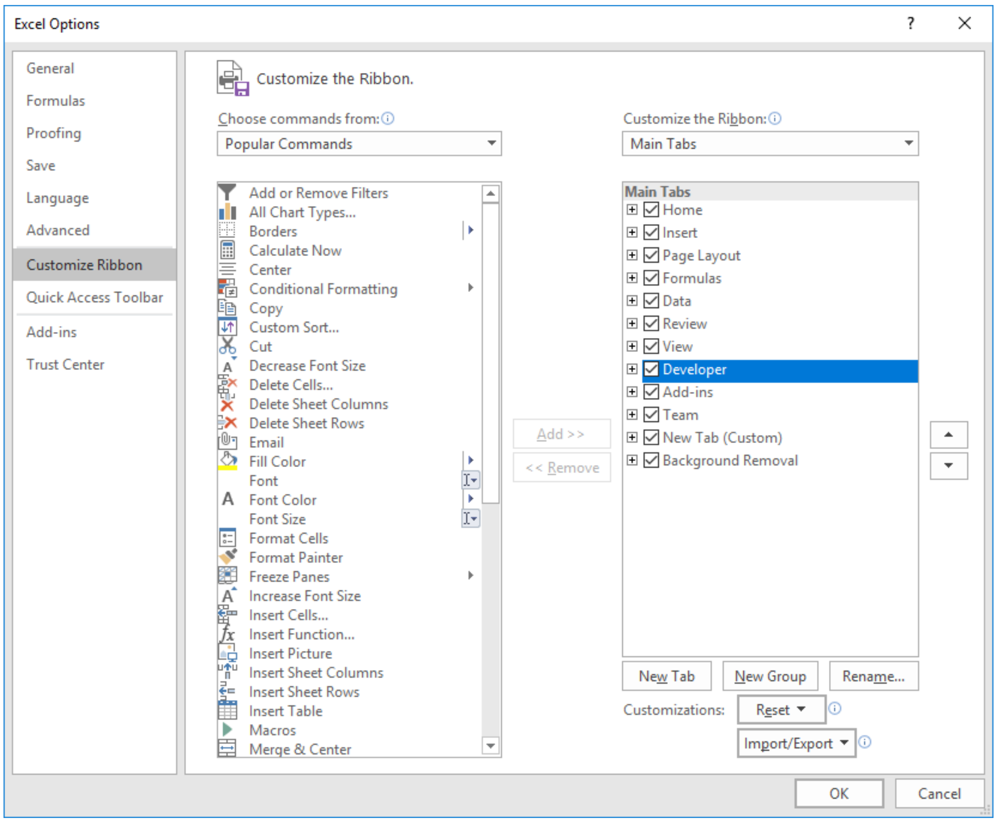
1. Macro

1.1 Overview

If you perform a task repeatedly in Microsoft Excel, you can automate the task with a macro. A macro is a series of commands and functions that are stored in a Microsoft Visual Basic module and can be run whenever you need to perform the task. For example, if you often enter long text strings in cells, you can create a macro to format those cells so that the text wraps.

1.2 Enable Developer Tab in Ribbon

1. Click the **Microsoft Office Button**, and then click **Excel Options**.
2. Click **Popular**, and then select the **Show Developer** tab in the **Ribbon** check box, and press **[OK]** to confirm.



Excel Options

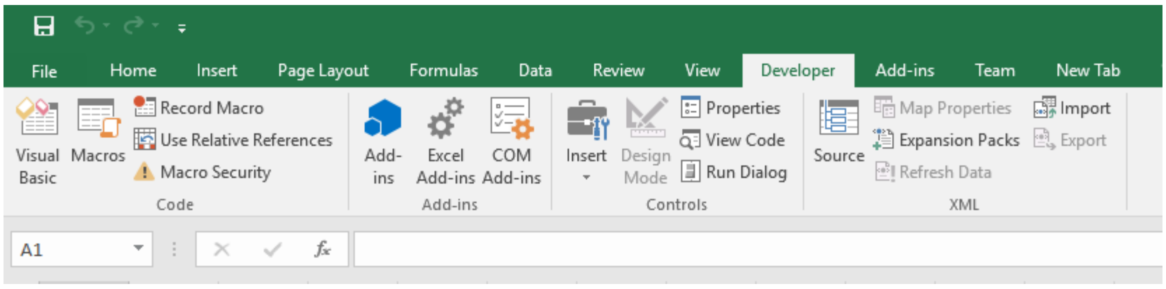
Customize the Ribbon.

Choose commands from: Popular Commands

Customize the Ribbon: Main Tabs

Main Tabs: ☑ Home, ☑ Insert, ☑ Page Layout, ☑ Formulas, ☑ Data, ☑ Review, ☑ View, ☑ Developer, ☑ Add-ins, ☑ Team, ☑ New Tab (Custom), ☑ Background Removal

3. The Developer tab will be enabled.

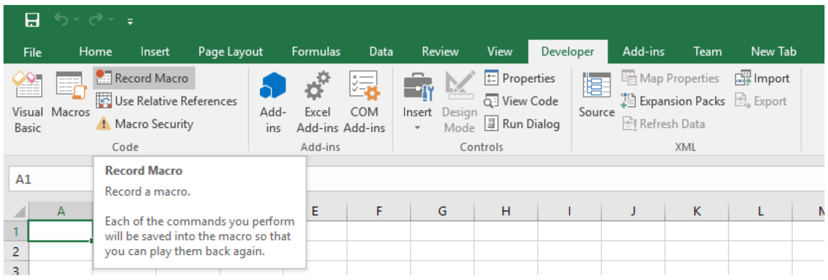


1.3 Recording Macro

1.3.1 Record Macro using Macro Recorder

When you record a macro, Excel stores information about each step you take as you perform a series of commands. You then run the macro to repeat, or play back, the commands. If you make a mistake when you record the macro, corrections you make are also recorded. Visual Basic stores each macro in a new module attached to a workbook.

