ADVANCE EXCEL ASSIGNMENT 16

1. **What is a Macro? How is it useful in excel or in your daily work?**

   A macro is a set of instructions used to execute repetitive tasks. It is an action or a set of actions that can be recorded, named, saved and executed as many times as required and whenever desired. You can record a set of commands and then play them back with one or two keystrokes. By using macros, we are able to automate repetitive tasks associated with data manipulation and data reporting that must be accomplished repeatedly. That means that you can save a lot of time when doing routine and repetitive tasks.

2. **What is VBA? Write its full form and briefly explain why VBA is used in excel?**

   VBA stands for Visual Basic Analysis. Excel VBA is Microsoft's programming language for Office applications such as MS-Excel, MS-Word, and MS-Access. Macros are what most people who write VBA code use. Visual Basic for Application is a human-readable and editable programming code that gets generated when you record a macro.

   VBA is effective and efficient when it comes to repetitive solutions to formatting or correction problems. If you have a change that you have to make more than ten or twenty times, it may be worth automating it with VBA. There are times when you want to encourage or compel users to interact with the Office application or document in a particular way that is not part of the standard application. For example, you might want to prompt users to take some particular action when they open, save, or print a document.

   Use VBA programming to interact with the details of two or more Office applications at the same time and then modify the content in one application based on the content in another.

3. **How do you record a macro? Write detailed steps to create a macro to automatically make the following table in bold and to create borders for it in excel.**

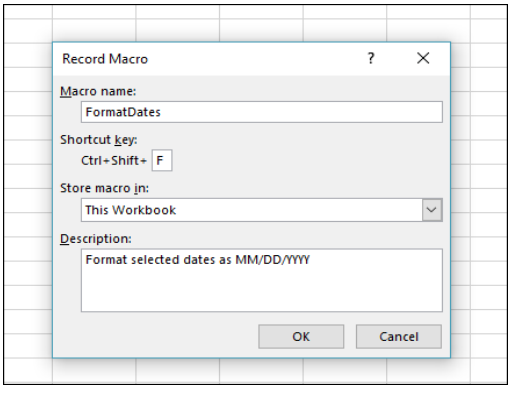
   hi 78

   hello 69

   ineuron 45

   Follow these steps to record a macro along with the example given in the question:-

   1. On the Developer tab, in the Code group, click Record Macro OR Press Alt+T+M+R .



2. In the Macro name box, enter a name for the macro. Make the name as descriptive as possible so you can quickly find it if you create more than one macro.

   Note: The first character of the macro name must be a letter. Subsequent characters can be letters, numbers, or underscore characters. Spaces cannot be used in a macro name; an underscore character works well as a word separator. If you use a macro name that is also a cell reference, you may get an error message that the macro name is not valid.

   Let's give the name as Table_format for the macro we are creating.
3. To assign a keyboard shortcut to run the macro, in the Shortcut key box, type any letter (both uppercase and lowercase will work) that you want to use. It is best to use Ctrl + Shift (uppercase) key combinations, because the macro shortcut key will override any equivalent default Excel shortcut key while the workbook that contains the macro is open. For instance, if you use Ctrl+Z (Undo), you will lose the ability to undo in that Excel instance.

   For our example, give the shortcut as Ctrl+F.
4. In the Store macro in list, select where you want to store the macro.

   In general, you'll save your macro in the This Workbook location, but if you want a macro to be available whenever you use Excel, select Personal Macro Workbook . When you select Personal Macro Workbook, Excel creates a hidden personal macro workbook (Personal.xlsb) if it does not already exist, and saves the macro in this workbook.

   We will store the new macro in This Workbook location only.
5. In the Description box, optionally type a brief description of what the macro does.

   Although the description field is optional, it is recommended you enter one. Also, try to enter a meaningful description with any information that may be useful to you or other users who will be running the macro. If you create a lot of macros, the description can help you quickly identify which macro does what; otherwise you might have to guess.

   We can give the description as "Macro to format the table with Bold and Border."
6. Click OK to start recording.
7. Perform the actions that you want to record.

   Initially, select the dataset and click the Bold option in the Home tab. As the next step, select the full border option from the dropdown list of table borders option.
8. On the Developer tab, in the Code group, click Stop Recording Button image OR Press Alt+T+M+R.

