

14. DIALOG File

DIALOG FILE: USER MESSAGES

Introduction

The VA FileMan DIALOG file is used to store dialog that would normally appear on a screen during interaction with a user. This dialog may include error messages, user help, and other types of prompts. FileMan distributes a set of entries in the DIALOG file.

The VA FileMan calls, `BLD^DIALOG` or `$$EZBLD^DIALOG`, are used to move text from the DIALOG file into arrays. The text can then be displayed using the display mode of choice.

Developers may add entries to the DIALOG file. Entries such as error messages, help messages and other general prompts can be placed in the file. The DIALOG file should not be used for storing alternate synonyms either for data or for fields in the data dictionary such as field labels or descriptions.

Note: If you wish to add entries to the DIALOG file, you must apply to the DataBase Administrator for a numberspace.

Advantages of the DIALOG file for user interaction are:

- User interaction can be easily separated from the other program functionality, a necessary step in creating alternate interfaces to roll-and-scroll, such as GUI.
- Text stored in the DIALOG File can be re-used.
- Package error lists can be identified and listed by error number in documentation.
- Text can be returned in multiple languages without changes to developers' code. (See "Internationalization" section of the "DIALOG File" chapter in this manual.

Use of the DIALOG File

VA FileMan controls and distributes entries in the DIALOG file in the number range 0 through 10000. These entries should not be edited by other package developers, with the exception of adding foreign language equivalents for text (see the "Internationalization and the Dialog File" section of the "DIALOG File" chapter in this manual for details). Some of the FileMan error messages are available for retrieval by other package developers, using the FileMan program calls. These messages are listed in the "Error Codes" appendix in this manual. Entries within

the FileMan number range that are not in the Error Codes listing should not be used as they are subject to change.

Other packages may make entries in the DIALOG file for their own use. The VHA Database Administrator will assign number ranges to a subscribing package.

If your package or site already has a file numberspace assigned by the DBA, you can use that number (or numbers) multiplied by 10000 (plus any decimal value between .001 and .999) for adding entries to the DIALOG file (e.g., Kernel owns file 200, so they can use numbers 2000000 through 2000000.999. If I'm site 665, I own numberspace 665000 for files, so I can use 6,650,000,000 through 6,650,000,000.999 in the DIALOG file).

If developers do not follow these guidelines, their DIALOG entries may be overwritten when new packages are installed. Note that an entry number does not have to be an integer--up to 3 decimal places can be used to identify an entry.

Creating DIALOG File Entries

Developers may enter or edit entries to the DIALOG file using VA FileMan Enter/Edit. The only required fields are the DIALOG NUMBER which uniquely identifies the entry, the TYPE (Error, Help or General Message), and the dialog TEXT.

Dialog text can contain parameter windows delimited by vertical bars. Within a pair of vertical bars, the developer puts a value that will correspond to a subscript in a parameter list. This subscript need not be numeric, but may be meaningful alpha characters such as "FIELD". When the dialog text with windows is retrieved using a call to either BLD^DIALOG or \$\$EZBLD^DIALOG, a subscripted parameter list is input to the call. The parameters are matched by subscript to the windows in the text, and the values from the parameter list are inserted into the corresponding windows in the text. If parameters are included in the text, the INTERNAL PARAMETERS NEEDED field should be set to YES. A multiple field called PARAMETER is used in documenting these parameters.

For error messages only, a list of output parameters can also be passed to BLD^DIALOG or \$\$EZBLD^DIALOG. This list is returned by the routine in a standard format. Output parameters might be, for example, file or field numbers which the calling routine may then use to make a decision. Output parameters should also be documented in the PARAMETER multiple described immediately above.

Another important optional field is the POST-MESSAGE ACTION field. If the developer wishes to perform some special action whenever a message is retrieved, M

code is simply inserted into this field. The code will then be executed whenever the associated message is retrieved with a call to BLD^DIALOG or \$\$EZBLD^DIALOG.