1. Select **Developer** tab, **Code** Group, **Record Macros**

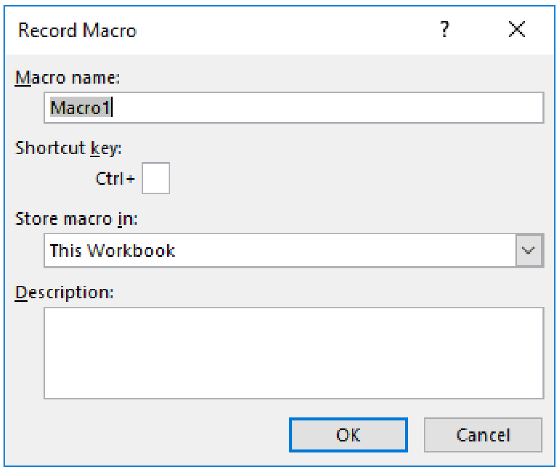


Record Macro

Record a macro.

Each of the commands you perform will be saved into the macro so that you can play them back again.

2. Perform the following action in the **Record Macro** dialog box.



Record Macro

Macro name: Macro1

Shortcut key: Ctrl+

Store macro in: This Workbook

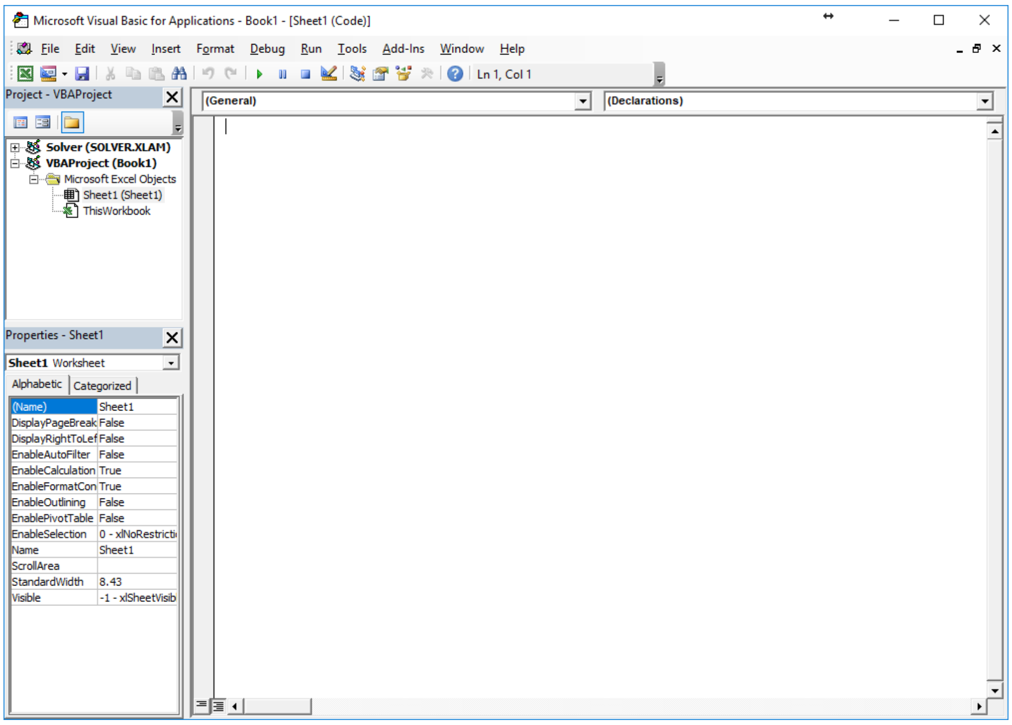
Description:

- In the **Record Macro** dialog box, enter a name in the **Macro Name** box
- If you want to run the macro by pressing a keyboard shortcut key, enter a letter in the **Shortcut** key box. The shortcut key will override any equivalent default Excel shortcut keys while the workbook that contains the macro is open
- In the **Store macro in** box, click the location where you want to store the macro. If you want a macro to be available whenever you use Excel, select **Personal Macro Workbook**.
- If you want to include a description of the macro, type it in the **Description** box.
- Click **[OK]**. If you want the macro to run relative to the position of the active cell, record it using relative cell references. On the **Stop Recording** toolbar, click **Relative Reference** so that it is selected. Excel will continue to record macros with relative references until you quit Excel or until you click **Relative Reference** again, so that it is not selected.

3. Carry out the actions you want to record.
4. Select **View** tab, **Code** Group, **Stop Macro** when finish the record.

1.3.2 Create Macro using Visual Basic Editor

1. Select **Developer** tab, **Code** Group, **Visual Basic**
2. Select the **Insert ➔ Module** in the **Microsoft Visual Basic Editor**.
3. Type or copy your code into the code window of the module.
4. If you want to run the macro from the module window, press **[F5]**.
5. When you're finished editing, click **File ➔ Close and Return to Microsoft Excel**.



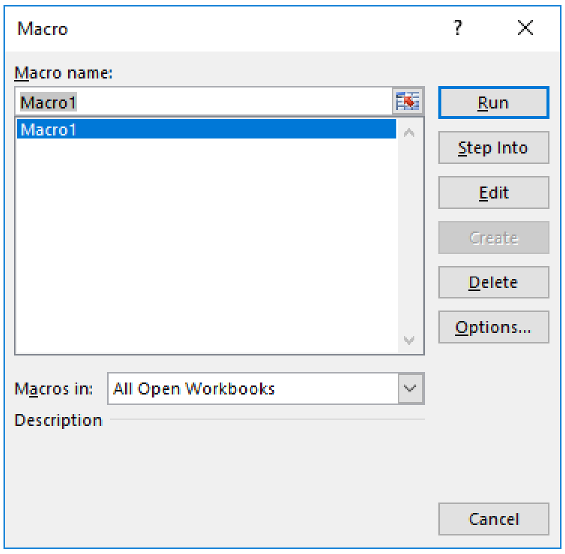
(Name)	Sheet1
DisplayPageBreak	False
DisplayRightToLef	False
EnableAutoFilter	False
EnableCalculation	True
EnableFormatCon	True
EnableOutlining	False
EnablePivotTable	False
EnableSelection	0 - xlNoRestricti
Name	Sheet1
ScrollArea	
StandardWidth	8.43
Visible	-1 - xlSheetVisib

1.4 Executing Macro

The next time you need to flag a cell, you can run the macro. If you're going to use the macro frequently, you can create a toolbar button for it, or assign a keystroke for it, or both.

