

Easy Host File Selection Via Dialog Box for the SAS® System, Version 6.08, for Windows™

Randy C. Finch, Tennessee Valley Authority, Muscle Shoals, AL

ABSTRACT

The SAS System, Version 6.08, for Windows provides no mechanism for easily selecting a host file using a selection list window (standard SAS/AF® method) or a dialog box (standard Windows method). The Microsoft® Visual Basic™ Programming System for Windows (VB) provides a Common Dialog control that makes its very easy to add a Windows file selection dialog box to a SAS/AF application. A VB program that displays a file dialog box can be called from a SAS/AF application. After selecting a file, the path and filename are passed back to the application via a file on a hard disk or RAM drive. The path and filename can then be used by the SAS/AF application to access the chosen host file. The SAS System, Version 6.10, for Windows has built in commands for creating file dialog boxes.

INTRODUCTION

Shortly after receiving the SAS System, Version 6.08, for Windows, I began updating some old SAS programs, giving them an (inter)face lift using SAS/AF and SAS Screen Control Language (SCL). One feature I needed in the updated programs was the ability to select an ASCII file from the host file system via a Windows-style dialog box. Ultimately, this file was to be converted to a SAS data set. After scanning the SCL manual, I discovered that there were many "list" functions available such as CATLIST, FILELIST, etc., but none applicable to the host file system. Since SAS itself uses a Windows file dialog box for its Program Editor and other windows, I thought there might be a way to tap into it. However, I was told by SAS technical support that there was no way to do this. Therefore, I set out to find a way to create my own file dialog box.

The first thing that came to mind was to use Visual Basic. The standard and professional editions of Visual Basic Version 3.0 have a dialog box control that makes it easy to create file dialog boxes. I had a copy of the older Visual Basic Professional Edition Version 2.0 which also has the dialog box control available (the standard edition of Version 2.0 does not have the control). I decided to give it a try.

THE SAS/AF INTERFACE

First, I created the SAS interface for my program (see Figure 1). It is quite simple, having entry fields for the ASCII filename and its host path. There is a push-button to the right of the filename field that activates the VB file dialog box program. There are also three push-buttons at the bottom of the window: one to convert the ASCII file to a SAS data set, one to allow viewing of the ASCII file via the Notepad window, and another to exit the program. The display information and SCL source code for the interface is shown in Listing 1.

Notice the first statement after the IF statement at the beginning of section MAIN. Here, a Windows program named FILEDLG, created with VB, is executed by the SCL SYSTEM function. Under the Windows environment, the SYSTEM function (or the equivalent X command in SAS) can execute either a DOS or a Windows program.

This is nice in that older programs written for DOS can still execute without change, but the added functionality of executing Windows programs is available.

THE VISUAL BASIC INTERFACE AND CODE

VB is a programming tool that allows you to design an application interface graphically and then write code to add functionality. This is generally called event driven programming because the program modules are only executed when some event, such as a button being selected or a timer expiring, occurs. SAS/AF works in a similar way but the interface is not as easy to create and modify as it is in the VB environment (see Figure 2).

VB works with forms, which is nothing more than user-defined windows. Since I only wanted to use VB to display a file dialog box, get a file selection, and then pass this information back to the SAS/AF program, I just used the default form, reduced in size, that is available when VB loads. VB has available an interface control called a Common Dialog control. By double-clicking on the Common Dialog control icon in the toolbox to the left of the VB screen, this control is inserted into the default form. The Common Dialog control can be used for opening and saving files, selecting colors and fonts, printing, and displaying help files. The particular function of the control is determined by the Action property of the control which is set at run time. Some properties, such as the dialog box title and the initial host directory to display, can be set at design or run time.

When the SAS application invokes the VB program, the default form, which is the only form in the program, loads. This causes the Form_Load event to occur. By writing code for this event, the file selection dialog box can be displayed. Listing 2 shows the short piece of code that is needed to accomplish this. First, the Common Dialog control's Action property is set to 1 which causes it to function as a File Open Dialog Box. At this point the dialog box appears (see Figure 3) and waits for a file to be selected. Then control returns to the VB code. The name of the selected file, including its path, resides in the Common Dialog control's Filename property. If the Cancel button is selected on the dialog box, the Filename property will be a null string. The filename is assigned to the variable FName. A file named FName.TXT is opened on drive F:, which is a RAM drive on my computer, and the name of the selected file, FName, is written to it. Finally, the file is closed, and the VB program terminates, returning control to the SAS/AF program.