The TRANSLATION (LANGUAGE) multiple in the DIALOG file allows a developer to enter text in a language other than English. See the "Internationalization and the DIALOG File" section of the "DIALOG File" chapter in this manual for additional information on this feature.

Finally, there is a place to enter documentation for the ROUTINE names and LINE TAGS which use the dialog entries. This is optional internal documentation for the use by developers only.

Following is an example creating a new entry in the DIALOG file.

```
Select DIALOG: 10001
  Are you adding '10001' as a new DIALOG (the 239TH)? Y <RET> (YES)
TYPE: ?
  Enter code that reflects how this dialogue is used when talking to
the users.
  Choose from:
    1      ERROR
    2      GENERAL MESSAGE
    3      HELP
TYPE: 3 <RET> HELP
PACKAGE: VA FILEMAN <RET>          DI
DESCRIPTION:
  1>Here we enter a description of the help message itself. This
  2>description is for our own documentation.
  3>
EDIT Option: <RET>
INTERNAL PARAMETERS NEEDED: Y <RET> YES
TEXT:
  1>Here we enter the actual text of the help messages, with
  2>parameters designated by vertical bars |1| as shown.
  3>
EDIT Option: <RET>
Select PARAMETER SUBSCRIPT: 1
  Are you adding '1' as a new PARAMETER SUBSCRIPT (the 1ST for this
DIALOG)? Y <RET> (YES)
  PARAMETER DESCRIPTION: Brief description of parameter 1 goes here. For
  documentation only.
Select PARAMETER SUBSCRIPT: <RET>
POST MESSAGE ACTION: ? <RET>          This is Standard MUMPS code. This
code will be
executed whenever this message is retrieved through a call to BLD^DIALOG
or $$EZBLD^DIALOG
POST MESSAGE ACTION: S MYVAR="HELP #10001 WAS REQUESTED"
Select LANGUAGE: <RET>
Select ROUTINE NAME: DIKZ// <RET>
  ROUTINE NAME: DIKZ// <RET>
  LINE TAG: // <RET>
```

INTERNATIONALIZATION AND THE DIALOG FILE

Role of the VA FileMan DIALOG File in Internationalization

The VA FileMan DIALOG file is used to store dialog that would normally appear on a screen during interaction with a user. The DIALOG file becomes especially important in assisting developer support for non-English speaking users because it allows easy entry and retrieval of non-English dialog without making any changes to code that is already using the DIALOG file.

Use of the DIALOG File in Internationalization

A system variable, DUZ(LANG), identifies to VA FileMan the language currently in use. This system variable is set equal to a number that corresponds to the ID NUMBER of an entry in the LANGUAGE file (see discussion of the VA FileMan LANGUAGE file). This number is also used as a subscript for the TRANSLATION (LANGUAGE) multiple in which non-English text can be stored. For users running Kernel V. 8.0 or later, this variable is set automatically during signon.

For every entry needing translation in the DIALOG file, the developer should populate the FOREIGN TEXT field for the desired language. When either of the text retrieval routines, BLD^DIALOG or \$\$EZBLD^DIALOG, is called, if DUZ("LANG") is greater than one (1), FileMan will look at the language location specified by DUZ("LANG") to find the text. If text cannot be found at that location, FileMan defaults to use the English equivalent from the TEXT field. As with English text, parameters to be inserted into the text can be passed to the call.

See also the programmer calls BLD^DIALOG and \$\$EZBLD^DIALOG.

Creating Non-English Text in the DIALOG File

Once an entry exists in the DIALOG file, developers may enter or edit non-English equivalents for the TEXT field, using FileMan Enter/Edit.

Example

Select DIALOG: **10001 <RET>** This is English text for a test message.

.
.
.

Select LANGUAGE: **?**

Answer with TRANSLATION LANGUAGE

You may enter a new TRANSLATION, if you wish

Enter the number or name for a non-English language.