Now if you choose the newly created macro from Developer tab or just by pressing Ctrl+F, you can do the specified formatting in the given table.

4. What do you mean when we say VBA Editor?

Microsoft Visual Basic for Applications (VBA) enables non-programmers to record, create, and edit macros that can automate tasks in Office applications. The first step to working with VBA in excel is familiarized with the Visual Basic Editor also called the VBA Editor, VB Editor or VBE Editor. Visual Basic Editor is a separate application. It is a part of Excel and opens whenever you open an Excel workbook. By default, it's hidden, and

you have to activate to access it. Visual Basic for Applications Editor is a very powerful tool. It is an interface for creating scripts. VBA Editor is the place where you keep the VBA code.

There are multiple ways to get the code in the VBA Editor, such as:

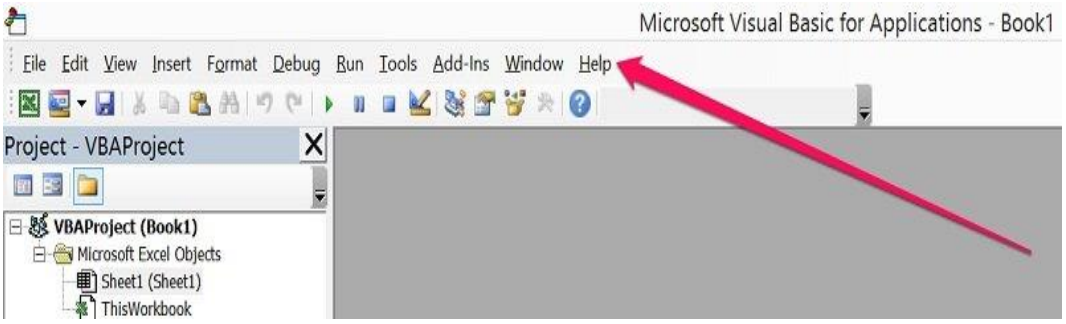
- When you record a Macro, it creates a new module in the VBA Editor automatically and inserts the code in that module.
- You can manually write VBA code in VBA Editor.
- Also, you can copy a code from some other workbook or some other source such as the internet and paste it in the VBA Editor.

5. **Briefly describe the interface of a VBA editor? What is properties window? And what is watch window? How do you display these windows?**

The basic VBE window can be divided in the following 6 sections, all of which is explained below. In reality, there are more components than those which appear in this screenshot (such as the Locals and Watch Windows).

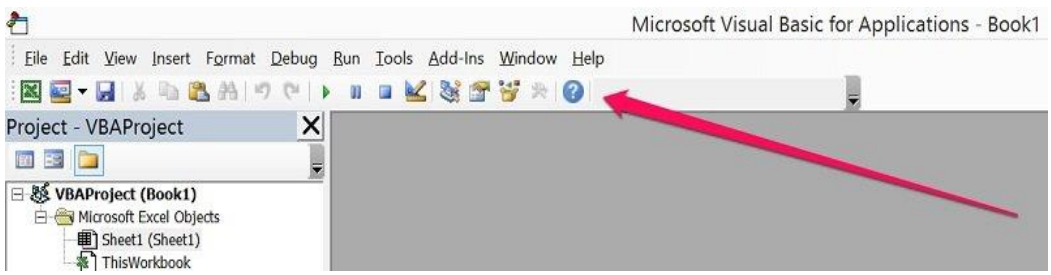
The Visual Basic Editor has several windows and is highly customizable.

#1: Menu Bar



The menu bar, basically, **contains several drop-down menus**. Each of the drop-down menus contains commands that you can use to interact and do things with the different components of the Visual Basic Editor. One thing you'll notice when clicking on any menu, is that several commands have a keyboard shortcut that is displayed at that point.