1.4.1 Running Macro using Tools Menu

1. Select **View** tab, **Code** Group, Macros to call the **Macro** dialog box.
2. Click the name of your macro, and then click **Run**.



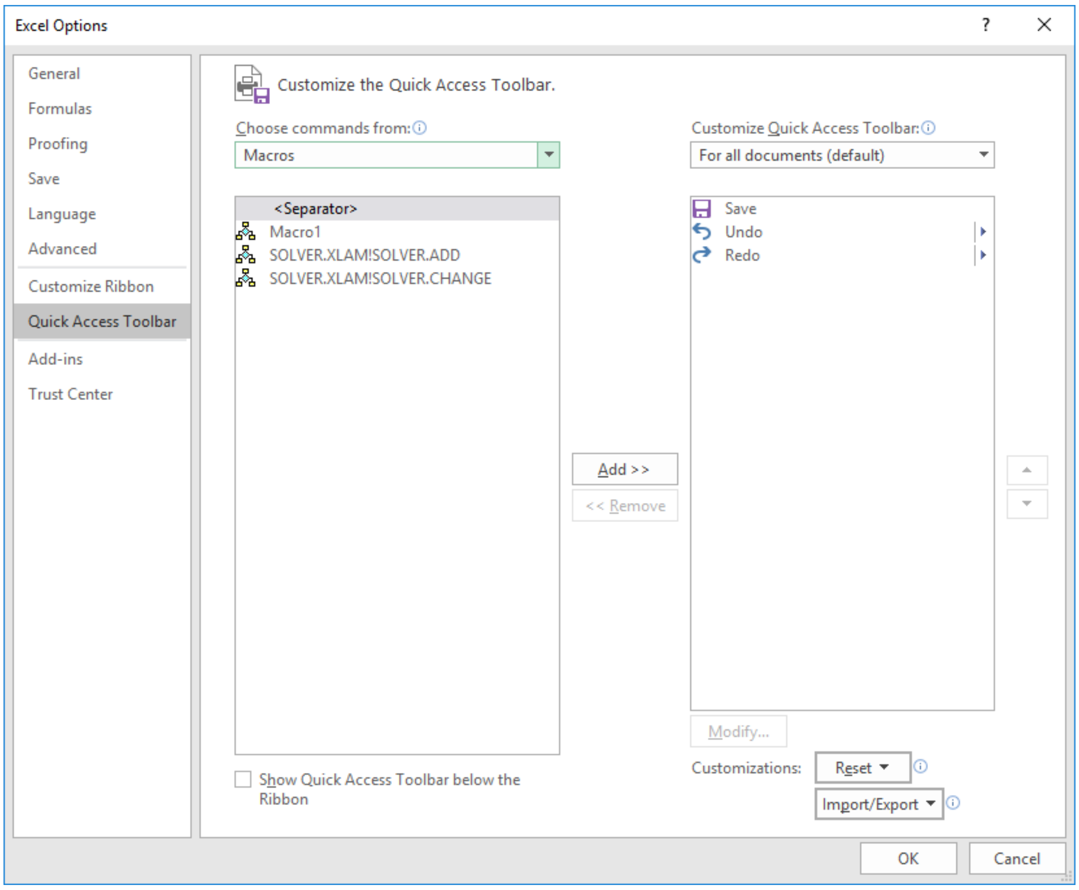
Macro name: Macro1

Macro1

Macros in: All Open Workbooks

1.4.2 Create Toolbar Button for Runing Macro

1. Click the Microsoft Office Button, and then click **Excel Options**, and in the **Quick Access Toolbar** tab.
2. Select **Macros** in the **Choose command from**
3. Select the macro you want to assign and press the **Add** button, and then click **[OK]**.



1.4.3 Assign Keystroke for Running Macro

1. Click the worksheet, and then select **Developer** tab, **Code** Group, **Macro**.
2. Select the name of your macro, and then click **[Options]**.
3. In the Shortcut key box, type the key to use along with **[Ctrl]** button to run your macro.

1.5 Managing Macros

After you record a macro, you can view the macro code with the Visual Basic Editor to correct errors or change what the macro does. For example, if you wanted the text-wrapping macro to also make the text bold, you could record another macro to make a cell bold and then copy the instructions from that macro to the text-wrapping macro.

The Visual Basic Editor is a program designed to make writing and editing macro code easy for beginners and provides plenty of online Help. You don't have to learn how to program or use the Visual Basic language to make simple changes to your macros. With the Visual Basic Editor, you can edit macros, copy macros from one module to another, copy macros between different workbooks, and rename the modules that store the macros, or rename the macros.

1.6 Macro Security

In Microsoft Office Excel, you can change the macro security settings to control which macros run and under what circumstances when you open a workbook. For example, you might allow macros to run based on whether they are digitally signed by a trusted developer.

