



**Text Editor**  
**Release 3.60**

8 Merthyr Mawr Road, Bridgend, Wales UK CF31 3NH

Tel: +44 (1656) 65 2222  
Eml: [support@cbl.com](mailto:support@cbl.com)

**CBL Web Site - [www.cbl.com](http://www.cbl.com)**

This document may be downloaded from [www.cbl.com/documentation.php](http://www.cbl.com/documentation.php)



# Contents

<b>Documentation Notes.....</b>	<b>1</b>
<b>Summary of Changes.....</b>	<b>2</b>
First Edition (October 2013).....	2
Second Edition (March 2015).....	3
Third Edition (January 2017).....	3
Fourth Edition (July 2020).....	4
Fifth Edition (February 2024).....	5
<b>Introduction to the CBL Text Editor.....</b>	<b>6</b>
CBLe Main Window.....	6
CBLe Main Window Menu Bar.....	7
CBLe Main Window Status Bar.....	9
CBLe Text Edit Document Window.....	9
CBLe Current Window.....	10
CBLe SDE Document Window.....	10
CBLe List, DB2 SQL & CBLVCAT Document Windows.....	11
CBLe Open Dialog Window.....	12
CBLe Find Dialog Window.....	13
CBLe Change Dialog Window.....	14
CBLe Sort Dialog Window.....	15
CBLe Fill Dialog Window.....	15
Text Edit Line/Box Block Options.....	16
Text/Data Edit GO Options.....	17
Editing HFS files.....	18
HFS File Records.....	19
HFS File Name Specification.....	19
Data Protection.....	19
Editing Large files.....	20
Editing VSAM files.....	20
Editing RECFM=U Files.....	21
Focus Line and Column.....	21
Using Marked Blocks.....	21
Targets.....	22
Line Targets (line-target).....	22
Absolute Line Number Target.....	22
Relative Line Number Target.....	22
String Target.....	22
Named Line Target.....	23
Line Class Target.....	23
Column Targets (column-target).....	24
Group Targets (group-target).....	24
FileKit Support For Regular Expressions.....	25
Usage.....	25
Syntax.....	25
Description.....	26
Examples.....	28
Alternation (or) operator.....	28
Not operator.....	28
Minimal closure.....	28
Maximal closure.....	28
Minimal plus.....	28
Maximal plus.....	29
Syntax Elements and Operators Summary.....	29
Special Characters Used in Regular Expressions.....	29
Text Specifiers.....	29
Operators.....	30
Predefined Expressions.....	30
Symbolic Escaped Characters.....	30
Save/Restore Text and Data Edit Document Windows.....	31
<b>Environment Variables.....</b>	<b>32</b>
Variable Types.....	32
Variable Substitution.....	33
<b>ISPF Edit Interface.....</b>	<b>34</b>
ISPF Edit Features.....	34
ISPF and ISPF Edit Primary Commands.....	34
ISPF Edit Line Commands.....	35
ISPF Edit Display Fields.....	35
ISPF PFKey/command line concatenation.....	35
ISPF Interface Initialisation.....	35
ISPF/XEDIT Command Precedence.....	36
<b>CMX (CoMmand eXecution) Files.....</b>	<b>37</b>

# Contents

<b>REXX Macros.....</b>	<b>38</b>
ISPF Edit Macros.....	40
CBLe PROFILE Macro.....	42
<b>Primary (Command Line) Commands.....</b>	<b>43</b>
Command Reference Syntax Conventions.....	43
How to read the Syntax Diagrams.....	43
ACTION.....	44
ADD.....	47
ALL.....	47
ALLOCATE.....	48
BACKWARD.....	49
BOTTOM.....	50
ISPF Interface.....	50
XEDIT Interface.....	50
BOUNDS, BNDS.....	51
BOX.....	52
BROWSE.....	52
CANCEL.....	53
ISPF Interface.....	53
XEDIT Interface.....	53
CAPPEND.....	54
CAPS.....	54
CDELETE.....	55
CFIRST.....	55
CHANGE.....	56
ISPF Interface.....	56
XEDIT Interface.....	59
CHANGEDIALOG.....	60
CINSERT.....	61
CLAST.....	61
CLIPBOARD.....	61
CLOCATE.....	63
CMMSG.....	63
COMMAND.....	64
COMPARE.....	64
COPY.....	65
ISPF Interface.....	65
XEDIT Interface.....	66
COUNT.....	67
COVERLAY.....	68
CREATE.....	68
CREATE BUTTON.....	69
CREPLACE.....	71
CURSOR.....	71
CUT.....	72
DEFINE.....	73
DELETE.....	74
ISPF Interface.....	74
XEDIT Interface.....	75
DESTROY BUTTON.....	76
DIALOG.....	77
DOWN.....	78
ISPF Interface.....	78
XEDIT Interface.....	79
DSN.....	80
DUPLICATE.....	81
ECOMMAND.....	81
EDIT.....	82
EDITV.....	85
EMSG.....	86
END.....	87
EQU.....	87
EXCLUDE, X.....	88
EXTRACT.....	90
FILE, FFILE.....	91
FILEKIT.....	92
FILLBOX.....	92
FILLDIALOG.....	93
FIND, FINDUP.....	93
ISPF Interface.....	93
XEDIT Interface.....	96
FINDDIALOG.....	97
FLIP.....	97
FORWARD.....	98
FREE.....	98
GEN.....	99

# Contents

## Primary (Command Line) Commands

GENCOMP.....	101
GENORPH.....	102
GET.....	103
GO.....	104
HELP.....	104
HEX.....	105
HIDE.....	105
ICOMMAND.....	106
IMMEDIATE.....	106
INPUT.....	107
JOIN.....	107
LABEL.....	108
LEFT.....	109
ISPF Interface.....	109
XEDIT Interface.....	110
LESS.....	110
LINE.....	111
LINE_AFTER.....	112
LINE_BEFORE.....	112
LIST.....	113
LOCATE.....	115
ISPF Interface.....	116
XEDIT Interface.....	117
LOWERCASE.....	118
MACRO.....	119
MARK.....	119
MORE.....	120
MOVE.....	121
ISPF Interface.....	121
XEDIT Interface.....	121
MSG.....	122
NFIND, NFINDUP.....	123
NOMSG.....	124
NOND.....	124
NORECALL.....	124
ONLY.....	125
OPEN.....	126
OPTIONS.....	126
OVERLAY.....	127
OVERLAYBOX.....	127
PASTE.....	128
POPUP.....	128
PRESERVE.....	129
PURGE.....	129
QUERY.....	130
QUEUE.....	130
QUIT, QQUIT.....	131
RCHANGE.....	131
REDO.....	132
REFRESH.....	132
REPLACE.....	133
ISPF Interface.....	133
XEDIT Interface.....	134
RESET.....	134
ISPF Interface.....	135
XEDIT Interface.....	136
RESTORE.....	136
RFIND.....	137
RIGHT.....	137
ISPF Interface.....	137
XEDIT Interface.....	138
RUNSELCOPY.....	139
RUNSLC.....	139
SAVE, SSAVE.....	140
SDATA.....	141
SET.....	141
SETPT.....	142
SHIFT.....	142
SORT.....	143
ISPF Interface.....	143
XEDIT Interface.....	145
SOS.....	146
SPLIT.....	146
ISPF Interface.....	147
XEDIT Interface.....	147
SPLTJOIN, SJOIN.....	148

# Contents

## Primary (Command Line) Commands

STEMINSERT.....	148
SUBMIT.....	149
SYNEX.....	149
SYSCOMMAND, TSO, CMS, DOS.....	150
SYSEdit.....	150
TAG.....	151
TFIND.....	151
TFLOW.....	152
TOP.....	152
ISPF Interface.....	152
XEDIT Interface.....	153
TSPLIT.....	153
UNDO.....	154
UP.....	154
ISPF Interface.....	154
XEDIT Interface.....	155
UPPERCASE.....	156
VIEW.....	156
VIGNORE.....	157
VRESPECT.....	158
WINDOW.....	159
WINDOWCOMMAND.....	161

## SET/QUERY/EXTRACT Options..... 162

ACTION - QUERY/EXTRACT.....	163
ACTIONCOMMENT - SET/QUERY/EXTRACT.....	164
ACTIONCURSOR - SET/QUERY/EXTRACT.....	164
ACTIONDELIM - SET/QUERY/EXTRACT.....	165
ALT - SET/QUERY/EXTRACT.....	166
ARBCHAR - SET/QUERY/EXTRACT.....	166
AUTOSAVE - SET/QUERY/EXTRACT.....	167
BEEP - SET/QUERY/EXTRACT.....	168
BLOCK - QUERY/EXTRACT.....	169
CASE - SET/QUERY/EXTRACT.....	169
CHANGE - EXTRACT.....	170
CLIPBOARD - QUERY/EXTRACT.....	171
CMDDEF - SET/QUERY/EXTRACT.....	171
CMDLINE - SET/QUERY/EXTRACT.....	172
COLOR, COLOUR - SET/QUERY/EXTRACT.....	173
COLUMN - QUERY/EXTRACT.....	174
COUNT - EXTRACT.....	175
CURLINE - EXTRACT.....	175
CURSOR - QUERY/EXTRACT.....	176
DEFPROFILE - SET/QUERY/EXTRACT.....	176
DIALOG - EXTRACT.....	177
DISPLAY - SET/QUERY/EXTRACT.....	177
DSN - SET/QUERY/EXTRACT.....	178
DSORG - SET/QUERY/EXTRACT.....	179
ENVVARS - SET/QUERY/EXTRACT.....	180
EOLIN - SET/QUERY/EXTRACT.....	181
EOLOUT - SET/QUERY/EXTRACT.....	182
FIDCHANGED - SET/QUERY/EXTRACT.....	183
FILEID - SET/QUERY/EXTRACT.....	184
FLSCREEN - QUERY/EXTRACT.....	185
FMODE - SET/QUERY/EXTRACT.....	185
FNAME, MBR - SET/QUERY/EXTRACT.....	186
FPATH - SET/QUERY/EXTRACT.....	187
FTYPE - SET/QUERY/EXTRACT.....	188
GENSAVE - SET/QUERY/EXTRACT.....	188
HCOLOR, HCOLOUR - SET/QUERY/EXTRACT.....	190
HEXSTRING - SET/QUERY/EXTRACT.....	191
HILITE, HILIGHT - SET/QUERY/EXTRACT.....	192
HSCROLLCURSOR - SET/QUERY/EXTRACT.....	193
IMPMACRO - SET/QUERY/EXTRACT.....	194
INIFILE - QUERY/EXTRACT.....	194
INIVAR - SET/UNSET/QUERY/EXTRACT.....	195
INSTANCE - SET/QUERY/EXTRACT.....	196
INTERFACE - SET/QUERY/EXTRACT.....	197
ISPFMODE - SET/QUERY/EXTRACT.....	198
KEY - SET/QUERY/EXTRACT.....	199
LASTMSG - QUERY/EXTRACT.....	199
LCOLOR, LCOLOUR - SET/QUERY/EXTRACT.....	200
LENGTH - QUERY/EXTRACT.....	201
LINE - QUERY/EXTRACT.....	202
LINEFLAG - SET/QUERY/EXTRACT.....	202
LINEND - SET/QUERY/EXTRACT.....	203

# Contents

<b>SET/QUERY/EXTRACT Options</b>	
LISTFILEACTION - SET/QUERY/EXTRACT.....	204
LOADWARNING - SET/QUERY/EXTRACT.....	205
LRECL - SET/QUERY/EXTRACT.....	206
LSCREEN - QUERY/EXTRACT.....	206
MACRO - QUERY.....	207
MACROPATH - SET/QUERY/EXTRACT.....	208
MBR - SET/QUERY/EXTRACT.....	209
MSGLINE - SET/QUERY/EXTRACT.....	209
MSGMODE - SET/QUERY/EXTRACT.....	210
NBWINDOW - QUERY/EXTRACT.....	211
OPSYS - QUERY/EXTRACT.....	211
PFKEY - SET/QUERY/EXTRACT.....	212
POINT - SET/QUERY/EXTRACT.....	213
POPUP - EXTRACT.....	214
PREFIX - SET/QUERY/EXTRACT.....	214
PSCOPE - SET/QUERY/EXTRACT.....	215
RANGE - SET/QUERY/EXTRACT.....	216
RECFM - SET/QUERY/EXTRACT.....	217
REDO - QUERY/EXTRACT.....	218
RESERVED - SET/QUERY/EXTRACT.....	218
RING - QUERY/EXTRACT.....	219
SAVEOPTIONS - SET/QUERY/EXTRACT.....	219
SCALE - SET/QUERY/EXTRACT.....	220
SCOLOR, SCOLOUR - SET/QUERY/EXTRACT.....	221
SCOPE - SET/QUERY/EXTRACT.....	222
SCREEN - QUERY/EXTRACT.....	223
SELECT - SET/QUERY/EXTRACT.....	223
SHADOW - SET/QUERY/EXTRACT.....	225
SIZE - QUERY/EXTRACT.....	225
SIZEWARNING - SET/QUERY/EXTRACT.....	226
STAY - SET/QUERY/EXTRACT.....	227
STREAM - SET/QUERY/EXTRACT.....	228
SYNONYM - SET/QUERY/EXTRACT.....	228
THIGHLIGHT - SET/QUERY/EXTRACT.....	230
UNDO - QUERY/EXTRACT.....	231
UNDOING - SET/QUERY/EXTRACT.....	231
USERNAME - QUERY/EXTRACT.....	232
VARBLANK - SET/QUERY/EXTRACT.....	233
VERSION - QUERY/EXTRACT.....	233
VIEW - SET/QUERY/EXTRACT.....	234
WINNAME - SET/QUERY/EXTRACT.....	235
WINPOS - SET/QUERY/EXTRACT.....	236
WINSIZE - SET/QUERY/EXTRACT.....	237
WRAP - SET/QUERY/EXTRACT.....	238
ZONE - SET/QUERY/EXTRACT.....	238
<b>Prefix Commands.....</b>	<b>240</b>
<b>Function Keys.....</b>	<b>241</b>
<b>Glossary.....</b>	<b>243</b>

# Documentation Notes

---

Fifth Edition, April 2024

Information in this document details general features and functionality of the Text Editor application which is part of the **CBL Product Suite** component, **FileKit**.

Copyright in the whole and every part of this document and of the CBL Product Suite system and programs, is owned by Compute (Bridgend) Ltd (hereinafter referred to as CBL), whose registered office is located at 8 Merthyr Mawr Road, Bridgend, Wales, UK, CF31 3NH, and who reserve the right to alter, at their convenience, the whole or any part of this document and/or the CBL Product Suite system and programs.

CBL Product Suite for z/OS, z/VM (CMS) and z/VSE operating systems, which includes SELCOPY, FileKit and CBLVCAT, is available for download and install from [www.cbl.com/selcdl.php](http://www.cbl.com/selcdl.php).

The following publications for CBL Product Suite and its component products are available in Adobe Acrobat PDF format at CBL web page [www.cbl.com/documentation.php](http://www.cbl.com/documentation.php):

- CBL Product Suite Customisation Guide
- SELCOPY User Manual
- SELCOPY C++ (SLC) Language Reference
- CBLVCAT User Manual
- FileKit Reference and User Guide
- FileKit Text Editor
- FileKit Data Editor (SDE)
- FileKit Quick Reference
- FileKit REPORT Utility
- FileKit SMF Utilities
- FileKit Training Manual

No reproduction of the whole or any part of the CBL Product Suite system and programs, or of this document, is to be made without prior written authority from Compute (Bridgend) Ltd.

At the time of publication, this document is believed to be correct. Where the program product differs from that stated herein, Compute (Bridgend) Ltd reserve the right to revise either the program or its documentation at their discretion. CBL do not warrant that upward compatibility will be maintained for any use made of this program product to perform any operation in a manner not documented within the user manual.

The following generic terms are used throughout this document to indicate all available versions and releases of IBM mainframe operating systems:

<b>z/OS</b>	-	z/OS, OS/390, MVS/ESA, MVS/XA, MVS/SP, OS.
<b>z/VSE</b>	-	z/VSE, VSE/ESA, VSE/SP, DOS.
<b>z/VM CMS</b>	-	z/VM, VM/ESA, VM/XA, VM/SP.
<b>All</b>	-	All z/OS, z/VSE and z/VM CMS operating systems.



# Summary of Changes

---

## First Edition (October 2013)

---

This section is a summary of significant new features provided in SELCOPYi Release 3.20.

### ISPF Edit Rexx Macros

Support execution of ISPF Edit macros that have been written in the Rexx programming language and exist in the SYSUEXEC or SYSEXEC library concatenations.

For details, see:

- ◇ [ISPF Edit Macros](#)

### Syntax Colour Highlighting

Introduce syntax colour highlighting of text displayed in a text edit view for the following programming languages:

<b>ASSEMBLER</b>	<b>JCL</b>
<b>C</b>	<b>PL1</b>
<b>CMX</b>	<b>REXX</b>
<b>COBOL</b>	<b>SELCOPY</b>
<b>HTML</b>	<b>XML</b>

For details, see:

- ◇ [SET HILITE / HILIGHT](#)
- ◇ [SET HCOLOR / HCOLOUR](#)

### Regular Expressions

Regular expressions may be used as arguments to commands that involve a string search for complex pattern matching. These commands are LOCATE, CHANGE and ALL for the CBLi Interface, and FIND, CHANGE and EXCLUDE for the ISPF Interface.

For details, see:

- ◇ [Regular Expressions](#)
- ◇ [ALL](#)
- ◇ [CHANGE](#)
- ◇ [LOCATE](#)

### Environment Settings for ACTION (CMDTEXT)

The preferred synonym ACTION is now used for primary command CMDTEXT.

Settings introduced to govern the significance of underscore ('\_') and OR symbol ('|') in command string text.

For details, see:

- ◇ [ACTION](#)
- ◇ [SET ACTIONCURSOR](#)
- ◇ [SET ACTIONDELIM](#)
- ◇ [Text Edit Settings \(=0.3\)](#)

### Non-Displayable Characters

Control appearance of displayable character data in text edit lines which also contain non-displayable characters.

For details, see:

- ◇ [NOND](#)
- ◇ [SET COLOR / COLOUR](#)

**Text Edit <-> Structured Data Edit/Browse**

Easily switch display of the loaded data between the text edit and SDE data edit utilities.

For details, see:

- ◇ [GO](#)
- ◇ [Text/Data Edit GO Options](#) panel

**Unset User INI variables**

The user's current configuration is saved automatically on closing SELCOPYi and used to initialise the user's environment on subsequent start up of SELCOPYi. A method has been introduced to allow unsetting of specific user initialisation variables if required.

For details, see:

- ◇ [SET INIVAR](#)

**Second Edition (March 2015)**

This section is a summary of significant new features provided in SELCOPYi Release 3.30.

**Regular Expression Octal Escape Sequences**

Support specification of characters as octal escape sequences in regular expressions. e.g. Octal 121 (\121) is equivalent to hex 51 (\x51) is equivalent to decimal 81.

For details, see:

- ◇ [Regular Expressions](#)

**ACTION Facility Comment Indicator String**

Support option ACTIONCOMMENT to specify a character (or string of characters) to represent the start of comment data in a line of text processed by the ACTION facility. (Default F16)

For details, see:

- ◆ [SET ACTIONCOMMENT](#)

**Syntax Colour Highlighting - AUTO**

Update syntax colour highlighting to perform automatic language detection.

For details, see:

- ◇ [SET HILITE / HILIGHT AUTO](#)

**Third Edition (January 2017)**

This section is a summary of significant new features provided in SELCOPYi Release 3.40.

**EDIT**

The behaviour of the EDIT primary command has been updated so that use of parameters "+" (plus) and "-" (minus) perform the same operation as MDINEXT (EDIT +) and MDIPREV (EDIT -).

Previously, EDIT and EDIT - (minus) would place focus on the next or previous text edit document view respectively within the Text Editor application window frame. This did not include any non-text edit document views (panels, lists, data edit views, etc.) that were also opened within the Text Editor frame window.

Performing MDINEXT and MDIPREV operations means that EDIT can navigate from a text edit view to a document window of any window class. Note that primary command WINDOW (assigned to F4 by default) or WINDOW - (minus) is the preferred method of navigating between document windows since WINDOW is supported by all classes of application windows.

For details, see:

- ◇ [EDIT](#)

## EQU

Include documentation for the EQU edit macro. EQU may be used to set and unset text and data editor environment variables in place of the EDITV primary command.

For details, see:

- ◆ EQU

## LIST

Primary command LIST has been updated to support list type parameter CLD. CLD will list details of the first occurrence of a member name (including the library in which it was found) within a concatenated library directory search path.

For details, see:

- ◆ LIST

## RUNSELCOPY and RUNSLC

Primary commands RUNSELCOPY and RUNSLC have been updated to support parameters that specify the name of the SELCOPY or SLC executable program and an input parameter string to be passed to the program.

For details, see:

- ◆ RUNSELCOPY
- ◆ RUNSLC

## WINDOWCOMMAND

Obsolete primary command WINDOWCOMMAND specification changed so that it may now be used to direct command strings to named window views.

For details, see:

- ◆ WINDOWCOMMAND

---

## Fourth Edition (July 2020)

---

This section is a summary of significant new features provided in SELCOPYi Release 3.50.

