

Z/VM

Facilities Manipulator

for z/VM and z/OS

Dynamic Manager of Screens - DMS

3270 Displays from REXX

Copyright © Cestrian Software 2008

Features and capability

<\$I2>

At the heart of z/FM is the Dynamic Screen Manager - DMS.

No seperately defined Panels

All displays are constructed from REXX commmands executed in the EXEC, No need to synchronize EXECs and Panels.

Execution time support

Displays may be defined to use features that may or may not be available at execution time. These featyres will not be used if the execution time device does not support then. Features such as Color, Graphic Escape, PSS and screen size.

Virtual Area

A display may occupy and area of up to 32768 rows by 32768 columns. The position of the displayed area may be navigated with no EXEC involvement. Available navigation commands include Left, Right, Up, Down, Home and Here.

Stem display

Stems may be displayed with a single command, regardless of size. Partial stems may be displayed. Stems may be displayed in a sequence other than its definition.

Box Display

A Box may be defined with a single command.

Display overlays

A display may be shown overlaying an existing display.

PF Keys

Up to 48 PF keys may be used. A profile file may be specified to assosiate PF Keys with command strings. An EXEC may examine actions by key name or command.

Field Naming

When input is recieved the cursor position is returned to the EXEC in normal screen coordinates, displacement within the Virtual Area and, if the cursor was in a named field, the name of the field and displacement in the field.

Timed Displays

A display may be shown for a specified time. Upon return to the EXEC a special key name 'TIMER' is set if ther time expired.

How to write Full Screen EXECS

This tutorial explains the z/FM commands to display full screen displays. Unless otherwise noted the syntax is the same for CMS and TSO.

Before any z/FM commands can be used the z/FM environment must be established. Several EXECS are shown followed by explanatory comments.

Basic EXEC "Hello World"

The Hello World EXEC

```
1. 'zfm dms'  
2. if rc<>0 then exit rc  
3. address zfm  
4. 'use screen1'  
5. "Define cen cen 'Hello World' white "  
6. "show"
```

1. Executes the program zfm and creates the z/FM subcommand environment. The function DMS is used in this example but any function can be used.
2. A non-zero return code on initialize indicates a fatal error. Execution cannot continue, a message has been displayed to describe the error.
3. Directs all future commands to z/FM.
4. Assign a name to the screen to be used. An arbitrary name unique to this screen set in this execution.
5. Places the character string -Hello World- in the CENter of the screen vertically and CENtered horizontally.
6. Displays the screen on the users terminal.

The EXEC will terminate after the users depresses any action key

Multiple Screens

<\$I2>

The Hello-Goodbye World EXEC (Multiple screens)

```
1.  'zfm dms'  
2.  if rc<>0 then exit rc  
3.  address zfm  
3a. 'use screen2'  
3b. "Define cen cen 'Goodbye World' white "  
4.  'use screen1'  
5.  "Define cen cen 'Hello World' white "  
5a. do while rstatus<>'PF3'  
6.   "show"  
6a.   if rstatus='PF4' then 'use screen1'  
6b.   if rstatus='PF5' then 'use screen2'  
6c. end
```

NOTE: only additional lines are explained

3a. Assigns a second screen with a new name

3b. Place -Goodbye World- in this screen

5a. Stay in loop till user depresses PF3

6a. If user depressed PF4 then change to screen1

6b. If user depressed PF5 then change to screen2

6c. Loop end

Displaying Stems

The Simple Browse EXEC

```
1.  'zfm dms'  
2.  if rc<>0 then exit rc  
3.  address zfm  
3a. parse arg fn ft fm  
4.  "exio * diskr "fn ft fm" (finis stem file."  
5.  "Define 1 1 stem exec." file.0 file.width "white"  
6.  do while rstatus<>'PF3'  
7.      "show va"  
8.  end
```

NOTE: Only additional lines are explained

3a. Get file specs to display

4. Read the file into stem -file.- sets file.0 to number of lines and file.WIDTH to the maximum line size

5. Place a stem -file.- in this screen beginning at row 1 col 1 for FILE.0 lines of width FILE.WIDTH

7. Show a virtual area. Default keys are

F6=Here - Moves cursor location to top left of real screen

F7=Back - Moves one screen towards beginning

F8=Forw - Moves one screen towards end

F9=Home - Moves beginning of virtual screen to top left of screen

F10=Left - moves one screen left

F11=Right - moves one screen right

Browse a file EXEC

The Browse EXEC with headers

```
1.  'zfm dms'  
2.  if rc<>0 then exit rc  
3.  address zfm  
3a. parse arg fn ft fm  
4.  "exio * diskr"fn ft fm" (finis stem file."  
4a. "Define res 1 1 'Displaying File:' green"  
4b. "Define res +0 +0 '"fn ft fm"' green"  
5.  "Define 1 1 stem exec." file.0 file.width "white"  
6.  do while rstatus<>'PF3'  
7.      "show va"  
8.  end
```

4a. Define a resident line starting at row 1 col 1, row 1 will now repeat on all displays of this screen. The number of screen lines available for other data is decreased by 1.