#2: Toolbar



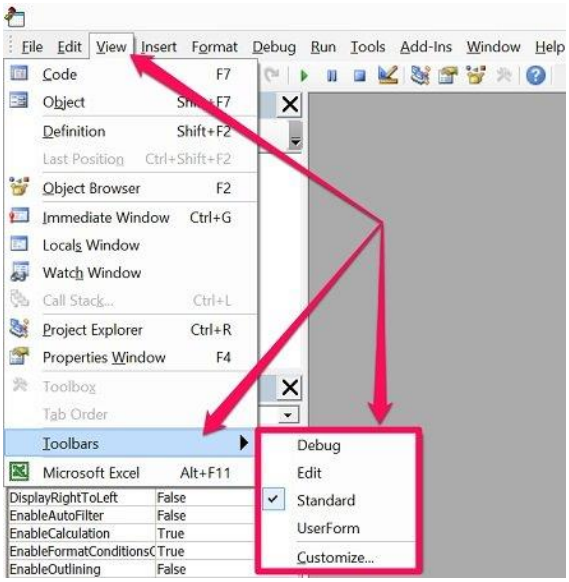
A toolbar contains on-screen buttons, icons, menus and other similar elements that you can use while working with the VBE. The toolbar that appears in the screenshot above is called the Standard toolbar. This

is the only toolbar that the Visual Basic Editor displays by default. There are, however, 3 other basic toolbars:

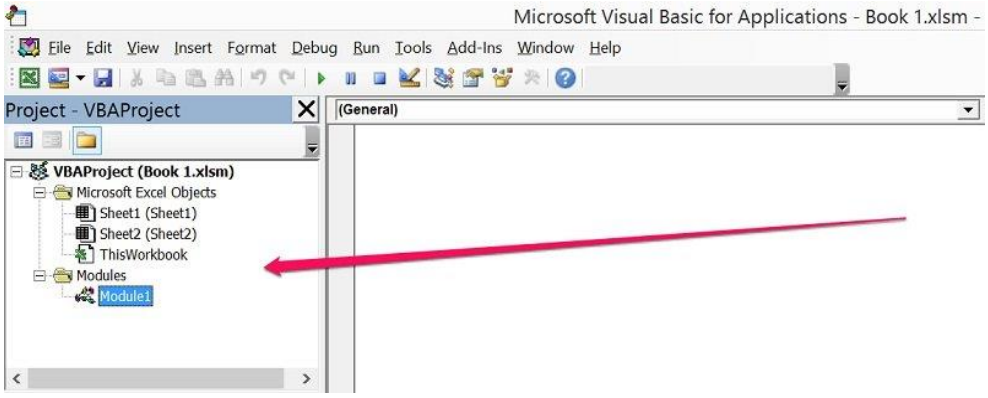
- The Debug toolbar.
- The Edit toolbar.
- The UserForm toolbar.

In addition to the above, the VBE gives you the possibility to customize the toolbars in several ways.

You can change all of these settings by going to the View menu and selecting “Toolbars”. The Visual Basic Editor displays a menu with the 4 different toolbars and the option to access the Customize dialog.



#3: Project Window / Project Explorer

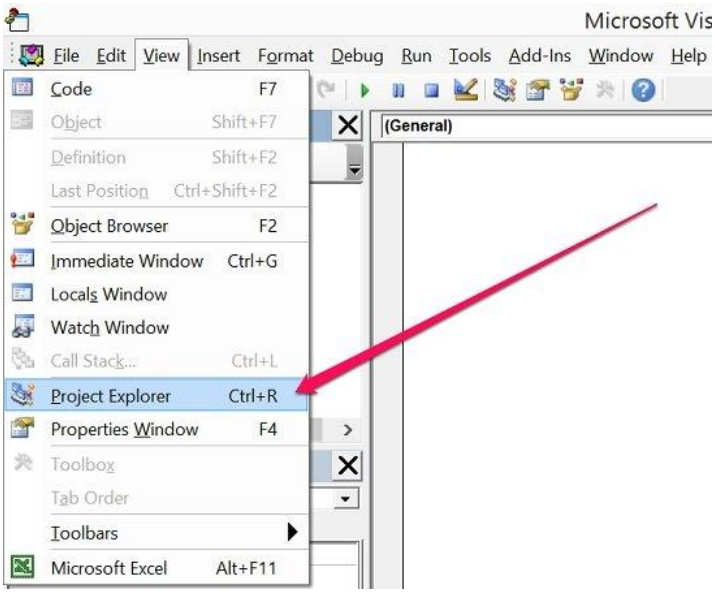


The Project Window, also known as the Project Explorer, is useful for navigation purposes.