### PDSE V2 Member Generations

General support for processing member generations in PDSE version 2 libraries allocated with a MAXGENS value. Utilities that support member generations include Text Editor VIEW and EDIT, Data Editor BROWSE and EDIT, File Copy, File Search & Update, File Compare.

For details, see:

- ◇ z/OS PDSE Library Member Generations in SELCOPYi reference documentation.
- ◇ GEN in CBL text editor documentation.
- ◇ GENCOMP in CBL text editor documentation.
- ◇ GENORPH in CBL text editor documentation.

### Macro Enhancements

Macros supplied in a compiled REXX format where possible, and support included for execution of a macro internationally for different local CCSIDs.

For details, see:

- ◇ REXX Macros
-

## Fifth Edition (February 2024)

---

This section is a summary of significant new features provided in FileKit Release 3.60.

### Product Rebrand

Starting at release 3.60 (and release 3.50 with PTF RS35003 applied), "**SELCOPYi**" is rebranded as "**FileKit**".

Product materials now have name and/or aliases that reflect the product name change. For example, the FileKit batch executable (ZZSSMAIN) has both alias names **SDEAMAIN** and **FILEKITB**.

---

# Introduction to the CBL Text Editor

The CBL text editor (CBL) may be used to perform text edit on the following:

- z/OS PDS/PDSE members.
- z/OS Sequential Data Sets.
- z/OS HFS Files.
- CMS files.
- VSE LIBR members.
- VSAM KSDS, ESDS, RRDS, VRDS, AIX & PATH.

The CBL edit environment is a hybrid of the IBM ISPF Editor and CMS XEDIT, supporting command syntax and data display indicative of both edit environments.

The user interface is modelled on PC style Multiple Document Interface (MDI) standards (although limited by restrictions imposed by 3270 display).

Since its inception, the CBL text editor has been expanded beyond the scope of simple text edit. In its current form, CBL also supports the CBLVCAT execution window and all other list and dialog windows that have traditionally been the domain of the FileKit main window only. See the "FileKit Reference and User Guide" for a complete description of these window types and their usage.

For users who have a SELCOPY licence, CBL also supports the Structured Data Environment (SDE), a more advanced type of edit for data sets whose records have a pre-determined structure as defined by a COBOL or PL/1 copy book. This feature enables the user to edit data within the strict confines of the defining structure with records being displayed as a number of individual fields. This subject is described in detail in the FileKit "Structured Data Environment (SDE)" documentation.

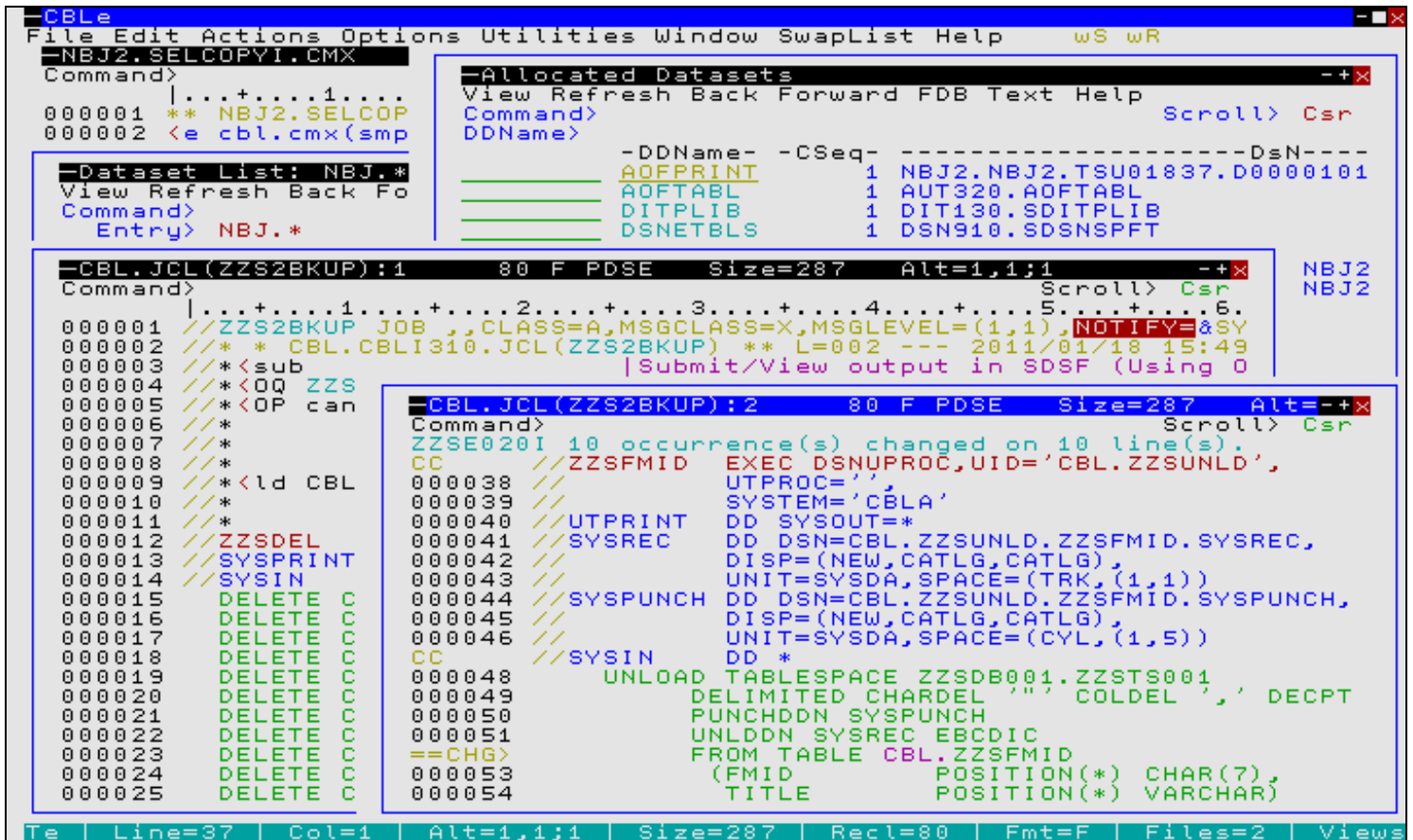


Figure 1. CBL Window.

## CBL Main Window

The FileKit text editor (CBL) main window (also called the CBL MDI frame window) defines the area in which supported client application MDI child windows are displayed.

The client area of the window includes a menu bar in the first line of the display and, if applicable, an information bar for the focus MDI child window in the last line of the display.

The remainder of the client area may be occupied by document (child) windows, one for each CBL text edit view, SDE structured data edit view, list window, IPO panel, etc.

If the FileKit User INI file variable **Edit.Instance** is set to **Multiple** or the CBL SET option **INSTANCE MULTIPLE** is in effect, then any number of CBL main windows may be opened within the FileKit session.

By default, the MDI frame windows of all MDI applications supported by FileKit (i.e. CBL and SELCOPY Debug) are opened in a maximised state.

```

File Edit Actions Options Utilities Window SwapList Help      wS wR      Scroll> Csr
Command>

(C)2010 Compute(Bridgend) Ltd                               User:  NBJ2
web:  http://www.cbl.com                                   Version: 3.20
email: support@cbl.com                                     Date:  2013/10/30
                                                    Time:  10:23:57
                                                    OpSys:  z/OS 1.11.0

Enter the H0me command to edit your
SELCOPY/i command text file.

[Title Bar: NBJ.INST.CBL1329]

Te | Line=1 | Col=1 | Alt=0,0;1 | Size=130 | Recl=32752 | Fmt=V | Files=2 | Vie

```

Figure 2. CBL Main Window.

## CBL Main Window Menu Bar

When MDI child windows are in a maximised state, the contents of this menu bar are governed by the **window class** of the focus child window.

When MDI child windows are in a non-maximised state, the CBL main window menu bar contains the following items:

### File

<b>New</b>	Edits a new document window.
<b>Open ...</b>	Opens the <b>Open dialog</b> window allowing the user to browse for the file to be edited.
<b>Close</b>	Close the current document window.
<b>Structured edit ...</b>	Open the <b>SDE Edit</b> dialog.
<b>Save</b>	Save the file in the current document window with the current fileid.
<b>Save As ...</b>	Save the file in the current document window and prompt for fileid.
<b>SELCOPY Debug/Dev ...</b>	Open the <b>SELCOPY/batch Debug and Development</b> panel.
<b>CBLVCAT Interactive ...</b>	Open the <b>Execute CBLVCAT (VCI)</b> window.
<b>DB2 ...</b>	Open the <b>DB2 Utilities</b> panels.
<b>Allocate NonVSAM ...</b>	Open the <b>Allocate NonVSAM</b> dialog.
<b>Define KSDS ...</b>	Open the <b>Define KSDS</b> dialog.
<b>Define ESDS ...</b>	Open the <b>Define ESDS</b> dialog.
<b>Define RRDS ...</b>	Open the <b>Define RRDS</b> dialog.
<b>Define GDG ...</b>	Open the <b>Define GDG Base</b> dialog.
<b>Define LDS ...</b>	Open the <b>Define LDS</b> dialog.
<b>Execute IDCAMS ...</b>	Open the <b>IDCAMS Command</b> window.
<b>Create Catalog ALIAS</b>	Open the <b>Create Catalog ALIAS</b> panel.
<b>Create Library ALIAS</b>	Execute the <b>ALIAS</b> FileKit CLI Command to open the Create ALIAS panel for Library members.
<b>Execute IEBCOPY ...</b>	Open the <b>IEBCOPY</b> utility panel.
<b>Exit</b>	Exit the current CBL main window.

### Edit

<b>Undo</b>	Undo change levels in the current document window.
<b>Redo</b>	Redo change levels that have been undone in the current document window.
<b>Cut</b>	Remove currently marked text and place in the FileKit clipboard.
<b>Cut append</b>	Remove currently marked text and append it to existing data in the FileKit clipboard.
<b>Copy</b>	Copy currently marked text to the FileKit clipboard.
<b>Copy append</b>	Append currently marked text to the FileKit clipboard.
<b>Paste</b>	Paste text from the FileKit clipboard to the last cursor position.
<b>Clear clipboard</b>	Clear the FileKit clipboard
<b>Select All</b>	Mark all text in the file area of the current document window.

<b>Reset Block</b>	Reset marked text in the file area of the current document window.
<b>Find ...</b>	Open the <b>Find dialog</b> to search the data in the current document window for a specified text string.
<b>Change ...</b>	Open the <b>Change dialog</b> to search the data in the current document window for a specified text string and replace occurrences with a new text string.
<b>System Edit ...</b>	Open the default system editor (XEDIT, ISPF Edit) to edit the current file.

**Actions**

<b>Sort ...</b>	Open the <b>Sort dialog</b> to sort data in the current document window.
<b>Fill ...</b>	Open the <b>Fill dialog</b> to fill a marked block.
<b>Uppercase</b>	Upper case all text in the currently marked block.
<b>Lowercase</b>	Lower case all text in the currently marked block.

**Options**

<b>Query SET Options</b>	Display settings of all <b>SET/QUERY/EXTRACT Options</b>
<b>Edit SET Options</b>	Edit a temporary CMX command file containing <b>SET</b> option commands.

**Utilities**

<b>List</b> - Open a sub-menu of List window items:	
<b>DASD Volumes ...</b>	Open the <b>DASD Volumes</b> list window.
<b>Cataloged files ...</b>	Open the <b>Cataloged files</b> list window.
<b>Dataset details ...</b>	Open the <b>Dataset details</b> list window.
<b>VTOC files ...</b>	Open the <b>VTOC files</b> list window.
<b>VTOC extents ...</b>	Open the <b>VTOC extents</b> list window.
<b>Allocated files ...</b>	Open the <b>Allocated files</b> list window.
<b>Library members ...</b>	Open the <b>Library members</b> list window.
<b>Enqueues ...</b>	Open the <b>Enqueues</b> list window.
<b>Job Enqueues ...</b>	Open the <b>Job Enqueues</b> list window.
<b>HFS Dir Path ...</b>	Open the <b>HFS Path</b> list window.
<b>SMS Storage Groups ...</b>	Open the <b>HFS Path</b> list window.
<b>SMS StorGrp Vols ...</b>	Open the <b>HFS Path</b> list window.
<b>System</b> - Open a sub-menu of System window items:	
<b>Operating system ...</b>	Open the <b>Operating System</b> window.
<b>FileKit storage stats ...</b>	Open the FileKit <b>Storage statistics Block</b> window.
<b>FileKit module list ...</b>	Open the FileKit <b>Module List</b> window.
<b>FileKit SVC ...</b>	Open the FileKit <b>CBLVCAT SVC</b> information window.
<b>CBLNAME ...</b>	Open the <b>CBLNAME</b> storage display window.
<b>Favourites ...</b>	Open the <b>Favourite Datasets/Commands</b> panel.
<b>File Copy ...</b>	Open the <b>File Copy</b> panel.
<b>File Search ...</b>	Open the <b>File Search</b> window.
<b>File Search Update Remap ...</b>	Open the <b>File Search/Update/Copy/Remap</b> panels.
<b>Create File Filter ...</b>	Open the <b>Create File Filter</b> panels for structured data record filtering.
<b>Search for Library Members ...</b>	Open the <b>Search for Library Members</b> panel.
<b>Compare Files ...</b>	Open the <b>Compare Files</b> panel.
<b>Compare Libraries ...</b>	Open the <b>Compare Libraries</b> panel.
<b>Calendar ...</b>	Open the <b>Calendar</b> window.
<b>Calculator ...</b>	Open the <b>Calculator</b> window.
<b>SDSF</b>	Under TSO or ISPF, call the SDSF JES2 spool access program.
<b>ISPF Dataset Utilities</b>	Under ISPF, call the ISPF dataset utilities menu panel.

**Window**

<b>New Window</b>	Open a document window containing a new view of the file in the current document window.
<b>All Windows</b>	Open the <b>Window List</b> window. Select new focus window.
<b>Cascade</b>	Cascade the document windows.
<b>Tile Horizontally</b>	Tile the document windows horizontally.
<b>Tile Vertically</b>	Tile the document windows vertically.
<b>Arrange..</b>	Open a popup menu to arrange current edit views horizontally/vertically. (Not activated.)
<b>Arrange Minimised</b>	Arrange the minimised document windows.
<b>title1 ... titleN</b>	Lists all document windows within the the CBL Main window. Select new focus document window.

## SwapList

Displayed only if ISPF is the 3270 screen manager. Execute the ISPF SWAPLIST command.

## Help

<b>Contents</b>	Open the help documents page.
<b>FileKit Environment</b>	Open the FileKit Reference and User Guide help contents page.
<b>CBL Text Edit</b>	Open the FileKit Text Editor (CBL e) help contents page.
<b>SDE Edit</b>	Open the FileKit Structured Data Environment (SDE) help contents page.
<b>SELCOPY Manual</b>	Open the SELCOPY user manual.
<b>CBLVCAT Manual</b>	Open the CBLVCAT user manual.
<b>Help Topic Index ...</b>	Open the <a href="#">FileKit Help Index</a> dialog.
<b>About FileKit ...</b>	Display FileKit Release and Build level information.

## CBL Main Window Status Bar

The CBL main window Status bar contains the following items:

<b>Te/Se</b>	Current (or last visited) edit view is a <b>Text Edit</b> (Te) window view or <b>Structured Edit</b> (Se) window view.
<b>Line=<i>n</i></b>	Line number of current line in current document window. "?" is displayed if the line number is unknown. (e.g. SDE KSDS edit by KEY)
<b>Col=<i>n</i></b>	Column number of current column in current document window.
<b>Alt=<i>n1,n2;x</i></b>	Alterations made since last save or autosave, last save and number of undo levels. An "*" (asterisk) indicates that Redo levels may be applied.
<b>Size=<i>n</i></b>	Number of records in the file within the current document window. <b>Size&gt;<i>n</i></b> is displayed if the SDE edit technique does not automatically read all records from the file and the data is displayed for edit before end-of-file is reached. For SDE segmented record edit, the size value will be suffixed by <b>(P)</b> , indicating number of physical records; or <b>(S)</b> , indicating number of segments.
<b>Recl=<i>n</i></b>	Lrecl of the file within the current document window.
<b>Fmt=<i>X</i></b>	Record Format (RECFM) of the file within the current document window.
<b>Files=<i>n</i></b>	Number of files being edited in this CBL main window.
<b>Views=<i>n</i></b>	Total number of views of all files edited in this CBL main window (equal to number of document windows)
<b>yyyy/mm/dd HH:MM:SS</b>	Current local date and time in columns 91 to 109. Since its position is beyond column 80, this is only visible on dynamic 3270 terminals with a greater number of columns than that provided by a standard 3270 terminal.

## CBL Text Edit Document Window

The CBL main (frame) window is an MDI parent window which supports a number of different types of document window, e.g. SDE Edit view, Execute CBLVCAT window, Data Set List window, etc. The format and capabilities of these types of window are discussed in relevant sections of the "[FileKit Reference and User Guide](#)" and "[FileKit Structured Data Environment \(SDE\)](#)" manuals.

It should be noted that the SELCOPY/Batch Debug and Development feature of FileKit is also an MDI frame window which supports all the same document windows supported by the CBL frame window, including CBL text edit documentation windows.

This CBL Text Edit documentation focuses exclusively on properties and command syntax applicable to CBL text edit document windows.

A CBL text edit document window (edit view) has a standard FileKit format title bar with [system menu](#), [minimise](#), [maximise](#), [restore](#) (if applicable) and [close](#) buttons. The title bar also contains the fileid, LRECL, RECFM, DSORG, file size (number of records) and alteration count for the file being edited. Furthermore, the literal **(Read Only)** is displayed in the title bar if the user does not have read/write authority for the file. Therefore, where the title bar of a non-maximised edit view is visible, the characteristics of the file being edited may be determined without having to make the edit view the focus window.

The text edit document window also contains a command prompt which are not considered part of the document window client area.

The client area of a document window contains a horizontal scale in the first line of the display and, optionally, a prefix area displaying line numbers. Before and after the editable text, the client area includes a "Top of File" and "End of File" marker.

The remainder of the client area is the file display area and contains the editable text.



Where edited record data includes non-printable text, then only the printable characters in that record may be edited. These editable areas of text occupy immovable fixed positions and lengths within the record, represented by underscores. These records are displayed in a different colour to records containing no non-printable text. Non-printable characters may only be updated using the CHANGE command or by setting HEX ON, so allowing oertype of the hex display of the data.

```

NBJ2.SELCOPYI.CMX      255 V SEQ      Size=1136      Alt=0,0;5
Command>
.....1.....2.....3.....4.....5.....6.....+...
.MISC      ** Miscellaneous Samples      *** .misc
000628      Issue operator command ...
000629 <Op d m=stor      | Check available memory.
000630
000631      Direct access to ISPF panels ... (assumes ISPF command delim=';')
000632 <;iex 3.4      | Access ISPF Data Set List Utility.
000633 <;iex m.5; da; scname SDSF#DA; pre *; sort cpu-time d | SDSF.
000634 <;iex panel(isrutil);=1;=i
..
000636      "Note:" SELCOPY/i's internal command delimiter is ";" by default
000637      but use of ";" as a command-prefix causes SELCOPY/i to
000638      suspend its own parsing of the command-line, allowing
000639      (in the above case) ISPF to handle the delimiter.
000640
000641      Issue AMS (IDCAMS) command ...
000642 <ams LISTCAT ENTRY(%dsn%)      | IDCAMS Listcat for current file.
000643
000644      Issue DB2 query ...
000645 <sql -limit=100
000646      select NAME,TBNAME,TBCREATOR from SYSIBM.SYSCOLUMNS
000647      where TBNAME='SYSCOLUMNS'
000648
000649 | System information ... (also available from 'Utilities->System')
000650 <SysI      Operating System information.
000651 <SysLPA      Link Pack Area.
000652 <SysLL      LinkList libraries.
000653 <SysAPF      APFList libraries.
000654 <SysTask      Task List.
000655 <SysStor      Storage statistics block.
000656 <SysPgm      Loaded Programs.
000657      "Note:" The display of System Information, and other features of
000658      SELCOPY/i may be restricted using RACF profiles.
000659 <help RACF
000660
000661 | List commands ...      ***
000662 <lvol      z*      | List DASD Volumes ...

```

Figure 3. CBL Text Edit Document Window.

## CBL Current Window

The current CBL window is the CBL text edit document view on which focus was last placed. This may be the focus window or, if the focus window is not a CBL text edit document view, the last CBL text edit view visited.

The concept of a current CBL window view is important when executing CBL CLI commands from an SDE window display via the **TEDIT** SDE CLI command or selecting CBL main menu bar items.

In both cases, the current CBL window will be the target of the operation.

## CBL SDE Document Window

In addition to CBL text edit document windows, the CBL frame window also supports Structured Data Environment **SDE window views** as one of its MDI child (document) windows.

SDE is supported in MVS environments only where a SELCOPY licence is active.

An SDE edit/browse window view may be opened to edit/display data set records that have an associated structure generated from a COBOL or PL1 copybook, using one of the following methods:

1. Select "Structured edit" to open the SDE edit dialog window from the "File" item of the CBL main menu bar.
2. From a CBL command prompt, execute the CBL **SDATA** CLI command to invoke an SDE **EDIT** or **BROWSE** CLI command.
3. Execute Prefix command "B" (SDE Browse) or "SE" (SDE edit dialog) from any file List or Execute CBLVCAT window.