4b. Define a field adjacent to the prior field (+0 rows, +0 cols)

Input from a Display

Simple input fields

```
1.  'zfm dms'  
2.  if rc<>0 then exit rc  
3.  address zfm  
4.  "Define 4 2 'Enter Name :' green"  
4b. "Define +0 +0 var name 8 white under"  
4c. "Define +0 +0 'Password:' green"  
4d. "Define +0 +0 var password 8 dark"  
5.  name='';password=''  
6.  do while rstatus<>'PF3'  
7.    "show"  
8.    valid=0  
9.    if rstatus='ENTER' then do  
10.     valid=checkpass(name,password)  
11.    end  
12.    if valid then leave  
13.  end
```

4. Define a prompt (protected field)

4b. Define an input field (unprotected) initial value is contents of the variable NAME, changed data placed into NAME for 8 bytes

4c. Define a prompt (protected field)

4d. Define an input field (unprotected, non display) initial value is content of the variable PASSWORD, changed data placed into PASSWORD - 8 bytes

5. Clear data fields. NOTE retrieve of variables is not performed at DEFINE statement but at SHOW.

8. Clear validation switch

9-11. If user pressed ENTER then call user supplied validation, returns 1 if valid

12. If valid name/password then leave loop, else retry

Filelist Sample

Simple File list and Browser

```
1.      parse arg fn ft fm .
2.      "pipe CMS listfile" fn ft fm "( date | stem list."
3.      "zfm dms"
4.      if rc<>0 then exit rc
5.      address zfm
6.      "use flst"
7.      "define res 1 1 var list.1 80 turq prot"
8.      "define 1 1 stem list. 2 to" list.0 " 80 cursor white keep"
9.      "pfkeys zfm dms"
10.     "pmap"
11.     do while rstatus<>'PF3'
12.         "use flst"
13.         "show va"
14.         if rstatus='PF2' then call disp_file
15.     end
16.     exit
17.
18.     DISP_FILE:
19.         parse var rfield nam '.' num
20.         if nam<>'LIST' then return
21.         parse var list.num fn ft fm .
22.         "exio * diskr "fn ft fm"(finis stem file."
23.         "use file"
24.         "clear"
25.         "define res 1 1 'File:" fn ft fm "Lines:" file.0"' green"
26.         "define 1 1 stem file." file.0 file.width" white keep"
27.         do while rstatus<>'PF3'
28.             "show va"
29.         end
30.         rstatus=''
31.     return
```

2. Pipe a listfile of requested files to stem LIST.

6. Name this screen

7. Use the first line in stem LIST. as a header

8. Place the stem LIST. on the screen starting at the second entry to the end.

FilelistSample Continued on next page

Filelist Sample continued

9. Name a set of predefined PF keys to use (zfmdms \$profile *)
10. Place the PF key settings on the bottom of the screen
12. Restate screen name
14. If the user depressed PF2 then call a procedure to display the file
19. RFIELD contains the stem name that the cursor was on.
20. Ensure stem name is LIST.
21. Get file specs from stem entry
22. Read the file to stem FILE.
23. Name a new screen
24. Discard prior defines, if any
25. Put header on screen
26. Put sthe stem FILE. to screen
28. Show the screen until user depresses PF3
30. Clean rstatus - so line 11 will not see it.
31. End of procedure

Display Stems Sorted

Sort a stem and display

```
1.  "zfm dms"
2.  if rc<>0 then exit rc
3.  address zfm
4.  state.1='Texas'
5.  state.2='California'
6.  state.3='New York'
7.  state.4='Arkansas'
8.  state.5='Alaska'
9.  state.6='New Mexico'
10. state.7='Washington'
11. state.8='Kentucky'
12. state.9='Nevada'
13. state.10='Florida'
14. state.11='Utah'
15. state.12='Virginia'
16. state.0=12
17. do i=1 to state.0
18.   state.i=left(state.i,14) right(i,2)
19. end
20. "use stemdemo"
21. "clear"
22. "vsort state." state.0 length(state.1)" into Idx"
23. "define res 1 1 zfm 'Demonstration of STEMBY' blue cest"
24. "define +2 22 'Unsorted' yellow"
25. "define +0 42 'Sorted on col 1' yellow"
26. "define +1 22 stem state. "state.0 length(state.1)" Keep cursor
white"
27. "define +0 42 stemiby Idx state. "state.0 length(state.1)" keep
white"
28. "define nl 1 ' '"
29. "define +2 1 'Idx ' white var Idx "length(idy)" yellow"
30. do while rstatus<>'PF3'
31.   "show"
32. end
```

1. - 3. Initialize Z/FM

4. - 19.; Create stem with State Name and Sequence number

20.; Create a screen named 'stemdemo'

21. Ensure screen cleared (not needed in this case)

22. Sort the stem. Store sequence into IDX. Stem does not change

23. Create a header line

24. - 25. Create subheader

26. Create a stem display. Place the cursor on the 1st entry, fields are not protected

Display Stems Sorted continued

(allowing tabbing) but input will be ignored.

