An Introduction to VBA

Alexander Baranov

ab155@le.ac.uk

- OUTLINE

- What is VBA
- Basic Concepts
- How to Record Macro
- VBA Editor Window
- Example 1

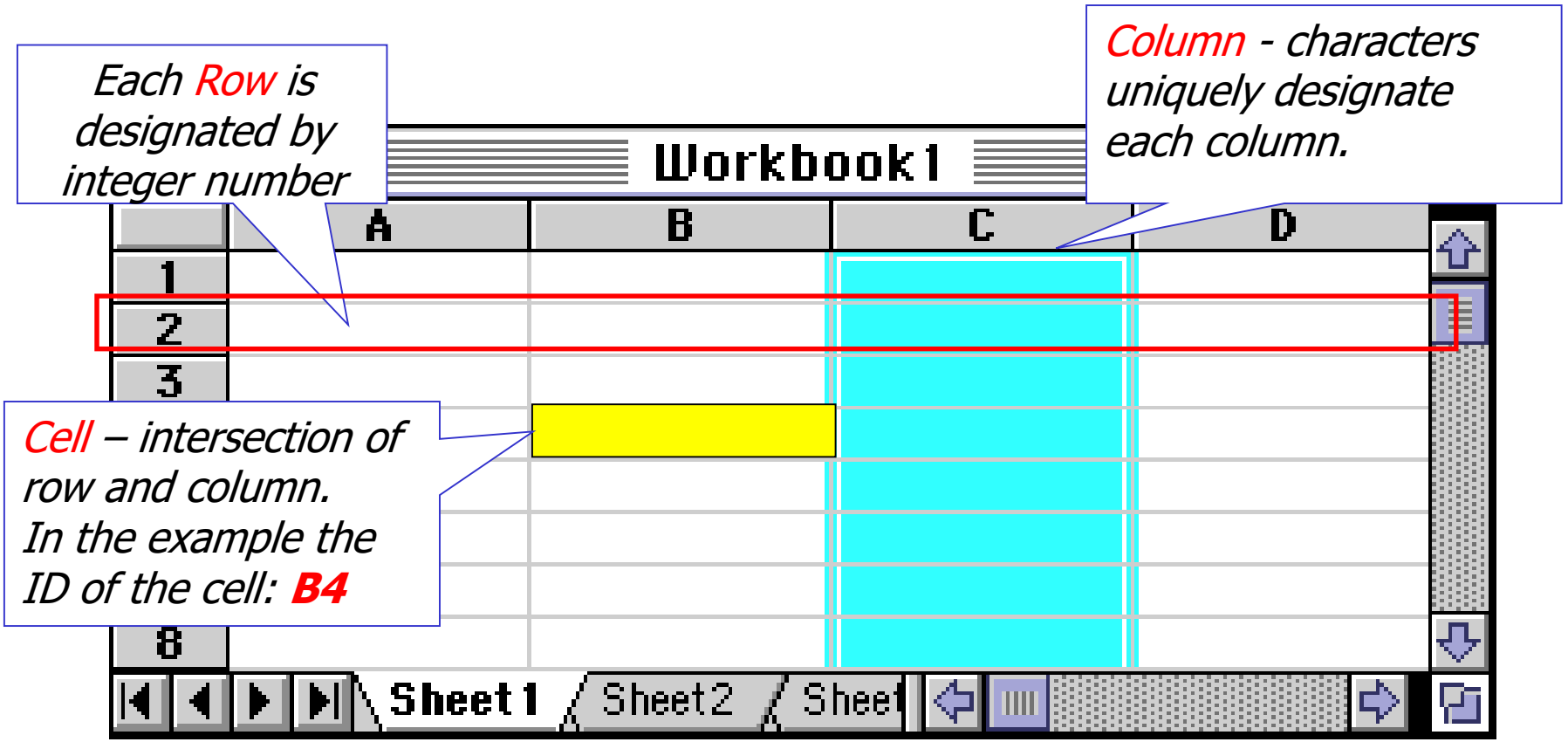
- What is VBA

- An abbreviation for Visual Basic for Applications
- Official name is "Visual Basic, Applications Edition."
- VBA is Microsoft's common application programming (macro) language for Word, Excel, Access, etc.
- Also being implemented in other Microsoft applications
- VBA and VB have a lot in common, but they are different. VB is a programming language that lets you create standalone executable programs.