1.6.1 Macro Security Settings and their Effects

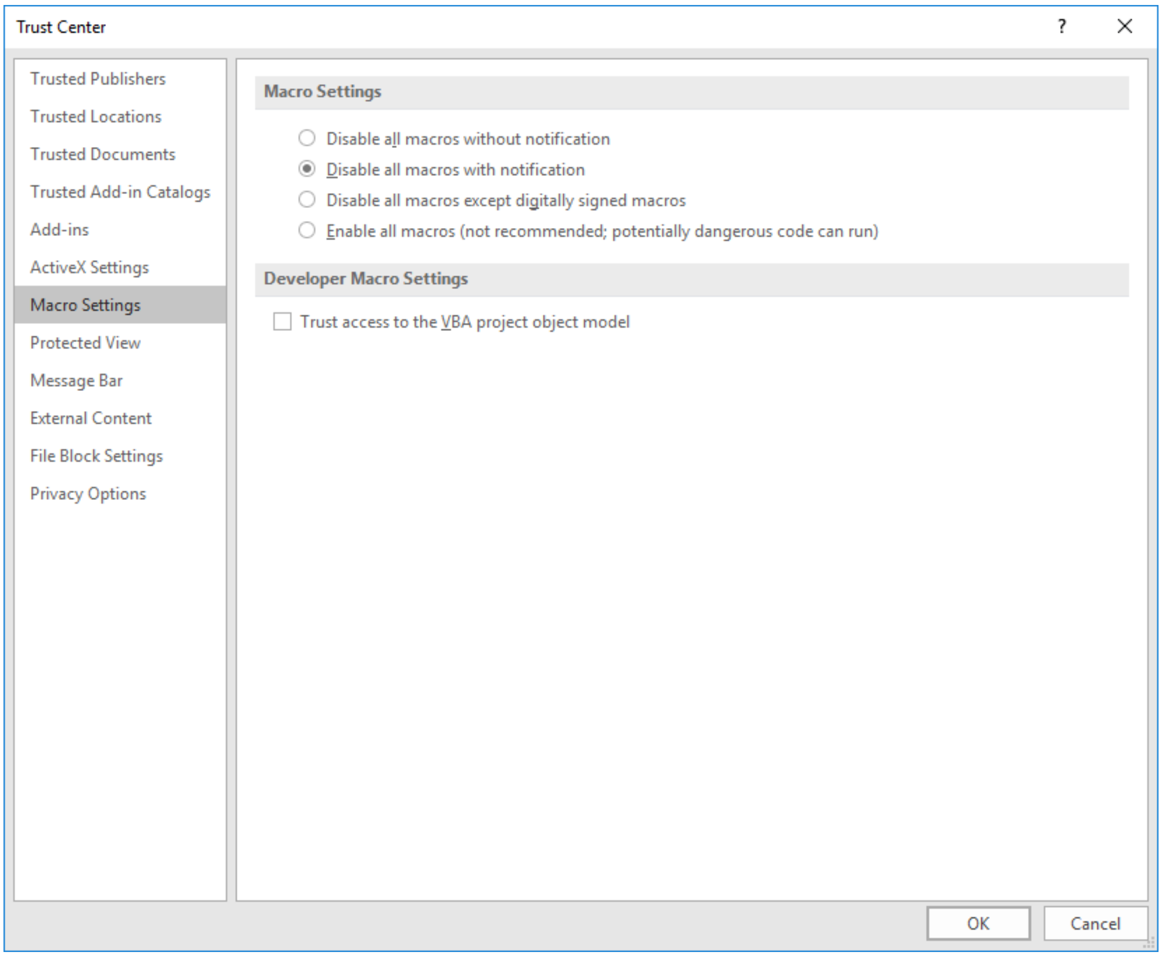
The following list summarizes the various macro security settings. Under all settings, if antivirus software that works with 2007 Microsoft Office system is installed and the workbook contains macros, the workbook is scanned for known viruses before it is opened.

- **Disable all macros without notification**. Click this option if you don't trust macros. All macros in documents and security alerts about macros are disabled. If there are documents that contain unsigned macros that you do trust, you can put those documents into a trusted location. Documents in trusted locations are allowed to run without being checked by the Trust Center security system.
- **Disable all macros with notification.** This is the default setting. Click this option if you want macros to be disabled, but you want to get security alerts if there are macros present. This way, you can choose when to enable those macros on a case by case basis.
- **Disable all macros except digitally signed macros.** This setting is the same as the Disable all macros with notification option, except that if the macro is digitally signed by a trusted publisher, the macro can run if you have already trusted the publisher. If you have not trusted the publisher, you are notified. That way, you can choose to enable those signed macros or trust the publisher. All unsigned macros are disabled without notification.
- **Enable all macros (not recommended, potentially dangerous code can run).** Click this option to allow all macros to run. Using this setting makes your computer vulnerable to potentially malicious code and is not recommended.
- **Trust access to the VBA project object model.** This setting is for developers and is used to deliberately lock out or allow programmatic access to the VBA object model from any Automation client. In other words, it provides a security option for code that is written to automate an Office program and programmatically manipulate the VBA environment and object model. This is a per user and per application setting, and denies access by default. This security option makes it more difficult for unauthorized programs to build "self-replicating" code that can harm end-user systems. For any Automation client to be able to access the VBA object model programmatically, the user running the code must explicitly grant access. To turn on access, select the check box.

1.6.2 Change Macro Security Settings

You can change macro security settings in the Trust Center, unless a system administrator in your organization has changed the default settings to prevent you from changing the settings.

1. On the **Developer** tab, in the **Code** group, click **Macro Security**.
2. In the **Macro Settings** category, under **Macro Settings**, click the option that you want. Any changes that you make in the Macro Settings category in Excel apply only to Excel and do not affect any other Microsoft Office program.