27. Create a stem display. stem is ordered by the sequence in IDX.
28. Create a dummy display on the 'NewLine' after the stem
29. Create a display of the index
30. - 32. Display the screen. Loop until PF3 depressed

Screen Name: STEMDEMO

```
z/FM 7.1v Demonstration of STEMBY Cestrian Software

          Unsorted                Sorted on col 1
Texas           1      Alaska           5
California      2      Arkansas         4
New York        3      California        2
Arkansas        4      Florida          10
Alaska          5      Kentucky          8
New Mexico      6      Nevada            9
Washington      7      New Mexico        6
Kentucky        8      New York          3
Nevada           9      Texas             1
Florida         10     Utah              11
Utah            11     Virginia          12
Virginia        12     Washington        7

Idx  5 4 2 10 8 9 6 3 1 11 12 7
***** 24 Blank lines were omitted for this view *****
```

Original Screen Size 43 x 80

CLEAR

DESCRIPTION

The CLEAR command discards any previously issued DEFINES and the fields associated with them. Additionally if just the operand SCREEN is specified then the physical device screen is cleared.

If the operands SCREEN ONLY is specified then the DEFINES are not cleared but the physical screen is cleared.

FORMAT

CLEAR
CLEAR SCREEN {ONLY}
CLEAR HOLD

RULES

CLEAR Discards all defines for this 'USE' screen

CLEAR SCREEN Discards all defines for this 'USE' screen and erases the physical screen.

CLEAR SCREEN ONLY Erases physical screen, keeps defines.

CLEAR HOLD Clears hold flag set by TYPE. .

MODES

EXEC mode only.

FUNCTIONS

Functions:- DMS

CURSOR

DESCRIPTION

The CURSOR command causes the cursor to be positioned at the designated place on the screen when the next SHOW command is executed.

FORMAT

CURSOR {row col {res}}

RULES

- row Must be numeric and within the bounds of the screen or omitted.
- col Must be numeric and within the bounds of the screen or omitted.
- res Must be 'R' or 'N' to indicate reserved line or not or 'T' for physical (true) location. Default is 'N'.

Example

```
"curs=''
"do while rstatus<>'PF3'
  "cursor" cs
  "show"
  cs=rrow rcol rres
end
```

Will retain cursor on screen

MODES

EXEC mode only.

FUNCTIONS

Functions:- DMS

DEFINE

DESCRIPTION

The DEFINE command designates the location of the attribute byte for a field in the VIEWABLE AREA (the area is 32767 rows by 32767 columns), the operands specify the contents of the field and the attribute byte. Multiple DEFINES may be issued before a SHOW command actually displays the entire screen, if EXEC variables are designated as the "data" source their contents are not extracted from REXX/EXEC2 until the SHOW command is issued. The CLEAR command discards all DEFINES. Colour attributes may be set, however unless the execution time terminal has the color feature installed colours will not be used.

FORMAT

DEFINE {RES} {INBOX} row col what how {what how}

RULES

RES Designates this defined line as a 'reserved' line meaning that this line is fixed in a VA show with multiple screens any nonRES lines will flow around RES lines.
e.g. If there are two RES lines on lines 1 and 2 then a nonRES line specifying line 1 will be placed on line 3).

RES line row designates a physical row number on the screen.

INBOX Designates this defined line as a line in the prior BOX.
Row and col are relative to the boundaries of the BOX.

ROW Designates the row on which to place this field. First row is 1.
It must be one of the following

- a. A numeric integer designates an absolute line number
- b. A signed integer is relative to the previous line defined.
NOTE: If +1 is the first define then line 1 is implied.
- c. The value +0 designates the same line.
- d. The value BOT to indicate the last line on the screen.
- e. The value BOT-x to indicate x lines above bottom.
- f. The value NL to indicate the next line not used.
- g. The value CEN to indicate the middle line vertically.

COL Indicates column position of the attribute byte for this field. First column is 1.
It must be one of the following

- a. A numeric integer designates an absolute column number.
- b. A signed integer is relative to the end of the previous field.
NOTE: May be positive or negative.
- c. The value +0 designates the column adjacent to the previous field.
- d. An equal sign (=) designates the same column as the previous field start. May be followed by a signed integer.
- e. The value !RIGHT may be used as the rightmost column
e.g !RIGHT - 15
- f. The value CEN to indicate the 1st field will be centered

DEFINE Continued on next page

DEFINE continued

horizontally.