- What Can You Do With VBA

- Automate Labor-Intensive and Repetitive Tasks
- Create User-Defined Functions to Achieve Complicated Functionality
- Create Standard Windows Menu/Tool Bars for Interface
- Interact with Other Windows Programs (e.g. Internet Explorer, Matlab, etc)
- I/O with External Files
- Access databases and Internet.

- Excel – Basic Elements



- VBA Object Based Programming Language

Basic Excel Objects:

- Workbook: an Excel file (e.g. “Book1.xls”)
- Worksheet: a single worksheet (e.g. “Sheet1”)
- Range: a range (e.g “A3:D10”)
- Cell: a single cell (e.g. “B7”)

- VBA Object Based Programming Language

Containers (or Collections):

- A Group of Similar Objects Share Common Properties
- Such as Workbooks, Worksheets, etc.
- Worksheets is a collection of all the Worksheet objects in the specified or active workbook.
- Worksheets(1) refers to the 1st worksheet of current active workbook.
- Worksheets(“Sheet A”) refers to the worksheet named “Sheet A”.

Referencing Cells

- Cells indexing format:
  - *Cells(row, column)*, where both row and column are given as integers (starting from 1)
- Following expressions are equivalent and refer to the cell B1 in the currently active sheet:
  - ActiveSheet.Cells(1,2)
  - Cells(1,2)
  - Range(“B1”)

- VBA Object Based Programming Language

Referring To:

Use brackets ( ) to refer to member object e.g. Worksheets(“Students”)

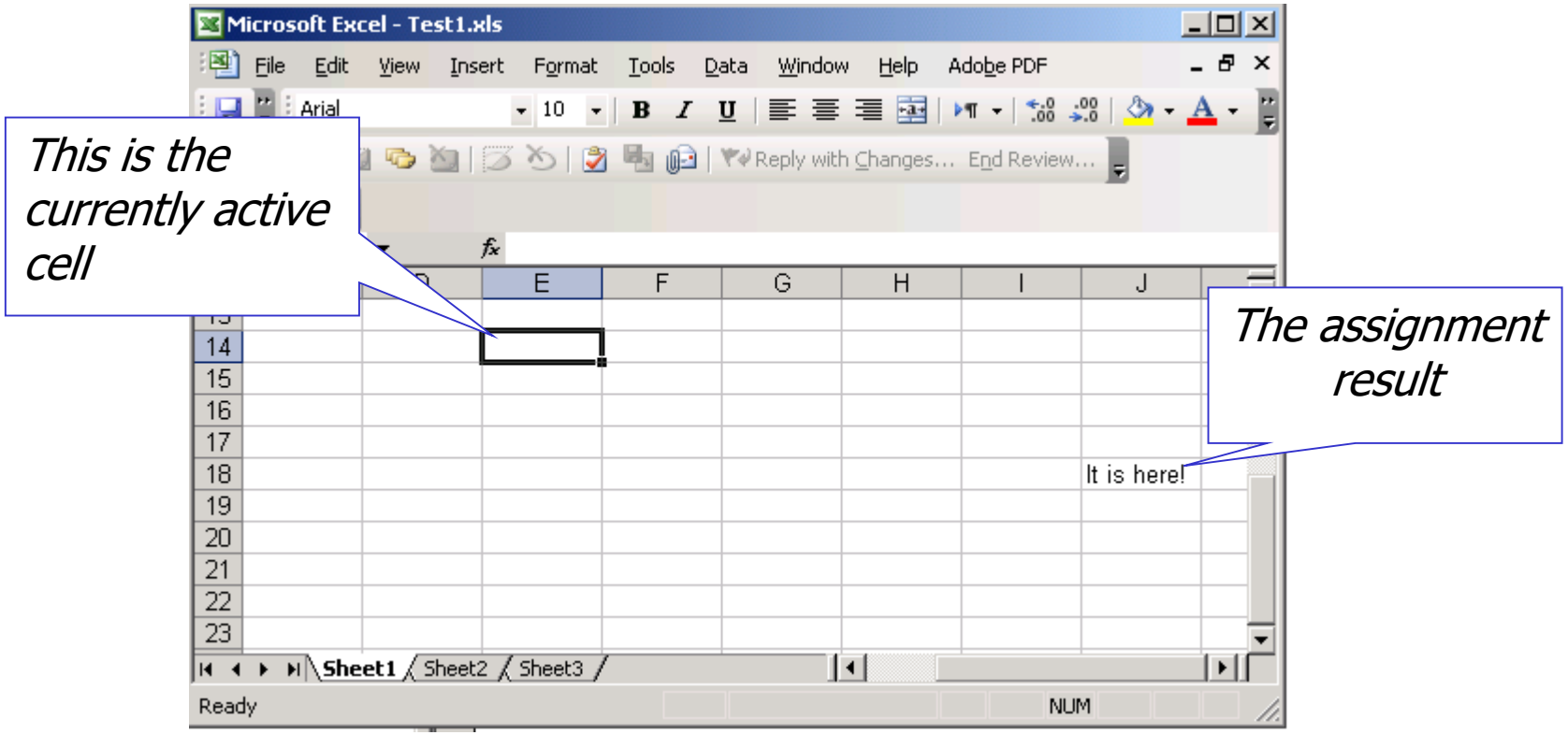
Use dot . to refer to child object or object’s properties and methods:

