

# Microsoft Excel VBA: Command Buttons

A Command Button is an object that you draw on to a worksheet and then program with VBA<sup>1</sup> to perform a specified task when the button is clicked. Command buttons offer a simple way to automate tasks in Excel and can provide the user with valuable interactivity in financial models.

## Getting Started

In Excel 2007 you must first enable the Developer tab of the Ribbon order to get access to the tools you need to work with worksheet controls and VBA. To do this click the **Office Button**, choose **Excel Options** and in the **Popular** section place a tick in the box next to **Show Developer tab in the Ribbon** (Fig. 1). (Note: In Excel 2003 (and earlier versions) you will find the tools you need on the Forms toolbar: **View > Toolbars > Forms**).

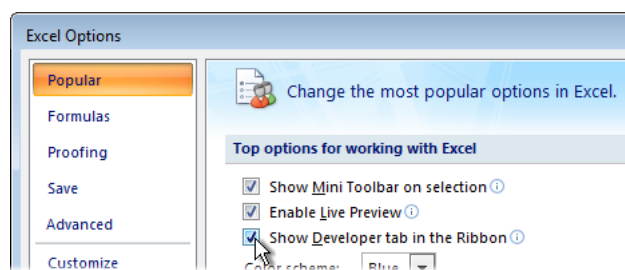


Fig. 1 Enable the Ribbon's Developer tab.

## Drawing the Command Button

It doesn't matter whether you create the command button first and then write its code or vice versa. In this example the button is created and then the code is written and assigned to the button.

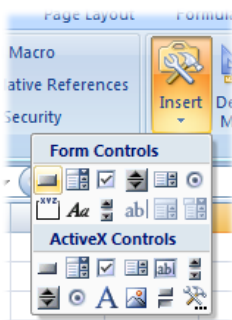


Fig. 2 The Controls Palette.

Move to the worksheet where you want to place the button and on the **Developer** tab of the Ribbon click **Insert** to display a palette of controls (Fig. 2). The controls are divided into two groups, *Form Controls* and *ActiveX Controls*.

Although similar in appearance these two groups of controls work in different ways. Form controls are simple with a restricted range of properties whilst ActiveX controls are highly programmable and customizable. However, owing to security issues ActiveX controls created in Excel 2007 do not work in earlier versions of Excel so it is important that you choose controls from the Forms section if there is any likelihood that your workbook will be viewed in Excel 2003 or earlier.

In the **Form Controls** section select the **Button** tool. The mouse pointer changes to a cross. Now **click and drag** on the worksheet to create the button (Fig. 3).

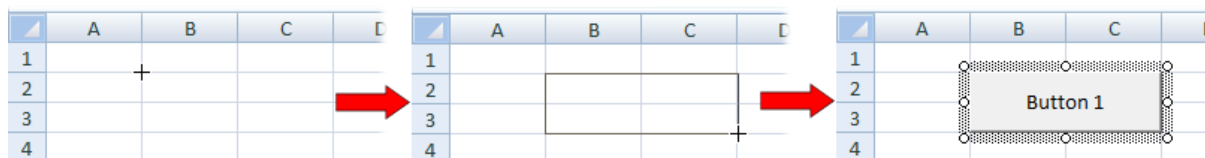


Fig. 3 Click and drag to create the button.

As soon as you release the mouse the **Assign Macro** dialog box appears inviting you to specify which macro code should run when the button is clicked but since you haven't created that yet you can dismiss the dialog by clicking its **Cancel** button. You can write and assign the code later.

When selected, the button has a broad shaded border with sizing handles at each corner and half-way along each side. Drag the border to move the button. Drag one of the handles to change its size. (HINT: Hold down the **[Alt]** key whilst dragging to snap the button to the borders of the underlying cells. This helps you to draw buttons of uniform size and shape.)

## Change the Caption

The button should have a meaningful caption so that the user knows what is going to happen when they click it. In this example we create a set of simple navigation buttons to help the user move quickly from one worksheet to another. Click the button to select it then drag the mouse across the existing caption (e.g. "Button 1") and type the caption you require (Fig. 4).