WHAT Indicates the data to be displayed. See below for details.

HOW Specifies the manner used to display the field including colours attributes and direction of the field. See below for detail.

Multiple sets of **WHAT** and **HOW** are allowed for a single **DEFINE** command, each set is delimited by a quoted string or one of the words "VAR", "STEMIBY", "STEM" "CEST" or "ZFM".

Subsequent sets in the same **DEFINE** are positioned at the next available position on the screen.

Field Definition Operands (WHAT).

A quoted string

Indicates a static data string to be displayed.

A hexadecimal quoted string preceded by x

Indicates a static data string to be displayed.

e.g. x'c1c2c3' Must be even number of valid hex characters

VAR variable-name #length

Indicates the name of an EXEC2/REXX variable containing the data to be displayed for #length.

STEMIBY iname stemnm #stems #l

Display a stem for #stems (default direction DOWN) each stem entry for #l bytes, stem tails are in variable iname.

STEM stemnm #stems #l

Display a stem for #stems (default direction DOWN) each stem entry for #l bytes.

STEM stemnm #a TO #b #l

Display a partial stem (default direction DOWN) each stem entry start at entry #a end at #b for #l bytes each.

STEM stemnm #a TO #b #l BY #i

Display a partial stem (default direction DOWN) each stem entry start at entry #a end at #b bump by #i for #l bytes each.

BOX #rows #cols { 'fillers' } { TOP 'top_title' } { BOT 'bot_title' } { EMPTY }

DEFINE Continued on next page

DEFINE continued

Display a box with #ros rows, #cols columns, fillers is a list of upto 8 characters for characters used in the box (topleft corner, top fill, topright corner, right side, bot right corner, bot fill, bot left corner, left side), default +++|+++|

Top_title centered on top line on box

Bot_title on bottom line of box.

EMPTY clears the box interior.

If the position of the box (ROW COL) is CEN CEN the center of the box is centered on the screen.

CEST

The string 'Cestrian Software'. NOTE if not first on define statement then column is set to right - length of string.

VMFM or ZFM

The string 'z/FM #y#xz', where # # is the current release
y is '.' if running normal module
y is '-' if running in the shared segment
y is ':' if running standalone

x is 'v' if running under VM

x is 'o' if running under MVS (OS/390 z/OS)

z is '' if running 24bit addressing in 24bit space

z is '-' if running 24bit addressing in 31bit space

z is '|' if running 31bit addressing in 24bit space

z is '+' if running 31bit addressing in 31bit space

DEFINE Continued on next page

Display Modifiers (HOW).

DEFINE Continued on next page

DRAFT

DEFINE continued

Colours *	BLUE RED PINK WHITE GREEN TURQUOS YELLOW DEFAULT NEUTRAL
More Color *1	BLACK DBLUE ORANGE PURPLE PGREEN PTURQ GREY RWHITE
named color	ZFM VMFM ALTDATA ARROW CMND DATA COPY HEAD CEST PROMPT VAR BORDER EOF PFKEYS PREFIX VIEWVA. See SET COLOR.
ASCII	Converts field from ASCII to EBCDIC prior to display. and, if an input field from EBCDIC to ASCII after read.
ASKIP	Place a closing attribute after this field.
BLINK *	Blink the field.
CANHI	Makes field eligible for highlighting (see: command SET HILITE)
CEN	Centers the field on the screen.
CICS	Changes the placement of the askip field to the start of the field.
CLDATA *	The data is preceded by three bytes specifying colour, extended highlight and PS set (see note 5 below).
COLOR clr-name	A named color. See - SET COLOR.
CURSOr IC	Places the cursor at the start of this field. May be overridden by the command 'CURSOR'.
DARK DRK	The field is not displayed.
DEFCOL	No colour
DEFHI	No highlighting
DSHIFT	The field may be shifted by the DSHIFT operand on the SHOW command. VLEN must be specified. (see note 6).
DOWN	The field is displayed downwards one byte at a time or the stem is placed downwards (default for stems).
FSET	Sets the MDT (Modified Data Tag) for this field.
GE *	Inserts a graphic escape X'08' before each character (except space and null) - Forces use of the APL character set (if available).
HIGH	Displays the field in high intensity.
JUSTR	Justifies the field to the right.
KEEP	Allows input into this field, but discards the input thus preserving the original data.
LEFT	The field is displayed right to left or the stem is placed right to left.

