Overview

How to Get Help

From the Help menu you can obtain help on the use of the OPEN Script Editor and on the OPEN Script language. Pressing F1 when the cursor is not on an OPEN Script language keyword in a script displays the contents of this help file.

If the cursor is on keyword in a script, pressing F1 displays information about that keyword.   You can also obtain help on keywords using the Right Mouse Button.

Editing

The OPEN Script Editor behaves in the standard Microsoft Windows fashion. Text entered from the keyboard or imported from a disk file can be selected, cut or copied to the clipboard, replaced by the contents of the clipboard, or cleared. Edited scripts may then be written back to the disk file.

Checking Your OPEN Script Program for Errors

The two flavors of errors that occur in OPEN Script programs are **Syntax** errors and **Execution** errors.

You can scan for syntax errors in the currently viewed OPEN Script program by clicking the ✓ icon on the toolbar or by using the **F6** function key. Lines containing syntax errors in your OPEN Script program will be highlighted in red. If there are syntax errors, you can scroll between them using the Edit Next Error or Edit Previous Error commands or the **Ctrl+N** and **Ctrl+P** keyboard accelerators.
Execution errors are corrected using the debugger.

Debugging

The debugger assists you in locating and correcting execution errors in your OPEN Script program. It allows you to slow or suspend execution of your program so that you can examine program flow and the contents of variables.

There are several ways to invoke the debugger:

- When in edit mode, use the step into button ( ) to debug the execution of the main procedure in the current buffer. Execution is suspended and the debugger is activated, highlighting the first line of the main procedure.
- Set breakpoints in the current buffer and begin execution by pressing the execute button ( ). Execution is suspended when one of the lines that contains a breakpoint is about to be executed. The debugger is activated and it highlights the line containing the breakpoint.
- When a program is executing and the Editor window is active, press the Pause ( ) button. Execution is suspended, the debugger is activated, and it highlights the line that is about to be executed.
- If an executing program encounters a runtime error, such as an unhandled OPEN Script error, Execution is suspended, the debugger is activated, and it highlights the line containing the error.

The Call Stack control displays the series of OPEN Script subroutine and function calls that got you to the current line. The Variable Window allows you to examine the contents of variables in the currently selected call frame.

Breakpoints

A breakpoint is a marker on a line of OPEN Script code that tells OPEN Script to suspend execution at that line so that the state of the program can be examined using the debugger. There are two ways to set breakpoints on lines of OPEN Script code when editing and there is an additional way to set a breakpoint when debugging.

When editing, place the cursor on the line of code on which you wish to set a breakpoint by clicking that line with the mouse or using the arrow keys. Then press the **F9** function key or the toolbar icon to toggle the breakpoint. The line's color will change to blue to indicate that when the program is run, OPEN Script will attempt to establish a breakpoint on that line. If the line already had a breakpoint on it, pressing **F9** or the toolbar icon will clear the breakpoint and the line's color will revert to black.

When debugging, clicking a line of OPEN Script code toggles a breakpoint on that line. Lines with breakpoints appear in blue while debugging.

Variable Window

The variable window is used to examine the contents of the variables associated with a particular call frame when debugging. Here is a description of the contents of the variable window:

```
+Globals
-TYPES
  +MType (NestedType)
-main
  i%: 0
  -s (NestedType)
    +tt (s2)
    s$:
    +t (s1)
    -tt1 (s2)
      +t (s1)
      i%: 0
      +t1 (s1)
    i%: 0
    +t1 (s1)
```

Lines in magenta indicate types or scopes that may be expanded or collapsed by pointing to them with the mouse and double-clicking. Those lines preceded with a **+** may be expanded, while those with a **-** may be collapsed.

In this example, there are three scopes: **Globals, TYPES** and **main** corresponding to the Global, Module and Procedural scopes of this call frame.

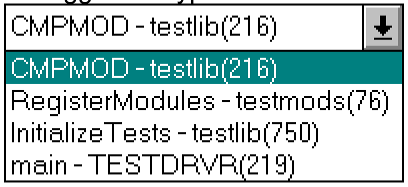
The **Globals** scope is not expanded, but when expanded the current global variables would be displayed.

The **TYPES** scope is the scope of module variables for this module (TYPES). There is one module variable in module TYPES: MType. MType is of type NestedType and is not expanded.

The procedure scope, **main** is expanded to list two local variables - i%, which is an integer and s, which is of type NestedType. The local variable s has been expanded to reveal six members: tt of type s2, s of type string, t of type s1, tt1 of type s2, and i of type integer.