Workbooks("book1.xls").Worksheets("sheet1").Range(“B1").Font.Bold = True

ActiveWorkbook.ActiveSheet.Cells(1, 2).ClearContents

Referencing Cells with Offset

ActiveCell.Offset(4, 5) = “It is here!”



Variables

- Variable is a “cell” in computer memory
- It has name and value

  - MyNewVar=56
    - x=x+1

Variables

- A named storage location that can contain data that can be modified during program execution
- A variable name
  - Must start with letter and can’t contain spaces and special characters (such as “&”, “%”, “\”)
  - Can’t be any excel keyword (“if”, “while”…)
  - Can’t have identical name to any existing class (“Wroksheet”, “Workbook”…)

The Variables Advantage by example

```
Sub NoVariable()
Range("A1").Value = Range("B2").Value
Range("A2").Value = Range("B2").Value * 2
Range("A3").Value = Range("B2").Value * 4
Range("B2").Value = Range("B2").Value * 5
End Sub
```

```
Sub WithVariable()
myValue = Range("B2").Value
Range("A1").Value = myValue
Range("A2").Value = myValue * 2
Range("A3").Value = myValue * 4
Range("B2").Value = myValue * 5
End Sub
```

VBA Operators

- Arithmetic: *+, -, *, /, ^*
- String Concatenation: *& (the same as +)*

  "Hello " & "World" gives *"Hello World"*
- Comparison: *=, <, >, <> (not equal), <=, >=*
- Logical: *And, Or, Not*

VBA Control Structures - If

- Short form (1 line):

```
If Age >= 18 Then Status = "Adult"
```

- Full form:

```
If Age >=18 Then
                Status = “Adult”
                Vote = “Yes”
Else
                Status = “Child”
                Vote = “No”
End If
```

VBA Control Structures – Loops

```
For i = 1 to 10
    Cells(i, 1) = i
Next i
```

```
i = 1
Do While i <= 10
    Cells(i, 1) = i
    i = i + 1
Loop
```

```
For i = 10 to 1 Step -1
    Cells(i, 1) = i
Next i
```

```
i = 1
Do
    Cells(i, 1) = i
    i  = i + 1
Loop While i < 11
```

Test yourself!
What does the procedure do?

```
Sub MTable()
    For i = 1 To 9
        For j = 1 To 9
            Cells(i, j) = i*j
        Next j
    Next i
End Sub
```

- First Step to VBA : Macros

Record Macro

- Similar to audio/video recorder
- Record all the steps you conduct and write them in VBA code
- Microsoft Excel 2010 don’t provide you the macro recording option by default for enabling macro recording at first you need to enable the Developer Tab.
- next slide defines step by step procedure to enable Developer Tab