DEFINE continued

NAME	Names a quoted string. Maximum size 16.
NOASKIP	Normally an unprotected file will automatically have an ASKIP, this inhibits the ASKIP.
NOCURS	Resets the CURSOR status (Used with MODIFY command).
NODARK	The field is displayed.
NOFSET	Unsets the MDT (Modified Data Tag) for this field.
NORM	No highlight, no colour, no PS.
NULL	Unused positions are set to nulls rather than spaces.
NUM	Numeric entry only.
OVER	Forces this field to be displayed even if it overlaps another DEFINEd field.
PERM	The field is permanent on the non-res portion of the screen. Row / col+length cannot exceed physical screen. Cannot be specified on a RES line. Permanent fields are written over any other fields.
PROT	Disallows input into this field.
PS opr *	Use Program symbol set 'opr' (see command PS).
REVerse *	Display the field in reverse display.
RIGHT	The field is displayed left to right (default) or stem is placed left to right.
SA *	Use Set Attribute in the data stream rather than Set Field Extended (see note 4).
SAClose *	Use Set Attribute in the data stream rather than Set Field Extended (see note 4) after the field the set attribute values are reset.
SELECT	Field is light-pen detectable.
SYNC	Synchronizes the buffer after this field.
UNDERli *	Underline the field.
UNPROT	Allows input into this field.
UP	The field is displayed upwards one byte at a time or the stem is placed upwards.
UPPER	Used on an entry field to upper case the reply.
VA	Only with STEMIBY. VA commands apply to stem. (see note 8)
VLEN	Species actual variable size (see note 6).

DEFINE Continued on next page

DEFINE continued

WRAP <V> # Stems only. The stem is placed left to right. When the right of the physical screen is reached the row is advanced and to column set to initial column. # is size of wrap if not datalength. V indicates vertical columns.

DEFINE Continued on next page

DEFINE continued

Notes on usage

* Extended operands are not used if the execution time device does not support the specific feature.

*1 More colors are available with some emulators. Use 'SET COLOR GRAPHIC ON' to use.

1. Some operands are mutually exclusive and if specified the last is used. e.g. BLUE, RED, PINK, WHITE, GREEN, TURQ and YELLOW or REVERSE, BLINK and UNDER.

2. The col and row may specify locations beyond the bounds of the logical screen in which the display is to occur, clipping of fields is automatic, and with the "VA" operand of the "SHOW" command the presentation screen may be positioned left, right etc.

3. This command allows an EXEC to simply format a large amount of data (up to 32767 columns by 32767 rows) for presentation on the display in one pass, and then allow the user (by the "VA" command) to select the portion of the display to be viewed, rather than multiple interactions with the EXEC to select the next page for example.

4. The operand SA allows adjacent fields to be displayed in different colors and highlighting without a position being taken for an attribute byte. The adjacent fields must have the same standard attribute else one will be inserted.

5. The operand CLDATA specified only with VAR, STEM or STEMIBY. Byte 1 is colour (B, R, P, Y, G, W or T), byte 2 is extended highlight (B, U or R), byte 3 is PS set_id. If characters are not valid then base colour is used. These three bytes are not included in length specified.

6. If a variable is larger than the size to be displayed the operand VLEN should be specified to keep the variable at its true length. If VLEN is not specified then the variable will be truncated to the displayed length. VLEN must be specified to use the DSHIFT operand. DSHIFT allows displaying parts of a variable.

7. When a screen is shown the defined entries are sorted in the following sequence.

a. Non-RES entries by row, col in catagories:

1. non-BOX entries

2. BOX Entries in sequence by box definition

I. BOX left, top and bottom entries

II. BOX empty entries

III. INBOX entries

VI. BOX right side entries

b. Perm entries by row, col.

c. RES entries by row, col in the same catagories as a.

Sorting may be suppressed by issuing 'SET SCRSORT OFF'.

8. VA allows VA commands FORW, BACK, HERE and HOME on the STEMIBY If multiple DEFINES use the same STEMIBY and VA then the VA command affects all matching DEFINES.

Examples

1.

DEFINE Continued on next page

DEFINE continued

```
"clear screen"  
"set color my_title white"  
"define 1 1 'At Location--' color my_title var address 8 blue prot"  
"define +1 1 'Data is -----' color my_title var myname 20 red keep"  
"define +1 = 'Hex is -----' color my_title var myhex 40 pink keep"  
"transfer current (into address"  
"char (into myname 20"  
"hex (into myhex 40"  
"show"
```

Will display, in full screen, the current location and its contents in character format.

2.

```
"exio * disk profile exec a (finis stem in."  
"define 1 1 stem in." in.0" 80 white prot"  
do while rstatus<>'PF3'  
"show va"  
end
```

Will display "profile exec a" in a virtual area

3.

```
"exio * disk profile exec a (finis stem in."  
"vsort in. "in.0 in.width "col 5 into sorted"  
"define 1 1 stemiby sorted in." in.0 in.width"white prot"  
do while rstatus<>'PF3'  
"show va"  
end
```

Will display "profile exec a" in a virtual area sorted on column 5. Not sure why anybody would want to but, it's a sample.

