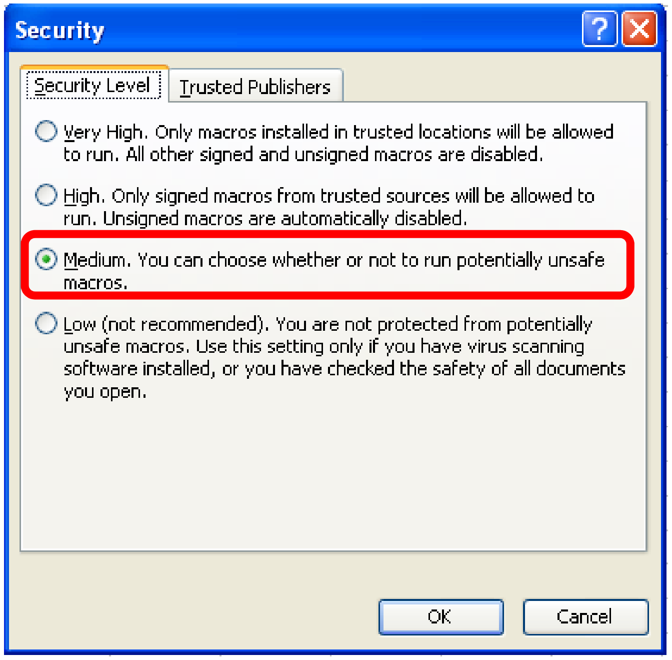
Macros may contain viruses. It is usually safe to disable macros, but if the macros are legitimate, you might lose some functionality.

- If no warning appears, then security settings must be adjusted to allow macros

Setting Security Level

- From the Tools menu, select Macro then Security
- Set the security level in the Security dialogue box to medium
- Restart the Excel file





○ Very High. Only macros installed in trusted locations will be allowed to run. All other signed and unsigned macros are disabled.

○ High. Only signed macros from trusted sources will be allowed to run. Unsigned macros are automatically disabled.

● Medium. You can choose whether or not to run potentially unsafe macros.

○ Low (not recommended). You are not protected from potentially unsafe macros. Use this setting only if you have virus scanning software installed, or you have checked the safety of all documents you open.

Next Steps

- We have introduced using VBA in MS Excel
  - □ How to record and run a macro
  - □ How to use VBE
  - □ How to debug and edit a recorded macro
  - □ How to set the security level to allow the saved macros
- How to practice:
  - □ Record macros while perfoming different Excel tasks and study the code
  - □ Modify the code in VBE and watch the results using step by step execution
- What do you need:
  - □ ***One or more*** Excel VBA programming books and the VBA help
  - □ Learn some general purpose VB statements
    - The ***IF – Then*** statement
    - The ***FOR – Next*** statement
    - Defining variables

Contact Information

- **Magdy Helal;** Research Associate, mhelal@mail.ucf.edu
- **Sandra Archer;** Director, archer@mail.ucf.edu

  University Analysis and Planning Support
  University of Central Florida
  12424 Research Parkway, Suite 215
  Orlando, FL, 32826-3207
  Phone: 407-882-0285
  http://uaps.ucf.edu

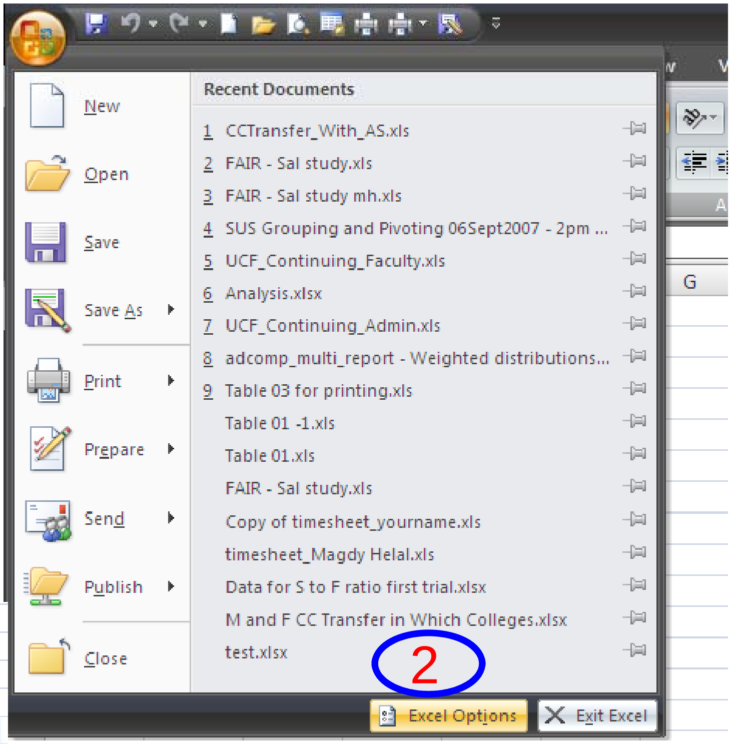
- Presentation will be available at http://uaps.ucf.edu

