Table of Contents

Stack Overflow

System Primitive Failed

Float invalid op

Invalid branch label [xyz]

Index (nnn) is outside collection bounds

Argument must be a collection

Collection is empty

IsEmpty

Protection violation

String can't hold string

Fast FAQ:

by -
RodBird

The following issues have been discussed by newcomers to the boards. Even if you have programming experience have a read, something will be relevant in the near future and little bells might ring when you feel perplexed.

First a few words about the environment you are working in, then some general concepts about Liberty BASIC and finally a discussion about some of the more cryptic runtime errors.

Windows Environment:

Liberty BASIC will by default install itself to the Program Files (x86) directory, usually C:\Program Files (x86). This is the protected Windows directory for 32Bit programs. You will not normally be allowed to save or write anything to this directory, it is locked. Support files and new .bas files are by default written to your appdata directory. Usually C:\Users\Yourname\AppData\Roaming\Liberty BASIC v4.04. Here you will find all of the media, bmp and .bas examples. Windows has a special name for this folder, it is %APPDATA%, this is a Windows environment variable name. Key %APPDATA% into any file dialog will give you quick access to the location of the default Liberty BASIC v4.04 folder.

The normal way to interact with Liberty BASIC is to open the Integrated Development Environment (IDE) and then select a .bas program to run or edit. Under the Preferences menu you can set it so that Liberty IDE always opens with the last file edited in view. Or, you can use the File menu to open another file.

You can set your system up to open any .bas file by clicking on it. To do this right click on any .bas file, choose "Open with" and then "Choose default program". Now navigate to and choose C:\Program Files (x86)\Liberty BASIC v4.04\liberty.exe. Click on "Always use this file". Now, anytime you click on a .bas program, Liberty BASIC's IDE will open with the .bas loaded ready to run or edit.

Its also a good idea to establish a project area like C:\MyBasicFiles and store projects in sub directories like C:\MyBasicFiles\ProjectName. This need not be in the %APPDATA% folder C:\MyBasic is just fine.

You need to know that paths like C:\MyBasicFiles\Project\Bmp can be hard coded or relative. If they are hard coded you must specify exactly where they are and quote the full path. Don't do this. It is much easier to use relative paths or none at all. If you have a bmp file you may keep it in the \Project directory in which case you just use the bmp's name "my.bmp" to load it. If you wish to keep things tidy create a sub directory inside your \Project directory like \Project\Bmp put the bmp file in this subdirectory. In your program use the relative path "Bmp\my.bmp" This way you can move or distribute the code and it will work as the directory structure remains intact and cares not where it resides. For example a user or your friend might put the \Project directory in C:\Temp which is not the same as C:\MyBasicFiles.

Final point about the Windows environment. When you start Liberty BASIC the IDE will open, be sure you can see the whole window. There is a status bar at the foot of the window that reports errors and offers help that is often hidden behind the Windows task bar. Resize your window, especially when debugging, so that you can see this status bar.

Tutorial and Help

Liberty BASIC v4.04 ships with interactive tutorials in the form of .lsn files, here you can read about and run code examples. To start the tutorial search for the .lsn file in %APPDATA%. Don't try to run it from the Help menu as it will fail.

The main help file ships as an HTML file that opens in your default browser. You can improve on this by using some of the search aids that Liberty coders have created. You can also create a .chm file which is more interactive and easier to navigate. Read to the end of this FAQ and then come back and click this link. http://alycesrestaurant.com/lb4chm.htm

Also searching this forum and Google often turns up interesting solutions to coding puzzles.

Write Run Compile Execute:

Liberty code is drafted in the IDE and stored as a .bas file. When you click on Run or Debug, your code is first compiled, any coding errors cause the compiler to stop and report in the status bar. When error free the compiled code goes to runtime and your instructions are executed and appear on screen in whatever window type you have specified. So, running a program is a two step process, seamless, but still two steps. Optionally a .tkn file can be created; it is a precompiled program that saves a little time when starting up. The .tkn hides your code from prying eyes but this is not its intended purpose, its simply a precompiled version of your code that will start instantly.

When your program is executing and comes across an error this is called a runtime error. you will get some form of error report or notice popup, it may simply point you at the error.log. You can look at the error.log by clicking run then selecting error.log. Be aware that the file may be large and that the errors are appended so you need to scroll to the bottom of the file. You may delete the contents or the file, Liberty will reinstate it. The error.log is complex, share it on the forums with the offending code if you want help.