- First Step to VBA : Macros

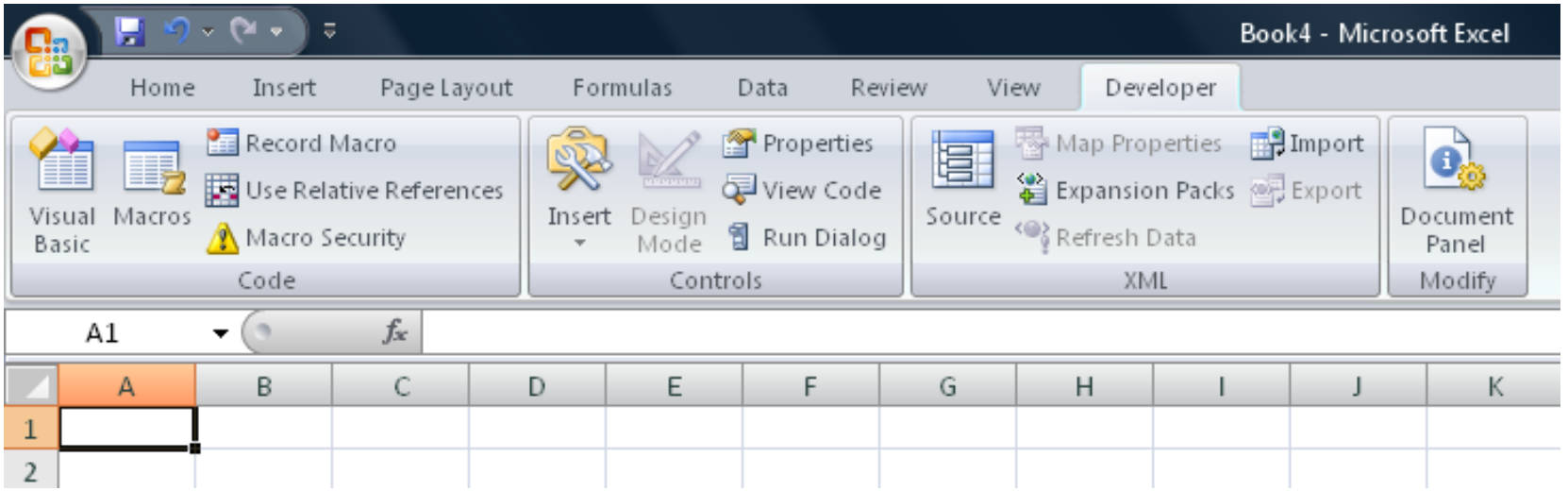
Enabling the Developer Tab in Excel 2010

- Click File, Options and select Customise Ribbon.
- Tick the Developer box (in the right column).
- Click OK
- Developer tab will appear on the Ribbon.