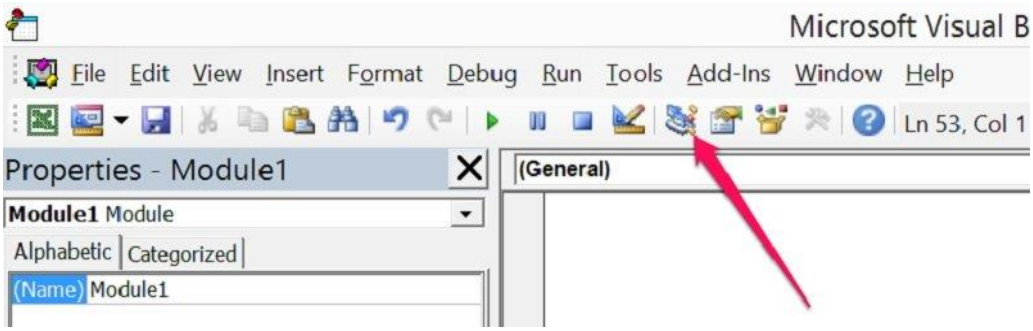
This is the section of the Visual Basic Editor where you'll be able to find every single Excel workbook that is currently open. This includes add-ins and hidden workbooks. More particularly, each Excel workbook or add-in that is open at the moment appears in the Project Explorer as a separate project.

If you can't see the Project Explorer, you can make the Visual Basic Editor display it by using any of the following methods:

- Clicking on “Project Explorer” in the View menu.

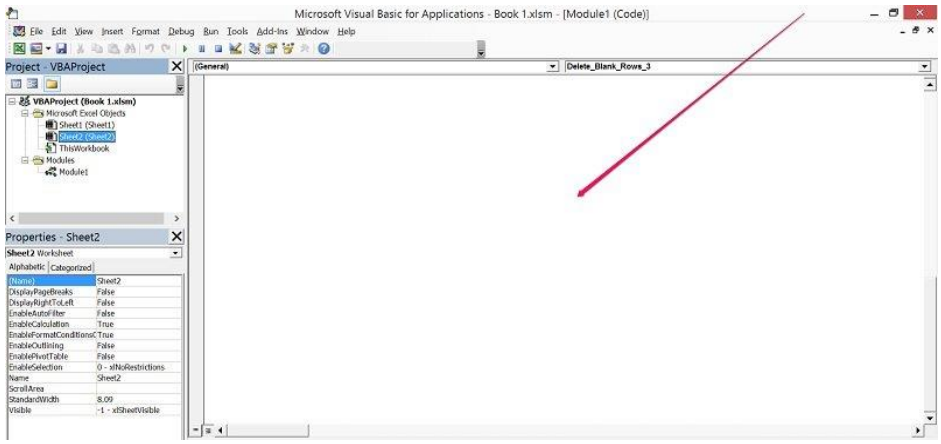


Clicking on the Project Explorer icon in the toolbar.



- Using the keyword shortcut “Ctrl + R”.

#4: Programming Window / Code Window / Module Window

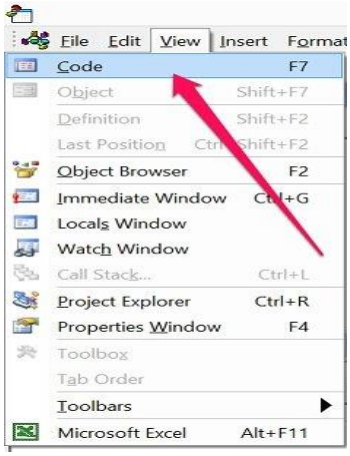


The Code Window of the Visual Basic Editor is where your VBA code appears, and where you can write and edit such code. At the beginning, though, the Programming Window is empty as in the screenshot above.

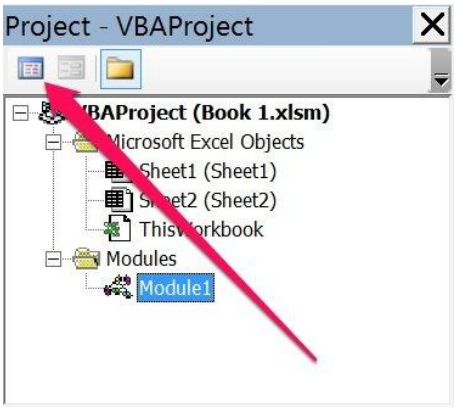
There is a Code Window for every single object in a project. You can access the window of a particular object by going to the Project Explorer and doing any of the following:

Double clicking on the object. The main exception to this rule is UserForms. If you double-click on a UserForm, the Visual Basic Editor displays the UserForm in Design view.