Macro Settings

- ○ 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

2. Visual Basic Editor

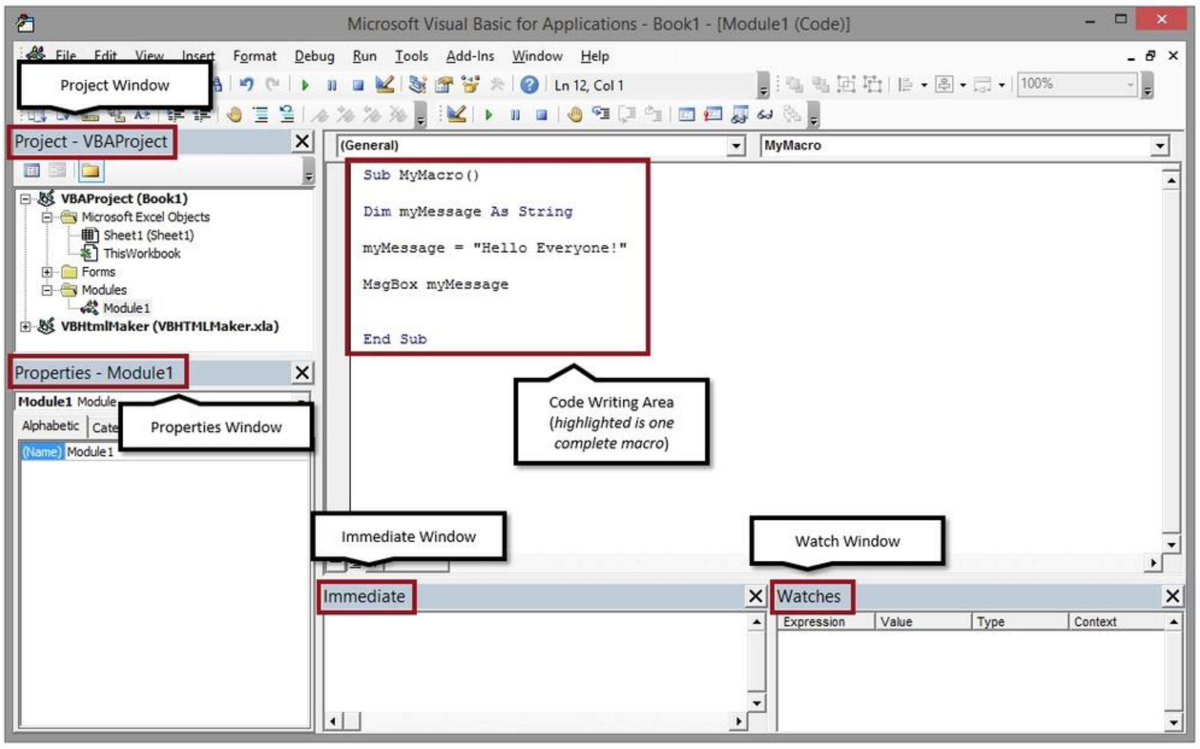
2.1 Introduction to Visual Basic Editor

The Visual Basis Editor is the workspace for creating VBA code. The editor can be accessed through your Developer Tab or by using the shortcut **[Alt]** + **[F11]**. The editor will display in a completely separate window than your Office Application and each one of the programs in the Office Suite has its own VBA Editor

There are several main areas in the VBA Editor:

- Project Explorer
- Code Window
- Immediate Window
- Watch Window
- Properties Window

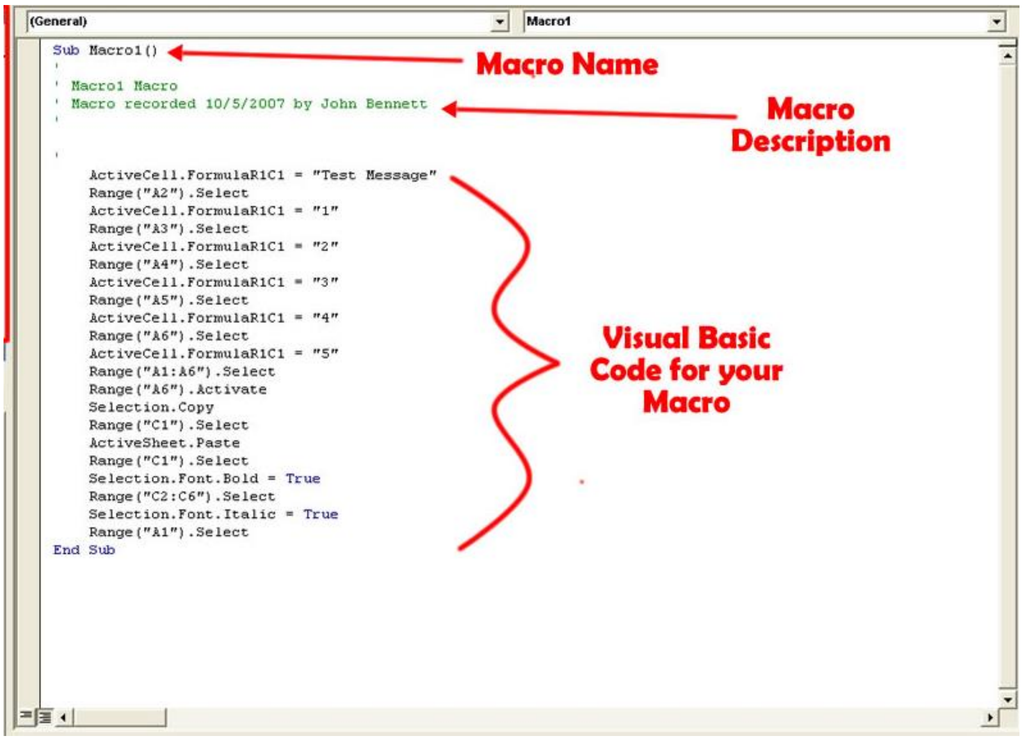
This is what is known as an Integrated Development Environment (which means everything you need to write programs and code are all in this one window)



```
Sub MyMacro()

Dim myMessage As String

myMessage = "Hello Everyone!"

MsgBox myMessage


End Sub
```

2.2 Using the Visual Basic Editor

The Visual Basic Editor is a powerful tool that lets you extend the power and versatility of macros beyond anything that can be done through recording alone.



```
Sub Macro1()
'
' Macro1 Macro
' Macro recorded 10/5/2007 by John Bennett
'

'
    ActiveCell.FormulaR1C1 = "Test Message"
    Range("A2").Select
    ActiveCell.FormulaR1C1 = "1"
    Range("A3").Select
    ActiveCell.FormulaR1C1 = "2"
    Range("A4").Select
    ActiveCell.FormulaR1C1 = "3"
    Range("A5").Select
    ActiveCell.FormulaR1C1 = "4"
    Range("A6").Select
    ActiveCell.FormulaR1C1 = "5"
    Range("A1:A6").Select
    Range("A6").Activate
    Selection.Copy
    Range("C1").Select
    ActiveSheet.Paste
    Range("C1").Select
    Selection.Font.Bold = True
    Range("C2:C6").Select
    Selection.Font.Italic = True
    Range("A1").Select
End Sub
```

2.2.1 Name of Macro

The basic macro is made up of three parts, its name, description and code. Every macro must have the keyword "**Sub**" before whatever you decide to name your macro and open and close parenthesis after. If you wanted to name your macro **Hello_World**, then the first line on your macro will be **Sub Hello_World()**.

2.2.2 Macro Description

The description is in green because it is a comment section. Any part of a macro that is commented out will be ignored when the macro is actually run, so you can type anything you want there.

To comment out a section, all you need to do is add a single quote before the line you want to comment ‘. You are then able to comment on anything you would like. Comments can be put anywhere within the macro code.

2.2.3 Code for Macro

The actual code is what the macro will be doing when you tell it to run. Anything you type in the section will be attempted to be run by the macro, so if you type something incorrectly that the macro doesn't understand, it will give you an error to debug. Over the next few posts we will be looking at some commands we can give the visual basic editor to make it do what we want.

2.2.4 End of Macro

After the code is complete you have to end the macro with the line **End Sub**. This just lets the macro know that it has reached the end of the executable code and its job is complete.