- First Step to VBA : Macros

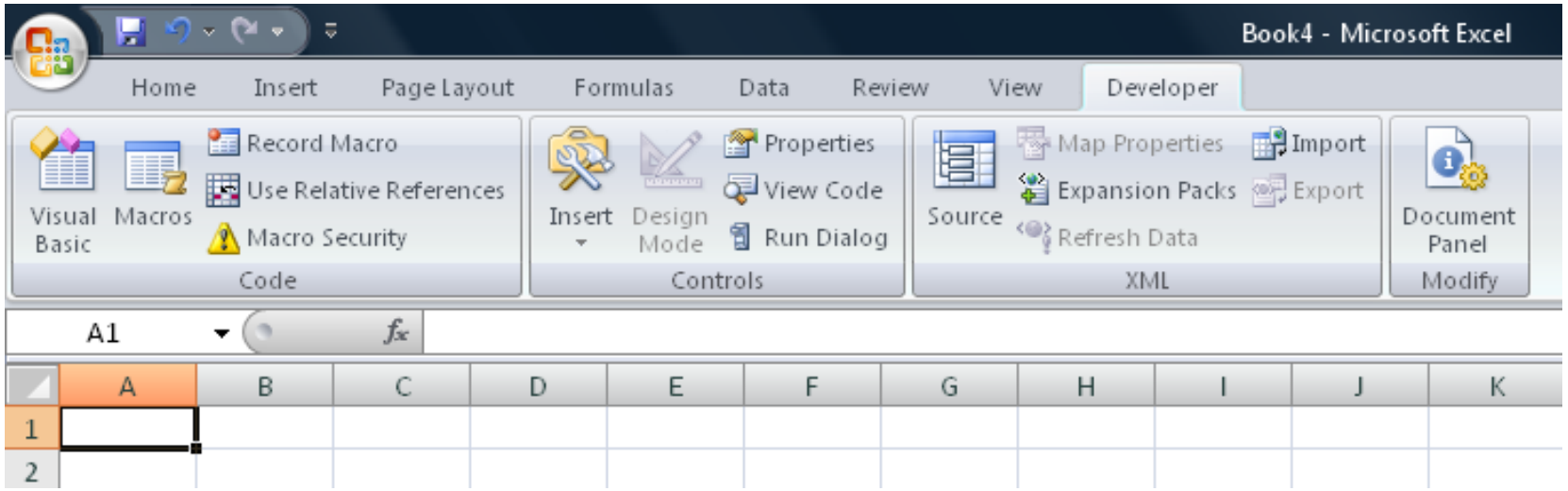
Enabling the Developer Tab

- Once you are done with the process you will have the Developer Tab ready



- First Step to VBA : Macros

Enabling the Macros

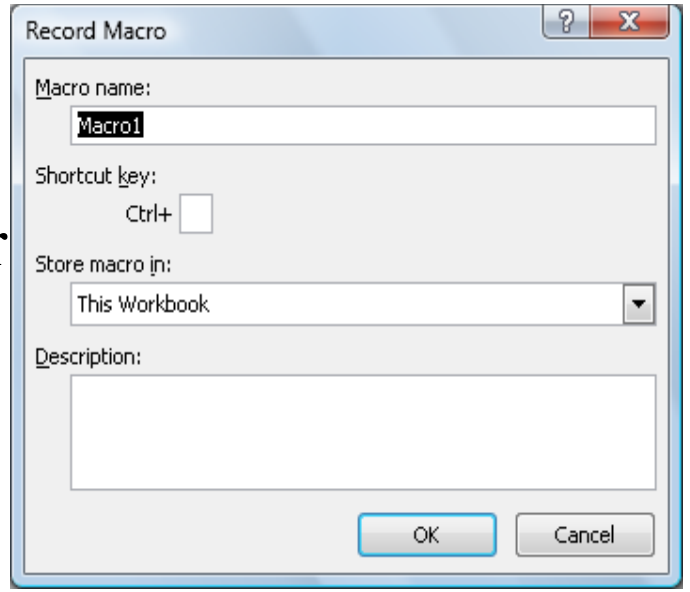


- Now, click on the Developer tab.
- Click on Macro Security.
- Select "Enable all macros" and press ok.
- Close and reopen spreadsheet.

- First Step to VBA : Macros

Record Macro

- Click Developer Tab
- Click on Record Macro
- The following screen will appear

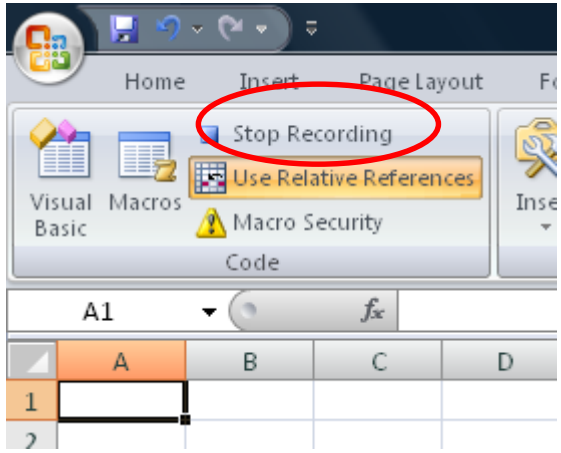


- Type the name of the macro
- Type short cut key which invoke the Macro if necessary
- Store macro (will be discussed later)
- You can also provide description of the macro

- First Step to VBA : Macros

Record Macro

- As soon as you press OK in the previous dialog box two things happens first record macro button changes to Stop Recording



- Secondly Excel starts looking` for your action and will record the same
- Do your actions and press Stop Recording