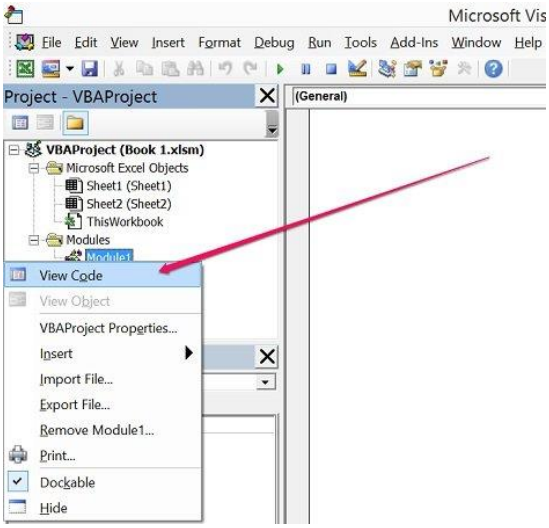
- Selecting the object and, then, clicking on "Code" in the View menu.



- Selecting the object and clicking on the View Code icon that appears at the top of the Project Explorer.



- Right-clicking on the object and selecting "View Code".



- Using the keyboard shortcut "F7".

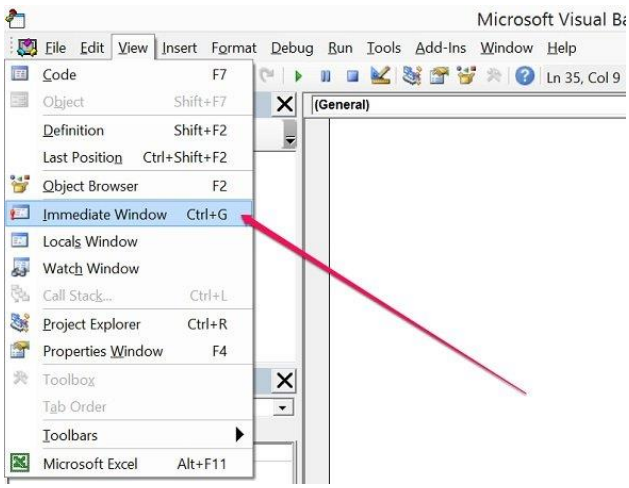
5: Immediate Window

The main purpose of the Immediate Window is to help you noticing errors, checking or debugging VBA code. The Immediate Window is, by default, hidden. However, as with most of the other windows, you can unhide it. Let' take a look at how you can do both the hiding and the un-hiding:

How To Unhide The Immediate Window:-

You can unhide the Immediate Window by doing either of the following:

- Clicking on “Immediate Window” in the View menu.



- Using the “Ctrl + G” keyboard shortcut.

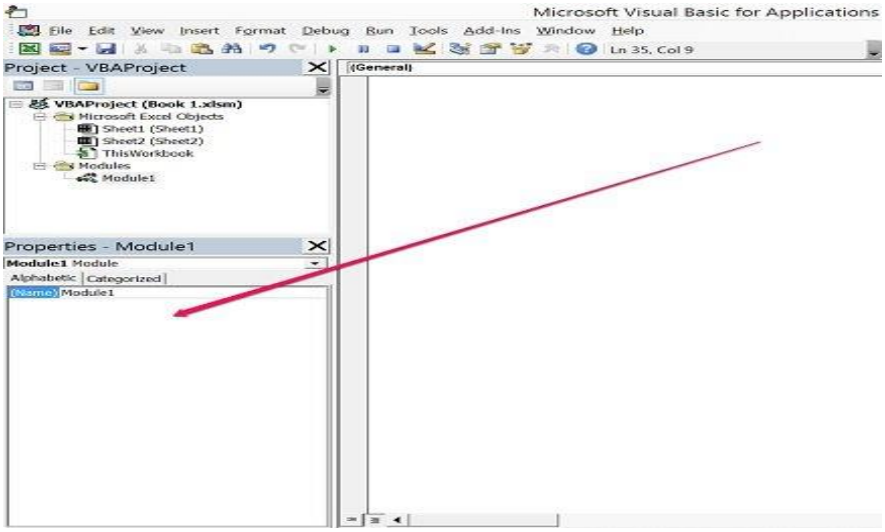
How To Hide The Immediate Window:-

You can hide the Immediate Window using either of the following methods:

- Click the Close button.