To distinguish an SDE edit document view from a CBL text edit document view, the edit type flag, displayed on the left of the CBL status bar, reflects the type of edit of the current (last in focus) edit view. This flag is either "Te" for CBL Text Edit or "Se" for SDE Structured Edit.

Concepts and functionality of the Structured Data Environment are documented at length in the [SDE Edit help](#) documentation.

```
File Edit Actions Options Utilities Window SwapList Help      wS wR
Edit CBL.DIST.CBLI.SDE.SAMP.VAR(DATSALES) using NBJ2.CBLI.SDO(COBSALE-+x
Command>
Record type: REC-PAYMENT      Fixed(26) Offset=0 Data elements=6
      CUST-ID RECEIVED-DATE      CARD-NUMBER      ORDER-REF      AMOUNT
      #2          #3          #4          #5          #6
      FB 1:4      PD 5:5      PD 10:9      FB 19:4      PD 23:4
      <---+---1>      <---+--->      <---+---1---+>      <---+---1>      <---+--->
00000028      9156      20070519 1282937860235632      22075050      1715.34

Record type: REC-CARD      Fixed(58) Offset=0 Data elements=11
      CUST-ID      SEQ CR-OR-DR      COMPANY      CARD-NUMBER      NAME
      #2          #3 #5          #6          #7 #8
      FB 1:4      FB 5:2 AN 7:1      AN 8:7      PD 15:9 AN 24:25
      <---+---1>      <---+>      <---+>      <---+>      <---+---1---+>      <---+---1--->
00000029      2991      0 X      Aqua      8627368960235435      Jacque Couste
00000030      6792      6 W      VISA      9463829326644146      BRUCE W HOWAR

Record type: REC-ORDER      Variable(92,112) Offset=0 Data elements=20
      CUST-ID      ORDER-REF      QTY      ITEM-CODE      UNIT-PRICE      DELIVERY
      #2          #3          #4          #5          #6          #7
      FB 1:4      FB 5:4 FB 9:2      FB 11:4      PD 15:4      PD 19:3
      <---+---1>      <---+---1>      <---+>      <---+---1>      <---+--->      <---+>
00000031      4169      497047541      6      16967      35.28      3.06
00000032      9156      122075050      9      14757      190.37      2.01
00000033      2532      819485175      8      20508      -117.70      -5.84
00000034      1427      104613891      7      12951      168.96      3.03
00000035      4484      938490727      10      29782      93.75      7.86
00000036      9156      512510042      9      16036      57.24      6.82
00000037      8348      276846581      8      14305      106.95      3.82
00000038      4046      802966435      8      29565      167.47      9.74

Record type: REC-PAYMENT      Fixed(26) Offset=0 Data elements=6
      CUST-ID RECEIVED-DATE      CARD-NUMBER      ORDER-REF      AMOUNT
      #2          #3          #4          #5          #6
      FB 1:4      PD 5:5      PD 10:9      FB 19:4      PD 23:4
      <---+---1>      <---+--->      <---+---1---+>      <---+---1>      <---+--->
00000039      2991      20071129 1098717650442152      980782594      730.01
00000040      2991      20070620 8627368960235435      95335803      -3.09
00000041      9928      20060120 9463829326644146      319314355      168.77

Se | Line=28 | Col=1 | Alt=0,0;0 | Size=63 | Recl=16384 | Fmt=V | Files=1 | Vie
```

Figure 4. CBL SDE Document Window.

## CBL List, DB2 SQL & CBLVCAT Document Windows

In addition to FileKit Text-Edit and SDE edit document windows, the Text-Edit frame window also supports [FileKit List windows](#), [DB2 Dynamic SQL windows](#) and [CBLVCAT Interactive Windows](#) as one of its MDI child (document) windows.

This may be done as follows:

1. Select "List" and the required list type from the "Utilities" item of the CBL main menu bar.
2. Select "DB2 Dynamic SQL" from the "File" item of the CBL main menu bar.
3. Select "Execute CBLVCAT" from the "File" item of the CBL main menu bar.
4. From within the CBL environment, execute a FileKit line command. (e.g. LD, LC, LVOL, LA, SQL, VCAT.)
5. From an existing List or Execute CBLVCAT window within the CBL environment, execute a prefix command that opens another list class window. (e.g. "M" - member list, "L" - List Data set, "VC" - Execute CBLVCAT.)

The format and functionality of a List, DB2 SQL or Execute CBLVCAT window opened as a CBL MDI child window is no different to that when it is opened as a child of the FileKit main window. By default, the FileKit environment command **WINDOW** is assigned to function key **F4** in all CBL child windows. This allows the user to pass focus from the current CBL child window to the next opened CBL child window which may be a CBL text edit, SDE structured edit, DB2 Dynamic SQL, Execute CBLVCAT or a list type window.

List, DB2 SQL and Execute CBLVCAT windows do not support the same concepts as CBL text edit and SDE structured edit document windows (e.g. text update, focus line, etc.). Therefore, CBL and SDE CLI commands and macros may not be successfully issued from List and Execute CBLVCAT windows.

Similarly, all CBL main menu bar items are specific to CBL text edit windows (e.g. "Fill", "Uppercase", "Lowercase") and, if selected, will apply to the **current CBL window**. Each List, DB2 SQL and Execute CBLVCAT window has its own menu bar items applicable to that individual window, displayed immediately below the document window's title bar.

```

-Execute CBLVCAT
View Refresh Back Forward FDB Raw Text Help
Command>
VCAT Command> report vcat dsn type vol2 ! listcat type=u ref=
                > CATALOG.Z111.MASTER
                >
VCAT Program> CBLV
-----
1CBLVCAT REL 3
-----
report vca
listcat t
CATALOG.Z1
-----
ICF CAT ZBSYS
-----
USERCAT.CBLCA
USERCAT.Z111.
USERCAT.Z111.
USERCAT.Z111.
USERCAT.Z111.
USERCAT.Z111.
USERCAT.Z111.
CBLVCAT 3.10.
**
Line 1 of 22 | C
000035 |

-Allocated Datasets
View Refresh Back Forward FDB Text Help
Command>
DDName>
-----
-DDName- -CSeq- -----DsN-----
AOFPRT   1 NBJ2.NBJ2.TSU03296.D0000101.7
AOFTABL  1 AUT320.AOFTABL
DITPLIB  1 DIT130.SDITPLIB
DSNETBLS 1 DSN910.SDSNSPFT
IHVCONF  1 AUT320.IHVCONF
ISPCLT1  1 SYS12110.T125749.RA000.NBJ2.R
ISPCLT2  1 SYS12110.T125749.RA000.NBJ2.R
ISPEXEC  1 ISP.SISPEXEC
"        2 SYS1.SBPXEXEC
"        3 CSQ700.SCSQEXEC
"        4 EUV.SEUVEXEC
ISPLLIB  1 CBL.SSC.EXE
"        2 CBL.ISPLLIB
"        3 DSU.ISPLLIB
"        4 GEN.ISPLLIB
"        5 GDDM.SADMMOD
"        6 FMN910.SFMNMOD1
"        7 CSQ700.SCSQAUTH
"        8 AUT320.SINGMOD1
"        9 DSN910.DB9G.SDSNEXIT
Line 1 of 173 | Col 1 of 115 | Views 1 | select * sort
  
```

Figure 5. CBLLe List &amp; CBLVCAT Document Window.

## CBLLe Open Dialog Window

The CBLLe text edit **Open File** panel may be opened via the following:

- Select **Open** from the **File** menu item in the **CBLLe Main Menu Bar**.
- Select **Text Edit** from the **Primary Option Menu** panel.

```

-Open File
File Help
Command>
ZZSGOPEN
-----
Open File: PDS(E) member, Sequential, or HFS path
Dsn/Path> cbl.cmx
Volume> If dataset is uncataloged.
Use (TSO) Prefix
E
-Dataset List: CBL.CMX
View Refresh Back Forward FDB Text Help
-----
-Library List: CBL.CMX
View Refresh Back Forward FDB Text Help
Command> where lastmod >= 2009 sort created desc
Library> CBL.CMX
-----
-Member- Alias VV MM -Created-- ----LastMod----- CurSize IniSize
TMP      N    1  3 2012/03/28 2012/04/02 17:28 1000 1000
SMPEINST N    1 15 2012/03/15 2012/04/11 17:41 885 884
IQ003044 N    1  1 2012/03/15 2012/03/15 11:40 58 58
VMNBJ    N    1  1 2012/03/08 2012/03/08 17:46 1566 1566
LAC5     N    1 12 2012/02/06 2012/02/07 10:29 29 4
SMPEVSVC N    1 26 2012/01/26 2012/04/12 15:10 1745 1642
NB2      N    1 27 2012/01/16 2012/04/13 16:33 6305 6002
Line 1 of 47 | Col 1 of 96 | Views 4 | select * where lastmod >= 2009 so
  
```

Figure 6. CBLLe text edit OPEN Dialog Window.

The Open File panel allows the user to enter the name of a cataloged or uncataloged data set or HFS file path for edit. A relevant file list window is opened that enables the user to browse, sort and filter file entries before selecting the required file for edit.

See [Text Edit](#) in "[FileKit Reference and User Guide](#)" for description of the Open File panel.

## CBLLe Find Dialog Window

The CBLLe text edit Find dialog window may be opened via the following:

- Select **Find** from the **Edit** menu item in the **CBLLe Main Menu Bar**.
- Enter the CBLLe command **FINDDIALOG** at the command line of any document window.

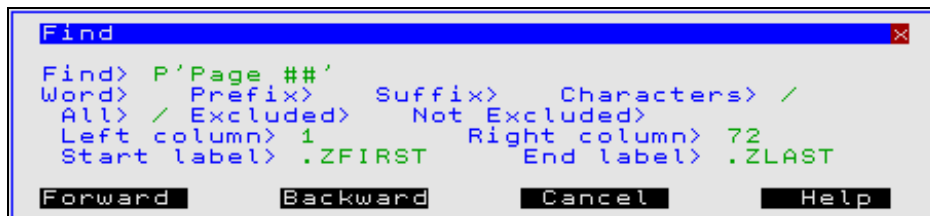


Figure 7. CBLLe text edit FIND Dialog Window.

The Find dialog allows the user to find occurrences of a character string in the current CBLLe text edit view.

The dialog generates an ISPF edit style **FIND NEXT** or **FIND PREV** command and, for repeated execution, an **RFIND** command. Therefore, the search string entered at the **Find>** prompt may be of any format supported by ISPF FIND.

Following the search, the dialog window remains open for repeated backwards or forwards search for the string.

The position of the found search string within the file area is identified by the find cursor. The appearance of the find cursor is controlled using the **SET COLOUR FCURSOR** command. (Default is WHITE REVVIDEO.)

Note that FIND/RFIND will only locate a string that occurs in its entirety within the current boundaries.

When the Find dialog is opened, fields are populated with the values specified by the last FIND operation, otherwise the default values are used.

### Word, Prefix, Suffix and Characters

These fields are mutually exclusive and define whether the search string represents:

- ◊ WORD - A word delimited by blanks or non-alphanumeric characters.
- ◊ PREFIX - A string at the beginning of a word.
- ◊ SUFFIX - A string at the end of a word.
- ◊ CHARACTERS - A string that may be anywhere in the text. (Default)

Enter any non-blank character in one of these fields to override the existing option.

### All, Excluded and Not Excluded

These fields are mutually exclusive and define the areas of text to search.

- ◊ ALL - All lines of data (Excluded and non-excluded). (Default)
- ◊ EXCLUDED - Excluded lines only.
- ◊ NOT EXCLUDED - Non-excluded lines only.

Enter any non-blank character in one of these fields to override the existing option.

### Left column and Right column

These fields define the first and last columns to be searched. If a value is entered in only one of the column fields or the values in both column fields are equal, then the string is only found if it starts in the specified column. Default is the left and right boundaries.

### Start label and End label

These fields define the first and last lines of the group of lines to be searched. Default is the system assigned labels **.ZFIRST** and **.ZLAST** indicating the first and last lines of the range respectively.

## CBLLe Change Dialog Window

The CBLLe text edit Change dialog window may be opened via the following:

- Select **Change** from the **Edit** menu item in the **CBLLe Main Menu Bar**.
- Enter the CBLLe command **CHANGEDIALOG** at the command line of any document window.

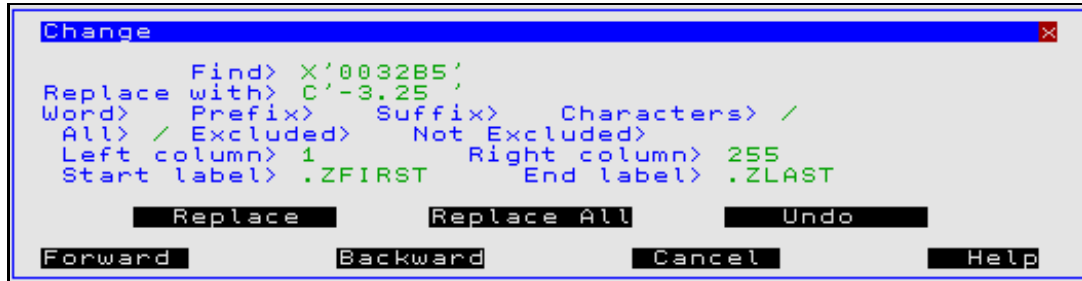


Figure 8. CBLLe text edit CHANGE Dialog Window.

The Change dialog allows the user to find occurrences of a character string in the current edit view and optionally replace them with the change string.

The dialog generates an ISPF edit style **FIND NEXT** or **FIND PREV** command and, for repeated execution, an **RFIND** command. Subsequently, if **Replace** is selected, an ISPF edit style **CHANGE .ZCSR .ZCSR NEXT** is executed, or, if **Replace All** is selected, an ISPF edit style **CHANGE ALL** is executed for the specified columns and/or group of lines.

Therefore, the search and change strings entered at the **Find>** and **Replace with>** prompts respectively, may be of any format supported by ISPF CHANGE.

Following the search, the dialog window remains open for repeated forward searches for the string.

The position of the found search string within the edit view text is identified by a find cursor. The appearance of the find cursor is controlled using the **SET COLOUR FCURSOR** command. (Default is WHITE REVVIDEO.) Note that FIND/RFIND will only locate a string that occurs in its entirety within the current boundaries.

When the Change dialog is opened, fields are populated with the values specified by the last FIND operation, otherwise the default values are used.

If a replace is actioned and the length of the replace string is greater than that of the find string, then the data following is shifted to the right.

### Word, Prefix, Suffix and Characters

These fields are mutually exclusive and define whether the search string represents:

- ◊ WORD - A word delimited by blanks or non-alphanumeric characters.
- ◊ PREFIX - A string at the beginning of a word.
- ◊ SUFFIX - A string at the end of a word.
- ◊ CHARACTERS - A string that may be anywhere in the text. (Default)

Enter any non-blank character in one of these fields to override the existing option.

### All, Excluded and Not Excluded

These fields are mutually exclusive and define the areas of text to search.

- ◊ ALL - All lines of data (Excluded and non-excluded). (Default)
- ◊ EXCLUDED - Excluded lines only.
- ◊ NOT EXCLUDED - Non-excluded lines only.

Enter any non-blank character in one of these fields to override the existing option.

### Left column and Right column

These fields define the first and last columns to be searched. If a value is entered in only one of the column fields or the values in both column fields are equal, then the string is only found if it starts in the specified column. Default is the left and right boundaries.

### Start label and End label

These fields define the first and last lines of the group of lines to be searched. Default is the system assigned labels **.ZFIRST** and **.ZLAST** indicating the first and last lines of the range respectively.

## CBLe Sort Dialog Window

The CBLe text edit Sort dialog window may be opened via the following:

- Select **Sort** from the **Actions** menu item in the **CBLe Main Menu Bar**.
- Enter the CBLe command **SORTDIALOG** at the command line of any document window.

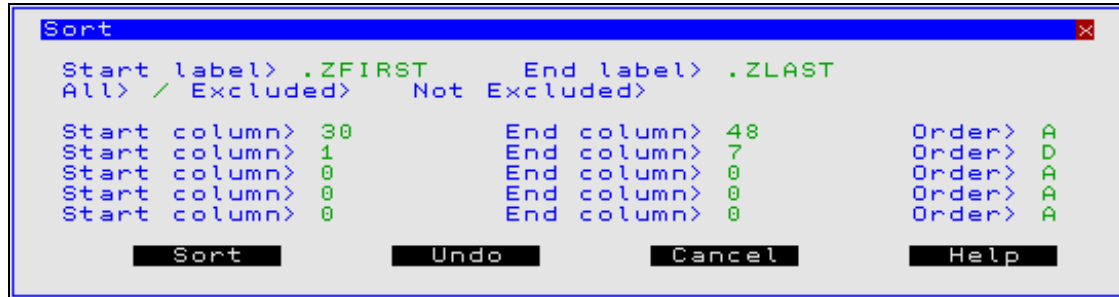


Figure 9. CBLe text edit SORT Dialog Window.

The Sort dialog allows the user to arrange data in the current edit view.

The dialog generates an ISPF edit style **SORT** command, sorting only data that falls within the current boundary settings.

Following the search, the dialog window remains open to allow further sorting.

### Start label and End label

These fields define the first and last lines of the group of lines to be sorted. Default is the system assigned labels **.ZFIRST** and **.ZLAST** indicating the first and last lines of the range respectively.

### Start column and End column

These fields define the start and end of each sort field. A maximum of 5 sort fields may be specified that must fall within the current boundaries and must not overlap. If a start column is specified on the last sort field without an end column, then the right boundary is used by default.

### Order

This field defines the order (Ascending or Descending) in which the data is to be arranged within each individual sort field.

## CBLe Fill Dialog Window

The CBLe text edit Fill dialog window may be opened via the following:

- Select **Fill** from the **Actions** menu item in the **CBLe Main Menu Bar**.
- Enter the CBLe command **FILLDIALOG** at the command line of any document window.



Figure 10. CBLe text edit FILL Dialog Window.

The Fill dialog allows the user to fill a marked block with text.

The dialog generates a CBLe **FILLBOX** command.

Before the Fill dialog can be activated, a block of text must first be marked within the text display of an edit view. This is done using <F17> or <F18> which are respectively assigned to **MARK BOX** and **MARK LINE** by default.

The **Fill Char/String** field may contain a single character or a character string. If no parameter is specified, the marked block is filled with blank characters.

If a single character is specified, the character is repeated to occupy all marked positions in the text.

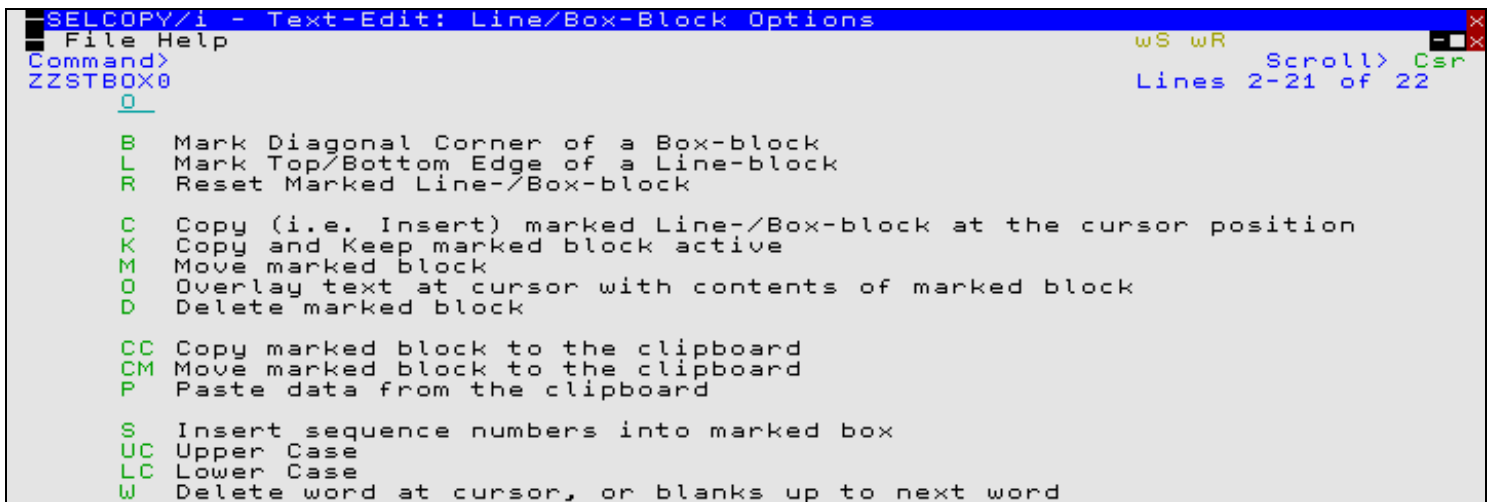
If a character string is specified, the string is repeated once at the rightmost column for each line of the marked block. Truncation or blank padding will occur if the length of the string is respectively greater than or less than the marked block width.

