# FILBR$ - Display File Browse Dialogue

The FILBR$ routine is available to generate a File Browse dialogue box. This function can be used by both Speedbase and Cobol programs.

## 1. Invocation

This routine is called as follows:

```
CALL FILBR$ USING fiblock  filefilter
```

where fiblock is a control block of the following format:

```
01    FI
  02  FIFLG1      PIC 9(2) COMP     * 1st flag byte
                                     * 0 = "Open" file dialogue box
                                     * 1 = "Save as" file dialogue box
  02  FIFLG2      PIC 9(2) COMP     * 2nd flag byte
                                     * 0 = Return full path name
                                     * 1 = Return file name
                                     * 2 = Return directory name
                                     * 3 = Return file name, <CR>,
                                     * and <directory>
  02  FILENG      PIC 9(4) COMP     * Length if file filter text
```

and filefilter is the text used as the file type filter which must be of the format:

```
descriptive text (*.ext)
```

For example, "Text files (*.txt)", "CSV files (*.csv)" etc. Note that the supplied file extension is used as the default file extension.

## 2. STOP Codes and Exception Conditions

No STOP codes are generated by FILBR$.

The following EXIT codes may be returned by FILBR$:

| EXIT code | $$COND | Description |
|---|---|---|
| 24902 | 2 | FILBR$ has been called on a client that is not running GX (or GSMWIN32.EXE). |

## 3. Programming Notes

FILBR$ sends a control block to GX which results in the display of a standard File Dialogue window. The selected filename/pathname string returned to GX by the File Dialogue window is transmitted back to the server so the FILBR$ call must be followed by an immediate ACCEPT operation (or 2 consecutive ACCEPT operations, if FIFLG2 = 3).

The file extension is only returned as part of the reply string if FIFLG2 is set to 0. If FIFLG2 is set to either 1 or 3 the filename **without** the file extension is returned.

Note that length of the file type filter text must be supplied in FILENG and must be nonzero i.e. a zero-terminated text-string cannot be supplied. This is, perhaps, a shortcoming of FILBR$.

## 4. Examples

```
FRAME FILBR "File Browse Demo"
************************************************************************
*                                                                      *
*                         FILE BROWSE SUBROUTINE                       *
*                         ======================                       *
*                                                                      *
* This routine allows a speedbase program to call the windows file     *
* browse dialogue box.                                                 *
*                                                                      *
************************************************************************

DATA DIVISION

77 FILE-FILTER  PIC X(17)
                VALUE "Text file (*.txt)"

01 FI-BLOCK
   02 FIFLG1    PIC 9(2) COMP   * 0 = "Open" file dialogue box
                                * 1 = "Save As" file dialogue box
   02 FIFLG2    PIC 9(2) COMP   * 0 = Return full path name
                                * 1 = Return file name
                                * 2 = Return directory name
                                * 3 = Return file name, <CR>, directory
   02 FILENG    PIC 9(4) COMP   * Length of file filter text
                VALUE 17

WINDOW W1
EDT
SEQUENCE EXIT CLW, EXIT CLW
BASE AT  7 14
02 02 "[ File browse                                    ]"
04 03 "File"
                04 08 W1PATH      X(40)        UF1 NUL
                05 09 W1NULL      X(0)         NUL
ROUTINES SECTION

V-W1PATH.
        IF $FUNC = 1
           CALL FILBR$ USING FI-BLOCK FILE-FILTER
           ON EXCEPTION EXIT WITH 1
           CALL ECHO
        END
EXIT
ENDWINDOW

LOAD DIVISION
        CALL TITLES USING W1
EXIT

ENDFRAME
ENDSOURCE
```

# 5. Copy-Books

See copy-book "F$" in copy-library S.SYS32. Note that this copy-book **MUST** be expanded using a SUBSTITUTING clause. For example:

```
      COPY "F$" USING "FI"
```

# 6. See Also

None.