MODES

EXEC mode only.

FUNCTIONS

Functions:- DMS

DROP

DESCRIPTION

The DROP command deletes screens and all associated allocated memory.

FORMAT

DROP `scrname` `<scrname2 scrname..>`

RULES

A screen name cannot start with ZFM. ZFM..... screens are reserved for internal use and cannot be dropped.

MODES

EXEC mode only.

FUNCTIONS

Functions:- DMS

MODIFY

DESCRIPTION

The MODIFY command changes characteristics of a previously DEFINEd screen entry.

FORMAT

MODIFY {STEMIBY iname} what how {FROM varname}

RULES

STEMIBY Limit modification to variables used in a DEFINE STEMIBY with iname as the index.

WHAT Indicates the data to be modified. Must be a variable name used in a previous DEFINE var, stem or stemiby.

If FROM specified then what is a stemname and varname consists of numeric tails to be appended to stemname.

To modify an entire stem use stemname.*

Special case "modify * nocurs" clears cursor setting from all fields.

HOW Specifies the manner used to display the field including colours attributes and direction of the field. See DEFINE definition of HOW.

FROM varname contains a list of tails to append to what.
i.e 'modify cmd. prot from list'
where list = '3 4 6'
will modify CMD.3 CMD.4 and CMD.6 to PROT

Examples

```
"modify liststembyrevn25x12ibblue25r02"pink prot"
```

Will change the entries LIST.6 in both stems to reverse red. The remainder of the stems LIST. will be displayed in blue or pink.

```
"modify stemiby index list.4 green"
```

Will change the entry LIST.4 in the STEMIBY only to green.

```
"modify stemiby index list.* white"
```

Will change all LIST. entries in the STEMIBY to white.

```
"modify list.* yellow"
```

Will change all LIST. entries in all stems to yellow.

MODES

EXEC mode only.

FUNCTIONS

Functions:- DMS

MODIFY Continued on next page

DRAFT

PFKEYS <\$I1>

DESCRIPTION

The PFKEYS command causes named \$PROFILE to be used to set PF keys to commands to be used on the current screen defined by USE.

FORMAT

PFKEYS name {DEFINE | NAME newname}

RULES

If the name (either name or newname) has been specified in a prior PFKEYS then that PFkey table is used.

name identifies a file with a filetype of \$PROFILE containing commands to set PF keys to command. Any valid SET command may be contained in a \$PROFILE file.

If DEFINE is specified then no file is read, an empty PF table id defined as name.

If the NAME option is used the file name is loaded and the PF table is named newname.

Format for \$PROFILE statements

Bytes 1 & 2 must be PF, AL, PA or AA

Bytes 3 & 4 must be numeric

Starting in byte 5 are three fields seperated by /*

Field one is the command to be associated with this function key

If field3 exists then field2 is ignored and field3 is description.

If field3 does not exist then field2 is the description

Sample \$PROFILE file

PF1 HELP CMS /* /*Help

PF2 UPDATE /* /*Update

PF3 QUIT /* /*Quit

PF4 PREV /* /*Prev

PF5 FUTURE /* /*Future

PF6 EX CURSPNT A # /* /*A # at cursor

PF7 A-SCREEN /* /*Back

PF8 A+SCREEN /* /*Forw

PF9 AA@ /* /* AA

PF14ex vmfdcode /* /* Decode instructions

PF18EX CURSPNT A @ /* /* A

at cursor

PF22? /* RETRIEVE /* Recall

Examples

`"use screen1"`

PFKEYS Continued on next page

PFKEYS continued

```
"pfkeys mypfs1"  
"set PF1 OPT1 /* Option1"  
"define ....."  
"define ....."  
"use screen2"  
  
"pfkeys mypfs2"  
"set PF1 OPTX /* OptionX"  
"define ....."  
"define ....."  
if screen_decide=1 then "show name screen1"  
                        else "show name screen2"
```

If the file 'myfile1 \$PROFILE' exists it will be loaded likewise 'myfile2 \$PROFILE'. If nonexistent then a null PF key table will be defined.

SHOW will display, depending on the value of screen_decide, either 'screen1' or 'screen2'. If 'screen1' is shown and PF1 is depressed then RCOMMAND will be set to 'OPT1' and if 'screen2' is shown and PF1 is depressed then RCOMMAND will be set to 'OPTX'.

Related commands

See PFMAP and SET MAPPF

MODES

EXEC mode only.

FUNCTIONS

Functions:- DMS

SHOW

DESCRIPTION

The SHOW command displays a named screen (previously set by USE), or current USE screen, this screen can contain any number of DEFINES.

