Trapclose

-
 Alyce

Closing a Window

All windows must be closed when a program ends. If you do not close a window, Liberty BASIC will close the window for you when you are running the program from within the Liberty BASIC editor, and it will notify you that this has been done so that you can add the appropriate CLOSE statement to your code. Most programs have a menu item or button that causes the window to close and the event handler (branch label) for the menu or button will contain the CLOSE statement. When a window is opened with the OPEN statement, it is given a handle. Possible handles are: #1, #win, #main, etc. This handle is used with the CLOSE statement as well.

```
Open "My Window" for window as #main
Close #main
```

Here is a small example that has both a menu item and button to close the window:

```
nomainwin
menu #1, "&File","E&xit",[quit]
button #1, "Exit",[quit],UL,10,10
Open "Close Me!" for window as #1

wait

[quit] close #1  :end
```

The User Bypasses Your Exit Button!

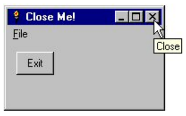
In the code above, it looks like you've got the exit routine handled quite well so that the window is properly closed when the program ends. The program's user can get around your carefully planned routine, though! There are three ways to close a window.

Keystrokes to Close

At any time, when the user presses the ALT key, holds it down, then presses the F4 key, he closes the window. This is built into the Windows interface.

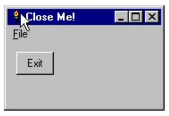
Closing with the X

Every window with a titlebar has a button at the right side of the titlebar that will close the window. It is generally represented by an X. It looks like this:

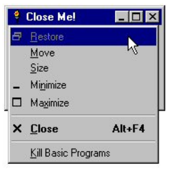


Closing with the System Menu

Windows with titlebars often have an icon on the left side of the titlebar.



If a user clicks on the icon, he activates a system menu that allows him to do several things, one of which is to close the window:



Trapping the System Close Events

When the user types ALT-F4, or clicks the X button or the "close" item on the system menu, the program shuts down improperly, not allowing you to close open handles and issue an END statement. Try running the code below. First, close it with the program's button or menu item. You'll get the notice that "You are quitting" that is contained in the [quit] branch. Try running the program again, but this time close with the X button. You won't get the notice, indicating that your closing routine is not accessed. In this small demo it doesn't matter, but in a larger program, you may need to close other windows or files, or unload bitmaps, or write information to ini files, etc.

```
nomainwin
menu #1, "&File","E&xit",[quit]
button #1, "Exit",[quit],UL,10,10
Open "Close Me!" for window as #1

wait

[quit]
notice "You are quitting."
close #1  :end
```

TRAPCLOSE to the Rescue!

One simple statement will allow you to trap the event of the user closing a window with the system commands listed above. It is the TRAPCLOSE statement. Send it to the window like this:

```
print #main, "trapclose [branchLabel]"
#main "trapclose [branchLabel]"
```

Here is an improved version of the little program that traps the system close event:

```
nomainwin
menu #1, "&File","E&xit",[quit]
button #1, "Exit",[quit],UL,10,10
Open "Close Me!" for window as #1
#1 "trapclose [quit]"

wait

[quit] close #1  :end
```

Trapclose | Closing a Window | The User Bypasses Your Exit Button! | Keystrokes to Close | Closing with the X | Closing with the System Menu | Trapping the System Close Events | TRAPCLOSE to the Rescue!