THE SAS/AF SCL CODE

Looking back at the SCL code in Listing 1, you will see that after the SYSTEM function is executed, the file on drive F: is opened and the text in it is assigned to the string variable INSTR using the FGET function. If this is a null string, then no file was selected, and no further action is taken. If, however, it is not a null string, then the path and filename are extracted. This is accomplished by reversing string INSTR, finding the position of the first backslash in the reversed string,

and then using this information to split INSTR into the path portion and the filename portion. These sub-strings are assigned to the input fields RAWPATH and RAWFILE, respectively.

When the CONVERT or VIEW push-button is selected, the RAWPATH and RAWFILE entries are combined to create a complete filename including the path (see the appropriate sections of Listing 1). Note that the path has to be checked for an ending colon or backslash in order to determine if a backslash needs to be added between the two values.

For my program, I always wanted to display the same initial host directory in the dialog box. If, however, you need the SAS/AF program to control the initial directory, you can write the path name to the RAM drive within the SCL code and let the VB program read this information. Before displaying the dialog box, the path can be assigned to the InitDir property of the Common Dialog control.

THE SAS SYSTEM, VERSION 6.10, FOR WINDOWS

If you have made the move to the SAS System, Version 6.10, for Windows, you will find that there is a much easier way to produce a file dialog box. You can simply use the new function, FILEDIALOG. An Open or SaveAs dialog box can be generated as follows:

```
FN=FILEDIALOG("OPEN");
FN=FILEDIALOG("SAVEAS");
```

This is obviously a better and easier way to generate file dialog boxes, and I am pleased to see it added to the SAS System.

CONCLUSION

Visual Basic provides an easy way for SAS programmers to add a host file selection dialog box to SAS/AF applications. Other similar visually oriented programming tools can probably provide equivalent functionality. However, if you really want to make your life easy, get the SAS System, Version 6.10, for Windows and use the FILEDIALOG function. If you already have this version, I hope that this paper will help you think about ways to overcome the limitations you have found in SAS.