Note that the **FILLBOX** command is influenced by the setting of **HEXSTRING** and **ZONE**.

## Text Edit Line/Box Block Options

The **Text Edit Line/Box Block Options** panel (ZZSTBOX0) is opened on executing the **BOX** text editor primary command (assigned to F20 by default).

This panel displays a number of actions relating to marked blocks including block marking, copying, moving, deleting and number sequencing.



```
SELCPY/i - Text-Edit: Line/Box-Block Options
File Help
Command>
ZZSTBOX0
  O
  B Mark Diagonal Corner of a Box-block
  L Mark Top/Bottom Edge of a Line-block
  R Reset Marked Line-/Box-block

  C Copy (i.e. Insert) marked Line-/Box-block at the cursor position
  K Copy and Keep marked block active
  M Move marked block
  O Overlay text at cursor with contents of marked block
  D Delete marked block

  CC Copy marked block to the clipboard
  CM Move marked block to the clipboard
  P Paste data from the clipboard

  S Insert sequence numbers into marked box
  UC Upper Case
  LC Lower Case
  W Delete word at cursor, or blanks up to next word
```

Figure 11. Text-Edit: Line/Box-Block Options Panel View.

### Options:

- B** Mark Diagonal Corner of a Box-block  
Mark a corner of a box block at the focus line and focus column (cursor position.)
- L** Mark Top/Bottom Edge of a Line-block  
Mark the focus line so that it identifies the top or bottom of a line block (cursor position.)
- R** Reset a Marked Line-/Box-block  
Reset an existing marked block of text.
- C** Copy (i.e. Insert) marked Line-/Box-block at the cursor position  
Copy an existing marked block of text to the focus line and, if a box block, focus column (cursor position.)
- K** Copy and Keep marked block active  
Same as C (Copy) except that the block remains marked at the target location following the copy operation.
- M** Move marked block  
Move an existing marked block of text to the focus line and, if a box block, focus column (cursor position.)
- O** Overlay text at cursor with contents of marked block  
Overlay text at the focus line and, if a box block, focus column (cursor position) with text from an existing marked block.
- D** Delete marked block  
Delete an existing marked block of text.
- CC** Copy marked block to the clipboard  
Copy an existing marked block of text to the FileKit text/data editor clipboard.
- CM** Move marked block to the clipboard  
Move (CUT) an existing marked block of text to the FileKit text/data editor clipboard.
- P** Paste data from the clipboard  
Not strictly related to marked blocks, this option will paste (copy) all text saved to the FileKit text/data editor clipboard to the focus line and focus column (cursor position.)
- S** Insert sequence numbers into marked box  
Opens the "Generate Sequence Numbers in Marked Box" sub-panel (ZZSTBXSQ). This panel allows the user to populate each line of an existing marked block with sequence numbers or to adjust existing numerical values.

### Input fields:

#### Base:

Identifies the numerical format of the start value, increment value or existing values within the marked block. Valid entries are **DEC** (BASE 10) or **HEX** (BASE 16).

**Start Value:**

For number sequencing only, this is the value to be inserted in the first line of the marked block. Subsequent lines in the marked block will be populated with values that are sequenced from this number until all lines in the block have been exhausted.

If DEC is selected in the Base: input field, this value may be any positive or negative real numeric value (e.g. -3.23). If HEX is selected, this value must be a valid hexadecimal value (e.g. 2A).

**Increment:**

Specifies the numeric value to be added to the previous value in the numeric sequence or existing values in the marked block.

If DEC is selected, this value may be any positive or negative real numeric value (e.g. -1.1). A negative value will result in a descending sequence. If HEX is selected, this value must be a valid hexadecimal value (e.g. 0F).

**Leading Zeros:**

Values are right adjusted in the marked block. This field identifies whether or not blanks to the left of the inserted or updated values are to be replaced with leading zeros.

**Options**

Mutually exclusive options that determine the operation performed.

- **Use above specified start value.**  
Create sequence using the value in the **Start Value:** field as the initial value in the sequence.
- **Use the first line existing value as start value.**  
Create sequence using the value already entered on the first line of the marked block as the initial value in the sequence.
- **Adjust each existing value by the increment number.**  
Do not create a sequence but instead add the value in the **Increment:** field to values already entered in each line of the marked block.

```

SELCOPY/i - Generate Sequence Numbers in Marked Box
File Command Help
Command>
ZZSTBXSQ
wS wR
Scroll> Csr
Lines 1-20 of 20

          --- BoxSeq ---

Use this panel to insert or modify a numeric column defined by a marked
"Box-block" in a Text-Edit view.

Use the "MrkBox" key (default is Shift-F5) to mark the corners of a
box-block before entering this panel.

The box corners define the start/end line and columns to receive
the sequenced numbers, incremented/decremented for each line.

      Base:  DEC          DECimal or HEXadecimal.

Start Value:  1          Number inserted on 1st line.
Increment:    1          +/- number adjustment to 2nd and subsequent lines.
Leading Zeros: YES

  / Use above specified start value.
  - Use the first line existing value as start value.
  - Adjust each existing value by the increment number.

```

Figure 12. Generate Sequence Numbers in Marked Box Panel View.

**UC** Upper Case

Convert alpha characters within an existing marked block of text to upper case.

**LC** Lower Case

Convert alpha characters within an existing marked block of text to lower case.

**W** Delete word at cursor, or blanks up to next word

Not related to marked blocks, this option will delete text at the focus line and focus column (cursor position).

If the focus position is non-blank, all non-blank text to the right of the focus column is deleted up to, but not including, the first blank character. Similarly, if the focus position is blank, all blank text to the right of the focus column is deleted up to, but not including, the first non-blank character.

---

## Text/Data Edit GO Options

---

The **Text/Data Edit GO Options** panel (ZZSGEDGO) is opened on executing the **GO** text editor primary command.



This panel prompts for the parameter to be passed to the GO primary command in order to switch the application used to process the data belonging to the current text or data editor window view. The current view will be closed and the data redisplayed in a view appropriate to the selection.

```

SELCOPY/i - Text/Data Edit: GO Options
Help
Command>
ZZSGEDGO
FileId: NBJ.JCL(SSXV05)
GO B      SE | SU | B | E | V
GO SE     Switches to Data-Edit (Full-Edit)
GO SU     Switches to Data-Edit (Upd-in-Place)
GO B      Switches to Data-Edit (Browse)
GO E      Switches to Text-Edit
GO V      Switches to Text-Edit (View)

```

Figure 13. Text/Data Edit: GO Options Panel View.

#### Options:

- SE** Display the data in a data editor **Full Edit** view.
- SB** Display the data in a data editor **Update-in-place Edit** view.
- B** Display the data in a data editor **Browse** view.
- E** Display the data in a text editor **Edit** view.
- V** Display the data in a text editor **Read-Only Edit** view.

---

## Editing HFS files

---

z/OS UNIX System Services (USS) provides a UNIX style hierarchical file system where files are organised in a directory tree structure. The z/OS hierarchical file system may comprise HFS, ZFS and NFS file system types.

Throughout FileKit documentation, these file systems are collectively referenced as the HFS file system, and entries in these systems are referenced as HFS files or HFS paths.

CBLe text edit and SDE edit views may be opened to Edit and Browse binary or text data in any HFS file.

To access HFS files for CBLe or SDE edit/browse:

- Open a List HFS Path window, using the HFS Dir Path item of the List drop-down menu or the LISTPATH command, and select an entry.
- Execute EDIT or BROWSE with a fileid which is either an absolute or relative HFS path name.
- Open the Structured Data Browse/Edit dialog window to specify an absolute or relative HFS path name.

The following FileKit functions support HFS files:

- CBLe EDIT & BROWSE.
- SDE EDIT & BROWSE - Full or In-place edit with Copy Book overlay.
- FSU - File Search/Update.
- LISTPATH - HFS Path lists including Prefix command support.
- USS Commands for Data and Environment Management. (LINK, UNLINK, MKDIR, RMDIR, CHDIR, etc.)
- ERASE
- RENAME

## HFS File Records

---

Unlike MVS data sets which are record oriented, HFS files are byte oriented and so records within the byte stream are identified as being in one of the following formats:

- EOL (End-of-Line) character delimited. (NL, CR, LF, CRLF, LFCR, CRNL or a 2-byte, user supplied *string*.)
- Fixed length.
- Variable length. (Records include a length field at a pre-determined offset.)

The format used is determined by user supplied parameters on the EDIT, BROWSE, FSU commands or via the equivalent dialog windows.

By default, the EOL delimiter format is used with an EOL delimiter character being that stored in the file's directory entry information. If undefined, the EOL delimiter NL is used.

## HFS File Name Specification

---

FileKit maintains the concept of the user's home directory, defined by RACF, and the **current working directory**. The current working directory may be updated using the FileKit command, USS CHDIR.

An HFS file may be referenced via an **absolute** path, starting at the root directory, or a path **relative** to the current working directory. Furthermore, it is case sensitive and, if it contains special characters, blanks or commas, should be enclosed within single quotes (apostrophes) or double quotes.

FileKit imposes no restriction on the length of an HFS path name and so is subject only to those restrictions imposed by z/OS USS standards.

The **name portion** of the HFS path is the character string that follows the last "/" (slash) or, if no "/" exists, is the entire relative HFS path name. The name portion may contain wild card characters, thus allowing generic HFS file names to be specified for HFS Path Lists and FSU (File Search/Update) utilities.

By default, where specification of a fileid may either be that of an MVS data set name or an HFS path name and interpretation of the type of fileid entered is ambiguous, then an MVS data set name is assumed. To avoid ambiguity, users should include a "/" (slash) within any relative HFS path names.

e.g. `EDIT ./dev.hfs.file` references a file in the current working directory, whereas `EDIT dev.hfs.file` references the MVS QSAM file DEV.HFS.FILE.

Any relative HFS path name or symbolic link is resolved to an absolute file path and displayed in the title bar of the CBL or SDE edit view. To conform with existing CBL concepts, the absolute HFS path name of a file may be split into the following components:

<b>FILEID or DSN</b>	The complete absolute HFS file path name.
<b>FPATH</b>	The directory path from the root directory up to, but not including, the last "/" (slash) character in the fileid.
<b>FMODE</b>	The first level directory name above the root directory in the fileid.
<b>FNAME or MBR</b>	The character string following the last "/" (slash) and immediately preceding the last "." (dot/period) in the fileid. If no "." exists, FNAME runs to the end of the fileid.
<b>FTYPE</b>	The character string following the last "." (dot/period) in the fileid. If no "." exists, FTYPE is a null string.

## Data Protection

---

User access to individual HFS files is dependent upon the file's permission bits and the configuration of OMVS settings within the user's RACF definition.

When an HFS file is opened for edit, an exclusive enqueue is established in the SPFEDIT queue which is of a format which is identical with the enqueue issued by ISPF for the edit of HFS files. The enqueue resource name is 12 bytes in length and comprises three integers that represent the file's inode number, device number and a sysplex indicator. (See: "ISPF Planning and Customizing", "z/OS UNIX file Enqueue" for full details.)

Although the manual states that this enqueue is compatible with that issued by the z/OS UNIX OEDIT command, this has been found to be incorrect in z/OS 1.9. The sysplex indicator flag in the OEDIT enqueue resource name is set as x'20' instead of x'01' as documented. As such, OEDIT and ISPF edit do not adhere to the same file enqueueing standard.

## Editing Large files

CBL always reads the whole of the file being edited into virtual storage. If a file is too big to fit in virtual storage then it cannot be edited.

However, support for edit of files that are too large to be loaded into the available region is available via the SDE (Structured Data Environment) edit feature. See the [SDE Edit Dialog Window](#) and [Structured Data Environment Manual](#) for help on starting an SDE edit view.

When a CBL edit window is loading a file and runs out of virtual storage an error message is issued (ZZSE063E) and the attempted edit is aborted. It can take a long time for CBL to exhaust storage and this time is wasted if the edit is eventually going to fail.

To give the user some control over this situation there are two [INI variables](#) which can be set in the (Edit) section of the System or User INI files:

### SizeWarning

The file size warning threshold. If a file is bigger than this value the user will be warned that the editor is about to attempt to load all records of a large file into storage. The message also prompts the user to either continue with the load or switch to using structured data edit which supports edit without all records having been loaded in storage. The default sizewarning value is 1M.

### LoadWarning

The file load increment warning threshold. When a file is being loaded a count is kept of the number of bytes loaded. If this count exceeds the LOADWARNING value then a popup message box is displayed to prompt the user to confirm whether or not the load is to continue. The default loadwarning value is 1M.

This feature helps with the case in MVS where the size of some files (e.g. PDS members) is not always known accurately when the file is opened.

## Editing VSAM files

More flexible editing of all types of VSAM data sets including copy book overlay is available via the SDE (Structured Data Environment) edit feature. See the [CBL SDE Edit Dialog Window](#) and [Structured Data Environment Manual](#) for help on starting an SDE edit view.

By default, when attempting to perform a text edit or view of a VSAM data set, the user will be prompted to select the type of edit to use.

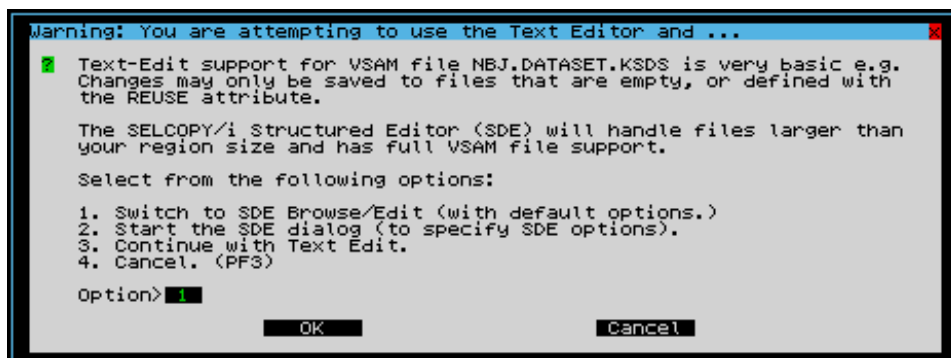


Figure 14. VSAM Edit Prompt.

The CBL text editor supports READ ONLY edit of most VSAM data sets. However, READ/WRITE edit is supported for VSAM data sets that have a high-used RBA of zero. This occurs when:

1. The data set is defined with parameter REUSE.  
Whenever CBL is directed to save data to disk (SAVE, FILE, etc.), it has to re-open the data set for output. For a VSAM file defined with REUSE, CBL re-opens the data set with the RESET attribute which resets the high-used RBA to zero.
2. Data has never been written to the data set.  
When attempting to save a new VSAM data set, the DEFINE VSAM dialog window is opened prompting the user to allocate it first.

Beware, having saved data to an empty data set defined with NOREUSE, the high-used RBA is no longer zero and subsequent attempts to save the file will fail with the following error message:

```
ZZSE085E VSAM PUT error for file dataset : return code x'8' reason code x'8'.
```

When editing KSDS data sets, care must be taken to preserve the primary key sequence. If an attempt is made to save a KSDS data set that is not in key sequence, then one of the following error messages is returned:

```
ZZSE086E Duplicate keys - records m and n have the same key.
ZZSE087E Sequence error - record m has a higher key than record n.
```

---

## Editing RECFM=U Files

---

FileKit Text and Data Editors both support edit or browse of undefined record format (RECFM=U) physical sequential datasets.

The maximum record length (LRECL) value used for edit of a RECFM=U dataset is the allocated physical record size (block size). If the dataset has not yet been allocated a block size (i.e. BLKSIZE=0) and the allocated LRECL value is not 0 (zero), then BLKSIZE=LRECL is used. Otherwise, if the LRECL value is also zero, an error is returned.

Edit of a RECFM=U format PDS/PDSE member is prohibited in order to protect module/program members of a load library from accidental update. If an attempt is made to edit this type of entry using the text editor, a read only edit view of the member is opened and warning ZZSE200W is returned. Similarly, if the data editor is used, a browse view of the data is opened and warning ZZSD667W is returned.

When editing text for a new, as yet unallocated dataset name, **DSORG** and **RECFM** options may be used to set the dataset organisation and record format respectively. An attempt to set DSORG=PO (partition organised) and RECFM=U will return error message ZZSE201E, ZZSE202E or ZZSD668E.

---

## Focus Line and Column

---

The concept of file focus is a very important one in CBL. The file focus line and/or column are implied input parameters to most CBL commands which refer to the contents of the file being edited.

For example, if you issue the DELETE command, which deletes lines from your file, the first line deleted is the focus line (more lines may be deleted, depending on the parameters supplied on the DELETE command). If you issue the CDELETE command, which deletes columns from a line in your file, the first column deleted is the focus column of the focus line (more columns may be deleted, depending on the parameters supplied on the CDELETE command).

As well as being the starting point for the effect or scope of many commands, the focus line and column may be changed as a result of executing the command. The documentation of each command will explain how the command uses and changes the focus.

The focus line and column are defined by the position of the cursor when a command is executed:

Cursor Location	Focus Line	Focus Column
In file area.	Line containing the cursor.	Column containing the cursor.
In prefix area.	Line containing the cursor.	Left column of current file area in the current edit view window.
Not in file or prefix area. (e.g. command line)	The current line. i.e. The top line of the current edit view window.	The current column. i.e. A column number which is, by default, column 1 of the file and is changed by the commands CLOCATE, CFIRST, CLAST and SET ZONE. The current column is indicated in the scale line by a vertical bar ( ).

---

## Using Marked Blocks

---

A marked block is a highlighted rectangular area of text in the file you are editing which can be used as the target of many edit commands. For example you can delete, copy, move or fill a marked block. There are two types of marked block:

Line blocks	For line blocks the marked rectangle is the width of the file (current LRECL). The left edge of the marked block is column 1 and the right edge is the last column in the file.
Box blocks	For box blocks the width of the marked rectangle can be any value from 1 to the width of the file.

Both line and box blocks may span any number of contiguous lines in the file display area.

You mark a block with the **MARK** command. You can also use the **prefix commands** MB (mark box) and ML (mark line). By default function keys **F17** and **F18** are assigned to MARK BOX and MARK LINE respectively.

When you first use the MARK command the marked block is always only one line deep (the focus line) . When you issue the next MARK command, the marked block is extended (or reduced) to the current focus line.

If you issue a MARK LINE command when a box block is already marked in the current file then the box block is changed to a line block before being extended (or reduced) to the current focus line. Similarly, if you issue MARK BOX when a line block is marked it is changed to a box block.

You can unmark the current marked block with the **RESET BLOCK** command which by default is assigned to function key F24.

Only one file in the ring of edited files can contain a marked block. If you mark a block in one file and a marked block already exists in another file you are editing then the existing marked block is unmarked before the new mark command takes effect.

## Targets

Targets are used as parameters to a large number of CBL commands in order to identify search criteria on data in the current file.

Successful target location can be influenced by the following SET options:

**ARBCHAR, CASE, HEXSTRING, STAY, STREAM, VARBLANK, WRAP, ZONE.**

### Line Targets (line-target)

Line targets identify one or more lines within the edited file for which the command will be actioned.

#### Absolute Line Number Target

:10 Target is line number 10.

Specify ":" (colon) immediately followed by an integer to indicate that the target is an absolute line number.

#### Relative Line Number Target

10 Target line is 10 lines down from the focus line.

-3 Target line is 3 lines up from the focus line.

-\* Target line is line preceding the first line of the current RANGE setting.

Specify an integer to indicate that the target is a line offset from the focus line.

Alternatively, specify "\*" (asterisk) to indicate the maximum offset should be applied. This will imply a line preceding the first line or following the last line of the current RANGE setting.

The direction of the offset is determined by a preceding "+" (plus) for a positive offset and "-" (minus) for a negative offset. By default, the offset is "+".

#### String Target

/Hello/ Target line is the first line following the focus line that contains string "Hello".

#ab/c# Target line is the first line following the focus line that contains string "ab/c".

-/Hello/ Target line is the first line previous to the focus line that contains string "Hello".

/Hello / Target line is the first line following the focus line that contains string "Hello" immediately followed by a blank.

~/Hello/ Target line is the first line following the focus line that does not contain the string "Hello".

--/Hello/ Target line is the first line previous to the focus line that does not contain the string "Hello".

`word /ask/`

Target line is the first line following the focus line that contains the word "ask". **Note:** "asking" or "flask" will not match.

`-suffix /ion/`

Target line is the first line previous to the focus line that contains a word ending in "ion".

`/hello/ & /welcome/`

Target line is the first line following the focus line that contains both the strings "hello" and "welcome".

`-/hello/ | /goodbye/`

Target line is the first line previous to the focus line that contains either the string "hello" or "goodbye".

A string target is a sequence of characters enclosed in delimiters. CBLe will scan one line at a time starting at the line following the focus line for a forward scan, or the line preceding the focus line for a backward scan.

On encountering bottom of file (or top of file for a backward scan), the WRAP setting will determine whether the scan will continue on lines at the opposite extreme of the file.

The "/" (slash) character is normally used as the delimiter character. However, any non-alphanumeric character that does not have a special meaning to CBLe may be used as a string delimiter (e.g. "%", "#", "."). The delimiter character used must not appear in the target string itself.

Leading and trailing blanks contained in target strings are respected.

"~" (not), "^" (carat) or "~" (tilde) preceding the target string may be specified to indicate scan for a line that does not include the target string.