English language cannot be selected.

Answer with LANGUAGE ID NUMBER, or NAME

Choose from:

2	GERMAN
3	SPANISH
4	FRENCH
5	FINNISH
6	ITALIAN
10	ARABIC
11	RUSSIAN

Select LANGUAGE: **2 <RET>** GERMAN

Are you adding '2' as a new TRANSLATION (the 1ST for this DIALOG)? **Y <RET>**

(Yes)

FOREIGN TEXT:

1>Here is where we enter the non-English text.

VA FILEMAN LANGUAGE FILE

Introduction

Certain types of data such as dates and numbers, should be formatted differently for display depending on the language of the end user. The VA FileMan LANGUAGE file has been designed to help solve this problem for users of interactive VA FileMan. The LANGUAGE file stores M code used to perform language-specific conversions on such data. A system variable identifies to FileMan the language currently in use.

At this time, VA FileMan distributes in the LANGUAGE file only the English equivalent of language-specific data conversions specified below.

Use of the LANGUAGE File

A system variable, DUZ("LANG"), identifies to VA FileMan the language currently in use. This system variable is set equal to a number that corresponds to the ID NUMBER of an entry in the LANGUAGE file. It tells VA FileMan where to find the appropriate data conversion code from the LANGUAGE file at the time the code needs to be executed (for example, when printing a date). For users running Kernel V. 8.0 or later, this variable is set automatically during signon.

Developers may enter or create their own entries in the LANGUAGE file. The VHA Database Administrator will assign an ID NUMBER for each unique language entry in the LANGUAGE file. If developers do not follow these guidelines, their language entry may be overwritten when VA FileMan is installed.

The following Language file entries have been assigned and are distributed with VA FileMan:

- | | |
|---|---------|
| 1 | English |
| 2 | German |
| 3 | Spanish |
| 4 | French |
| 5 | Finnish |

6	Italian
10	Arabic
11	Russian

Creating LANGUAGE File Entries

Developers may enter or edit entries in the LANGUAGE file using VA FileMan Enter/Edit. The only required fields are the ID NUMBER that uniquely identifies a language and the language NAME. If M code is not found within the current language for a specific conversion, VA FileMan will default to use the English equivalent.

The other fields that can be entered for any LANGUAGE file entry are described below. At the time the code in any of these fields is executed, the data to be converted will be in the local variable Y. The M code in the field should put the transformed output back into Y, without altering any other local variables. More detail can be found in the field description for each field. Looking at the English equivalent in entry number 1 may also be helpful.

ORDINAL NUMBER FORMAT	Changes 1 to 1ST, 2 to 2ND, etc.
CARDINAL NUMBER FORMAT	Changes 1234567 to 1,234,567
UPPERCASE CONVERSION	Converts text to uppercase
LOWERCASE CONVERSION	Converts text to lowercase
DATE/TIME FORMAT	Converts date in internal VA FileMan format to MMM,DD,YYYY@HH:MM:SS
DATE/TIME FORMAT	Does other date conversions from date in internal VA FileMan format. This call has an additional input flag that

(FMTE) indicates the conversion to be done.

The flags are:

- | | | |
|----------|---|-------------------------------|
| 1 | MMM DD, YYYY@HH:MM:SS | Space before year |
| 2 | MM/DD/YY@HH:MM:SS | No leading zeros on month,day |
| 3 | DD/MM/YY@HH:MM:SS | No leading zeros on month,day |
| 4 | YY/MM/DD@HH:MM:SS | -- |
| 5 | MMM DD,YYYY@HH:MM:SS | No space before year |
| 6 | MM-DD-YYYY @ HH:MM:SS | Special spacing for time |
| 7 | MM-DD-YYYY@HH:MM:SS | -- |
| S | Always return seconds | |
| U | Return uppercase month (use only with 1 or 5) | |
| P | Return time with am,pm | |
| D | Return only date without time | |