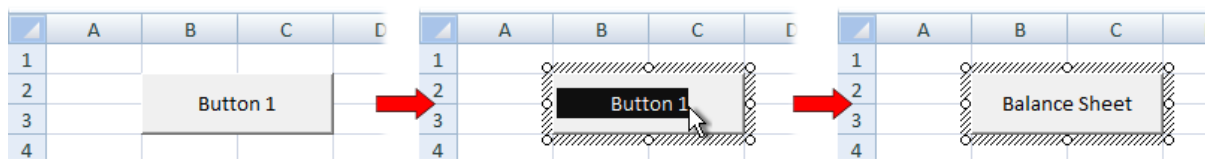


Fig. 4 Change the button's caption.

(HINT: After you have assigned a macro to the button simply clicking it runs the macro so, if you want to select the button without running its macro, hold down the **[Ctrl]** key as you click it.)

## Write the Macro

Any macro can be assigned to run when the button is clicked. You can record a macro or write the code yourself. Here's how to write a macro that takes the user to a specific worksheet:

On the **Developer** tab of the Ribbon click the **Visual Basic** button (or use the keyboard shortcut **[Alt]+[F11]**) to open the Visual Basic Editor<sup>2</sup>. Make sure that the current workbook is selected in the **Project Explorer** then choose **Insert > Module** to create an empty code module. The module opens in the Visual Basic Editor and also appears in the Project Explorer. This is where you write your code.

In the code window type the word *Sub* followed by a space and then a name for your macro (maximum 32 characters with no spaces) such as *GoToBalanceSheet* then press **[Enter]**. The editor adds a pair of brackets after the name, creates the line *End Sub* and places your cursor in between. Your code should go between the *Sub* and *End Sub* lines. To take the user to a specific worksheet and also make sure they go to the top of the sheet by selecting cell A1 you need two lines of code:

```
Worksheets("Balance Sheet").Activate
Range("A1").Select
```

Your code window should now look something like this (Fig. 5). (HINT: Code looks neater and is easier to read if you indent each line by pressing the **[Tab]** key before you start typing.)

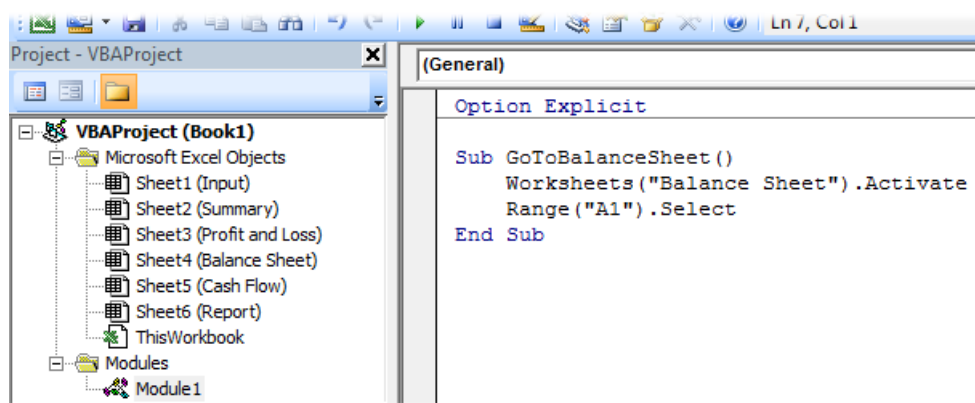


Fig. 5 The Visual Basic Editor showing the completed macro.

## Assign the Macro to the Button

Return to the main Excel window and **right-click** on the button you created and choose **Assign Macro** to open the Assign Macro dialog. You will see the name of the macro you just created. Select the macro then click **OK**. Test the macro by clicking the button.

<sup>1</sup> VBA: Visual Basic for Applications. The programming language used for creating macros in Microsoft Office applications.

<sup>2</sup> Visual Basic Editor: A program embedded into most Microsoft applications where you write VBA code to create macros.