The string target may be prefixed by one of the following special keywords:

<code>Word</code>	The target string must match a complete word.
<code>Prefix</code>	The target string must match the leading characters of a complete word.
<code>Suffix</code>	The target string must match the trailing characters of a complete word.

A word is considered to be a string of characters with the dlimiter characters blank or any non-alphanumeric character.

String targets may be combined to form an expression using logical operators. The "and" operator is represented by "&" (ampersand) and the "or" operator is represented by "|" (vertical bar).

The direction of the scan is determined by a preceding "+" (plus) for a forward scan and "-" (minus) for a backward scan. By default, the direction is "+".

**Note:** , for expressions comprising multiple string targets, direction may only be specified once on the first string target.

## Named Line Target

`.curr`

Target line is the line previously named ".curr" via a SET POINT command.

A named line target is a line that has been named using the SET POINT command.

Specify "." (dot) immediately followed by the line name to indicate that the target is a named line target.

## Line Class Target

`blank`

Target line is the first line following the focus line that is completely blank.

`-altered`

Target line is the first line previous to the focus line that has been altered (updated or added) during the current CBLe session.

`~new`

Target line is the first line following the focus line that has not been added during the current CBLe session.

The following line class targets are supported:

<code>BLAnk</code>	Lines that are blank.
<code>TAGged</code>	Lines that have been tagged using the TAG command.
<code>CHAnged</code>	Lines that have been updated.
<code>NEW</code>	Lines that have been added.
<code>ALTered</code>	Lines that have been added or updated.
<code>SELECT n</code>	Lines that have selection level n or lines that have selection level between n and m. See option <b>SELECT</b> for description of selection levels.

## Column Targets (column-target)

---

:66	Target column is column number 66.
10	Target column is 10 columns to the right of the focus column.
-3	Target column is 3 columns to the left of the focus column.
-*	Target column is the column to the left of the current ZONE setting.
/abc /	Target column is the first column following the focus column that contains string "abc" immediately followed by a blank.
-/John/	Target column is the first column previous to the focus column that contains string "John".
word /just/	Target column is the first column following the focus column that contains the word "just". <b>Note:</b> "adjust" or "justify" will not match.

Column targets identify a column number within the edited file for which the command will be actioned. CBL commands requiring the column-target parameter are GLOCATE and CDELETE.

The column-target may be an absolute column number, relative column number or a string target.

An absolute column number is indicated using a ":" (colon) immediately followed by the column number.

A relative column number acts in the same way as a relative line number except that the specified integer indicates an offset to the right or left of the focus column for positive or negative offsets respectively.

Similarly, specifying "\*" (asterisk) as the offset value implies a column preceding the first column or following the last column of the current ZONE setting.

String target syntax is supported in same way as for line targets. CBL will scan one column at a time starting at the column to the right of the focus line for a forward scan, or the column to the left of the focus line for a backward scan.

For string targets only, the SET STREAM command determines whether the scan will continue over multiple lines should the target string not be found on the focus line.

## Group Targets (group-target)

---

:26	Target lines are the focus line and all lines up to, but not including line number 26.
-8	Target lines are the focus line and the 7 lines immediately preceding it.
*	Target lines are the focus line and all lines that follow it.
/dummy/	Target lines are the focus line and all lines up to, but not including, the first line following the focus line to contain the string "dummy".
ALL	Target lines are all lines in the file.
BLOCK	Target is all characters in the currently marked line or box block.

Group targets identify a group of lines (target area) for which the command will be actioned. A group-target parameter is required for CBL commands such as CHANGE, UPPERCASE, LOWERCASE and DELETE.

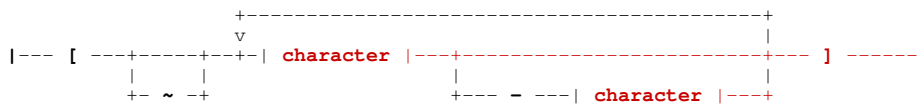
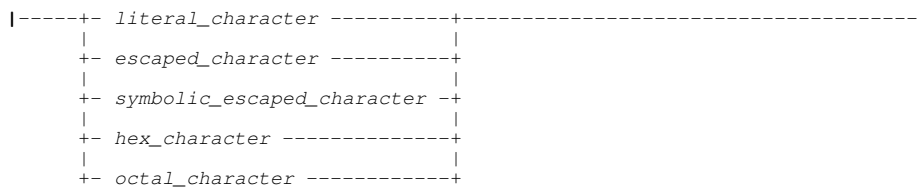
The focus line is treated as the first line of the group. Any form of the **line-target** syntax is used to flag the end of the target area, however, the target line itself is not included as part of the target area.

The group target may also be one of the following special keywords:

ALL	The group-target is all lines in the file.
BLOCK	The group-target is the currently marked line or box block.





**character\_class:****character:****Note:**

1. In regular expressions blanks represent themselves as literal text specifier characters. Only use a blank when it is part of the pattern.

**Description**

A regular expression is a string which represents the definition of a pattern of characters. It is a terse description of a pattern recognition algorithm made up of text specifiers and operators.

A text specifier represents a character to match.

Operators are used to combine text specifier matches so as to define the required pattern.

~

The **tilde** character represents the logical **NOT** operator. It applies to the term of the regular expression which follows it.

**character**

A single character text specifier. The character can be expressed in the following ways:

*literal\_character*

A character which represents itself. This can be any character other than the regular expression **special characters**.

*escaped\_character*

An escaped character is the escape operator \ (backslash) followed by one of the regular expression **special characters**. This is used to define one of these syntactically significant characters as a text specifier.

*symbolic\_escaped\_character*

A **symbolic escaped character** is the escape operator \ (backslash) followed by a lower case letter. These represent alternative ways of defining apostrophes, quotes and certain EBCDIC control characters as text specifiers.

*hex\_character*

A hex character is the escape operator \ (backslash) followed by **x** or **X** and two hexadecimal digits for example **\x00**. This provides a way of defining text specifier characters which cannot be input from the keyboard.

*octal\_character*

An octal character is the escape operator \ (backslash) followed by one, two or three octal digits for example **\072** or **\72** (equivalent to **\x3A**).

**character\_class**

A character class is a set of one or more characters in the range **\x00-\xff**. It is a text specifier which matches a single character in the target string when that character is one of those in the class.

The class definition is a list of character specifications enclosed within delimiters. It is **inclusive** when delimited by **[ ]** (square brackets) or **exclusive** when delimited by **[~ ]** (a logical not operator ~ immediately following the opening square bracket). When the class definition is **exclusive** the characters in the class are all those in the range **\x00-\xff** **except** those in the definition list.

The class definition list consists of single characters and character ranges.

Single characters can be given in any of the forms described above for single character text specifiers.

Character ranges are defined as a range start character followed by a hyphen followed by a range end character both of which can be given in any of the forms described above for single character text specifiers. Ranges can be specified in either order, so **0-9** and **9-0** both define the set **0123456789** of numeric digits.

The list of characters included in a range depends of the type of the start and end characters:

- ◇ When both ends are lower case letters the range consists only of the lower case letters alphabetically between the range ends. So **h-k** defines the characters **hijk** even though in EBCDIC there are code points between **i** (**x89**) and **j** (**x91**) which do not represent letters.
- ◇ Similarly when both ends are upper case letters the range consists only of the upper case letters alphabetically between the range ends.
- ◇ When the range ends are numeric digits the range consists of the numeric digits between the range ends.
- ◇ When the range ends are of different types the range consists of all code points between the range ends considered as hexadecimal values. So **a-Z** defines all characters with hexadecimal code points between **x81** and **xE9** inclusive, regardless of whether they represent letters.

**?**  
The **question mark** character is the wild card character text specifier. It matches any character and so is equivalent to the character class **[\x00-\xff]**.

**^**  
The **caret** character is the start of string text specifier. When this text specifier is encountered in applying a regular expression to a string it is only considered a match if the current position in the target is the start of the string.

**\$**  
The **dollar sign** character is the end of string text specifier. When this text specifier is encountered in applying a regular expression to a string it is only considered a match if the current position in the target when matching is complete is the end of the string.

#### *predefined\_expression*

Several commonly used regular expressions have been supported as **predefined expressions** and can be referred to with a shorthand consisting of the predefined expression operator : (colon) followed by a single lower case letter. They can be used as the regular expression itself or as a sub-expression in a more complex regular expression.

**( )**  
Parentheses delimit a sub-expression.

Parentheses can be used to group terms together to which the logical NOT operator **~** or any of the pattern repetition operators minimal closure **\***, minimal plus **+**, maximal closure **@**, maximal plus **#**, and power **^** apply.

For example:

```
ab#
```

matches an "a" followed by one or more "b"s such as "ab" "abbb"

```
(ab)#
```

matches one or more "ab"s such as "ab" "ababab"

Parentheses must be used to enclose an **alternation** which is a list of regular expressions separated by logical OR signs | (vertical bar). When an alternation is encountered in applying a regular expression to a string each of the sub-expressions is tried in turn at the current position in the target string until one matches or they all fail.

For example:

```
(a|b|c)
```

would match an "a" or a "b" or a "c" at the current position in the target string.

**{ }**  
Braces delimit a tagged sub-expression. Up to 9 sub-expressions may be tagged.

Tagged sub-expressions are used to enable reference to be made to that part of the target string which was matched by the sub-expression. When the regular expression is parsed from left to right each tagged sub-expression is given a reference number starting with 1. When the regular expression is being matched, if the tagged sub-expression matches part of the target string, the matching part is saved and may be used in a tag reference.

**&n**  
The tagged expression reference operator **&** (ampersand) is used to refer to the string matched by a previous tagged sub-expression. The reference number *n* must be in the range 1-9.

For example in the expression:

```
{[a-zA-Z]}[0-9]&1
```

the term **[a-zA-Z]** defines a character class of the lower and upper case alphabet, the term **[0-9]** defines a character class of the numeric digits, and the term **&1** refers back to the characters that were matched by the first tagged expression in braces, **{[a-zA-Z]}**.

This regular expression would match 3-character strings such as **a1a** and **X9X** where the first and last characters are the same alphabetic character and the middle character is a numeric digit.

The following operators are used to specify repetitions of a pattern. They apply to the preceding term or sub-expression of the regular expression.

**\*** The **asterisk** is the minimal closure operator. It applies to the term preceding it and specifies that zero or more occurrences be included in the match, but only the minimum number.

**+** The **plus sign** is the minimal plus operator. It applies to the term preceding it and specifies that one or more occurrences be included in the match, but only the minimum number.

**@** The **at sign** is the maximal closure operator. It applies to the term preceding it and specifies that the maximum of zero or more occurrences be included in the match.

**#** The **hash sign** is the maximal plus operator. It applies to the term preceding it and specifies that the maximum of one or more occurrences be included in the match.

**^n** The **caret** character when followed by an integer is the power operator. It applies to the term preceding it and specifies that exactly **n** occurrences be included in the match. The repetition factor **n** must be in the range 1-999.

## Examples

### Alternation (or) operator

The alternation (or) operator **|** is used to choose between a series of possible matches. The alternative sub-expressions are separated by or signs and enclosed in parentheses.

Expression	String	Match
(u v)	xxxabbbbvyy	No match
(u v)	xxxvccccccvyy	v

### Not operator

The not operator **~** applies to the following term in the expression. It specifies that the regular expression fails to match if the negated term matches at the current position in the target string.

Expression	String	Match
a~b	xxxabbbbvyy	No match
a~b	xxxacccccvyy	a

### Minimal closure

The minimal closure operator **\*** specifies zero or more repetitions of the term it applies to but a minimal number. As such it is only useful when used for a term which is not the last in the pattern.

Expression	String	Match
ab*	xxxabbbbvyy	a
ab*c	xxxabbbbccvyy	abbbbc
ab*c	xxxacccccvyy	ac

### Maximal closure

The maximal closure operator **@** specifies zero or more repetitions of the term it applies to but a maximal number.

Expression	String	Match
ab@	xxxabbbbvyy	abbbb
ab@c	xxxacccccvyy	ac

### Minimal plus

The minimal plus operator **+** specifies one or more repetitions of the term it applies to but a minimal number.

Expression	String	Match
ab+	xxxabbbbvyy	ab

ab+c	xxxacccccccvyy	No match
ab+c	xxx <b>abbbbc</b> ccyy	abbbbc

### Maximal plus

The maximal plus operator # specifies one or more repetitions of the term it applies to but a maximal number.

Expression	String	Match
ab#	xxx <b>abbbb</b> yy	abbbb
ab#c	xxxacccccccvyy	No match
ab#c	xxx <b>abbbbc</b> ccyy	abbbbc

## Syntax Elements and Operators Summary

The regular expression character string consisting of delimiters, operators and text specifiers. Apart from characters which represent themselves as text specifiers, regular expression syntax uses special characters as delimiters, operators and special sorts of text specifiers.

### Special Characters Used in Regular Expressions

The following characters have a special meaning as operators or delimiters in regular expression syntax:

Character	Name	Description
\	Backslash	The escape operator
(	Left Parenthesis	Delimits the start of a sub-expression or alternation
)	Right Parenthesis	Delimits the end of a sub-expression or alternation
[	Left Bracket	Delimits the start of a character class definition
]	Right Bracket	Delimits the end of a character class definition
{	Left Brace	Delimits the start of a tagged sub-expression
}	Right Brace	Delimits the end of a tagged sub-expression
?	Question Mark	The wild card character text specifier
~	Tilde	The NOT operator
^	Caret	The power operator and the start of line text specifier
\$	Dollar Sign	The end of line text specifier
*	Asterisk	The minimal closure operator
@	At Sign	The maximal closure operator
#	Hash	The maximal plus operator
	Vertical Bar	The alternation (or) operator
:	Colon	The predefined expression operator
&	Ampersand	The tagged expression reference operator
+	Plus Sign	The minimal plus operator
-	Minus Sign	The character range operator

### Text Specifiers

Text specifiers are used to match characters:

Text Specifier	Name	Description
<i>character</i>	Literal character	Any character (including blank) which is not a regular expression <b>special character</b> .
[...] or [~...]	Character class	A set of characters and/or character ranges enclosed in square brackets.
?	Wild card	The wild card character which matches any character.
\character	Escaped Character	The escape operator followed by any one of the <b>special characters</b> used in regular expression syntax represents that character as a literal.
\letter	Symbolic Escaped Character	A number of control characters or special characters can be specified using a <b>symbolic escaped character</b> for convenience. This consists of the escape operator followed by a single lowercase letter.
\xnn	Hex character	The escape operator followed by x or X and a two digit hexadecimal number defines a literal character by its hexadecimal code value.

<b>^</b>	Match start	The caret character matches the start of the compare string
<b>\$</b>	Match end	The dollar sign matches the end of the compare string

## Operators

Operators are used to affect the way text specifiers or sub-expressions are processed or to combine sub-expressions to build more complex regular expressions.

Operator	Name	Description
*	Minimal Closure	Match zero or more occurrences of the preceding expression but only as many as necessary.
+	Minimal Plus	Match one or more occurrences of the preceding expression but only as many as necessary.
@	Maximal Closure	Match zero or more occurrences of the preceding expression matching as many as possible.
#	Maximal Plus	Match one or more occurrences of the preceding expression matching as many as possible.
<b>^n</b>	Power	Match n occurrences of the preceding expression.
~	Not	The not operator applies to the following expression. It allows the specification of a pattern in terms of the negation of a match.
()	Parentheses	Parentheses are used to define sub-expressions.
	Alternation	The alternation (or) operator matches one of a series of expressions enclosed in parentheses.
{}	Tagged Expressions	Braces are used to identify a tagged sub-expression. When the tagged expression is matched, the matching text can be referred to with a tag reference. Tagged expressions are identified by a sequence number assigned left to right starting at 1.
&n	Tag Reference	A tag reference represents the matched text of the associated tagged expression.
:letter	Predefined Expression	A <b>predefined expression</b> is represented by a colon followed by a lowercase letter. It represents a shorthand form of commonly used regular expressions which can be used by themselves or as sub-expressions.

## Predefined Expressions

The following predefined expressions are supported:

Name	Definition	Description
:a	[a-zA-Z0-9]	Alphanumeric character
:b	(\t x40)#	White space (a string of blanks and tabs).
:c	[a-zA-Z]	Alphabetic character
:d	[0-9]	Numeric digit
:g	(("[~"] "@"))(["~"] "@')	Quoted string (in single or double quotes).
:w	([a-zA-Z]#)	Word (a string of alphabetic characters).
:z	([0-9]#)	Integer (a string of numeric digits).

## Symbolic Escaped Characters

The following symbolic escaped character sequences are supported:

Symbol	Character	Description
\a	'	Apostrophe (single quote)
\g	"	Double quote
\b	X'16'	EBCDIC control BS backspace
\f	X'0C'	EBCDIC control FF formfeed
\r	X'0D'	EBCDIC control CR carriage return
\l	X'15'	EBCDIC control NL new line
\n	X'25'	EBCDIC control LF linefeed
\t	X'05'	EBCDIC control HT horizontal tab

---

## Save/Restore Text and Data Edit Document Windows

---

When operating in a windowed (i.e. non-maximised) FileKit window display, it is often desirable to resize and move a window to a fixed location within the CBL Main Window, and then for FileKit to use this size and location whenever the window is re-opened.

For windows that are of **interactive panel** window class (e.g. FCopy, FSU utilities and DB2 lists), this is done automatically.

For Text Editor or Data Editor document windows, the size and location of individual edit/browse window views may be saved and later restored across FileKit sessions using the WINX utility macro.

Simply selecting the wS (window save) and wR (window restore) items, displayed to the right of the CBL Main Window menu bar items, will invoke WINX to save and restore the document window view used to display data belonging to a specific file. By default, window save and restore operations are also assigned to F13 and F14 respectively when the cursor is positioned in the title bar of any edit or browse document window view.

When window save is performed, the edit/browse document window's physical attributes are saved to an individual record within the WINX control data set. This is a physical sequential data set with a DSN of "*userprefix*.FILEKIT.WINX", where *userprefix* is the string defined by INI variable SYSTEM.USERDSNPREFIX. Each saved record entry has a unique name reference which defaults to be the **fileid** of the data being edited/browsed in the current document window. If an entry with this name already exists within the WINX control data set, it will be replaced by the latest window save operation.

On restoring the window size and location of an edit/browse document, a WINX reference name is used to select the required window attributes record entry. Like window save, this name defaults to be the fileid assigned to the current edit/browse document window.

Accepting the default for both save and restore is a convenient method of managing the window size and location of any file displayed by the Text or Data Editor. However, using different document window attributes for multiple edit/browse views of the same data (i.e. views with the same fileid) may be achieved by assigning specific WINX reference names for the save/restore operation.

Execution of the WINX edit macro supports parameters PROMPT and RESTORE \*, both of which prompt the user for a reference name to be used on the save and restore respectively. Execute EM WINX to display the macro source which includes a description of supported parameters.

# Environment Variables

The text editor supports a set of system determined and user defined environment variables that may be used in primary commands and macros.

The types of variables supported and they way in which they may be utilised by the user are documented in this section.

## Variable Types

Text editor environment variables may be one of the following four types:

1. User defined environment variables set via the text edit EDITV primary command. Use **ERU** for a simple method of setting user environment variables.
2. Standard environment variables, as follow:

VarName	Description
<b>user</b>	User name as reported by 'Query USERNAME'
<b>datetime</b> <b>dtme</b>	Current local date and time in format 'yyyy/mm/dd hh.mm'. e.g. 2006/08/09 23.59
<b>timestmp</b>	Current local date and time in format 'yyyymmddhhmns'. e.g. 20060809235942
<b>date</b>	Current local date in format 'yyyy/mm/dd'. e.g. 2006/08/09
<b>yyyy</b>	Current 4 digit year. e.g. 2006
<b>yy</b>	Current 2 digit year. e.g. 06
<b>mm</b>	Current 2 digit month. e.g. 08
<b>dd</b>	Current 2 digit day of month. e.g. 09
<b>ddd</b>	Current 3 digit day of year. e.g. 221
<b>time</b>	Current local time in format 'hh:mm:ss'. e.g. 23:59:42
<b>tme</b>	Current local time in format 'hh.mm'. e.g. 23:59
<b>tsoprefix</b> <b>tsopfx</b>	For MVS, the defined TSO prefix. In many TSO environments, this has the same value as the %USER% userid environment variable.
<b>hh</b>	Current local 2 digit hour of day. e.g. 23
<b>mn</b>	Current 2 digit minute of hour. e.g. 59
<b>ss</b>	Current 2 digit second of minute. e.g. 42
<b>fi</b> <b>fid</b>	Current file's fileid as reported by 'Query Ffileid'
<b>fm</b>	Current file's filemode as reported by 'Query FMode'
<b>fp</b>	Current file's filepath as reported by 'Query FPath'
<b>fn</b>	Current file's filename as reported by 'Query FName'
<b>ft</b>	Current file's filetype as reported by 'Query FType'
<b>ds</b> <b>dsn</b>	Current file's DSname as reported by 'Query DSname'
<b>mcat</b>	The local Master Catalog DSN.