SAS and SAS/AF are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

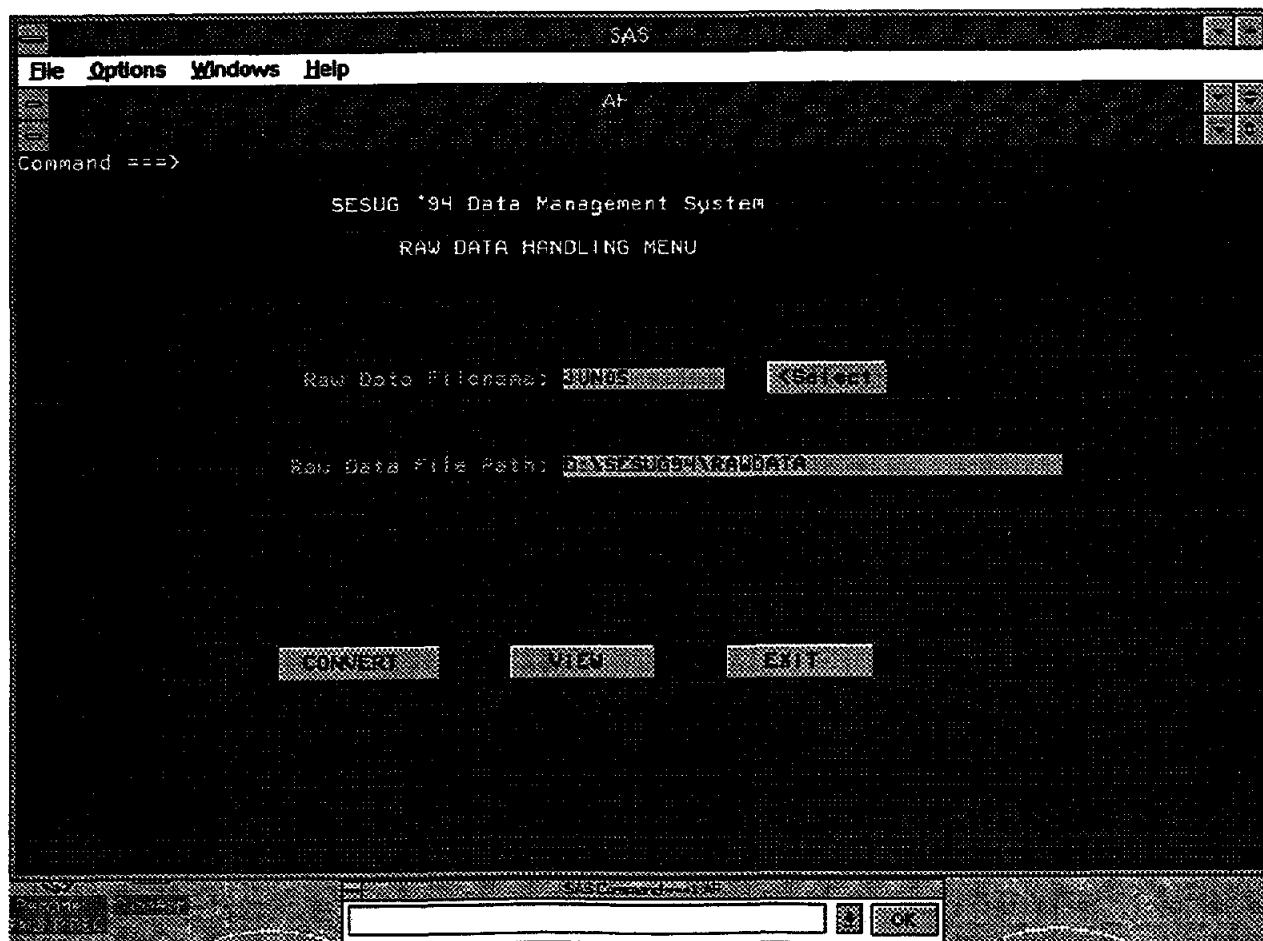


Figure 1. SAS/AF Application Interface

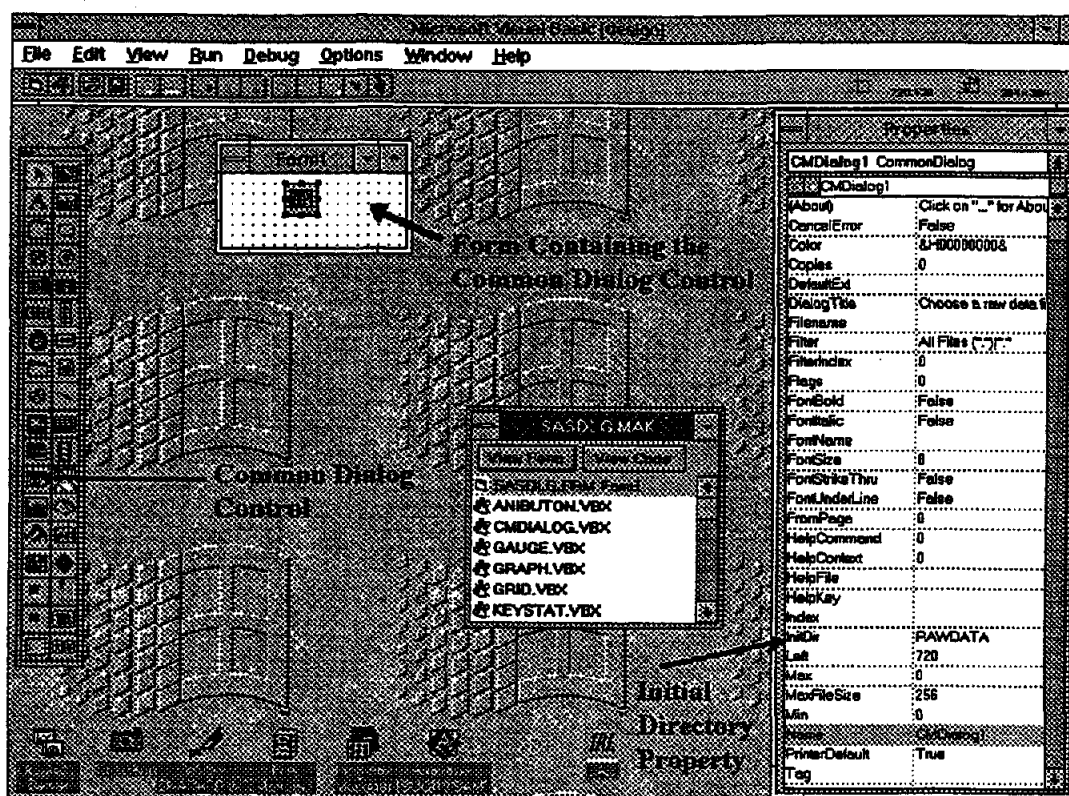


Figure 2. Visual Basic Interface

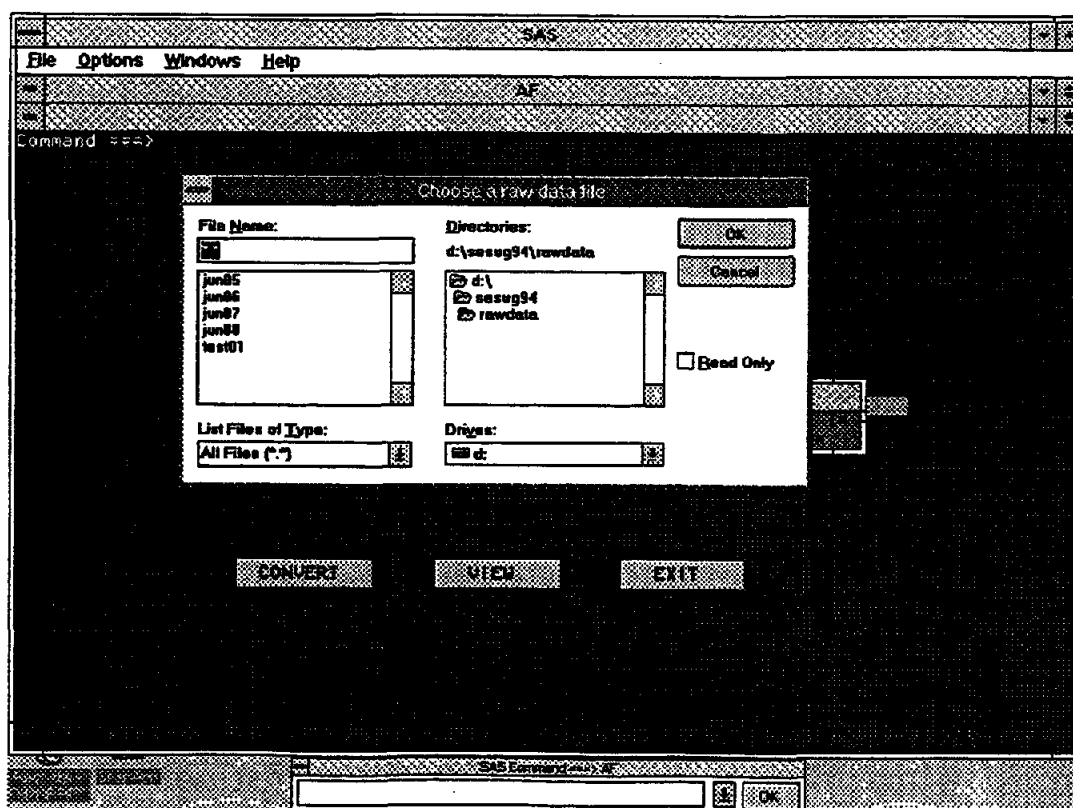


Figure 3. SAS/AF Application with File Dialog Box Displayed

Listing 1. SAS/AF Display Information and SCL Program Listing

***** DISPLAY *****

```

+-----+
|                                     |
|               SESUG '94 Data Management System               |
|               RAW DATA HANDLING MENU                         |
|                                     |
|               Raw Data Filename: &RAWFILE_____ &RAWSEL     |
|                                     |
|               Raw Data File Path: &RAWPATH_____             |
|                                     |
|               &CONVERT___ &VIEW___ &EXIT___                 |
|                                     |
+-----+

```

***** SOURCE *****

INIT:

```

/* SET LOGICAL TRUE AND FALSE IN CASE THEY ARE NEEDED */
TRUE=1;
FALSE=0;

```

```

/* SET MESSAGE COLOR TO PINK. IT SHOWS UP BETTER THAN RED */
CALL EXECCMD('COLOR MESSAGE PINK');

```

```

/* SET SETCR TO EXECUTE MAIN ON 'ENTER' EVEN IF FIELD UNMODIFIED */
CALL SETCR('STAY', 'RETURN');