Call Stack Control

The call stack control contains a list of OPEN Script subroutine and function calls that got you to the current line. The control is a drop-down list box which appears in the toolbar when you are running the debugger. A typical call stack control looks like this:



CMPMOD - testlib(216)
CMPMOD - testlib(216)
RegisterModules - testmods(76)
InitializeTests - testlib(750)
main - TESTDRVR(219)

This control indicates that we are currently suspended in the debugger in the OPEN Script routine *CMPMOD* on line 216 of the module *testlib*. *CMPMOD* was called by *RegisterModules* on line 76 of module *testmods*, etc. Selecting an entry in this list box will cause the current call frame to shift to that entry: the debugger window will display the line of code that made the call and the Variable Window will display the variables that are associated with the procedure and that made the call.

Controls

Here is a brief description of the controls of the OPEN Script Editor:

Icon	Accelerator	Action
🗋	Ctrl+N	Create a new script file.
📂	Ctrl+O	Open an existing script file.
💾	Ctrl+S	Save current script file to disk.
		Invoke the Dialog Editor to edit an OPEN Script dialog box.
	Ctrl+V	Paste current editor selection with the contents of the clipboard.
✂	Ctrl+X	Cut current editor selection and put it on the clipboard.
	Ctrl+C	Copy current editor selection to the clipboard.
↶	Ctrl+Z	Undo the previous editor action.
🔍	Ctrl+F	Find text in the editor.
	F3	Find next occurrence of last text searched for.
	Shift+F3	Find previous occurrence.
	Alt+F3	Find and replace text.
🗋	F6	Check the syntax of the current OPEN Script program.
🗋	F5	Execute the main procedure of the current OPEN Script program if editing, or continue executing the current OPEN Script program if debugging.
▶	Ctrl+F5	Execute in Animate mode.
🗋	Esc Ctrl+Break	Pause the currently executing OPEN Script program and activate the debugger on the current line of execution.
■	Esc (twice) Ctrl+Break (twice)	Stop executing current OPEN Script program. If you were debugging the program, the debugging session is ended.
	F10	Step over current line in the debugger. If the current line is a call to an OPEN Script subroutine or function, the debugger stops at the next line in the current procedure.
🗋	F8	Step into current line in the debugger or begin debugging current file at main procedure.
	Ctrl+F8	Step out of current procedure in the debugger.
	F7	Set a temporary breakpoint on the current line. Execution continues until a breakpoint is encountered or the program finishes. When either of these occurs, the temporary breakpoint is cleared.
🗋	F9	Toggle a Breakpoint on the current line.
	Shift+F2	Toggle the Console window.
🗋	F2	Toggle the Variable window.
	Ctrl+N	Scroll edit window to next error returned from a syntax check.
	Ctrl+P	Scroll edit window to previous error returned from a syntax check.

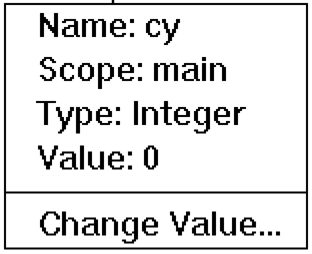
Also see the discussion on the Right Mouse Button and the Variable Window for other OPEN Script

Editor controls.

The Right Mouse Button

The right mouse button is used in the OPEN Script Editor to obtain help on OPEN Script language keywords.   Clicking an OPEN Script language keyword (such as **sub**) in a script displays the online help associated with that keyword.

While debugging, the right mouse button can be used to examine and modify the contents of variables in the running progrm.   Clicking a variable with the right mouse button creates a pop-up menu near the mouse's position:



This popup menu describes an integer variable in the procedure **main**, whose value is zero.   Clicking the **Change Value...** presents a dialog box that allows you to change **cy**'s value.

Using the right mouse button in the edit window allows you to modify the value of variables of string, integer, single, and double.   In the variable window, you can modify the value of any expanded field that is of type string, integer, single, or double.   Application Datatype fields cannot be modified.

Console Window

The console window is used to display the output of the OPEN Script print statement. It is a scrollable text display that appears at the bottom of the OPEN Script Editor window and remembers 100 lines of printed output.

Animate Mode

Animate mode causes the debugger to do repeated step into procedures.   It is useful for visually examining the flow of an OPEN Script program.

Step Into

Step into is used to execute one line of code in the debugger. If the current line is a call to an OPEN Script subroutine or function, the debugger stops at the first line of that function. Otherwise, the debugger stops at the next line of the current procedure.

Call Frame

A call frame is an entry on the Call Stack.   Changing the current call frame using the call stack control causes the debugger window to view and highlight the line of code associated with the call frame and causes the variable window to display the variables associated with the call frame.

Dialog Editor

With the Dialog Editor you can graphically edit the layout of the dialog boxes you create in the OPEN Script Editor.   You can invoke the Dialog Editor with the Edit Dialog command or the Edit a Dialog Box toolbar button.