3. MVS system symbols. e.g. &SYSNAME.
4. INI variables that have been explicitly set in the System or User INI files.

These types of variables have the form "SYSTEM.category.varname" and "USER.category.varname" where:

<b>SYSTEM</b> <b>USER</b>	Variable is defined in either the System or User INI file.
<b>category</b>	The variable category name represented by a sub-section within the INI file. e.g. SYSTEM, EDIT, SELCOPY, RACF, etc.
<b>varname</b>	The variable's descriptive name within the specified category. e.g. InitialSize, CmdText, ProgramName, etc.

See [INI File Help](#) for all standard INI variables that are significant to the FileKit program. e.g. `SYSTEM.CBLVCAT.SVC`, `USER.EDIT.SizeWarning`, `SYSTEM.HELP.DefaultPath`

In addition to the standard INI variables, user defined variables may also be entered in the System and User INI files and referenced in the same way as the standard INI variables. User defined INI variables may be inserted using either of the following methods:

- ◆ Execute the CBL CLI command, **SET INIVAR**. The variable is set with immediate effect and is automatically inserted in the User INI file when the FileKit session is ended normally.
- ◆ Manually edit and update the relevant (User or System) INI file. The alterations are not immediate and will take effect the next time FileKit is started.

## Variable Substitution

FileKit supports use, and subsequent substitution, of text editor environment variables referenced in edit REXX macros or specified within any command string. This includes:

1. Commands in an edited file (typically a CMX file) that are executed using the **ACTION** facility.
2. Commands executed from a command prompt.

The text editor primary command **SET ENVVARS** switches variable substitution ON or OFF and also defines the variable delimiter character. By default, variable substitution is switched on with '%' (percent - X'6C') as the variable delimiter character.

By default, the environment variables, **user** and **system.edit.macropath**, will be translated in the following command.

```
set MACROPath %user%.CBLE.MACROS %system.edit.macropath%
```

The following example of concatenated command strings could be saved in your command (CMX) file for execution via **ACTION** (F16), to output a lists of library members updated by your userid today. Note that ';' (semi-colon - X'5E') is the command separator character.

```
<LL %user%.cbli.cble; WHERE USER = %user% & LASTMOD => '%date%' \
;ll %user%.cbli.cmx; WHERE USER = %user% & LASTMOD => '%date%'
```

The prevailing SET ENVVAR status (ON/OFF) may be temporarily overridden by prefixing the command with **VIgnore** or **VRespect**. VIgnore bypasses variable translation and VRespect performs variable translation regardless of the SET ENVVAR status.

The following command could be issued from a CBL REXX macro to temporarily bypass variable translation when ENVVAR is ON. (Upper case characters in the keywords indicate minimum abbreviation.)

```
<VIgnore Input '<ld %&SYSNAME%.Z16.** | List system data sets.'
```

The LISTDATASET command string, beginning '<ld ', will be inserted following the focus line of the edited data with %&SYSNAME% unchanged.



# ISPF Edit Interface

The text editor can execute in one of two operating interface modes namely, XEDIT and ISPF.

- The XEDIT interface is based on IBM's CMS XEDIT and Mansfield's PC Kedit.
- The ISPF interface is an operational mode based on IBM's MVS ISPF Edit.

Unless otherwise stated, features of the text editor documented in this manual are equally applicable to text edit views running in either operating interface mode.

This section of the manual deals specifically with features supported by the ISPF operating interface and the contrasts between the two operating interface modes.

```

SELFCOPY/i - CBL.CMX(ISPF)      252 V PDSE      Size=352  Alt=3,3;10
File Edit Actions Options Utilities Window SwapList Help  wS wR
Command>
.....+.....|<.....+.....>.....+.....3.....+.....4.....+.....5.....+.....6.....+.....7..
==CHG> xyz          ZZZ
==CHG> xyzd         ZZZd
==CHG> xyz          zZZZ
000035
=BND>
000036 <synex bounds * 30 !f xyz          Asterisk means current setting.
000037 <synex bounds 1 * !f xyz          Asterisk means current setting.
000038 <synex bounds 1 99999 !f xyz      Reset it.
000039 <synex bounds 33 !f xyz          Omit param to use defaults.
=BND>
.....+.....1.....+.....2.....+.....3.....+.....4.....+.....5.....+.....6.....+.....7..
000040 <synex bounds !f xyz
000041 | Default BOUNDS depends on the filetype.
000042
000043
000044
000045
.CAN .can          *** CANCEL          - (CBLe equiv: QQ)          ***
000047 <c can XXX all !cancel
000048
000049
000050
.C .c          *** Change/CHG          - (CBLe simil: Change)          ***
000052 <synex bounds 11 20 ;c xyz ZZZ          all
000053 <synex bounds 1 9999 ;c xyz ZZZ          all
000054 < chang xyz ZZZ .c .ce          first
000055 < chg xyz ZZZ .c .ce          last
000056 < c xyz ZZZ .c .ce          next
000057 xyz          def hij
==CHG> xyz          ZZZ
==CHG> xyzd         ZZZd
==CHG> xyz          zZZZ
000061 < c xyz ZZZ .c .ce          prev
000062 < c xyz ZZZ .c .ce          all chars
000063 < c xyz ZZZ .c .ce          all prefix
000064 < c xyz ZZZ .c .ce          all suffix
000065 < c xyz ZZZ .c .ce          all word
==CHG> <synex x zZZZ .c .ce first
000067 < c xyz ZZZ .c .ce          all
Te | Line=32 | Col=10 | Alt=3,3;10 | Size=352 | Recl=252 | Fmt=V | Files=2 | Vi

```

Figure 15. Text Edit View running ISPF interface mode.

## ISPF Edit Features

Wherever possible, the ISPF Interface is intended to mirror functionality included in the ISPF editor.

The functionality provided by ISPF features are documented fully in IBM manuals "ISPF: Edit and Edit Macros", "ISPF: User's Guide Vol 1" and "ISPF: User's Guide Vol 2".

Although not all features of ISPF Edit are included, the CBLe ISPF interface supports the following:

### ISPF and ISPF Edit Primary Commands

AUTOSAVE	EDIT	LINE_AFTER	SORT
BOTTOM	END	LINE_BEFORE	SUBMIT
BOUNDS (BND)	EXCLUDE (X)	LOCATE	TFLOW
CANCEL	FIND	MOVE	TOP
CAPS	FLIP	RCHANGE	TSPLIT
CHANGE (CHG)	HEX	REPLACE	UNDO
COPY	HIDE	RESET	UP
CREATE	LABEL	RFIND	VIEW
DELETE	LEFT	RIGHT	
DOWN	LINE	SAVE	

See [ISPF/XEDIT Command Precedence](#) for co-existence of ISPF and XEDIT commands.

## ISPF Edit Line Commands

(	BNDS	I	R
((	BOU	L	RR
)	BOUND	LC	S
))	BOUNDS	LCC	TF
<	C	LCLC	TS
<<	CC	M	UC
>	COL	MASK	UCUC
>>	COLS	MM	UCUC
A	D	O	X
B	DD	OO	XX
BND	F	R	

In addition to these standard ISPF line commands, the ISPF interface also supports the following CBLLe text editor [prefix area](#) line commands:

"	/	MB	SCALE
""	HEX	ML	SJ

## ISPF Edit Display Fields

- The ISPF edit enterable SCROLL field with valid entries: 0-9999, CURSOR, DATA, HALF, MAX and PAGE.
- Line labels.  
Note that the SETPT REXX macro may be used to set multiple labels corresponding to line markers in the file's text.
- Line command (prefix) area flagged text for changed lines (==CHG>), error lines (==ERR>) and special lines for boundary definition (=BNDS>) and column identification (=COLS>).

## ISPF PFKey/command line concatenation

The contents of the edit view command line are concatenated to the PFKey definition. The result is executed as a single command.

As for ISPF, the caveat exists whereby the PFKey definition is a scroll command and the concatenation of the command line contents results in an invalid scroll command. In this case, if the word following the PFKey scroll command definition begins with a non-numerical character, the contents of the command line are executed prior to the PFKey scroll function.

## ISPF Interface Initialisation

The prevailing operational interface mode is defined via the following methods:

1. The INI variable EDIT.INTERFACE may be set in either the System or User INI file. e.g.

```
(Edit)
Interface=ISPF      * ISPF or CBLLe (XEDIT).
```

This method defines the default interface for any new invocation of the text editor main application window.

If the INI variable EDIT.INTERFACE is not set, then INTERFACE=ISPF is the default on MVS type systems, whereas INTERFACE=XEDIT is the default for VM/CMS and VSE.

2. The primary command SET INTERFACE, allows the user to define the interface type to be used at the specified level. Supported levels are:

VIEW	The current edit view only.
FILE	All new and existing edit views of the current file.
GLOBAL	All new and existing edit views.

The parameter **Initialise** may also be specified on SET INTERFACE to initialise the edit view(s). This redefines the PFKeys to be the defaults for the specified interface and also updates the appearance of the edit view (e.g. SCROLL field on for ISPF). e.g.

```
SET INTERFACE ISPF VIEW INI
```

---

## ISPF/XEDIT Command Precedence

---

The set of text editor primary commands include most, commonly used ISPF Edit and XEDIT primary commands.

Where a primary or (prefix area) line command verb is unique to ISPF or XEDIT, then it may be executed successfully, regardless of the prevailing interface type. e.g. The XEDIT command, ALL, may be executed when in INTERFACE=ISPF. Conflict occurs when a command verb is common to both ISPF and XEDIT command sets.

The following are commands or SET options that exist for both command interfaces.

CANCEL CHANGE UP DOWN BOTTOM	TOP LEFT FIND DELETE COPY	MOVE RESET LOCATE REPLACE RIGHT	SPLIT SORT
--	---------------------------------------	---	---------------

When executing one of these commands, then, by default, the function and syntax of the command will be that associated with the active command interface. e.g. If INTERFACE=ISPF is active, the ISPF primary command, CHANGE, would be executed instead of the XEDIT form of the command.

The active command interface may be temporarily overridden by prefixing the command with **ICommand** or **ECommand**. ICommand passes the command to the ISPF command interface, ECommand to the XEDIT command interface.

The following command will use the XEDIT version of the CHANGE command when INTERFACE=ISPF is active. (Upper case characters in the keywords indicate minimum abbreviation.)

```
ECommand Change /ABC/DEF/ * *
```

---

# CMX (CoMmand eXecution) Files

---

Also referred to as command centres, CMX files are non-executable, plain text files that contain groups of associated TSO, ISPF, FileKit and CBL commands and comment data. These groups of commands may be used to perform regular tasks and procedures that would ordinarily be issued from a command prompt.

Execution of commands within a CMX file is achieved using FileKit's **ACTION** (CMDTEXT) facility. Simply place the cursor on the line of text containing the required command (usually a line starting with '<' to signify immediate execution) and pressing the F16 key (assigned to **ACTION** by default.) e.g.

```
<edit      DEV.XG80.JGE001.COB(XCC532)      | Edit COBOL source.
<submit    DEV.XG80.JGE001.JCL(XCC)        | Compile COBOL source.
<browse    DEV.XG80.JGE001.LST.XCC         | Browse SYSPRINT output.
```

## Note:

Use of ACTION is not restricted to files of type CMX. Commands may be stored, and subsequently executed, from any editable plain text file. (e.g. REXX procedures and SELCOPY, Assembler, C/C++, COBOL source files.)

Benefits of CMX files include:

- Productivity. Faster than using menus and dialog panels or re-typing commands from scratch.
- Session. Launch edit of frequently used files.
- Management. Single point of reference for related commands and procedures.
- Environment. Set and query the local operating environment options.
- Program Development. Submit jobs, display output and invoke debuggers.
- Education. Users can keep a record of supported command syntax and working examples.

The first time a user starts FileKit following initial install, the FIRSTUSE procedure is executed to establish the user's individual working environment (User INI file) and default command centre (CMX) files. During this process the user will be prompted to allocate these 2 new files.

The user's CMX file, also referred to as the user's HOME file, is generated from the FIRSTUSE skeleton CMX file distributed by CBL and is automatically opened for edit each time the user starts FileKit. This is the user's personal springboard into FileKit's functions and it is intended that the user add his or her own commands and comments.

This file also acts as a basic tutorial for functions provided by FileKit and it is recommended that the user take some time to read the comments and execute some of the commands in order to become familiar with the software's features and operation. Following an upgrade of FileKit, the user's CMX file may be automatically updated to reflect new features that have been added to the product since the last release.

---

# REXX Macros

User macros may be written to perform functions within a CBL e text edit or SDE data edit window using the REXX procedure language.

The name associated with the CBL e text edit environment is **CBLEEDIT**, whereas the equivalent name associated with the SDE data edit environment is **CBLSDATA**.

The appropriate environment name should be specified on the REXX instruction, ADDRESS, if commands within the macro are to be directed to a particular (text or data) edit environment. If a macro is executed from within a CBL e text edit document window, CBLEEDIT is automatically set as the default environment. Similarly, CBLSDATA is automatically set as the default environment if a macro is executed from within an SDE data edit document window.

The current environment may be identified using the REXX built-in function ADDRESS().

The string "**RXCCSID=ccsid**" may be entered in position 3 of any comment line within the macro to identify the CCSID of the system on which the macro was written. This will trigger automatic CCSID conversion on the REXX macro text to convert it from the specified CCSID (*ccsid*) to that of the local system.

This is necessary if a macro may be executed on systems with different local CCSIDs to that of the system on which the macro was written. For all CBL supplied macros that reference certain special characters, RXCCSID="285" is specified.

Unlike ISPF Edit, FileKit uses the same primary edit commands in macros as would be entered at a text edit command prompt (i.e. it does not employ different edit command syntax for use in edit macros.) This makes understanding the supplied macros and developing new edit macros much easier for users of all abilities.

A number of REXX macros are supplied with the SELCOPY Product Suit. A detailed description on the use of each macro is documented in comment data within the macro itself. (Execute "**EM macroname**" from an edit window command prompt to edit the macro source.) Where possible, **compiled versions** of these macros are included to improve execution performance.

BLOCK	Scan for next or previous highlighter line and make it the focus line.
BOX	Restrict operation of a CBL e command/macro to the <b>ZONE</b> and <b>RANGE</b> limits determined by the marked block.
BOXSEQ	Insert a numbers in sequence, one in each line of a marked block.
BOXTOT	Display the sum of all numeric values in a marked block.
CBLIINI	Display the fileid of the FileKit System and User <b>INI</b> files and the INI variables they assign.
CBLITU	Check current user is a trusted user. Executed once only by PROFIRST.
CBLIZAPL	Report all zaps that are applied to the FileKit modules being executed.
CCDATE	Output today's date at the cursor in the format ccy/mm/dd.
CMDX	Execute <b>ACTION</b> for all commands in the currently displayed text between the focus line and the specified line target.
CMEN	Generate a temporary CMX file containing all retrievable commands.
CMXAMS	Convert IDCAMS DEFINE input to FileKit <b>AMS</b> command format for execution using <b>ACTION</b> .
COLSET	Set colour scheme for the current display window.
CURALL	Generate ALL command for text at the cursor position.
CURSCALE	Use <b>SET RESERVE</b> to insert a scale line at the cursor position.
CUT	Simulate ISPF Edit primary command, CUT.
DELALL	Delete lines from the current file that match the specified line-target.
DELNOT	Delete lines from the current file that do not contain the specified string.
DIR	Generate edited list of library entries belonging to multiple libraries suitable for CMDFUNC execution.
DIRCMD	Convert DIR macro output or FileKit command FL, LL, LC, LD Edit output to CMX file format.
DIRSORT	SORT DIR macro output.
DSN	Provides utility functions to be executed for a DSN on which the cursor is positioned within the Text edit view.
EINI	Edit the System (site-wide) or User INI file.
EM	Edit the first macro in the macro path with the specified file name.
EQU	Set a CBL e user environment variable using EDITV.
ERA	Erase the file in the current display window.
ERASEALL	Create a CMX file containing an ERASE command for all files matching a supplied mask.
FILES	Create a temporary CMX file containinf EDIT commands for all files edited in this CBL e session.
FILESADD	Add the current fileid to the %FILES% environment variable. (Used by the FILES macro.)
FIRSTUSE	Install macro. Executed automatically for first ever logon to FileKit by a user to initialise the user's environment.
FSX	Search multiple MVS PDS(E) libraries for a specified search string.
GETAVRL	Get the average record length of the specified file. (Uses SELCOPY)
HD	Insert a CBL style header line in line 1 of the current file. (Suitable for SV macro)

IEX	For FileKit on MVS ISPF only, open a split screen containing ISPF panel nominated by an ISPF FastPath.
JCLCMX	Opens a temporary CMX file containing <b>ALLOCATE</b> commands generated from DD statements in an MVS batch job. Its usage is deprecated by support for SELCOPY Debug JCL input.
JEM	For FileKit on MVS ISPF only, sample FileKit interface to a JEM JCL validation ISPF-Edit macro.
JOBCARD	Insert a skeleton MVS jobcard in line 1 of the current file.
KBxxxxxx	Edit macros designed to be invoked by 3270 emulator keyboard macros that may be assigned to individual key strokes. See CBL web page <a href="#">3270 Terminal Emulation Software</a> to download the CBL supplied keyboard map configuration and keyboard macros for use with some popular 3270 emulators. Keyboard macros included in these .zip archives utilise the KB text edit macros.
LDIFF	Compares text in the current file with text in the next file in the ring and highlights the 1st difference found following the current line.
LLX	List members from multiple MVS PDS(E) libraries.
LM	Open a List Library window, one for each library in the macro path.
LVX	List data sets entries from multiple MVS DASD VTOCs.
MDL	Set current window's size to dimensions imposed by standard terminal hardware models.
MI	<b>SET CASE M I</b> prior to executing the supplied command and reset it when the command completes.
NAM	Place the current file's fileid on the command line for subsequent editing.
MOVEBLKR	Move a marked block and reset instead of leaving it marked.
MR	<b>SET CASE M R R</b> prior to executing the supplied command and reset it when the command completes.
OP	For FileKit on MVS ISPF only, gives quick access to the SDSF Operator System Log (LOG) panel and optionally execute an operator command.
OQ	For FileKit on MVS ISPF only, gives quick access to the SDSF Output Queue (ST Status of jobs) panel.
PASTE	Simulate ISPF Edit primary command, PASTE.
PROFILE	The CBL default <b>PROFILE</b> macro, executed on edit of a file.
PROFIRST	Called by PROFILE macro to perform once only operations for the Text Edit environment.
PROSITE	Called by PROFILE macro to apply Site wide overrides to default CBLE settings.
PROUSER	Called by PROFILE macro after PROSITE to apply User overrides to default CBLE and PROSITE settings.
QX	Display REXX stem variables returned by CBLe command <b>EXTRACT</b> for the specified extract option.
RENAME	For FileKit on MVS only, intercept CBLe RENAME and check for current fileid removing the ENQ if necessary.
RESTREAM	Re-Stream a file UNSTREAMed from a VSE LIBR RECFM=S member.
RINGL	Display the current ring of files as a popup menu. A file may be selected from the menu and made the current file. CIU option displays changed files only.
RST	Re-edit the current file discarding all unsaved changes.
SDBPOPOP	Display the SELCOPY Interactive Popup menu for the focus operation.
SDBTRACK	Performs SELCOPY Interactive command <b>TRACK</b> and colour the tracked expression in all edit views.
SDBWINX	Save and restore the size and locations of all edit view windows in the SELCOPY Interactive MDI frame window.
SDECOB	Structured Data Environment - Generate a COBOL copybook from SDE structure.
SDECOPYF	Structured Data Environment - Copy data in selected columns to new data set.
SDEPROF	Structured Data Environment - The CBL default SDE Edit profile macro.
SDERTALL	Structured Data Environment - Issue a command against all available record types.
SDESEL	Structured Data Environment - Generate SELECT command for all columns in the default record type.
SDEZOOMW	Structured Data Environment - Open a new Edit view containing the ZOOMed record.
SETPT	Scans the file for strings beginning with "." within specified zone columns and uses <b>SET POINT</b> to name the line.
SHOW	Un-exclude specified number of excluded lines.
SV	Save the current file if it has alterations and update the level in the CBL style header line. (See macro HD)
TRA	Capture TRACE output from a REXX procedure.
TRB	Insert REXX TRACE commands around a marked block of REXX statements in a macro file.
TSOC	Execute the specified TSO command then store and display any TSO messages returned in a temporary file.
UNNUM	Simulate ISPF Edit primary command, UNNUM.
UNSTREAM	Convert a VSE LIBR RECFM=S character file to RECFM=F.
VSECBLN	Insert CBLNAME ASSEMBLE and tailor for VSE specific variables. (VSE CBL product install from CMS.)
VSEINCL	Replaces 'INCLUDE module' records with the data from 'module TEXT *'. (VSE CBL product install from CMS.)
VSESITEV	Update Site-dependent variables in various VSE .Z skeleton JCL decks. (VSE CBL product install from CMS.)
VSESNAM	Insert SELCOPY NAM and tailor for VSE specific variables. (VSE CBL product install from CMS.)
WINX	Save and Restore the size and location of the current edit view within the MDI frame window.

WW	Start a new CBL or SDE window view of the current file and optionally execute a command.
----	--

All edit macros used in a FileKit session must exist within a library referenced by the macro path.

The macro path is a list of directories assigned to the **Edit.MacroPath** variable which may be set via the System or User **INI** file or via the CBL **SET MACROPATH** command. The macro path is searched in order until a file name that matches the macro name is found.

