

The ABCs of APIs Lesson 2

Using Multiple Arguments to Move a Window

Table of Contents

[The ABCs of APIs Lesson 2](#)

[Review](#)

[Expanding Liberty BASIC](#)

[Moving a Control by API](#)

[API Functions Can Perform Actions](#)

[Moving a Window by API](#)

[Recap](#)

[Do it Yourself](#)

Review

[In Lesson 1](#) we learned how to make a simple API call, passing a numeric argument and retrieving a value.

Expanding Liberty BASIC

Liberty BASIC has many built-in capabilities. It can move and resize controls with the **LOCATE** command. The **LOCATE** command looks like this:

```
#1.button "!locate 200 100 120 50"  
#1 "refresh"
```

The control is given the **LOCATE** command and the window receives a **REFRESH** command to cause

the display to be repainted and show the new location of the control. This only works in a window of type **window**. It does not work in a graphics or dialog window.

We can add this feature to other types of windows with the API function **MoveWindow**.

Moving a Control by API

A single API call allows us to relocate and resize a control. It looks like this:

```
hButton = hwnd(#1.move)      'handle of button

call dll #user32, "MoveWindow", hButton as ulong, _
x as long, y as long, w as long, h as long, _
1 as long, result as long
```

This function requires **multiple arguments**. In Lesson 1, we showed how to use a single argument. We can use many arguments in an API call by separating them with commas, as in this example that has 6 arguments.

Before we call the function we need to retrieve the handle of the button with **HWND()** just as we described in Lesson 1. Since it is a handle, it must be passed as type **ULONG**. The handle is the first argument of the **MoveWindow** API call.

In Lesson 1 we said that **CALLDLL** requires the statement to be on a single line, but we can break it into multiple lines with the line continuation character. That makes it easier to read and it allows us to add comments. Here is the **MoveWindow** function fully commented. You can see that each of the six arguments has a specific purpose. Each argument gives a bit of information to the **MoveWindow** function.

```
hButton = hwnd(#1.move)      'handle of button
call dll #user32, "MoveWindow", _
    hButton as ulong, _      'handle of button
    100 as long, _           'x location of button
    100 as long, _           'y location of button
    200 as long, _           'width of button
    100 as long, _           'height of button
    1 as long, _             'repaint flag, 1=yes, 0=no
    result as long           'nonzero = success
```

API Functions Can Perform Actions

In Lesson 1, we used an API call to discover the enabled state of a control. The API call gave us some information, but it did not perform an action. The **MoveWindow** API moves the window or control according to the information in the arguments.

Copy the code below and paste it into Liberty BASIC, then run it. When you click the button, you'll see it move. This is done in a graphics window. Liberty BASIC doesn't have a native capability to move a control in a graphics window, so the API function extends the capabilities of Liberty BASIC.

```
nomainwin
WindowWidth=400:WindowHeight=400
button #1.move, "Move Button API",[moveButton],UL,10,10
open "Moving Test" for graphics_nsb as #1
#1 "down; fill cyan; flush"
#1 "trapclose [quit]"
wait

[quit] close #1:end

[moveButton]
    'this code shows how to move a button
    'in a graphics type window
    'with the MoveWindow API function
    hButton = hwnd(#1.move)      'handle of button
    calldll #user32, "MoveWindow",_
        hButton as long,_        'handle of button
        100 as long,_            'x location of button
        100 as long,_            'y location of button
        200 as long,_            'width of button
        100 as long,_            'height of button
        1 as long,_               'repaint flag, 1=yes,0=no
        result as long            'nonzero = success
    wait
```

The x and y locations for the control are relative to the upper left corner of the window's workspace. That work space is the part of the window inside the frame and under the titlebar and menubar if there is one. It is also called the **ClientArea** of the window.

The repaint flag updates the appearance of the window to show the new location of the control. If the repaint flag is passed as 0, the button may not appear to move, or it may appear in both the old and new locations, so be sure to specify a repaint flag of 1.

Moving a Window by API

Liberty BASIC allows us to specify a location and size for a window when it is created. We do this by setting **UpperLeftX UpperLeftY WindowWidth WindowHeight**.

```
UpperLeftX = 10
UpperLeftY = 50
WindowWidth = 400
WindowHeight = 200
```

There is not a way to change the location or size of the window after it has been opened. The same **MoveWindow** API function allows us to move a window as easily as we moved the button in the previous example.

When a Window is moved with **MoveWindow** the x and y locations are relative to the upper left corner of the screen.

Here is a program that shows how to move a button with the native **LOCATE** command and how to move the window with the **MoveWindow** API call.

```
nomainwin
WindowWidth=400:WindowHeight=200
button #1.movewin, "Move Window API",[moveWindow],UL,10,10
button #1.movebtn, "Move Button LB",[moveButtonLB],UL,10,100
open "Moving Test" for window as #1
#1 "trapclose [quit]"
wait
```

```
[quit] close #1:end
```

```
[moveWindow]
    'this code shows how to move the window
    'with the MoveWindow API function
    hWnd = hwnd(#1)          'handle of window
    callDll #user32, "MoveWindow",_
        hWnd as long,_      'handle of window
        10 as long,_        'x location of window
        10 as long,_        'y location of window
        550 as long,_       'width of window
        350 as long,_       'height of window
        1 as long,_         'repaint flag, 1=yes,0=no
```

```
        result as long        'nonzero = success
wait
```

```
[moveButtonLB]
    'this code shows how to move a control
    'with the native LOCATE command
    #1.movebtn "!locate 200 100 120 50"
    #1 "refresh"
wait
```

Recap

Some API calls perform actions based on the arguments they receive. Many API calls require **multiple arguments**. Arguments are the instructions for the API function. Arguments are separated by commas.

As we learned in Lesson 1, the last item in the CALLDLL statement is always the value returned by the function. The last item is not an argument. It is the return from the function.

Do it Yourself

Do you think you understand the basics of making API calls? Do you understand how **MoveWindow** works? Why not write your own program that demonstrates this function? Move a control other than a button. Move a control in a dialog type window. Relocate a program window. If you can do it properly on your own, you are well on your way to mastering the skill of the API!

[The next tutorial](#) in this series will discuss the different numeric types used in API calls.

Table of Contents

[The ABCs of APIs Lesson 2](#)

[Review](#)

[Expanding Liberty BASIC](#)

[Moving a Control by API](#)

[API Functions Can Perform Actions](#)

[Moving a Window by API](#)

[Recap](#)

[Do it Yourself](#)