FORMAT

```
SHOW {NAME scname}  
    {AT row col}  
    {WAIT ##}  
    {VA {row col}}  
    {NOIN}  
    {NOCON}  
    {CLEAR}  
    {NOCLEAR}  
    {DSHIFT #}  
    {OVERbuff {PROT}}  
    {ONCEOVER {PROT}}
```

```
SHOW BUFFER FROM varname {STRUC} {NOW} {OVERBUFF  
{PROT}}  
SHOW BUFFER DROP
```

RULES for SHOW

NAME scname	Identifies the screen, defined by USE, to display. Defaults to current USE screen.
AT row col	Adjusts the top left corner of the display to the physical position on the screen.
WAIT ###	Displays the screen, waits for an action key or then ## of hundredths of a second to elapse. If time expires RSTATUS='TIMER'
VA {row col}	Display of a VIEWABLE AREA is to be performed. The fields specified by previous DEFINE commands may not be within the bounds of the physical device, that is the TRUE DISPLAY set by the DEFINES is 32768 by 32768 characters. The displayed view is set such that the specified row and column (0 0 assumed) of the TRUE DISPLAY is at the top left corner. The command "VA" may be used to change the position of the physical display in relationship to the TRUE DISPLAY.
NOIN	Displays the screen does not wait.
NOCON	Suppresses display the users terminal, and writes only to dialed devices 100-11F if the command "ALT" has been entered. ALT terminal require 'CP 370ACCOM ON'.
CLEAR	Clears the physical screen prior to display.
NOCLEAR	Does not clear the physical screen.
DSHIFT	Moves the start point to display # bytes into the variable. the option

SHOW Continued on next page

SHOW continued

DSHIFT must be specified on the DEFINE statement.

OVERbuff	The screen is appended to the current screen and remains appended.
ONCEOVER	The screen is appended to the current screen for this SHOW only.
PROT	Used in conjunction with OVERbuff and ONCEOVER. All fields in the current screen are set to protected. NOTE: All extended highlighting will be suppressed.
NOW	Used only with SHOW BUFFER, displays the buffer immediately.
STRUC	Indicates that the buffer is a structured field.
SHOW BUFFER FROM	Allows the display of a 3270 data stream from an EXEC variable. This command sets the buffer content, a subsequent SHOW is required to actually display the buffer if NOW is not specified. A SHOW OVER may be placed in between.
SHOW BUFFER DROP	Clears a previously shown buffer.

View status

If the viewable area will not fit on the physical screen then the lower right corner of the screen will contain the character string "xxx,yyy....." (unless SET VIEWVA OFF has been issued), xxx and yyy show the number of rows and columns respectively that the displayed screen is offset from the TRUE DISPLAY, may be replaced by "+" "-" "<" or ">" to indicate that data exists above, below, left or right of this screen.

Results and status returned

REXX variables are set to reflect the status of the SHOW and any data entered into fields defined by the VAR operand of a DEFINE is transferred into those variables.

RSTATUS The key "name" (e.g. ENTER PF1 PA2 etc.)
If the key executed a z/FM command the key name is ZFx or ZAx

If, however, the time has elapsed then the EXEC variable "RSTATUS" will contain "TIMER"; and, as above, data is transferred to EXEC variables.

If IUCV is active and a buffer is available EXEC variable "RSTATUS" will contain "IUCV"; and, as above, data is transferred to EXEC variables. An "IUCV RECEIVE" command should be issued.

RCURSOR	The cursor location, 3 digit Row 3digit column (upper left corner = 001001).
RROW	The row number of the cursor location on the physical screen.
RCOL	The column number of the cursor location on the physical screen.
RRES	'R' if on a reserved line else 'N'.

SHOW Continued on next page

SHOW continued

RVAVIEW The cursor location relative to the virtual area (upper left corner = 001001).

RVROW The row number relative to the virtual area.

RVCOL The column number relative to virtual area.

RFIELD The name of the field at the cursor location, if the cursor position is within the field. For fields **DEFINED** by **VAR** or **STEM** this is the variable name, for quoted strings with a **NAME** option it is that name. If a name cannot be determined the value 'UNKNOWN' is returned

ROFFSET The offset within the named field of the cursor. The attribute byte is 0 offset.

RINDEX For a **STEMBY** field the associated index name.

RCOMMAND If a PF key is depressed then the corresponding command from the current \$PROFILE file.

In a VA display the user has the ability to move the viewpoint at will so, after display the **EXEC** variable **RVCURSOR** will be set to the row and column of the final viewpoint.

Note that in a "VA" display the cursor location is adjusted to the coordinates of the true display.

Usage

The use of the **SHOW** command using **NOIN** and **VA** repeatedly while changing the viewpoint allows for animated effects.

MODES

EXEC mode only.

FUNCTIONS

Functions:- DMS

USE

DESCRIPTION

The USE command names the screen to be used by DEFINE and SHOW commands. Multiple screens may be in existence at one time.

FORMAT

USE scname
USE (INTO scname)

RULES

The screen name is in effect for a single EXEC execution only.