If no macro path is defined then the following occurs:

- For MVS systems, no action is taken. In order to execute CBL macros a macro path is mandatory.
- For CMS systems, CBL will search for a matching file name with a file type of **CBL**, **CBL****EDIT** or **XEDIT** (in order) on each accessed minidisk.  
Note that macro ABC.XEDIT.A will be found before ABC.CBL.B.
- For VSE systems, CBL will search the LIBDEF PROC library SEARCH chain for a matching member name.

If CBL is unable to locate a CBL macro then use CBL command **QUERY MACROPATH** to verify your macro path, otherwise, please contact your Systems Programmer.

Users should refer to IBM REXX documentation for assistance when writing edit macros in the REXX language. CBL supplied macros are also a good starting reference for examples of REXX procedures that use CBL commands.

```

SELCOPY/i - CBL.CBLI320.CBLE(DLALL) 255 V PDSE Size=56 Alt=0,0;2
File Edit Actions Options Utilities Window SwapList Help wS wR
Command> |.....1.....+.....2.....+.....3.....+.....4.....+.....5.....+.....6.....+.....7.....
/* ** CBL.CBLI.DIST.CBLE(DLALL) *** L=006 --- 2008/05/01 11:08:11 (JGE)
-----
Copyright: Compute (Bridgend) Ltd 2005 Tel: +44(1656) 652222
Eml: support@cbl.com
Web: www.cbl.com
-----
Format: DELALL string
e.g. DELALL /abc def/
Delete from current file all lines containing standard delimited 'stri
Displayed lines only, starting with the current line, are eligible for
<pcop %fid% s a g (ascii crlf lrecl 222 | -> VM.
<pcop %fid% s c %d% i | -> MVS.
L=005 2006/10/31 -jge- Needed a return value for strict CMS.
L=004 2006/08/17 -nbj- ECommand for CBLi < 1.3B.
L=003 2006/07/12 -nbj- ECommand on CBL commands if conflict with ISPF
L=002 2005/10/12 -jge- Prepared for distribution.
L=001 2005/08/16 -jge- Cribbed from .kex version L=006.
* /
env=address(); if env='CBLSDATA' then do; 'msg Macro DLALL is not SD
parse arg targ
'extract /version/'; if version.2 < '1.3B' then ECommand = '' /* EC
if targ = '' then do
'emsg DLALL.. No parameter given'
return 22
end
'extract /wrap/line/'; 'wrap off'
startlin = line.1
delcount = 0

```

Figure 16. CBL REXX edit macro - DLALL

## ISPF Edit Macros

In addition to its support for macros written using standard FileKit text edit and SDE data edit primary commands, CBL text edit also supports most ISPF Edit macros written in the REXX programming language. Note that this support must be activated by the individual user.

ISPF edit has traditionally been the editor of choice for most users of TSO and so many installations already make use of REXX macros written for the ISPF editor. To avoid the inconvenience of having to re-write these macros for FileKit text edit, most of the ISPF Edit macro (ISREDIT) command set and assignment statements are supported by the FileKit text editor. Therefore, in most cases, text edit will execute these same ISPF edit macros with no change to the macro syntax. This feature also allows users to write new FileKit edit macros using established ISPF edit macro techniques.

Commands passed to the ISREDIT environment, either directly or via the TSO or ISPEXEC environments, are intercepted and processed by FileKit. Calls via ISPEXEC to any other ISPF service are passed unchanged to the ISPEXEC environment.

To activate use of ISREDIT commands and the ISPF Edit Rexx macro search path, the ISR Macros option of the **Text Edit Settings (=0.3)** must be set to "YES" and FileKit restarted. When this option is set, any command entered at a text edit command prompt that is not recognised as a FileKit text edit command and is not identified as a FileKit text edit macro within the edit macro path, will result in a search of the **SYSUEXEC** and **SYSEXEC** library concatenations for a member of the same name. If found, the member will be loaded and executed as an edit macro.

The status for support of ISREDIT macro commands in FileKit is as follows:

Command	Supported	Command	Supported	Command	Supported
BROWSE	Yes	HILITE	Yes	RESET	Yes
BUILTIN	No	INSERT	No	RFIND	Yes
CANCEL	Yes	LEFT	Yes	RIGHT	Yes
CHANGE	Yes	LF	No	SAVE	Yes
COMPARE	No	LINE_AFTER	Yes	SEEK	Yes
COPY	Yes	LINE_BEFORE	Yes	SHIFT (	No
CREATE	Yes	LOCATE	Yes	SHIFT )	No
CUT	No	MACRO	Yes	SHIFT <	No
DEFINE	No	MEND	No	SHIFT >	No
DELETE	Yes	MODEL	No	SORT	Yes
DOWN	Yes	MOVE	No	SOURCE	No
EDIT	Yes	NONUMBER	No	TENTER	No
END	Yes	PASTE	No	TFLOW	Yes
EXCLUDE	Yes	PROCESS	No	TSPLIT	Yes
FIND	Yes	RCHANGE	Yes	UNNUMBER	No
FLIP	Yes	RENUM	No	UP	Yes
HIDE	Yes	REPLACE	Yes	VIEW	Yes

The status for support of ISREDIT macro Set and Query operations in FileKit is as follows:

Assignment	Query Supported	Set Supported	Assignment	Query Supported	Set Supported
AUTOLIST	Yes	No	MACRO MSG	No	No
AUTONUM	Yes	No	MASKLINE	No	No
AUTOSAVE	Yes	Yes	MEMBER	No	-
BLKSIZE	Yes	-	NOTES	No	No
BOUNDS	Yes	Yes	NULLS	No	No
CAPS	Yes	Yes	NUMBER	Yes	No
CHANGE COUNTS	Yes	-	PACK	No	No
CURSOR	Yes	Yes	PRESERVE	No	No
DATA CHANGED	Yes	-	PROFILE	No	No
DATA WIDTH	No	-	RANGE CMD	No	-
DATAID	No	-	RECFM	Yes	-
DATASET	Yes	-	RECOVERY	No	No
DISPLAY COLS	Yes	-	RMACRO	No	No
DISPLAY LINES	Yes	-	SAVE LENGTH	No	No
EXCLUDE COUNTS	Yes	-	SCAN	No	No
FIND COUNTS	Yes	-	SEEK COUNTS	Yes	-
FLOW COUNTS	No	-	SESSION	Yes	-
HEX	No	Yes	SETUNDO	Yes	-
IMACRO	No	No	STATS	No	No
LABEL	Yes	Yes	TABS	No	No
LEVEL	No	No	TABSLINE	No	No
LINE	Yes	Yes	USER STATE	No	No
LINE STATUS	No	-	VERSION	No	No
LINENUM	Yes	-	VOLUME	No	-
LRECL	Yes	-	XSTATUS	No	No
MACRO LEVEL	Yes	-			



---

## CBLLe PROFILE Macro

---

Unless option NOPROFILE has been specified, when a new file is opened for text edit, CBLLe searches the macro path libraries for the first occurrence of the profile macro name as specified on the EDIT or VIEW primary command, or defined in the INI file by variable **Edit.DefProfile**. If unspecified, CBLLe uses a profile macro name of **PROFILE**.

The profile macro may be used to define the user's CBLLe environment. Any System or User INI file options that correspond to CBLLe SET command options may be overridden by including the SET command in the PROFILE macro.

A default profile macro, PROFILE, is distributed with FileKit. This macro includes a call to other text edit macros as follow:

- PROFIRST - Perform once only operations for the Text Edit environment. (e.g. Set command synonyms.)
- PROSITE - For each file edited, perform operations which are common to all FileKit users at the local installation.
- PROUSER - For each file edited, perform operations which are specific to the current user.
- COLSET - For each file edited, set window colours and initiate syntax colouring based on file type.

These macros have been configured to provide a standard appearance and behaviour for CBLLe text edited data. However, copies of these macros may be updated by the FileKit site administrator and saved in the site specific CBLLe library as identified by the macro path search chain. Additionally, any user may copy a macro to their personal CBLLe macro library and update it as required.

---

# Primary (Command Line) Commands

---

Text Editor primary commands may be issued from:

1. A text editor command line.
2. A text file using the **ACTION** facility.
3. A **text edit macro** (address CBLEDIT).
4. A programmable **function key** in a text editor document view.

Multiple commands may be issued in a single invocation by separating each command with the special line end character as defined by **SET LINEND** (set to "!" exclamation mark by default.)

Note that, if LINEND ON ! is set and the user wants to temporarily switch it off in order to execute a command that contains the linend character as text, then this may be achieved by prefixing the command with the line end character. e.g.

```
!c/Hello there!/Goodbye/ 10 1
    Change command to change the 1st occurrence of "Hello there!" to "Goodbye" on the focus line and the next 9 lines.

!/==!/
    Locate a line-target of "=="!

! nomsg all "!!READ THIS!!"
    ALL command to display all lines containging string "!!READ THIS!!".
```

---

## Command Reference Syntax Conventions

---

### How to read the Syntax Diagrams

The following rules apply to the syntax diagrams used in this command reference.

1. The diagrams should be read from left to right, from top to bottom, following the path of the line.

- ◆ The >>- symbol indicates the beginning of a statement.
- ◆ The ->< symbol indicates the end of a statement.

2. Required items appear on the horizontal line (the main path).

```
>>--- required_item -----><
```

3. Optional items appear below the main path.

```
>>--- required_item ---+-----+-----+-----><
                        |               |
                        +-- optional_item -----+
```

4. If an optional item appears above the main path, then that item has no effect on the execution of the statement and is used only for readability.

```
                +-- optional_item -----+
                |               |
>>--- required_item ---+-----+-----+-----><
```

5. If you can choose from two or more items, they appear vertically, in a stack.

6. If you must choose one of the items, one item of the stack appears on the main path.

```
>>--- required_item ---+--- required_choice1 ---+-----><
                        |               |
                        +-- required_choice2 ---+
```

7. If choosing one of the items is optional, the entire stack appears below the main path.

```
>>--- required_item ---+-----+-----+-----><
                        |               |
                        +-- optional_choice1 ---+
                        |               |
                        +-- optional_choice2 ---+
```

8. If one of the items is the default, it appears above the main path and the remaining choices are shown below.

```
                +-- default_choice -----+
                |               |
>>--- required_item ---+-----+-----+-----><
                        |               |
                        +-- optional_choice1 ---+
                        +-- optional_choice2 ---+
```



```

<EDIT  USERNBJ.FTP.INPUT  ; UP MAX  ; DELETE ALL  \
;INPUT  anonymous          \
;INPUT  dummypass         \
;INPUT  ASCII              \
;INPUT  cd mvs/smpe/smpnts/CBL13295 \
;INPUT  get RFNJOB.TXT 'SYSV.SELC320.INIT.JCL(RFNJOB)' (replace \
;INPUT  quit               \
;SAVE \
;ALLOC  FILE(INPUT) REUSE SHR DSN('USERNBJ.FTP.INPUT') \
;TASK   ftp -parm cbl.ftp.com \
;FREE  FILE(INPUT) \

```

**Note:** The selected command text may extend beyond the length and depth of text visible in the window display area. i.e. Command strings are not limited by the size of the window display area.

The start of comment data within a line of text processed by the ACTION facility may be indicated by a 1-4 character string assigned by option **ACTIONCOMMENT**. This is of particular use where the delimiter character ('|' OR symbol) is part of the executable command text.

The following special characters may exist within the line of text and control how the ACTION facility interprets the command string to be executed. These characters are excluded from the command string.

#### | (OR symbol)

If option **ACTIONDELIM** is set on, '|' is the delimiter character used to partition a single line of edited text.

The ACTION facility will operate only on text at the focus column (cursor position) bounded on the left by '<', '>', '|' or the start of the line and bounded on the right by the '|' or the end of the line. Text outside these boundaries is ignored by the ACTION facility.

If '||' (2 consecutive OR symbols) follow '<' or '>' at the start of the line of text then the setting of ACTIONDELIM is temporarily reversed for that line of text only. i.e. If ACTIONDELIM ON is set, occurrences of '|' will not partition the line of text but will be treated as part of the executable command text.

#### < (less than)

Treated as a special character only if found within the first 4 characters of text at the start of a line or following the delimiter character '|' when ACTIONDELIM is set on. At all other locations within the text, it is treated as part of a command or comment string.

'<' indicates that, by default, the ACTION facility is to execute the command string immediately as opposed to placing it at the command prompt. However, if parameter EDIT or EDITALL is specified, then the command string will be placed at the command prompt.

Text preceding '<' at the start of the line or following a delimiter character is ignored. This allows special comment indication characters to be inserted before the '<' character without disrupting the command string interpretation and thus enable commands to be inserted in JCL, Assembler, COBOL, PL1 source, etc. e.g.

```

/**<sub ;OQ %JobName% | Submit and wait in SDSF for output.

```

#### > (greater than)

'>' has the same specification as '<' except that, instead of executing the command string immediately, it will be placed at the command prompt. Similarly, if parameter EDIT is specified, the command string will be executed immediately.

The action taken when '>' is found within the first 4 characters of text is identical to that taken when the ACTION facility is executed with no leading '<' character.

#### \_ (underscore)

If option **ACTIONCURSOR** is set on, the first occurrence of '\_' following '<', '>', '|' or the start of the line is treated as a special character. All other occurrences will be treated as being part of the command or comment string.

'\_' indicates the location at which the cursor is to be positioned when the command string is placed at the command prompt. e.g.

```

>edit USERNBJ.JCL(Job_01) | Change the job number and <Enter> to EDIT it.

```

#### ` (grave accent)

All occurrences of '`' are treated as null characters and are excluded from the command string. Its purpose is simply to allow alignment of text within the command strings on different lines of text. e.g.

```

>edit USERNBJ.JCL````(Job01) | Edit DSN 'USERNBJ.JCL(Job01)'.
>edit USERNBJ.OUTPUT(Job01) |

```

#### \ (backslash)

Treated as a special character only if found as the last non-blank character in the line of text. At all other locations within the text, it is treated as part of a command or comment string.

'\ indicates that the command string continues at the first character of the next line of text. There is no restriction on the number of lines over which the command string may be continued. e.g.

```

>alloc reuse f(OUTDD) new dsn('CBL.TEST.OUTPUT') \
space(1,1) cyl unit(3390) vol('DATT0B') \
recfm(F,B) lrecl(80) blksize(0)

```

```
>sub USERNB.JCL` `` (Job01) \
;e USERNB.OUTPUT (Job01) \
;e CBL.TEST.OUTPUT
```

Note: In the above example, ';' (semi-colon) is the LINEND command delimiter.

### Cursor Position following LOCATE:

If a LOCATE command is executed via the ACTION facility that causes the display to scroll, then the cursor position within the display area is preserved if the focus line occupies one of the first 10 lines of the display area. i.e. Having scrolled the display of text, the position of the cursor in the 3270 display is unchanged. In all other cases, scrolling will reposition the cursor in the current (1st) line of the display.

This behaviour allows use of a label name LOCATE command to scroll the display of edited text and have the cursor positioned on another label name LOCATE command which scrolls the display back to its original position. In this way the display may be toggled between two fixed points in the text simply by pressing F16 (ACTION).

e.g. In the following, executing ACTION on command '.main' will scroll the display leaving the cursor positioned on command '.data' which, when executed with ACTION, will scroll the display once again leaving the cursor positioned on command '.main'.

```
000269
.DATA ** .data ** Data Fields. ***
000271 <-- |.main | | | | | -->
000272
000273 <data>
000274 <field type="char" length="0002" id="Action" />
000275 <field type="enum" id="Formatted" enumid="eForm" />
000276 <file id="RptFid" dsn="YES" member="YES" title="Report" />
000277 </data>
000278
000279
000280
000281
.MAIN ** .main ** Basic File Search. ***
000283 <-- |.data | | | | | -->
000284
000285 <view id="main" width="78" depth="21" help="zzsiFSU9" >
000286 <title>FSU: Basic File Search</title>
000287
```

### Parameters:

EDIT  
If preceded by special character '<', the command string is placed at the command prompt for edit by the user before execution. Otherwise the command is executed immediately when the ACTION facility is run.

EDITALL  
Place the complete focus line and any continued lines at the command prompt. EDITALL disregards the delimiter character '|' regardless of the setting for ACTIONDELIM.

### Example:

The following command centre excerpt demonstrates how TSO, ISPF and FileKit primary commands may be saved in a text file for execution using the ACTION facility.

Note that TSO CONSOLE and LISTUSER commands require the appropriate RACF authorisation.

Command	Description
<tso console syscmd(d j,l)	Execute the TSO CONSOLE command to list active jobs.
<tso listuser *	Execute the TSO LISTUSER command to list all RACF users.
<vcat q cblname	Open the <b>CBLVCAT execution window</b> and list the contents of CBLNAME.
<lvol *	Open the <b>list DASD volumes</b> window to list all volumes.
<lc sys1.h	Open the <b>list calatog window</b> to list all cataloged datasets whose names begin sys1.h (MVS only).
<lc %user%	Open the <b>list calatog window</b> to list all cataloged datasets whose names begin with the current userid.
<ll sys1.help	Open the <b>list library</b> window to list all members of the sys1.help library (MVS only).
<ll prd2.*	Open the list library window to list all sublibraries of the prd2 library (VSE only).
<wl	Open the <b>window list</b> window.
<cal	Open the <b>calendar</b> window.
<calc x2d('1000')	Convert X'1000' to decimal using the <b>calculator</b> window.
<ll %user%.cbli.cble; where User = %user% & LastMod => '%date%' \	
;ll %user%.cbli.cmx ; where User = %user% & LastMod => '%date%'	List all library members changed by me today.

### See Also:

SET/QUERY/EXTRACT Options: **ACTION ACTIONCOMMENT ACTIONCURSOR ACTIONDELIM**



**Parameters:**

- line-target*  
Display only those lines which satisfy the **line-target** condition.
- R /*regexp*/  
Display only those lines which satisfy the **regexp** condition.

**Example:**

- all /x/|/y/  
Display only those lines which contain either an "x" or a "y" within the current zone limits.
- all changed  
Display only those lines which have been changed in the current edit session.
- all r/IQ#/  
Display only those lines which contain 'IQ' followed by a numeric character.

**See Also:**

[MORE](#) [LESS](#)

## ALLOCATE

**Syntax:**

```
>>-- ALLOCate  --+-----+-----+-----+-----+-----+----->>
                |         |         |         |         |
                +- -Cat  ---+   +--- allocparms ---+
                |         |         |         |         |
                +- -FREE  --+
                |         |         |         |         |
```

**Description:**

The ALLOCate command may be used to:

1. Invoke the **Allocate Non-VSAM** Dialog window if no parameters are specified.
2. Dynamically define and/or allocate a data set.
3. Concatenate a list of data sets.
4. Concatenate a data set to an existing list of data sets.
5. Free (unallocate) a data set or override DISP/CLASS. (Same as FREE command.)

ALLOCATE allows users to allocate files whether or not a TSO environment is available. The syntax of the command closely matches that of the TSO ALLOCATE command so most ALLOCATE commands, executed without TSO as a prefix, will give the same results.

ALLOCATE is supported for MVS only.

**Parameters:**

- CAT  
The data set being allocated is concatenated to an existing data set, or list of data sets, allocated to the specified ddname.
- FREE  
Unallocates the data set(s) allocated to the specified ddname.
- allocparms*  
Any parameter supported by the TSO ALLOCATE command plus the following:
- SUBSYS (*subsys-name*<,> *subsys-parm*> . . . )  
Directs the allocation request to the specified subsystem name with optional parameters. Null parameters can be specified by omitting a *subsys-parm*.

Parameters supported by the TSO ALLOCATE command area as follow:

BLKSIZE( <i>blocksize</i> )	MOD
BLOCK	NEW
BUFNO( <i>buffers</i> )	OLD
CATALOG	OUTDES( <i>output-descriptor-name</i> )
COPIES( <i>copies</i> )	PATH( <i>pathname</i> )
CYLinders	PATHDISP(KEEP DELETE<,KEEP DELETE>)
DATACLAS( <i>data-class</i> )	PATHMODE( <i>path-mode-list</i> )
DAtaset(*   ' <i>dsn</i> '..) DSName(*   ' <i>dsn</i> '..) DUMMY	PATHOPTS( <i>path-options-list</i> )
DDname( <i>ddname</i> ) File( <i>ddname</i> )	RECFM( <i>format</i> <, <i>format</i> ...>)
DELETE	RECORGL(S)
DEST( <i>dest</i>   <i>node</i> <, <i>user</i> >)	REUSE
DIR( <i>directory-blocks</i> )	SHR
DSNTYPE(LIBRARY , 1 2 )IPDS HFS LARGE)	SPACE( <i>n</i> <, <i>m</i> >)
DSORG(PS PO DA)	SPIN(UNALLOC)
FILEDATA(TEXT BINARY)	STORCLAS( <i>storage-class</i> )
FORMS( <i>forms</i> )	SYSOUT<( <i>class</i> )>
KEEP	TRACKS
LIKE('model- <i>dsn</i> ')	UNCATALOG
LRECL( <i>record-length</i> )	UNIT( <i>unit</i> )
MAXGENS( <i>member generations</i> )	VOL( <i>volser</i> <, <i>volser</i> ...>)
MAXVOL( <i>volumes</i> )	WRITER( <i>external-writer-name</i> )
MGMTCLAS( <i>management-class</i> )	

**Examples:**

```
alloc f(sysin) dsn('cbl.ssc.ctl(ssdemo01)') shr reuse
    Allocate an existing data set.
```

```
alloc f(sysudump) da('nbj.sysudump') cyl space(100,20) lrecl(133) blksize(0) recfm(v,b,a) new catalog
    Allocate a new data set. Note that DCB information may be omitted.
```

```
alloc f(multdsn) da('cbl.ssc.ctl(ssdemom1)' 'cbl.ssc.ctl(ssdemom2)') shr
    Allocate a new data set list.
```

```
alloc -cat f(multdsn) dsn('cbl.cm(xnbj)')
    Allocate a new data set to the data set list allocated to ddname 'multdsn'.
```

```
alloc -free f(multdsn)
    Unallocate data set(s) allocated to ddname 'multdsn'.
```

**See Also:**

FREE

---

**BACKWARD**

---

**Environments:**

BACK primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Lists	List display windows.
Interactive Panels	Panel window view navigation.
Help	HTML format Help windows.