- Right-click on the Immediate Window and select “Hide”.

#6: Properties Window



The Properties Window displays the properties of the object that is currently selected in the Project Explorer and allows you to edit those properties.

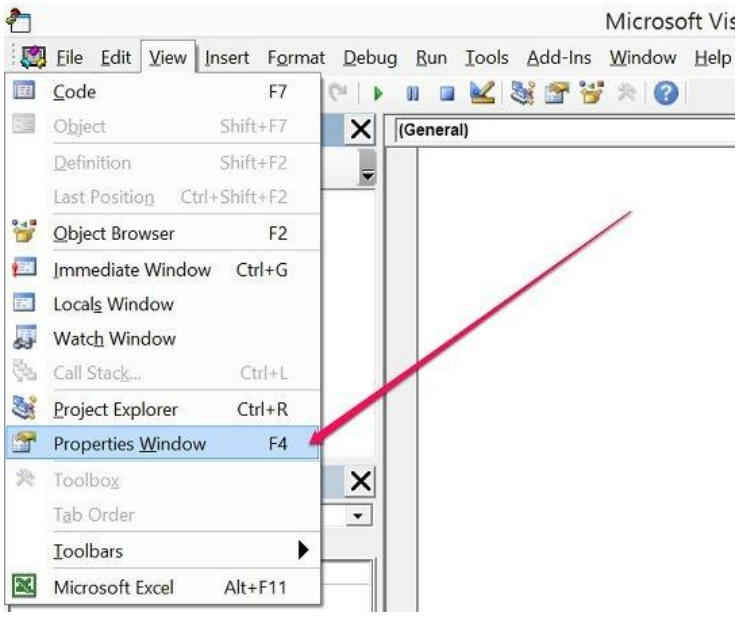
Just as with the Project Window, you can hide or unhide the Properties Window. You're likely to (eventually) work with the Properties Window, particularly in the context of creating UserForms. If you're just beginning to use the VBE, you probably won't need this window too much.

In any case, let's take a look at how you can hide or unhide the Properties Window.

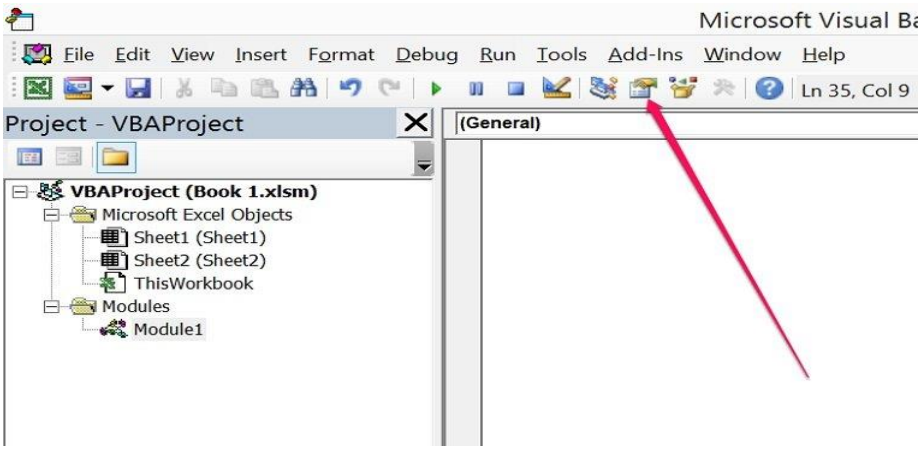
How To Unhide The Properties Window

You can get the Visual Basic Editor to show the Properties Window by using any of the following methods.

- Clicking on “Properties Window” within the View menu.



- Clicking on the Properties Window icon.

