```
Function ShowFileInfo(pathName, fileName)
   String pathName      // Name of symbolic path or "" for dialog.
   String fileName      // File name or "" for dialog.

   Variable refNum

   Open /D=2 /R /P=$pathName refNum as fileName    // Sets S_fileName

   if (strlen(S_fileName) == 0)
      Print "ShowFileInfo was canceled"
   else
      String fullPath = S_fileName
      Print fullPath
      Open /R refNum as fullPath
      FStatus refNum    // Sets S_info
      Print S_info
      Close refNum
   endif
End
```

In this case, we wanted to open the file for reading. To create a file and open it for writing, omit /R from both calls to Open.

Pause For User

The **PauseForUser** operation (see page V-738) allows an advanced programmer to create a more sophisticated semi-modal user interface. When you invoke it from a procedure, Igor suspends procedure execution and the user can interact with graph, table or control panel windows using the mouse or keyboard. Execution continues when the user kills the main window specified in the PauseForUser command.

PauseForUser is fragile because it attempts to create a semi-modal mode which is not supported by the operating system. Before using it, consider using an **Alternative to PauseForUser**.

Uses of PauseForUser

Pausing execution can serve two purposes. First, the programmer can pause function execution so that the user can, for example, adjust cursors in a graph window before continuing with a curve fit. In this application, the programmer creates a control panel with a continue button that the user presses after adjusting the cursors in the target graph. Pressing the continue button kills the host control panel (see example below).

In the second application, the programmer may wish to obtain input from the user in a more sophisticated manner than can be done using DoPrompt commands. This method uses a control panel as the main window with no optional target window. It is similar to the control panel technique shown above, except that it is modal.

Following are some examples of how you can use the **PauseForUser** operation (see page V-738) in your own user functions.

PauseForUser Simple Cursor Example

This example shows how to allow the user to adjust cursors on a graph while a procedure is executing. Most of the work is done by the UserCursorAdjust function. UserCursorAdjust is called by the Demo function which first creates a graph and shows the cursor info panel.

This example illustrates two modes of PauseForUser. When called with autoAbortSecs=0, UserCursorAdjust calls PauseForUser without the /C flag in which case PauseForUser retains control until the user clicks the Continue button.

When called with autoAbortSecs>0, UserCursorAdjust calls PauseForUser/C. This causes PauseForUser to handle any pending events and then return to the calling procedure. The procedure checks the V_flag variable, set by PauseForUser, to determine when the user has finished interacting with the graph. PauseForUser/C, which requires Igor Pro 6.1 or later, is for situations where you want to do something while the user interacts with the graph.

To try this yourself, copy and paste all three routines below into the procedure window of a new experiment and then run the Demo function with a value of 0 and again with a value such as 30.

```
Function UserCursorAdjust(graphName,autoAbortSecs)
   String graphName
   Variable autoAbortSecs

   DoWindow/F $graphName                  // Bring graph to front
   if (V_Flag == 0)                       // Verify that graph exists
      Abort "UserCursorAdjust: No such graph."
      return -1
   endif

   NewPanel /K=2 /W=(187,368,437,531) as "Pause for Cursor"
   DoWindow/C tmp_PauseforCursor          // Set to an unlikely name
   AutoPositionWindow/E/M=1/R=$graphName  // Put panel near the graph

   DrawText 21,20,"Adjust the cursors and then"
   DrawText 21,40,"Click Continue."
   Button button0,pos={80,58},size={92,20},title="Continue"
   Button button0,proc=UserCursorAdjust_ContButtonProc
   Variable didAbort= 0
   if( autoAbortSecs == 0 )
      PauseForUser tmp_PauseforCursor,$graphName
   else
      SetDrawEnv textyjust= 1
      DrawText 162,103,"sec"
      SetVariable sv0,pos={48,97},size={107,15},title="Aborting in "
      SetVariable sv0,limits={-inf,inf,0},value= _NUM:10
      Variable td= 10,newTd
      Variable t0= ticks
      Do
         newTd= autoAbortSecs - round((ticks-t0)/60)
         if( td != newTd )
            td= newTd
            SetVariable sv0,value= _NUM:newTd,win=tmp_PauseforCursor
            if( td <= 10 )
               SetVariable sv0,valueColor= (65535,0,0),win=tmp_PauseforCursor
            endif
         endif
         if( td <= 0 )
            KillWindow tmp_PauseforCursor
            didAbort= 1
            break
         endif

         PauseForUser/C tmp_PauseforCursor,$graphName
      while(V_flag)
   endif
   return didAbort
End

Function UserCursorAdjust_ContButtonProc(ctrlName) : ButtonControl
   String ctrlName

   KillWindow/Z tmp_PauseforCursor           // Kill panel
End

Function Demo(autoAbortSecs)
```