The screen name cannot start with ZFM. ZFM..... screens are reserved for internal use.

The into option returns the current screen name.

MODES

EXEC mode only.

FUNCTIONS

Functions:- DMS

VSORT

DESCRIPTION

The VSORT command will sequence a stem of REXX variables, replacing the STEM unless INTO is specified.

The sequence of variables with equal sort fields is unpredictable.

The INTO or INTOSTEM variable will be defined if non-existent.

If maxlen is specified each variable in the stem must be less than maxlen, if not it will be truncated to maxlen.

FORMAT

VSORT stem <#stems <TO stem#> maxlen> <COL col. ><D|A><(out>

RULES

stem	Identifies the stem to sort
#stems	The number of stem entries or 1st stem entry if TO specified. If specified as 0 or omitted then stemname.0 has number of stems.
maxl	The maximum size, in bytes, of the largest stem. Does not have to be exact but cannot be too short. If INTO is specified the stem is allowed to truncate but not below sort column. If omitted the length will be calculated from the stem.
TO stem#	Ending stem number.
COL col.	Format COL beg <type> size beg = Beginning column (first column is 1) type = Type of field - optional UPR - Sort in upper case. UPRD - Sort in upper case, ignores leading blanks. HEX - Hex value in display format ADR - Hex value ignore hi-bit. size = Number of bytes to sort.

Up to 16 sets of fields may be specified. Total columns must not exceed 256. Default 1 maxlen.

A Ascending sequence (Default)

D Decending sequence

out INTO var - Stem is not replaced var will contain a series of numbers that when indexed against the original stem will produce a sequential stem.

INTOSTEM stemname - Stores the sorted stem in the new stem name. If a partial stem was sorted (stem x TO y) then the resulting INTOSTEM will contain only the entries sorted.

EXAMPLES

Assume REXX variable values:

VSORT Continued on next page

VSORT continued

```
data.1 = "Texas"  
data.2 = "TEXAS"  
data.3 = "Ohio"  
data.4 = "NEW YORK"  
data.5 = "....."
```

Then the command "VSORT data. 5 8 "

will change the variables to:

```
data.1 = "....."  
data.2 = "NEW YORK"  
data.3 = "Ohio"  
data.4 = "Texas" /* Note the sort IS case sensitive */  
data.5 = "TEXAS" data.0 = 5
```

However the command "VSORT data. COL 2 5 into theindex"

will not change the variables, but will create:

```
theindex = "5 1 3 4 2"
```

Which could be used in the following manner:

```
do i = 1 to 5  
  trui=word(theindex,i)  
  say data.trui otherdata.trui .....
```

or

```
"define x y stemiby theindex data. 5 10 white"
```

NOTES

VSORT performs an incore QuickSort. Size is limited by machine size only.

As a test VSORT was run against a stem from a large file (225000 recs, 60 bytes per rec). Elapsed time was 72 seconds. For comparison the same file was sorted using CMS SORT command, that comparison was aborted after 3 hours.

MODES

Applicable primarily to EXEC mode.

FUNCTIONS

Functions: - All

DESCRIPTION

Highlighting only applies to data displayed (DEFINEd) with the colors DATA or ALTDATA or DEFINE option CANHI

Displays the quoted string in the specified colour, highlighting and program symbol set.

Multiple strings may be highlighted by the use of '+', no limit.

+ DROP will drop all + hilites

The LOCATE command (/) uses the first HILITE string to highlight the found string.

The string should not start with a space.

```
{set} Hilite {+} 'string' {ASIS} {BETWEEN {NOT char}} colspec  
{set} Hilite {+} CLEAR  
{set} Hilite OFF  
{set} Hilite ON
```

RULES

ASIS	Data must match exactly regardless of case setting
BETWEEN	Data between occurrences of the string if highlighted.
NOT char	Data occurring between successive not characters will not be highlighted.
CLEAR	Drop highlighting. If + then all + highlighting dropped.
Colspec	Any combination of color, highlight and PS
Color	Must be a valid colour (RED, BLUE, GREEN, YELLOW, TURQ, WHITE)
Highlight	Must be valid (BLINK, REVerse, UNDER)
PS #	The Program Symbol Set # must be loaded.

Examples

```
"hil + '""' between not '""' pink"  
"hil + '""' between not '""' turq "
```

Will not hilite the ' in "it's the end" as the ' is between "s
Will not hilite the " in 'it"s the end' as the " is between 's

MODES

Applicable primarily to INTERACTIVE mode.

VIEWVA

DESCRIPTION

In a screen generated by 'SHOW VA' the locator block is in the bottom right corner of the display.
This command inhibits its display.

```
{set} VIEWVA ON (default)
{set} VIEWVA OFF
{set} VIEWVA (inverts current VIEWVA setting)
```

MODES

Display