B

CBL Compiler

Lookout*Direct* automatically compiles source files when you open the `.lks` file instead of the `.L4P` file. There is always the possibility that an automatic compile might fail, so you can also use Lookout*Direct*'s CBL compiler to manually compile a Lookout*Direct* source file from a DOS command line.

If the source file has been corrupted, you can view the error file for compilation errors and either repair the source file or delete corrupted sections and rebuild your process from an intact foundation.

This appendix explains how to use the CBL command-line compiler to compile an `.lks` file into an `.l4p` file.

You can re-compile an `.lks` file in Lookout*Direct* by using **File»Open** and choosing Lookout*Direct* Source Files (`*.lks`) as the type of file to open.

When you save a process in Lookout*Direct*, Lookout*Direct* creates an `.l4p` file and an `.lks` file. The `.lks` file is a source code file that includes object definitions, names, I/O configuration, communications, control logic, control panel layout, and other parameters. All `.lks` files are standard ACSII text files that you can print or view in any text editor. The `.l4p` file is a compiled executable that contains complete configuration information for a particular process control application. An `.lks` file can be compiled into an `.l4p` file with the CBL compiler. You can also generate an `.lks` file by hand or modify an existing `.lks` file and use the compiler to produce an `.l4p` file.

CBL is a command-line compiler that takes three arguments: sourcefile, targetfile, and error log file. To use it, type the following at the command line in a DOS window, where *`file`* is the name of your file:

`cbl` *`file`*`.lks` *`file`*`.l4p` *`file`*`.err`

This command compiles the `.lks` file into the `.l4p` file and writes any error messages into the error file. If `file.err` is not included in the command line, errors will appear in the DOS window. If the compiler generates error messages, you need to debug your object or your `.lks` file.

You can directly edit your `.lks` file with any text editor. In general, if there is no simple repair to be made as indicated by an error message, you can often repair a file by deleting the problem object and recreating it in Lookout*Direct*.

CBL Compiler Error Messages

You can use the `.err` file to track down compilation errors.

The error file records errors one error per line, with a number at the beginning of each error message. This number corresponds to the line number of your `.lks` file where this error was encountered. You can go to these line numbers on your `.lks` file to correct the errors before recompiling your `.lks` file.

Here is a partial list of the error messages that you might encounter and tips on what might be causing them and how they can be corrected.

Class not found: *class_name*

The `.lks` file referred to a class that Lookout*Direct* does not know about (for which no corresponding `.cbx` file exists). It might be that the desired class name was spelled incorrectly, or it might be that the Lookout`.dat` file needs to be removed so that Lookout*Direct* loads the `.cbx` file that defines the class. For example, the line

```
Foo1 = new Foo ();
```

will produce the error

```
Class not found: Foo
```

Not a member of *object name*: *member_name*

The name used is not recognized as a valid member name for the object. For example, the line

```
Pot1.fred = 3;
```

produces the error

```
error: Not a member of 'Pot1': 'fred'
```

Member is not writable for class *class name*: *member_name*

The data member is read only. For example, the line

```
Modbus1.update = 3;
```

produces the error

```
error: Member is not writable for class 'Modbus':
'update'
```

Name is not valid

The name is not valid. For example, the line

```
Foo@ = new Pot (0, 100, 1, yes, 0);
```

produces the error

```
error: Tag is not valid: Foo@
```

Member is not readable for class *class_name*: *member_name*

The data member is write only. For example, the line

```
Modbus1.1 = Pot1.increment;
```

produces the error

```
error: Member is not readable for class 'Pot':
'increment'
```