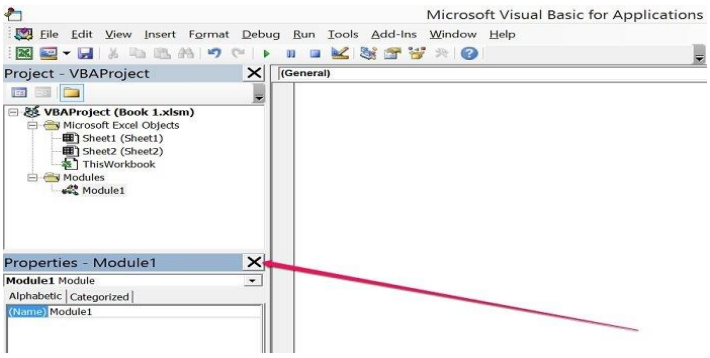


- Using the “F4” keyboard shortcut.

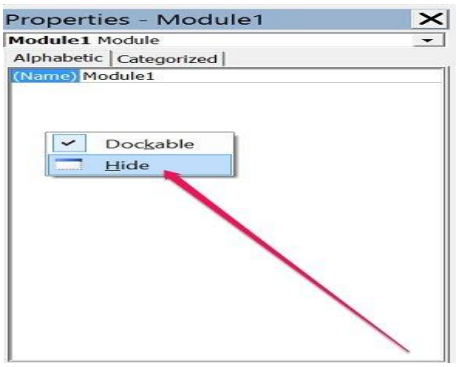
How To Hide The Properties Window:-

You can get the Visual Basic Editor to hide the Properties Window by doing either of the following:

- Click on the Close button of the Properties Window.



- Right-click on the Properties Window and select “Hide”.



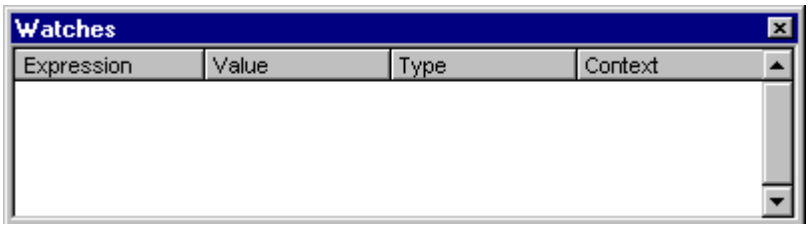
#7 Watch Window

A watch is a variable or expression that has been placed in the window to enable you to monitor its value. It lets you watch the values of variables and expressions as your code executes. When your application enters break mode, the watch expressions you select appear in a window allowing

you to observe their values etc. It is also possible to set up conditional watches. This window is automatically updated after each line of code is executed.

Watches Window:-

Although it is labelled as Watch Window the actual window displays Watches Window.



Expression - Lists the watch expression with the Watch icon, on the left.

Value - List the value of the expression at the time of the transition to break mode.

Type - Lists the expression type.

Context - Lists the context of the watch expression.

You can close the window by clicking the Close box. If the Close box is not visible, double-click the Title bar to make the Close box visible, and then click it.

This window shows all the watches that have been created. It lets you pause the program when a variable value changes. It appears automatically when watch expressions are defined in the project.

You can change the size of the column headers by dragging its border to the right to make it larger or to the left to make it smaller. Close the window by clicking the Close box. If the Close box is not visible, double-click the Title bar to make the Close box visible, and then click it.

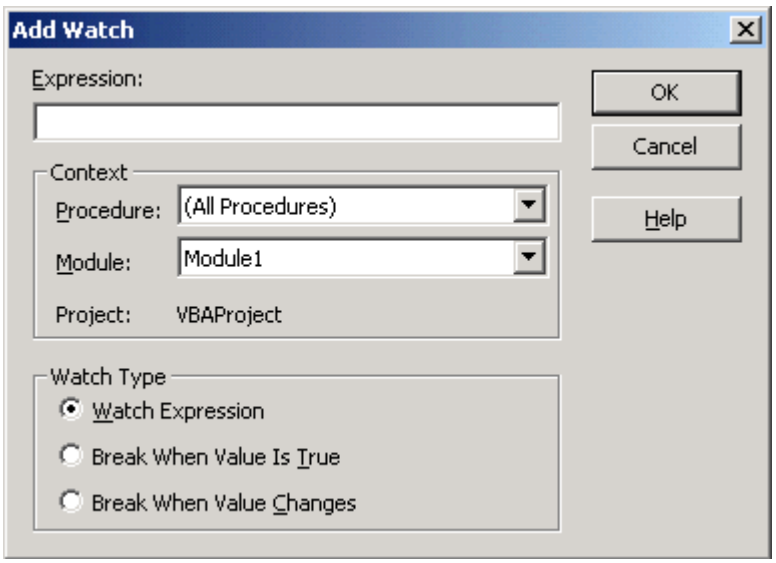
You can edit a value and then press ENTER, the UP ARROW key, the DOWN ARROW key, TAB, SHIFT+TAB, or click somewhere on the screen to validate the change. If the value is illegal, the Edit field remains active and the value is highlighted. A message box describing the error also appears. Cancel a change by pressing ESC.