- VBA Resources:
  - MSDN: http://msdn2.microsoft.com/en-us/isv/bb190538.aspx
  - John Walkenbach, “Excel 2003 Power Programming with VBA”, Wiley Publishing Inc., 2003

VBA in Excel 2007

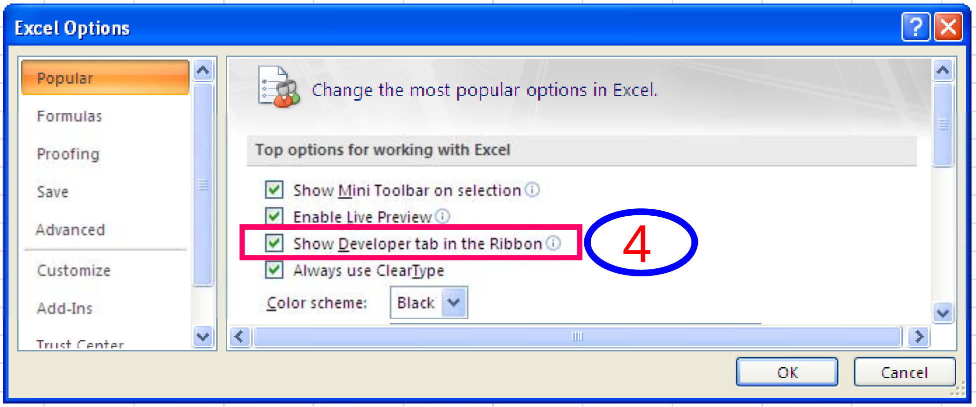
- The Visual Basic icon bar of 2003 is replaced by the Developer tab in 2007
- If the Developer tab is not shown on among Excel menus, add the Developer tab as follows:
  1. Click the Office Button
  2. Click Excel Options
  3. Select Popular
  4. Check Show Developer tab in the Ribbon

1



New
Open
Save
Save As
Print
Prepare
Send
Publish
Close
Recent Documents
1 CCTransfer_With_AS.xls
2 FAIR - Sal study.xls
3 FAIR - Sal study mh.xls
4 SUS Grouping and Pivoting 06Sept2007 - 2pm ...
5 UCF_Continuing_Faculty.xls
6 Analysis.xlsx
7 UCF_Continuing_Admin.xls
8 adcomp_multi_report - Weighted distributions...
9 Table 03 for printing.xls
Table 01 -1.xls
Table 01.xls
FAIR - Sal study.xls
Copy of timesheet_yourname.xls
timesheet_Magdy Helal.xls
Data for S to F ratio first trial.xlsx
M and F CC Transfer in Which Colleges.xlsx
test.xlsx
2
Excel Options
Exit Excel
G

3



Change the most popular options in Excel.

