10. BREAKPACKAGE

The Break Package is a part of Interlisp that makes debugging your programs much easier.

Break Windows

A break is a function either called by the programmer or by the system when an error has occurred. A separate window opens for each break. This window works much like the Executive Window, except for extra menus unique to a break window. Inside a break window, you can examine variables, look at the call stack at the time of the break, or call the editor. Each successive break opens a new window, where you can execute functions without disturbing the original system stack. These windows disappear when you resolve the break and return to a higher level.

Break Package Example

This example illustrates the basic break package functions. A more complete explanation of the breaking functions, and the break package will follow.

The correct definition of FACTORIAL is:

```
(defun factorial (x)
     (if (zerop x)
          1
         (* x (factorial (1- x)))))
```

To demonstrate the break package, we have edited in an error: DUMMY in the IF statement is an unbound atom, it lacks a value.

```
((defun factorial (x)
     (if (zerop x)
          dummy
         (* x (factorial (1- x)))))
```

The evaluated function

```
(FACTORIAL 4)
```

should return 24, but the above function has an error. DUMMY is an unbound atom, an atom without an assigned value, so Lisp will "break". A break window appears (Figure 10-1), that has all the functionality of the typing lisp expressions into the Executive Window (The top level), in addition to the break menu functions. Each consecutive break will move to another level "down".

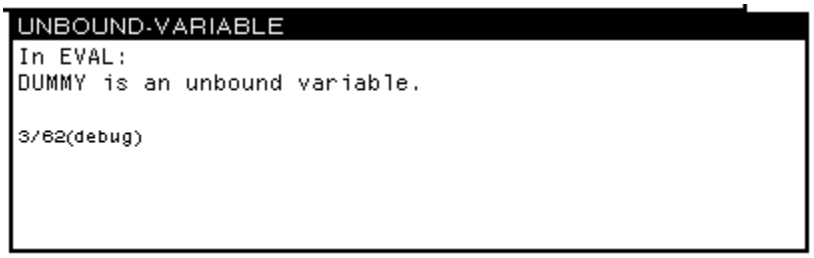

Figure 10-1. Break Window

Move the mouse cursor into the break window and hold down the middle mouse button. The Break Menu will appear. Choose BT. Another menu, called the stack menu, will appear beside the break window. Choosing stack items from this menu will display another window. This window displays the function's local variable bindings, or values (see Figure 10-2). This new window, titled FACTORlAL Frame, is an inspector window (see inspector Chapter 17).

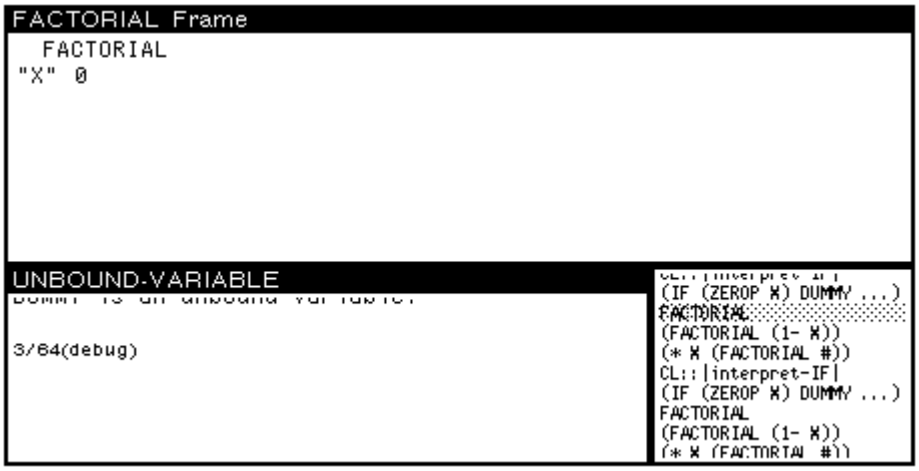

Figure 10-2. Back Trace of the System Stack

From the break window, you can call the editor for the function `FACTORIAL` by `middle-buttoning on the word FACTORIAL and selecting DisplayEdit from the menu that pops up.`

Replace the unbound atom `DUMMY` with 1. Exit the editor .

The function is fixed, and you can restart it from the last call on the stack. (It does not have to be started again from the Top Level.) To begin again from the last call on the stack, choose the last (top) `FACTORIAL` call in the BT menu. Select `REVERT` from the middle button break window, or type it into the window. The break window will close, and a new one will appear with the message: **`Breakpoint at FACTORIAL`**

To start execution with this last call to `FACTORIAL`, choose OK from the middle button break menu. The break window will disappear, and the correct answer, 24, will be returned to the top level.

Ways to Stop Execution from the Keyboard (Breaking Lisp)

There are ways you can stop execution from the keyboard. They differ in terms of how much of the current operating state is saved:

Control-G — Provides you with a menu of processes to interrupt. Your process will usually be "EXEC". Choose it to break your process. A break window will then appear.

Control-B — Causes your function to break, saves the stack, then displays a break window with all the usual break functions. For information on other interrupt characcers, see Chapter 30 in the *IRM*.

Break Menu

Move the mouse cursor into the break window. Hold the middle button down, and a new menu will pop up, like the one in Figure 10-3.

Figure 10-3. Middle Button Menu in Break window

Five of the selections are particularly important when just starting to use Medley:

BT — Back Trace displays the stack in a menu beside the break window. Back Trace is a very powerful debugging tool. Each function call is placed on the stack and removed when the execution of that function is complete. Choosing an item on the stack will open another window displaying that item's local variables and their bindings. This is an inspector window that offers all the power of the inspector. (For details, see the section on the Inspector, Chapter 17.)

? = — Before you use this menu option, display the stack by choosing BT from this menu, and choose a function from it. Now, choose ?=. It will display the current values of the arguments to the function that has been chosen from the stack.

↑ — Move back to the previous break window, or if there is no other break window, back to the top level, the Executive Window.

REVERT — Move the point of execution back to a specified function call before the error. The function to revert back to is, by default, the last function call before the break. If, however, a different function call is chosen on the BT menu, revert will go back to the start of this function and open a new break window. The items on the stack above the new starting place will no longer exist. This is used in the tutorial example (see the Break Package Example section above).

`OK` Continue execution from the point of the break. This is useful if you have a simple error, i.e., an unbound variable or a nonnumeric argument to an arithmetic function. Reset the variable in the break window, then select OK. (see the Break Package Example section above).

In addition to being available on the middle button menu of the break window, all of these functions can be typed directly into the window. Only BT behaves differently when typed. It types the stack into the trace window instead of opening a new window.)

Returning to Top Level

Typing Control-D will immediately take you to the top level from any break window. The functions called before the break will stop, but any side effects of the function that occurred before the break remain. For example, if a function set a global variable before it broke, the variable will still be set after typing Control-D.