Visual Basics

By Barry Seymour

Advice from your Uncle

No code this week; just a bunch of tips and good ideas.

Converting numbers to character strings: Here's the old way, which takes more typing but ensures you don't have any leading or trailing blanks..

```
Number%=145
STRING$=Ltrim$(Rtrim$(Str$(Number%)))
```

Here's the new way, which is a lot easier to type and yields the same results:

```
Number%=145
String$=Format$(Number%, "0")
```

This command formats String$ in the same pattern you enter as the second parameter. It performs multiple tasks; it converts the number (whatever type) to a string, it removes leading and trailing blanks and will format the number cleanly, with no commas, dollar signs, etc. If the expression evaluates to zero, you'll get "0" as the result instead of a null string with the other method.

This works equally well with actual values or variables. Bear in mind if you're converting non-integer numbers that they'll get rounded off unless you leave room for the numbers to the right of the decimal in your Format$ call.

Help with Error Messages: Whenever you're writing code and something crashes, Visual Basic gives you an error message. If you want more information on the nature of the error, press F1 before you dismiss the error message box. VB.HLP will be opened to the relevant topic describing the error. The help file often discusses many ways the error could have been caused, and can act as a convenient checklist of things to look for. There've been many times I've found this useful, even though I thought I knew all about a given error condition.

The Moral: **Don't be too macho to use F1!**

Maniuplating forms by remote control: The subs that control your forms don't have to be in the forms themselves! Remember that memory is set aside for all the subroutines and functions in a form for the entire time the form is loaded. If you're concerned about conserving on system resources, put your subs and functions in separate modules which are only loaded into memory when their code is called.

This can present a few problems with manipulating forms. For one, you need to reference the controlled form by name in **all** your code. For example, if you had a module called DISPLAY.BAS and it contained routines which controlled the display of SHOWOFF.FRM, (FormName = ShowOff), you'd need to use SHOWOFF's FormName in front of all calls controlling that form. Thus the following differences:

This...	**Becomes this...**

```
Height=600                          ShowOff.Height = 600
Label1.Caption = "Kick Me"          ShowOff.Label1.Caption = "Kick Me"
List1.ListIndex = 0                 ShowOff.List1.ListIndex = 0
```

And so on.

Getting to a Sub or Function Quickly: Highlight the sub or function call in your current code and press Shift F2. This will immediately jump you to that Sub or Function. This is a mixed blessing; when you're tracing code manually it's a big help; however there is no equally quick way to backtrack! I usually stick in a comment with some totally unique combination of characters (such as three pipe symbols or something) and then key a search on that after I'm done with the sub in question.

Keeeeeyyboard Shortcuts!: As a touch typist, I can't do without 'em. They may seem obvious when you read them in the following table (they sure look obvious to me as I write 'em!) but if you take a few minutes to memorize them and use them as you code you speed things up immensely and maybe even save yourself some heartache!

Command	What it does
ALT+F4	This closees your currently active code window This gets it out of the way fast prior to a program run, but it can be a pain in the butt if you made a mistake and have to open it again.
ALT+FV	**F**ile Sa**v**e Project: Updates your MAK file!. Also fools you into thinking you've saved everything else as well! You didn't.
ALT+FS	**F**ile **S**ave: Saves the form or code module you're currently working on. This is NEVER bad unless you just saved a change you can't easily undo. (More on this later)
ALT+FR or F5	**F**ile **R**un: Much faster than doing it from the menus with the mouse. Trust me.
F3	Repeat Last Search: This is convenient and can also be used to start a new search if there is no prior search. However you could also inadvertently lose your place.
ALT+WI	**W**indow **I**mmediate Window: At Break Time, this brings up the Immediate Window: If you're working on a large project or you have a screenful of cluttered large forms, this will cut to the IWindow quickly! On the other hand, it could just be simpler just to click on the darn thing. It all depends on your project. (Now I'm starting to sound like Tom Bodett!

Slingin' around them Controls: You can move, cut, copy and paste groups of controls using the **Ctrl** key. Click on the first control to highlight it; hold down the Ctrl key as you click on subsequent controls. You can drag 'em all by grabbing any one of them; If cut (Shift+Delete) or copy (Ctrl+Insert), you can then paste (Shift+Ins) the controls into the form, a frame or picturebox on the form or even to another form. I've even used this technique to copy controls from one project to another, but the code in the controls does not survive the ride.

Detecting Keyboard Input: Good Windows programs allow the user to control a lot of things via keyboard shortcuts, but that means you may have to write a keyboard handler. Many events can be put in a menu and assigned accelerator keys, which is the simplest method. When the accelerator keys are pressed the function or sub is called without the menu displaying.

If you have to trap non-ASCII keystrokes (arrow keys, PgUp, tabs, etc) your best bet is to write a KeyHandler routine which is accessible globally, then put calls to it in the KeyDown event of all controls which might be active when the user types. There are a few catches you should be aware of:

☞ Some controls don't have a KeyDown event. Try to avoid letting them get the focus.

☞ Don't put calls to KeyHandler in buttons; if you have several buttons on a form the left & right arrow keys will move the focus around on the button without calling the KeyDown event.

☞ You can disable pictures without affecting their display. This means you don't have to worry about or plan for their getting the focus and capturing/redirecting keystrokes.

☞ You can simulate Keyboard repeats with a While..Wend or Do..Loop loop. Set a form or global variable to TRUE and start looping in the KeyDown Event. Reset the variable to FALSE in that control's KeyUp event to stop the loop. This works either on the Control level or in the global KeyHandler routine.

The Unknown Form: Here's a trick many people probably don't know about; You can remove the title bar from your form! It's not easy, but it can be done by setting the ControlBox, MinButton and MaxButton properties of your form to ZERO and setting the Caption to a null string. At that point the blue title bar at the top of the form disappears with a sigh of resignation at run time. This can be helpful if you want to prevent the user from moving your form. A false title bar can easily be mocked up with a label control.

And finally...

Don't use END. You're better off just unloading everything. First of all, it's a good idea to unload everything anyway; if you have a multiple form application simply calling 'End' won't unload the other forms. Clean up after yourself, whydontcha!

The second (and more important) reason is that Windows and VB don't get along when it comes to ending a program. This becomes a problem when Windows or some external agency tries to send a "shut yourself down" message to a running VB application.

In the Form_Unload section of your main form, put code to clean up your application (prompt to save unsaved work, etc.) Then unload all **other** forms. After all this code completes, the form itself will bow out gracefully. Put a call to Form_Unload in the File Exit section of your form's menu and all will be done!

This column is available on the Windows Online BBS in Danville, California, phone 1 510 736-8343.   This column in Windows HELP format (VB018EX.HLP) is in VB017EX.ZIP, and may be distributed as freeware.

Marquette Computer Consultants
22 Sirard Lane
San Rafael, CA 94901-1066
(415) 459-0835

for Windows Online "the Weekly"