- First Step to VBA : Macros

Macros can be run from

- Main Menu: Developer – Macros - Run
- VBA Editor Window
- Keyboard shortcut (e.g. Ctrl-m)
- User Can Assign Macro to An Event, Normally a Button Click Event:
  - Easy to Execute and Easy to Remember
  - Give a Good Name for The Button and Click the Button to Run the Macro

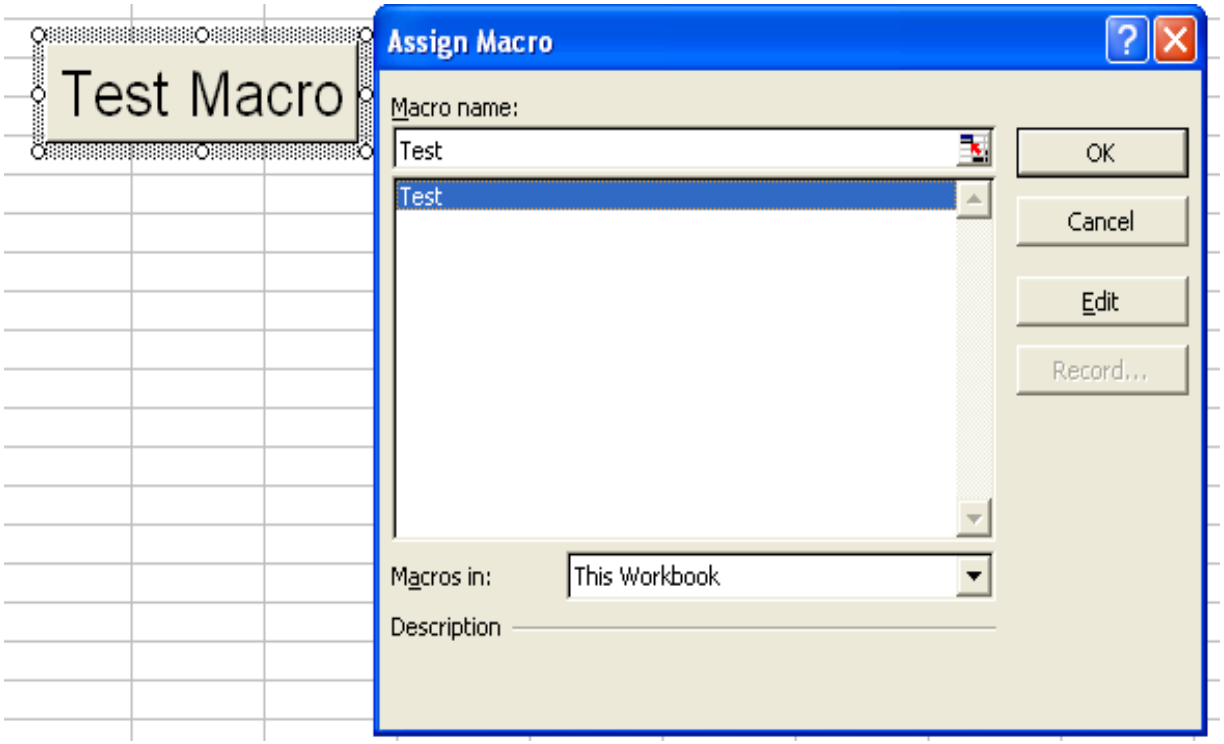
- First Step to VBA : Macros

Assign Macro to An Event

- Add button: Developer – Insert – click rectangular button and click anywhere on the sheet you want to place the button.
- A dialog will appear. Choose the macro you wish to assign to the button (e.g. Test)
- Rename the button, e.g. “Test Macro”