If the context of the expression isn't in scope when going to break mode, the current value isn't displayed.

Add A Watch:-

3 different types of watches highlight a variable and drag it straight in

To add a watch select (Debug > Add Watch).



Expression -

Context - After adding a watch you can use these drop-downs to change the scope to "Full scope" if originally entered as local scope.

Watch Expression - Adds the expression so the value can be watched during execution. It is same as Local Window.

Break When Value is True - Adds the expression so the value can be watched and execution will enter break mode if the value of the expression is true

Break When Value Changes - Add the expression so the value can be watched and execution will enter break mode if the value of the expression changes.

Drag and Drop:-

Variables can be added by dragging and dropping them into the window. It is easy to watch individual variables.

Drag a selected variable to the Immediate window or the Watch window.

Quick Watch:-

This is a feature you can use to check the value of a variable or expression quickly while in break mode.

Place the insertion point over the variable name and select (Debug > Quick Watch) or alternatively press (Shift + F9).

You can highlight complete expressions.

Another way to get values for expressions and variables quickly is to enable the Auto Data Tips from the (Tools > Options)(Editor tab).

With this feature enabled when you place the mouse over a variable or select an expression and place the mouse over the expression a tool tip will appear after a sort delay displaying the current value.

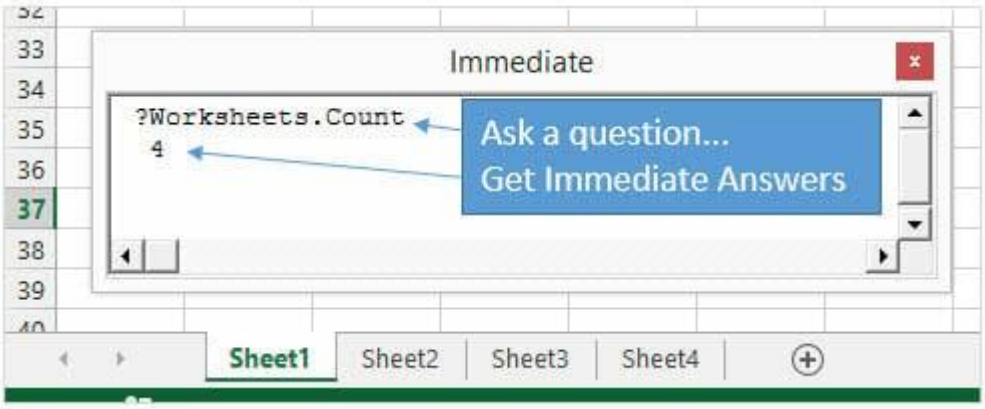
To view the value of variables during execution or break mode hover the cursor over the variable within the procedure or expression. Entire expressions can be evaluated by highlighting the expression with the cursor and hovering.

Remove a Watch:-

Delete it by clicking on the far left and pressing delete.

6. What is an immediate Window and what is it used for?

The VBA Immediate Window is an awesome tool that allows you to get immediate answers about your Excel files, and quickly execute code. It is built into the Visual Basic Editor, and has many different uses that can be very helpful when writing macros, debugging code, and displaying the results of your code.



Every Excel user can benefit from the Immediate Window, even if you're not writing macros. This post will explain 5 different uses for the Immediate Window.

The Immediate window displays information resulting from debugging statements in your code or from commands typed directly into the window.

To display the Immediate window:- From the View menu, choose Immediate window (CTRL+G).

To execute code in the Immediate window:- Type a line of code in the Immediate window. Press ENTER to execute the statement.

Use the Immediate window to:-

- Test problematic or newly written code.
- Query or change the value of a variable while running an application. While execution is halted, assign the variable a new value as you would in code.
- Query or change a property value while running an application.
- Call procedures as you would in code.
- View debugging output while the program is running.