



VBA Message Boxes

VBA message boxes provide a way to give information to a user and get information from a user while a macro is running. This tip describes the InputBox and MsgBox functions.

The InputBox function has the following syntax:

```
MyValue = InputBox(Prompt, [Title], [Default])
```

where

MyValue is a user defined variable to contain the value entered into the input box,
Prompt is a string defining the prompt or message the input box displays,
Title is the title for the input box,
Default is a default value that appears when the input box is opened

An example is shown in the following code:

```
' Use an InputBox to get the input value  
ptext1 = "Please enter the input value for the model."  
ptext2 = "The input must be a number in the range 1 - 50."  
ptext3 = ptext1 & vbNewLine & ptext2  
ans = InputBox(ptext3, "Need Input", 1)
```

This code produces the input box shown in Figure 1.

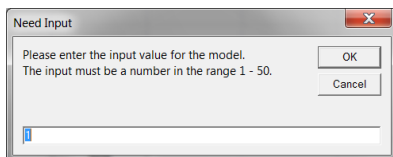


Figure 1. Example Input Box

An InputBox function interprets any value entered as a text string. If a user enters the number 7, the value assigned to the variable that contains the output of the InputBox function is actually the string "7". To convert the variable to a numerical value, the Val command can be used, for example,

```
' Convert a string to a number  
ans = Val(ans)
```

If the user presses the Cancel button a blank string, "", is returned.

The MsgBox function has the following syntax:

```
Myvalue = MsgBox(Message, [Buttons], [Title])
```

where

MyValue is a user defined variable to contain a code corresponding to the button that is clicked,

Message is a string defined the message the message box displays,

Buttons are the type of buttons selected for the message box plus the type of icons to be displayed,

Title is the title for the input box.

An example is shown in the following code:

```
' Give the user a warning about clearing the results  
mtext1 = "You are about to clear your saved results."  
mtext2 = "Are you sure you want to do this?"  
mtext = mtext1 & vbNewLine & mtext2  
ans = MsgBox(mtext, vbExclamation + vbYesNo, "Caution!")  
' If the user clicked the No button exit the sub  
If ans = vbNo Then  
    Exit Sub  
End If
```

This code produces the message box shown in Figure 2.

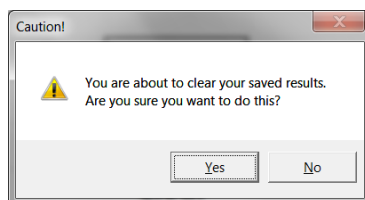


Figure 2. Example Message Box

In this example, the value in the variable is a number corresponding to the button that was clicked. VBA provides functions such as `vbYes`, `vbNo` and `vbCancel` that allows the user to checked whether a button has been clicked.

A list of all the available button styles and icons can be found in Help for Visual Basic. Search for the `MsgBox` function.