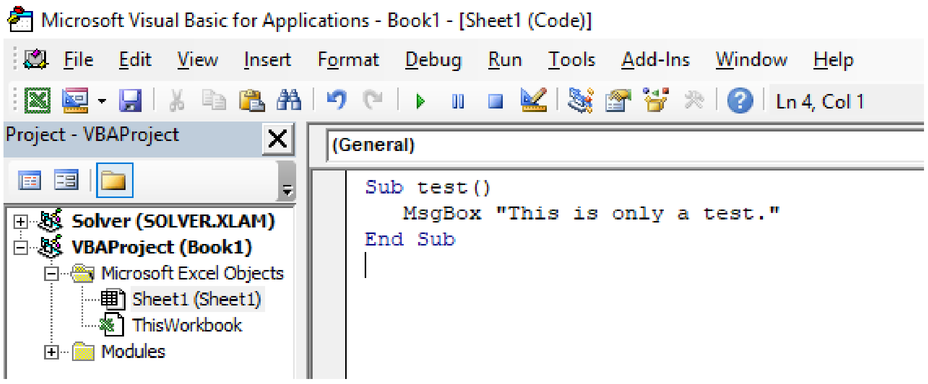
2.2.5 Example

Below is the example for creating an sample Macro program using VBA:

1. Start Excel and open a new, blank workbook.
2. Select **Developer** tab, **Code** Group, **Visual Basic**.
3. In the **Project** window, double-click **ThisWorkbook**.
4. Enter this code into the code window:

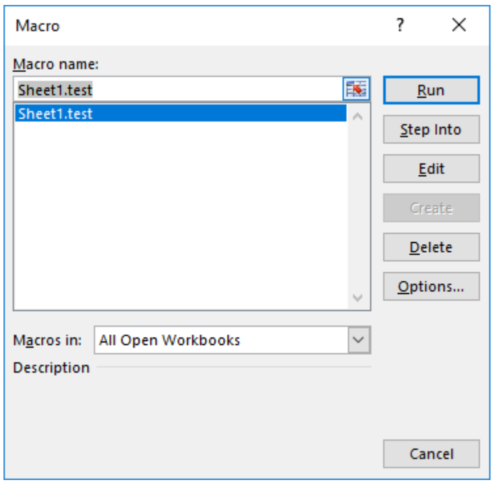
```
sub test( )
    MsgBox "This is only a test."
end sub
```

5. Save the file and then select **File ➔ Visual Basic Editor and close the workbook**.

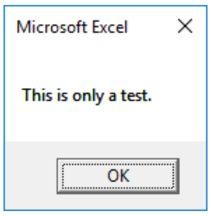


```
Sub test()
    MsgBox "This is only a test."
End Sub
```

6. Select **Developer** tab, **Code** Group, **Macros** in the Excel worksheet. Then select the previous macro "text" we created in the Visual Basic Editor.



7. Press **[Run]** to execute the macro and a message box will be displayed.

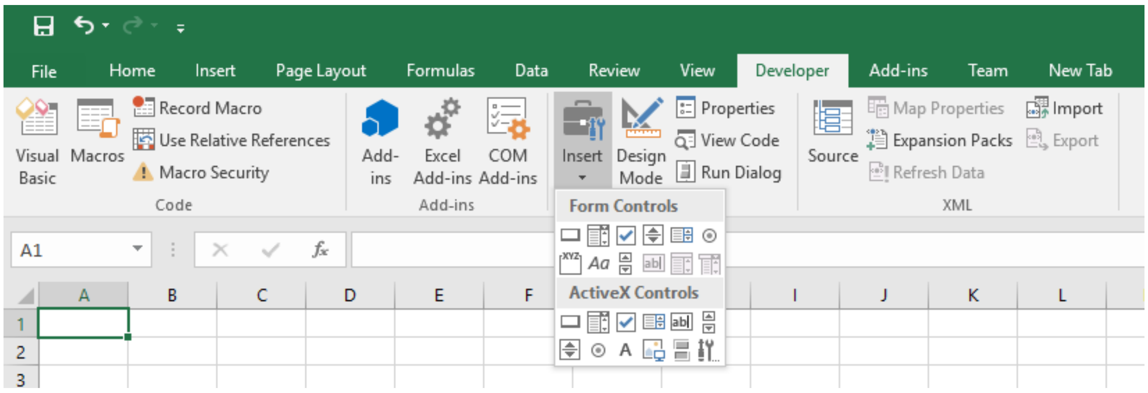


This is only a test.

3. Working with Excel VBA Control

3.1 Overview

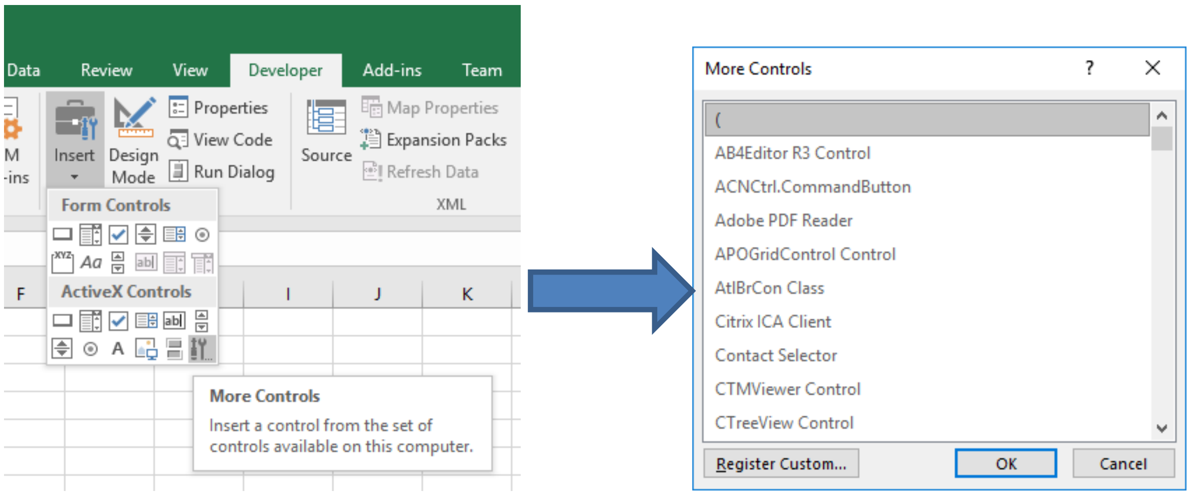
The main objects used to help a person interact with the computer are Windows controls. There are two main ways you can access these objects when working with Microsoft Excel. If you are working in Microsoft Excel, you can add or position some Windows controls on the document. To do that, on the Ribbon, click Developer. In the Control section, click Insert.