- First Step to VBA : Macros

Assign Macro to An Event



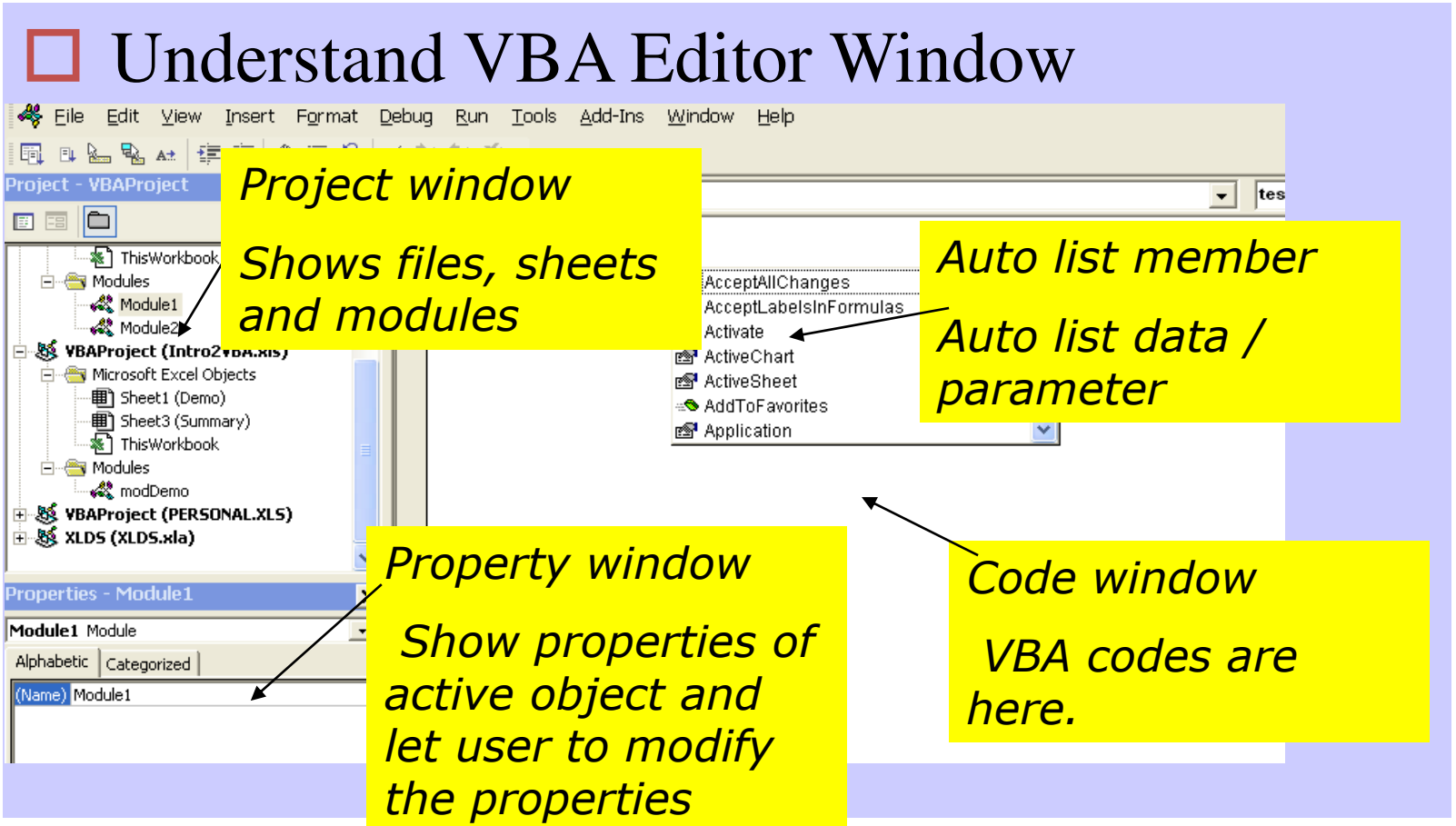
- Second Step to VBA : Edit Macro

VBA Editor Window

- Press Alt+F11 to Activate VBA Window
- Or Go to Developer > Visual Basic > Modules > Module1 (double click to expand)
- Project Explore Window
- Properties Window
- Code Window
- Tools>Options>Docking Window

- Second Step to VBA : Edit Macro

Understand VBA Editor Window



- Second Step to VBA : Edit Macro

VBA Editor Window

- Use Tools/Options to Enable Auto Syntax Check, Auto List Members, etc.
- Use Tools/Properties to Protect Your Code with Password – You Must Remember the Password
- Insert Module, Procedure, Form

- Second Step to VBA : Edit Macro

Example

- Record Macro
- Understand Macro
- Assign Macro to Button Click Event
- Modify Macro