Debug:

So the compiler did some error correction for you but now that you are executing the .bas file the Ladybug icon is your best friend. Go on, click it. Spend as much time with the debugger as the IDE. Click on the >>> animate icon and watch your program run line by line. If it gets a runtime error it stops on the offending line, reports in the status bar and you can see the content of all your variables at that moment in time. You may set break points by clicking on the left side bar of the debugger window.

Window Types:

Liberty displays everything on screen in a variety of window types. The first window type you will encounter is Liberty's mainwin. This is the default window type and is essentially a testing and debugging window. It always opens unless you switch it off with nomainwin, as soon as you are competent you will be using other window types and once debugged you will switch off the mainwin.

Mainwin is great for your first steps into programming and you can write quite complex programs in this environment, but it has limits. It cannot produce a GUI style interface. It's styled as a traditional DOS console and cannot produce colored text. Neither does it support the traditional inkey$() statement, only input$(1). The essential difference is that program flow stops and waits for input.

You will use mainwin for development and debugging and a standard window for almost everything else. A standard window provides GUI controls like listboxes, graphicboxes, buttons and checkboxes. There are specialized windows for graphics, text and dialogs. Be aware they exist and take time to choose the correct window type for the task in hand, but have a good reason to move away from the standard window type.

Looping Stop Break Kill:

Endless loops are an endearing feature of all programming languages. A simple loop like WHILE WEND can consume all of the processor cycles that Liberty can get hold of and if you don't have dual core your PC will freeze as well as the program.

If you are in trouble press Ctrl+Break Then navigate to the IDE and from the Menu select Run, Kill BASIC Programs. The program may persist in the kill list even though it has been killed, this is because you opened a window but it did not get closed. It will get corrected when you restart Liberty.

Do put SCAN inside loops. Scan is a very important command that causes Liberty to check for mouse or keyboard input. The trapclose event for example will never be seen if you don't SCAN inside a loop.

Don't code delay loops like FOR n=1 TO 50000 : NEXT. Liberty operates in a multi-threaded environment where processing cycles are shared. A delay loop like that hogs the processor and ignores all input and events till it is done. Use TIMER if you need a pause, or better still simply WAIT for the next user input event.

GUI program flow:

Know a bit about BASIC? Cool. To make the transition to GUI you need to have the concept of "events" firmly in your mind. You set up events in your program, the keyboard might trigger an event, the mouse might trigger an event, and the timer might trigger an event. A listbox might trigger an event. Each event has a handler, a code block that will be called when the event fires.Typically at the end of this code block there will be a WAIT statement. Program execution sits and waits here till another event fires then it shoots off to action that. You may also use Subs to handle events.

Events repeat, you need to switch them off if you don't want them to fire again. The timer statement catches a lot of folks out, switch it off immediately it has fired if you want it to be a one shot event.

Subs are self contained and don't know much about the outside world. If an event that has a branch label handler fires while you are in the sub it's going to fail. Switch off events going into a sub or ensure the event is handled by a sub itself. Best not to mix subs and [branching] to labels in GUI code.

Include:

Include is not too relevant to Liberty BASIC. For a start the system is complete in itself and needs no libraries attached. There are no restrictions on code size so there is no need to page in and out segments of code. Code banks are available that let you cut and paste blocks of reusable code directly to the program. The IDE allows you to jump about the program in a variety of ways, click the Jump To icon to see. Liberty also provides direct access to Windows API and third party dll files.

There are IDE's that provide include functionality if you really desire it.

Typing & Dimming variables:

If you are used to typing and defining variables forget it, this is automated within Liberty. Any number that has a fractional value beyond the decimal point is stored as a float, anything else is stored as an integer and they can be huge. Floats are precise to 16 significant figures and usually truncated when displayed. Optionally they can be viewed in full in the debugger (right click) or with the USING() command.

As in any computer language you have to be aware that comparing two float values is problematic, they are never really equal. > or
Variables initialise automatically to 0 or "" and don't need defined but you can if that's your habit. Arrays initiate automatically with 11 elements numbered 0 to 10 if you intend to use more than 11 elements you must dimension the array. Multi dimension arrays must be dimensioned and should be dimensioned at the start of the program. All arrays are global by default other variables may be specified as global if you wish.

Liberty can hold 70Mb of variable data at any one time. Beyond that you need to page data in and out from file.

Graphics:

Liberty has a drawing pen, if you don't have that Down you won't see any graphics. So set the pen down as the first step in any graphics operation.

Liberty has a Flush command that consolidates recent drawing operations into a segment. The segment resides in memory and can be redrawn instantly.
You must Flush if you want graphics to stick on screen and not be wiped out if another window passes over them. Be aware that Flush consumes memory, use it sparingly and delete any segments you don't need. You can use up your 70Mb of memory in a split second if you misuse the drawing commands.