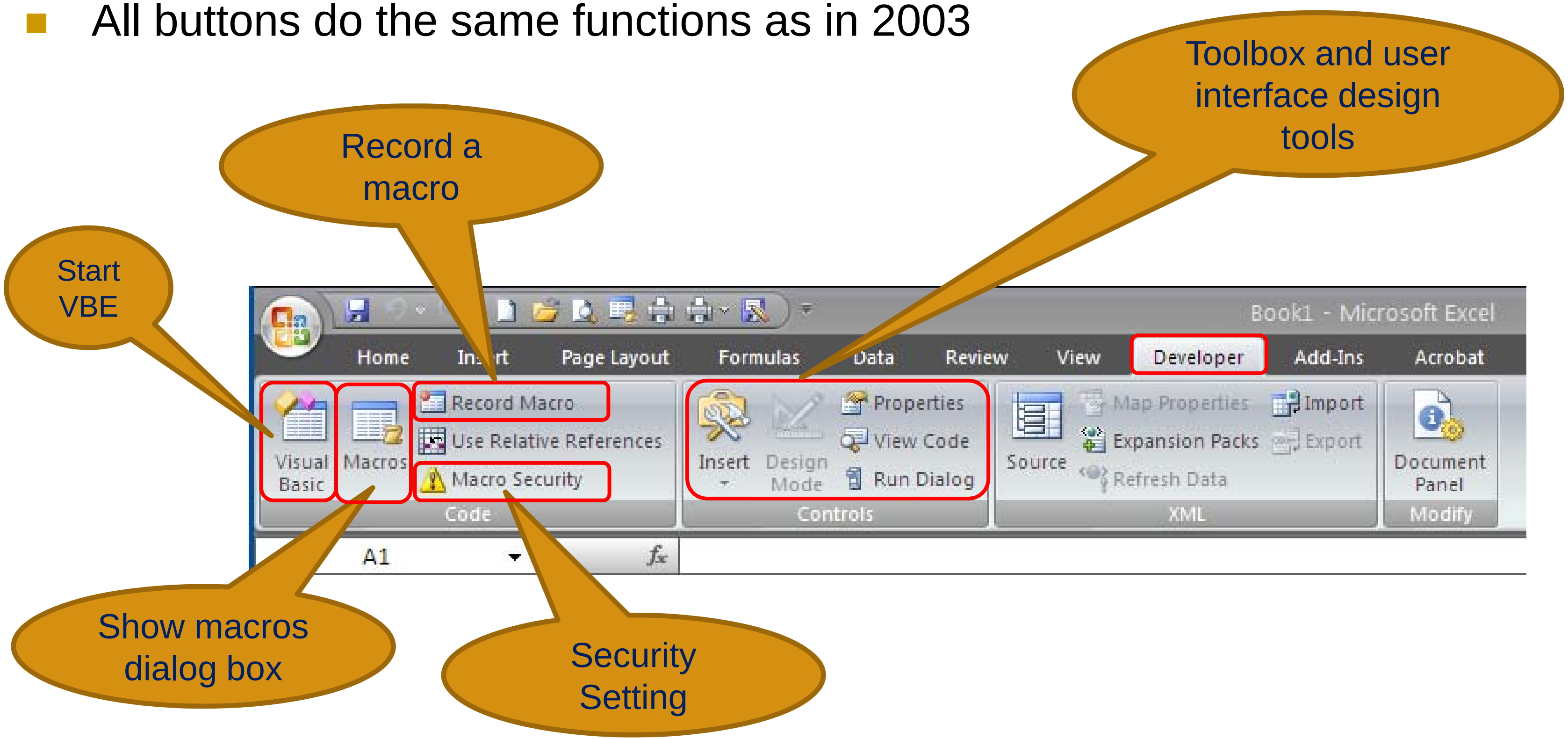
Top options for working with Excel

- ☑ Show Mini Toolbar on selection
- ☑ Enable Live Preview
- ☑ Show Developer tab in the Ribbon
- ☑ Always use ClearType

Color scheme: Black

The VBA Ribbon in 2007

- The Developer tab appears on the Excel ribbon
- VBA icons and buttons look different from those in 2003
- Buttons are pointed to
- All buttons do the same functions as in 2003