This would display the list of controls available in Microsoft Excel. The controls appear in two sections: Form Controls and ActiveX Controls. If you are working on a spreadsheet in Microsoft Excel, you should use only the controls in the ActiveX Controls section. If you are working on a form in Microsoft Visual Basic, a Toolbox equipped with various controls will appear.

3.2 Using Additional Objects

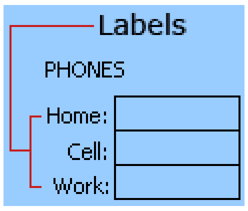
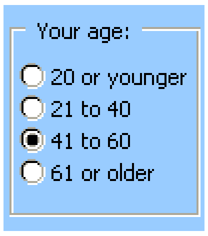
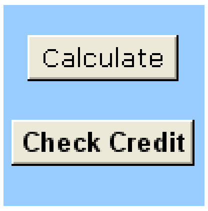
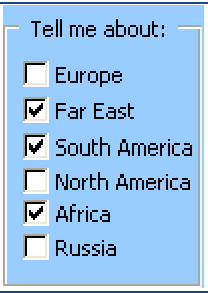
The Developer tab of the Ribbon in Microsoft Excel provides the most regularly used controls. These controls are enough for any normal spreadsheet you are developing. Besides these objects, other controls are left out but are still available. To use one or more of these left out controls, in the Controls section of the Ribbon, click Insert and click the More Controls button. This would open the More Controls dialog box. You can scroll up and down in the window to locate the desired control. If you see a control you want to use, click it and click [OK].

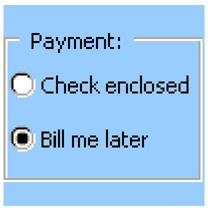
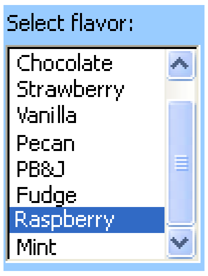
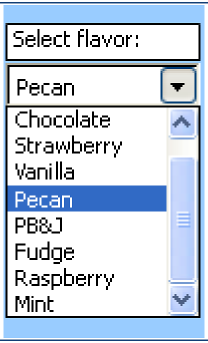
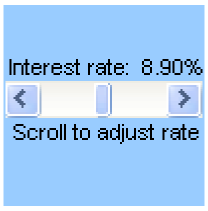
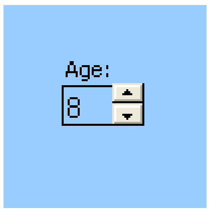


3.3 Form Controls

Form controls are the original controls that are compatible with earlier versions of Excel, starting with Excel version 5.0. Form controls are also designed for use on XLM macro sheets. You use Form controls when you want to easily reference and interact with cell data without using VBA code, and when you want to add controls to chart sheets. For example, after you add a list box control to a worksheet and linking it to a cell, you can return a numeric value for the current position of the selected item in the control. You can then use that numeric value in conjunction with the INDEX function to select different items from the list.

You can also run macros by using Form controls. You can attach an existing macro to a control, or write or record a new macro. When a user of the form clicks the control, the control runs the macro. However, these controls cannot be added to UserForms, used to control events, or modified to run Web scripts on Web pages.

Icon	Name	Example	Description
Aa	Label		Identifies the purpose of a cell or text box, or displays descriptive text (such as titles, captions, pictures) or brief instructions.
▭	Group box	 Your age: 20 or younger 21 to 40 41 to 60 61 or older	Groups related controls into one visual unit in a rectangle with an optional label. Typically, option buttons, check boxes, or closely related contents are grouped.
▭	Button		Runs a macro that performs an action when a user clicks it. A button is also referred to as a push button.
☑	Check Box	 Tell me about: Europe Far East South America North America Africa Russia	Turns on or off a value that indicates an opposite and unambiguous choice. You can select more than one check box on a worksheet or in a group box. A check box can have one of three states: selected (turned on), cleared (turned off), and mixed, meaning a combination of on and off states (as in a multiple selection).

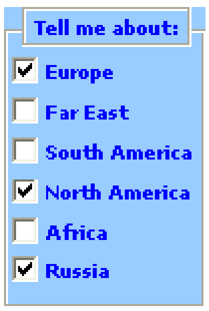
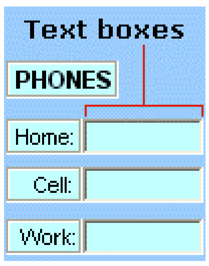
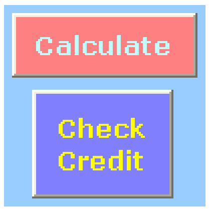
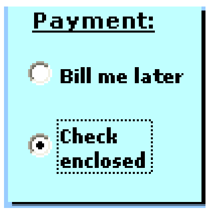
◉	Option Button		Allows a single choice within a limited set of mutually exclusive choices; an option button is usually contained in a group box or a frame. An option button can have one of three states: selected (turned on), cleared (turned off), and mixed, meaning a combination of on and off states (as in a multiple selection). An option button is also referred to as a radio button.
▤	List Box	 Select flavor: Chocolate Strawberry Vanilla Pecan PB&J Fudge Raspberry Mint	Displays a list of one or more items of text from which a user can choose. Use a list box for displaying large numbers of choices that vary in number or content. There are three types of list boxes: A single-selection list box enables only one choice. In this case, a list box resembles a group of option buttons, except that a list box can handle a large number of items more efficiently. A multiple-selection list box enables either one choice or contiguous (adjacent) choices. An extended-selection list box enables one choice, contiguous choices, and noncontiguous (or disjointed) choices.
▤	Combo Box	 Select flavor: Pecan Chocolate Strawberry Vanilla Pecan PB&J Fudge Raspberry Mint	Combines a text box with a list box to create a drop-down list box. A combo box is more compact than a list box but requires the user to click the down arrow to display the list of items. Use a combo box to enable a user to either type an entry or choose only one item from the list. The control displays the current value in the text box, regardless of how that value is entered.
⇳	Scroll Bar		Scrolls through a range of values when you click the scroll arrows or drag the scroll box. In addition, you can move through a page (a preset interval) of values by clicking the area between the scroll box and either of the scroll arrows. Typically, a user can also type a text value directly into an associated cell or text box.
⬍	Spin Button		Increases or decreases a value, such as a number increment, time, or date. To increase the value, click the up arrow; to decrease the value, click the down arrow. Typically, a user can also type a text value directly into an associated cell or text box.