Liberty can scroll, display, scale and print huge graphic resources. But Getbmp may only capture and save the visible screen as a .bmp.

Printing:

Dump only works with Lprint and is for line printing from the mainwin. Full high resolution graphics can be drawn and printed witht the graphics print command. You must issue a single Flush after all drawing if you want to print or scale graphics to the printer. The optional size argument is very powerful and spreads the number of pixels you state across the full width of the printed page.

The printer dialog is not able to change from the default printer.

Sorting:

Liberty sorts in Lexographic or Dictionary order. This is different from ASCII order which you may be more used to. It is possible to arrange for the sort to be in ASCII order just be aware of the difference.

Using():

If you use the Using(###.##) statement you will get a % in front of your number if it is too big to fit in the template.

Liberty Runtime Errors:

Tip, don't cry "bug”, you may feel foolish later :) try reading this and especially try running animate in the debugger.

Creating an .exe gives OS Error cannot find file

Start Liberty from the folder you installed it in to begin creating your .exe

Creating an .exe gives Problem Creating File

Don't create your .exe in Liberty's program folder, create it elsewhere.

Open file gives File System Access Denied

Either the file is already open, perhaps you forgot to close it, or it is in use by another program or you don't have Windows permission to access it.

Open file gives Error Opening File

The file is being written to by another program or, you don't have Windows permission to access it.

Open file, the file is literally invisible

you don't have Windows permission to access it.

You may need to change to a directory you do have access to or you may need to be an Administrator to gain permission to access the file.

Also be aware that Windows hides .extensions by default so the file may be named mytext.txt.txt when you see mytext.txt
use windows help system to show .extensions

Attempted to kill non-existant file

Using the Kill command you have named the file or path incorrectly or windows is denying you access.
Print the path and file name to the mainwin
to check.

Stack Overflow

This usually indicates that you have been jumping out of a programmed loop. Use exit for and exit while to leave loops early.

System Primitive Failed

You ran out of memory by exceeding the 70MB memory limit of LB.
Check your loops, you may be creating a very large number or string, which is bigger than the memory left.

Float invalid op

A function failed because of a negative value use abs(n) prior to sqr(n) or log(n)

Invalid branch label [xyz]

If it does exist you called it inside a sub that doesn't know where it is, a timer, mouse or keyboard event perhaps. Alternatively you have listed a [label] immediately after a [label]. Liberty expects code after a [label], don't pen another [label].

Index (nnn) is outside collection bounds

An index is a pointer to a position in an array. Indexes are used in array variables, in record pointers and by the system in bmps. Most often you will have asked for an array item that does not exist. So check what value is in array(index), array(index,index) or seek(index).

Sometimes this pops up for bmp files that are 32bit and have a transparency mask. Save your bmp from paint as a 24bit bmp.

Argument must be a collection

This is often associated with sending Liberty the wrong set of arguments in a command string. sending "refresh" to graphicbox for example or using $ values where numeric values are expected. Animate should stop on the offending line, double check the help file for the command and examine the values in the variables.

Collection is empty

Liberty expected some arguments but didn't get them, Print "!" to a textbox for example, Liberty expects arguments after the !command. Animate should stop on the offending line, double check the help file for the command.

IsEmpty

Liberty found itself without the information it needed. This is most often program flow.

Search for missing ends, end sub, end if, next. Practice indenting your code so that code blocks are easily visible. Make sure code does not run through to code blocks you didn't expect it to reach. Ensure Waits are placed where required.

Other reasons for this error are missing arguments or missing data, file access denial or Liberty finding Null info where it expected something.

This is sometimes the hardest to debug because program flow errors turn up anywhere but where the error was made. Use animate but don't assume the line shown is the cause.
Find the error.log and see what fragments of code are mentioned in it.

Start manual debugging, use step through a line at a time and make sure you are going where you expected.
Code in notices or use the run to break option in the debugger.
Hey debugging is part of the fun.

Up to date on info on everything that is discussed above is available in more detail within the Forums. Use the search features.

Protection violation

This is often associated with API calls and not having the correct data type specified in the Struct. Check the C definition of the API function.

If an API call fails randomly it is likely that a handle is of the wrong type. This can happen if HWND is used to obtain a handle that is subsequently typed as Long. The safest thing to do is type all handle variables as Ulong.

Lots of old API code uses Long, change this to Ulong and the problem will go away.

String can't hold string

You are using an UNC path (\\Server\Share) to access your program.
To make things work you must map the shared drive to a drive letter.

LB does not support UNC paths for the executables.