```

RETURN;

MAIN:

/* THE USER WANTS TO LOOK AT A LIST OF RAW DATA FILES */

```

IF ACTION = 'RAWSEL' THEN DO;
  RC=SYSTEM('FILEDLG');
  RC=FILENAME('FN', 'F:\FNAME.TXT');
  FID=FOPEN('FN');
  IF (FID>0) THEN DO;
    IF FREAD(FID)=0 THEN DO;
      RC=FGET(FID, INSTR);
      RC=FCLOSE(FID);
      IF INSTR^='' THEN DO;
        RINSTR=REVERSE(INSTR);
        RPOS=INDEX(RINSTR, '\');
        IF RPOS^=0 THEN DO;
          POS=LENGTH(INSTR)-RPOS+1;
          RAWPATH=SUBSTR(INSTR, 1, POS-1);
          RAWFILE=SUBSTR(INSTR, POS+1);
        END;
      END;
    END;
  ELSE DO;
    RC=FCLOSE(FID);
  END;
  ACTION='';
END;

```

```

/* THE USER WANTS TO CONVERT THE RAW FILE TO A SAS DATA SET */
ELSE IF ACTION = 'CONVERT' THEN DO;
  /* GET THE PATH AND FILE COMBO */
  ENDING=SUBSTR(RAWPATH, LENGTH(RAWPATH), 1);
  IF RAWPATH='' OR ENDING='.' OR ENDING='\ ' THEN PATH=RAWPATH;
  ELSE PATH=RAWPATH||'\';
  RAWDATA=PATH||RAWFILE;

```

```

/* MAKE SURE RAWDATA FILE EXISTS. IF NOT, PRINT MESSAGE */
IF FILEEXIST(RAWDATA) THEN DO;
  _MSG_='Please wait while converting '||RAWFILE;
  REFRESH;
  /* EXECUTE SAS CODE TO CONVERT RAW DATA FILE TO A SAS DATA SET */
  SUBMIT CONTINUE;
    %LET FNAME=&RAWFILE;
    FILENAME RAWDATA '&RAWDATA';
    %INCLUDE 'READRAW.SAS';
  ENDSUBMIT;
  ALARM;
  _MSG_='Conversion complete!';
END;
ELSE DO; /* ERROR MESSAGE */
  ALARM;
  _MSG_='File '||PATH||RAWFILE||' does not exist';
END;
ACTION='';
END;

/* THE USER WANTS TO VIEW THE RAW DATA FILE */
ELSE IF ACTION = 'VIEW' THEN DO;
  /* GET PATH AND FILENAME */
  ENDING=SUBSTR(RAWPATH,LENGTH(RAWPATH),1);
  IF RAWPATH='' OR ENDING=':' OR ENDING='\' THEN PATH=RAWPATH;
  ELSE PATH=RAWPATH||'\';
  /* MAKE SURE FILE EXISTS BEFORE DISPLAYING */
  IF FILEEXIST(PATH||RAWFILE) THEN DO;
    VIEWCMD='NOTEPAD; CLEAR; ZOOM; INC '||PATH||RAWFILE||'';
    CALL EXECCMD(VIEWCMD);
  END;
  ELSE DO; /* ERROR MESSAGE */
    ALARM;
    _MSG_='File '||PATH||RAWFILE||' does not exist';
  END;
  ACTION='';
END;

/* THE USER WANTS TO EXIT THE RAW DATA MENU */
ELSE IF ACTION = 'EXIT' THEN DO;
  CALL EXECCMD('END');
  ACTION='';
END;

/* THE USER PRESSED ENTER IN THE RAWFILE FIELD */
ELSE IF CURFLD()='RAWFILE' THEN DO;
  CURSOR RAWPATH;
END;

/* THE USER PRESSED ENTER IN THE RAWPATH FIELD */
ELSE IF CURFLD()='RAWPATH' THEN DO;
  CURSOR RAWFILE;
END;

RETURN;

TERM:
RETURN;

```

Listing 2. Visual Basic Code for the Form_Load Event

```

Sub Form_Load ()
  CMDialog1.Action = 1
  FName = CMDialog1.Filename
  Open 'F:\FNAME.TXT' For Output As #1
  Print #1, FName
  Close #1
End
End Sub

```