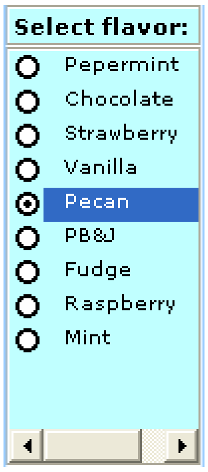
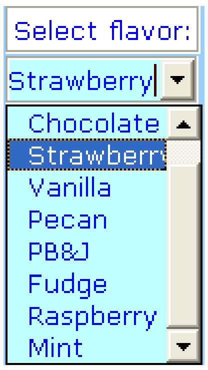
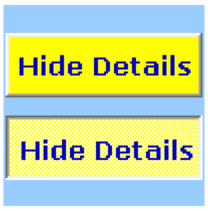
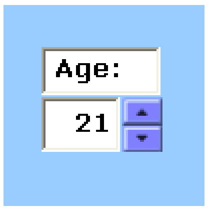
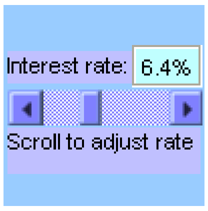
3.4 ActiveX controls

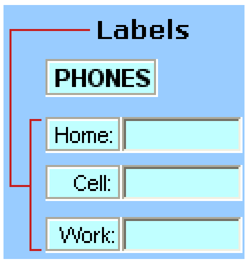
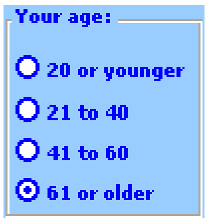
ActiveX controls can be used on worksheet forms, with or without the use of VBA code, and on VBA UserForms. In general, use ActiveX controls when you need more flexible design requirements than those provided by Form controls. ActiveX controls have extensive properties that you can use to customize their appearance, behavior, fonts, and other characteristics.

You can also control different events that occur when an ActiveX control is interacted with. For example, you can perform different actions, depending on which choice a user selects from a list box control, or you can query a database to refill a combo box with items when a user clicks a button. You can also write macros that respond to events associated with ActiveX controls. When a user of the form interacts with the control, your VBA code then runs to process any events that occur for that control.

Your computer also contains many ActiveX controls that were installed by Excel and other programs, such as Calendar Control 12.0 and Windows Media Player.

Icon	Name	Example	Description
☑	Check Box	 Tell me about: Europe Far East South America North America Africa Russia	Turns on or off a value that indicates an opposite and unambiguous choice. You can select more than one check box at a time on a worksheet or in a group box. A check box can have one of three states: selected (turned on), cleared (turned off), and mixed, meaning a combination of on and off states (as in a multiple selection).
abl	Text Box		Enables you to, in a rectangular box, view, type, or edit text or data that is bound to a cell. A text box can also be a static text field that presents read-only information.
▭	Command Button		Runs a macro that performs an action when a user clicks it. A command button is also referred to as a push button.
◉	Option Button		Allows a single choice within a limited set of mutually exclusive choices usually contained in a group box or frame. An option button can have one of three states: selected (turned on), cleared (turned off), and mixed, meaning a combination of on and off states (as in a multiple selection). An option button is also referred to as a radio button.

	List Box	 Select flavor: Pepermint Chocolate Strawberry Vanilla Pecan PB&J Fudge Raspberry Mint	Displays a list of one or more items of text from which a user can choose. Use a list box for displaying large numbers of choices that vary in number or content. There are three types of list boxes: • A single-selection list box enables only one choice. In this case, a list box resembles a group of option buttons, except that a list box can handle a large number of items more efficiently. • A multiple selection list box enables either one choice or contiguous (adjacent) choices. • An extended-selection list box enables one choice, contiguous choices, and noncontiguous (or disjointed) choices.
	Combo Box	 Select flavor: Strawberry Chocolate Strawberry Vanilla Pecan PB&J Fudge Raspberry Mint	Combines a text box with a list box to create a drop-down list box. A combo box is more compact than a list box, but requires the user to click the down arrow to display the list of items. Use to allow a user to either type an entry or choose only one item from the list. The control displays the current value in the text box, regardless of how that value is entered.
	Toggle button		Indicates a state, such as Yes/No, or a mode, such as On/Off. The button alternates between an enabled and disabled state when it is clicked.
	Spin Button		Increases or decreases a value, such as a number increment, time, or date. To increase the value, click the up arrow; to decrease the value, click the down arrow. Typically, a user can also type a text value into an associated cell or text box.
	Scroll Bar		Scrolls through a range of values when you click the scroll arrows or drag the scroll box. In addition, you can move through a page (a preset interval) of values by clicking the area between the scroll box and either of the scroll arrows. Typically, a user can also type a text value directly into an associated cell or text box.

	Label		Identifies the purpose of a cell or text box, displays descriptive text (such as titles, captions, pictures), or provides brief instructions.
	Image		Embeds a picture, such as a bitmap, JPEG, or GIF.
	Frame Control	 Your age: 20 or younger 21 to 40 41 to 60 61 or older	A rectangular object with an optional label that groups related controls into one visual unit. Typically, option buttons, check boxes, or closely related contents are grouped in a frame control. *Note: The ActiveX frame control is not available in the ActiveX Controls section of the Insert command. However, you can add the control from the More Controls dialog box by selecting Microsoft Forms 2.0 Frame.*
	More Controls		Displays a list of additional ActiveX controls available on your computer that you can add to a custom form, such as Calendar Control 12.0 and Windows Media Player. You can also register a custom control in this dialog box.