Toolbox and user interface design tools

Record a macro

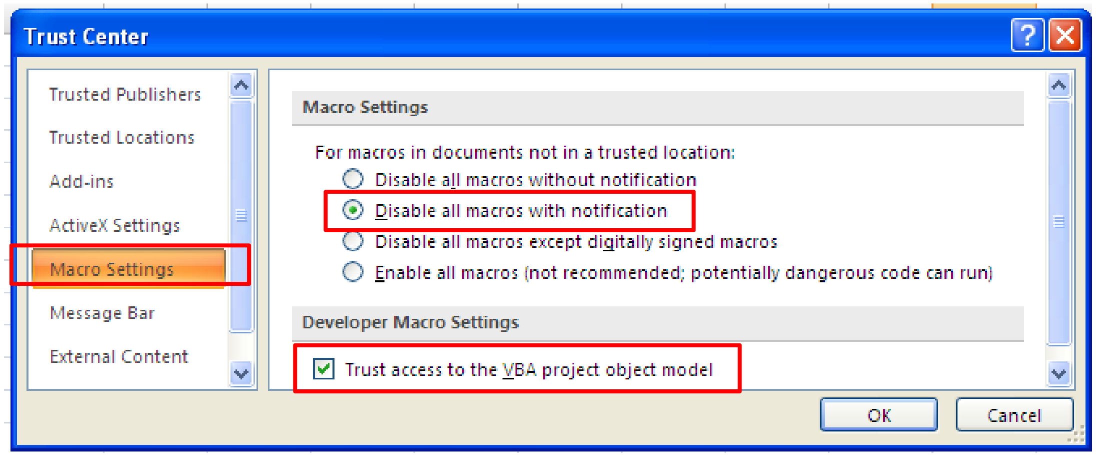
Start VBE

Show macros dialog box

Security Setting

Security Settings in 2007

- Click Macro Security in the Developer tab
- Trust Center dialog box appears
- Select Macro Settings
- Make the settings as shown



Macro Settings

For macros in documents not in a trusted location:

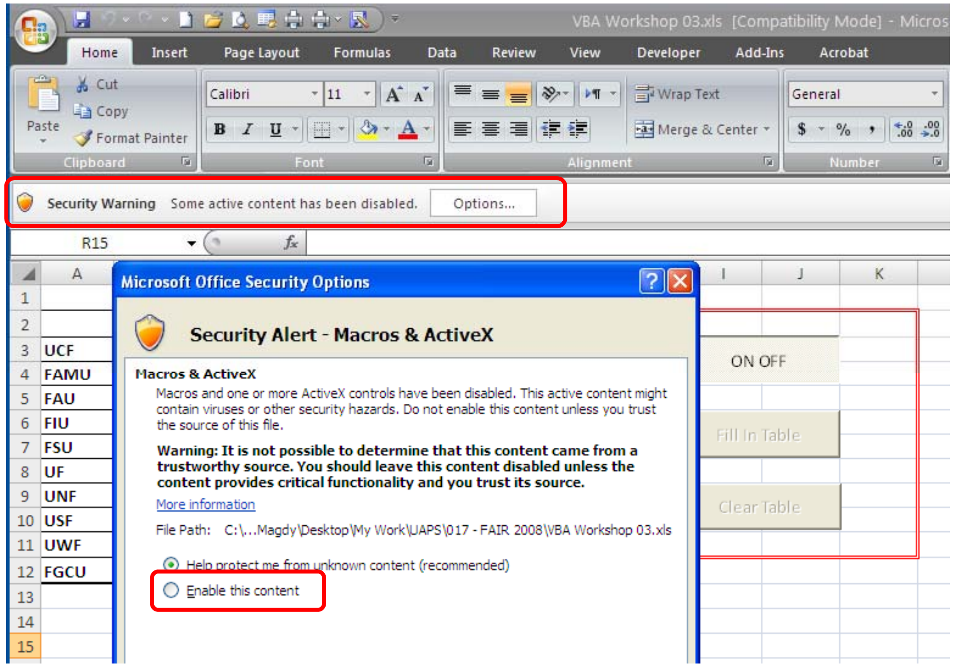
○ Disable all macros without notification
● Disable all macros with notification
○ Disable all macros except digitally signed macros
○ Enable all macros (not recommended; potentially dangerous code can run)

Developer Macro Settings

☑ Trust access to the VBA project object model

Security Settings in 2007

- When you open an Excel file with a macro a security warning appears below the ribbon with the **Options…** button
- Click **Options…** to show the Microsoft Office Security Options dialog box
- Click “Enable This Content” to all the macro



Security Warning Some active content has been disabled. Options...

Microsoft Office Security Options

Security Alert - Macros & ActiveX

Macros & ActiveX

Macros and one or more ActiveX controls have been disabled. This active content might contain viruses or other security hazards. Do not enable this content unless you trust the source of this file.

Warning: It is not possible to determine that this content came from a trustworthy source. You should leave this content disabled unless the content provides critical functionality and you trust its source.

More information

File Path: C:\...Magdy\Desktop\My Work\UAPS\017 - FAIR 2008\VBA Workshop 03.xls

● Help protect me from unknown content (recommended)

○ Enable this content