**Syntax:**

```
>>-- Backward -----<<
```



**Description:**

The BACKWARD command scrolls the focus window backwards 1 page towards the top of the file.

**See Also:**

FORWARD

## BOTTOM

**Environments:**

BOTTOM primary command exists in the following application environments.

Text Edit (ISPF)	Text Editor with INTERFACE=ISPF (default for z/OS).
Text Edit (XEDIT)	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
Data Edit	SDE Data Editor.
System	FileKit base windows system.

### ISPF Interface

If INTERFACE=XEDIT is in effect, the ISPF version may be invoked by prefixing the command with ICOMMAND. See ISPF/XEDIT Primary Command Precedence.

**ISPF Syntax:**

```
>>-- Bottom -----><
```

**Description:**

BOTTOM is an alias for DOWN MAX which scrolls the display of the data downwards to display the last page of edited data.

**See Also:**

DOWN  
TOP

### XEDIT Interface

If INTERFACE=ISPF is in effect, the XEDIT version may be invoked by prefixing the command with ECOMMAND. See ISPF/XEDIT Primary Command Precedence.

**XEDIT Syntax:**

```
>>-- Bottom -----><
```

**Description:**

BOTTOM makes the last line in the file the focus line.

**See Also:**

TOP

## BOUNDS, BNDS

### Syntax:

```
>>--+ BOUunds -----+----->>
      |                |                |
      +- BNDs -----+  +--+ left_col  --+--+ right_col  --+--+
                        |                |                |
                        +--+ * -----+  +--+ * -----+  +--+
```

### Description:

Set the left and right column boundaries between which certain search, sort and shift operations will operate. BOUNDS is functionally equivalent to SET ZONE option and QUERY/EXTRACT ZONE will report the current left and right boundary (zone) columns.

Boundary columns may also be set by overtyping the < and > markers in the a special =BNDS> line in the text edit display. Special =BNDS> lines may be displayed at a fixed line within the display area using line (prefix) command BOUNDS (or BNDS) and then removed using RESET SPECIAL.

The current left and right boundary (zone) columns are represented by < and > markers respectively in the scale line and in any =BNDS> line.

Text editor primary (and ISREDIT macro) commands affected by boundary (zone) column values are as follow:

ALL CAPPEND CDELETE CFIRST CHANGE CLAST CLOCATE COMPARE COUNT	DELETE (1) EXCLUDE FIND (2) LESS LOCATE (1) LOWERCASE MORE MOVE (1) RCHANGE	RFIND SEEK SHIFT SORT TFIND TFLOW TSPLIT UPPERCASE
---	---	---

### Notes:

1. XEDIT interface format only.
2. ISPF interface format only.

Text editor line (prefix) commands affected by boundary (zone) column values are as follow:

([n] ([n] ( <[n] <<[n] <<	) [n] ) [n] ) > [n] >> [n] >>	TF TS
------------------------------------	--	----------

Execution of BOUNDS with no parameters resets the boundary columns to their default column numbers.

### Parameters:

*left\_col*

The left boundary column number. This value must be less than or equal to the right boundary column number. If "\*" (asterisk) is specified, the left boundary column is unchanged.

*right\_col*

The right boundary column number. This value must be greater than or equal to the left boundary column number. If "\*" (asterisk) is specified, the right boundary column is unchanged.

### Example:

bou 7 72

Set text boundary left and right column to be 7 and 72. e.g. for COBOL copy book source.

bnd \* 80

Set the right text boundary column only. The left column boundary is unchanged.



Use the BROWSE command to open a Structured Data Environment (SDE) **BROWSE** window view to browse a page of data from the specified fileid.

Use BROWSE instead of VIEW to browse large data sets. Unlike EDIT and VIEW, BROWSE does not need to load the entire file into storage in order to display a page of records.

### Parameters:

*fileid*

The fileid of the file to be browsed.

*fileid* may be any of the following:

- ◇ The DSN of a physical sequential data set.
- ◇ The DSN of a VSAM (KSDS, ESDS, RRDS, VRDS or LDS) data set.
- ◇ The library DSN and parenthesised member name of a PDS or PDSE library member.
- ◇ The library DSN, parenthesised member name and absolute or relative number of a PDSE library member generation as described under **z/OS PDSE Library Member Generations**. (PDSE version 2 with MAXGENS only.)
- ◇ The name of a member to be browsed from the same PDS or PDSE library as the member displayed in the **current data edit window** view. If no data edit window view is open and *fileid* is an IDCAMS alias name, then data belonging to the aliased dataset is browsed. Otherwise, it is treated as an HFS/ZFS file name in the user's current working directory.
- ◇ The name of an HFS/ZFS file system fileid.

SDE **BROWSE** Opt.s

See SDE **BROWSE** for supported parameters.

---

## CANCEL

---

### Environments:

CANCEL primary command exists in the following application environments.

<b>Text Edit (ISPF)</b>	Text Editor with INTERFACE=ISPF (default for z/OS).
<b>Text Edit (XEDIT)</b>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
<b>Data Edit</b>	SDE Data Editor.

---

## ISPF Interface

---

If **INTERFACE=XEDIT** is in effect, the ISPF version may be invoked by prefixing the command with **ICOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

### ISPF Syntax:

```
>>-- CANCEL ----->>
```

### Description:

ISPF CANCEL closes all text editor views that display the same file data with the a bias towards discarding all unsaved changes.

If the text edit view contains unsaved changes so that either the alteration count (**ALT**) is non-zero or the fileid update flag (**FIDCHANGED**) is ON, then CANCEL will display the CBLEDIT Close prompt with the cursor positioned on the "No" field. Pressing <Enter> to select the "No" option will close the edit views without saving the data. Selecting "Yes" will save the data and selecting "Cancel" (or pressing <F3>) will cancel the CANCEL operation and return focus to the edit view.

### See Also:

**END**

---

## XEDIT Interface

---

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

**XEDIT Syntax:**

```
>>-- CANCEL -----><
```

**Description:**

XEDIT CANCEL causes the text editor to execute **END** for all text edit views in the text editor ring. It does not affect document windows that are not text edit views.

The action taken for text edit views that contain unsaved data is determined by the current setting of the **AUTOSAVE** option. If unsaved data belonging to a view is not saved, then that view will not be closed and so will still exist in the text editor ring of edit views when the CANCEL command has completed.

**See Also:**

**CBLICANCEL** **END**

## CAPPEND

**Syntax:**

```
>>-- CAppend --+-----+-----><
                |         |
                +- string -+
```

**Description:**

Set the focus column to be the column immediately following the last character of the focus line and append the specified text to the focus line, starting at the the focus column.

**Parameters:**

*string*

Text string to be appended.

Where more than 1 blank separates the text string from the command verb, the additional separating blanks are treated as part of the text string.

**Example:**

ca abc

Append "abc" to the focus line.

ca abc

Append a single blank followed by "abc" to the focus line.

ca abc def

Append "abc def" to the focus line.

## CAPS

**Syntax:**

```
>>-- CAPS ---+-----+-----><
                |         |
                +- ON ---+
                |         |
                +- OFF -+
```

**Description:**





will text to the right of the replaced text be shifted to the right.

The prevailing **BOUNDS** left and right column values define the area of the record within which the search occurs. i.e. the matched data must begin at or after the left bound and not exceed the right bound.

The BOUNDS columns may be overridden using *pos1* and *pos2* positional parameters.

When a CHANGE command is used within an edit macro, statistical information relating to the last execution of CHANGE may be obtained as REXX compound variables using **EXTRACT CHANGE**.

**Parameters:**

*string1*

The CHANGE search string. The search string may be any of the following:

- ◇ An unquoted character string containing no commas or blanks. The search for the character string will be case-insensitive so that uppercase and lowercase characters are treated as being the same. This type of search string must not match a FIND parameter keyword.
- ◇ A character string enclosed in apostrophes (') or quotation marks ("). The search string may contain embedded commas and blanks and the character string will be case-insensitive.

Two adjacent quotation mark characters that are embedded in a search string which is enclosed by the same quotation mark characters, will be treated as a single occurrence of the character. e.g.

```
CHANGE 'That''s Entertainment' 'She''s The One'
```

Find the character string "That's Entertainment" and replaces it with "She's The One".

- ◇ A character string enclosed in apostrophes (') or quotation marks (") with the prefix C. This is equivalent to specifying a quoted search string but that the string search will be case-sensitive. (e.g. C'Book')
- ◇ A hexadecimal string enclosed in apostrophes (') or quotation marks (") with the prefix X.
- ◇ A picture string enclosed in apostrophes (') or quotation marks (") with the prefix P.

Picture strings use special characters to represent a generic group of characters as described below. Any character in a picture string that is not one of these special characters is untranslated.

String	Description
P'='	Any character.
P'~'	Any non-blank character.
P'.'	Any non-displayable character.
P'#'	Any numeric character, 0-9.
P'.'	Any non-numeric character.
P'@'	Any uppercase or lowercase alpha character.
P'<'	Any lowercase alpha character.
P'>'	Any uppercase alpha character.
P'\$'	Any non-alphanumeric special character.

- ◇ A regular expression string enclosed in apostrophes (') or quotation marks (") with the prefix R. For example:

```
R'C[0-9]^2'
```

is a regular expression which would match the upper case character **C** followed by 2 digits. This expression would match **C00 C91 C22** etc. but not **c99 C 99** etc.

Regular expressions enable powerful string pattern matching at the cost of rather complex syntax and potentially extended command processing time. For syntax and usage see **Regular Expressions** in the text editor documentation.

*string2*

The CHANGE replace string used to replace *string1*. The replace string may be expressed using any of the notations described for *string1* with the following differences:

- ◇ If expressed as a picture string, it must be of the same length as *string1* and may contain only the following special characters. Any character in a picture string that is not one of the special characters supported by picture strings, is untranslated.

String	Description
P'='	Same as the corresponding character in the search string.
P'<'	Change the corresponding character in the search string to lowercase.
P'>'	Change the corresponding character in the search string to uppercase.

- ◇ Where *string1* is a regular expression, *string2* may contain tag references to tagged sub-expressions of the regular expression search pattern defined by *string1*.



Tagged sub-expression reference is documented under **Regular Expressions** in text editor documentation.

ALL	All occurrences of <i>string1</i> are changed to <i>string2</i> . A message is displayed providing the number of occurrences of <i>string1</i> that have been changed. If NX is not specified, all excluded records that contain an occurrence of the <i>string1</i> are made visible whether or not text is successfully replaced.
FIRST	Search forwards from the Top of File indicator to find the first occurrence of <i>string1</i> and attempt to replace it with <i>string2</i> .
LAST	Search backwards from the End of File indicator to find the last occurrence of <i>string1</i> and attempt to replace it with <i>string2</i> .
NEXT	Search forwards from the current cursor location to find the next occurrence of <i>string1</i> and attempt to replace it with <i>string2</i> . If the cursor is not within the window's data display area, the search begins at the first position of the first visible or excluded record within the display area.
PREV	Search backwards from the current cursor location to find the previous occurrence of <i>string1</i> and attempt to replace it with <i>string2</i> . If the cursor is not within the window's data display area, the backwards search begins at the first position of the first visible or excluded record within the display area.
CHARS	CHARS indicates that a successful match occurs if the <i>string1</i> is found anywhere within the text being searched.
PREFIX	PREFIX indicates that a successful match only occurs if <i>string1</i> is found at the start of a word within the text being searched. i.e. the matched text must precede an alphanumeric character and either be preceded by a non-alphanumeric character or be at the start of a text line.
SUFFIX	SUFFIX indicates that a successful match only occurs if <i>string1</i> is found at the end of a word within the data being searched. i.e. the matched text must be preceded by an alphanumeric character and either precede a non-alphanumeric character or be at the end of a text line.
WORD	WORD indicates that a successful match only occurs if <i>string1</i> is found to be a complete word within the data being searched. i.e. the matched text must either be preceded by a non-alphanumeric character or be at the start of a text line, and either precede a non-alphanumeric character or be at the end of a text line.
EX X	Search EXCLUDED data records only.
NX	Search only visible data records (i.e. not EXCLUDED).
<i>pos1</i>	The first position of a range of positions within the lines of text to be searched.  <i>pos1</i> must be a positive integer value (not zero) and must be a value that is less than or equal to the maximum length of the data records.
<i>pos2</i>	The last position of a range of positions within the lines of text to be searched.  Like <i>pos1</i> , <i>pos2</i> must be a positive integer value. If <i>pos1</i> references a position which is higher than that referenced by <i>pos2</i> , then the <i>pos1</i> and <i>pos2</i> values are swapped.  If <i>pos2</i> is greater than the maximum record length then <i>pos2</i> is set equal to the maximum record length.  Default is <i>pos1</i> plus the length of the search string minus 1.
<i>.name1</i>	A label name identifying the first record of a range of data records to be searched. The preceding "." (dot) is mandatory. Default is .ZFIRST.
<i>.name2</i>	A label name identifying the last record of a range of data records to be searched. The preceding "." (dot) is mandatory. If <i>.name2</i> occurs before <i>.name1</i> in the display, then the order is reversed. If CHANGE PREV is executed and <i>.name1</i> is specified, the default is .ZFIRST. Otherwise the default is .ZLAST.

### Examples:

change c'PGM=SELCOPY' c'PGM=SLC' all 16 71  
Change all occurrences of the uppercase string 'PGM=SELCOPY' to 'PGM=SLC' between text positions 16 and 71.

chg p'@===== ' p'><<<<<<<' 1  
Change 8 characters beginning with an alpha character at the start of each text line so that the first character is upper cased and any alpha characters in the remaining 7 positions are lower cased. e.g. 'alison ' becomes 'Alison ', 'nicHoLAS' becomes 'Nicholas'.

ch r'IQ{:d^6}' r'ZZS&1' word .Q1 .Q2

Within a range of lines delimited by labels .Q1 and .Q2, change the next occurrence of a word 'IQnnnnnn' to 'ZZSnnnnnn' (where nnnnnn is the same 6 digit integer value in both strings.)

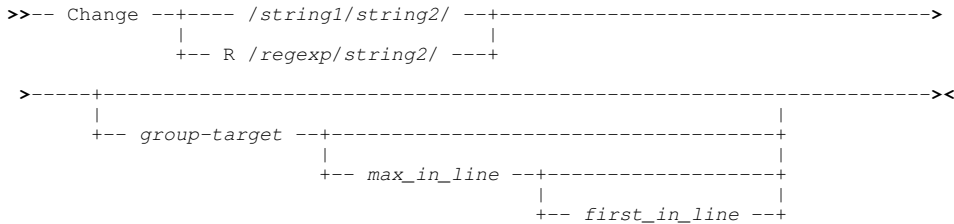
**See Also:**

EXCLUDE FIND RCHANGE RFIND

**XEDIT Interface**

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

**XEDIT Syntax:**



**Description:**

Change occurrences of edited text that matches *string1* or satisfies the condition defined by **regular expression** *regexp*, to be *string2*. This occurs for text in the focus line and in all lines up to, but not including, the line containing the first match for *group-target*.

The "/" (slash) character is normally used as the delimiter encompassing and separating the string arguments. However, any non-alphanumeric character that does not have a special meaning to CBLE may be used as a string delimiter (e.g. "%", "#", "."). The delimiter character used must not appear in the either of string arguments.

Where the *string1* and *string2* arguments do not contain special or blank characters, blanks may be used as the delimiter character. Where *regexp* is used, blanks can not be used as the delimiter character.

The length of the changed line may be increased or decreased where the length of text matched by *string1* or *regexp* is less than or greater than that of *string2*. Where the text length is increased, characters that extend beyond the truncation column are lost. By default, the truncation column is equal to the LRECL of the file.

The STAY setting determines which line is the focus line following a successful CHANGE command. If STAY is ON, the focus line remains unchanged. If STAY is OFF, the focus line is set to the last line of the target area (i.e. the line immediately preceding the group-target line.)

Where BLOCK is specified as the group-target, the CHANGE command will operate within the a currently-defined block. For line blocks, the CHANGE command operates on text within the current ZONE setting, whereas, for box blocks, the zone setting is ignored. Text is changed only if *string1* is contained entirely within the block boundaries. Text outside the box will remain unchanged, i.e. it will not be shifted left or right due to unequal string argument lengths. Similarly, truncation may occur at the block's right column boundary if *string1* < *string2*.

When a CHANGE command is used within an edit macro, statistical information relating to the last execution of CHANGE may be obtained as REXX compound variables using **EXTRACT CHANGE**.

**Parameters:**

/string1/string2/

Source string and update string.

**Note:** settings for ARBCHAR, CASE, HEXSTRING and ZONE affect the search for string1.

/string1/ may be prefixed by one of the following special keywords:

Word	String1 must match a complete word.
Prefix	String1 must match the leading characters of a complete word.
Suffix	String1 must match the trailing characters of a complete word.

R /regexp/string2/

Regular expression search string and text update string.

**Note:** settings for CASE and ZONE affect the search for *regexp*.

*string2* is not a regular expression but may contain tag references to tagged sub-expressions of the regular expression search pattern defined by *regexp*.

*group-target*

**Group-target** condition defining the end of the target area for the command. If the group-target is not satisfied then the CHANGE command will fail.  
Default is 1.

*max\_in\_line*

Integer specifying the maximum number of occurrences of *string1* that may be changed on an individual line in the target area. Alternatively, "\*" (asterisk) may be specified to indicate that all occurrences of *string1* are to be changed.  
Default is 1.

*first\_in\_line*

Integer specifying the first occurrence of *string1* in a line at which the change will be applied. Preceding occurrences of *string1* in the line remain unchanged.  
Default is 1.

**Example:**

```
change Hello Hi
```

Only change the 1st occurrence of the string "Hello" on the focus line to "Hi". The length of the line is reduced by 3 characters.

```
change /abc/def/ ALL 1
```

Change the 1st occurrence of the string "abc" on all lines.

```
change /x'0D0A'/x'0000'/ ALL 1
```

Where HEXSTRING is ON, the 1st occurrence of the 2 byte hex string x'0D0A' are changed to hex string x'0000' on all lines.

```
change /X Y/z/ :100 2 3
```

Change the 3rd and 4th occurrences of the string "X Y" on the focus line and all lines following, up to but not including line 100.

```
change #A/B#X/Y# ~/Ref:/ *
```

Change all occurrences of the string "A/B" on the focus line and all lines following, up to but not including the first line that does not contain the string "Ref."

```
c word /All/Most/ 1 *
```

Change all occurrences of the word "All" on the focus line. **Note:** "Allocate" will not be changed.

```
c r /[0-9]ABC/ABC&1/ 1 1
```

Uses a regular expression to search the focus line of text for any single numeric character followed by 'ABC'. A tagged sub-expression reference operator is used in *string2* to identify the numeric value in the text that satisfies *regexp*. e.g. '5ABC' will be changed to 'ABC5'.

**See Also:**

SET/QUERY/EXTRACT Options: [ARBCHAR](#) [CASE](#) [CHANGE](#) [HEXSTRING](#) [ZONE](#)

---

## CHANGEDIALOG

---

**Syntax:**

```
>>-- CHANGEDialog -----><
```

**Description:**

Open the [CBLed Change Dialog Window](#) to perform ISPF edit style FIND and CHANGE commands.

This dialog window may also be opened by selecting **Change** from the **Edit** menu item in the [CBLed Main Menu Bar](#).

**See Also:**

[CBLed Change Dialog Window](#)

---

## CINSERT

---

**Syntax:**

```
>>-- CInsert --+-----+-----><
                |         |
                +- string -+
```

**Description:**

Insert a text string into the focus line starting at the focus column. Existing text in, or to the right of the focus column will be shifted to the right for the length of the inserted text string.

**Parameters:**

*string* Text string to be inserted.

Where more than 1 blank separates the text string from the command verb, the additional separating blanks are treated as part of the text string.

**Example:**

```
ci abc
Insert "abc" in the focus line at the focus column.
```

```
ci  abc
Insert a single blank followed by "abc" into the focus line.
```

```
ci Hi Jane
Insert "Hi Jane" in the focus line at the focus column.
```

---

## CLAST

---

**Syntax:**

```
>>-- CLAst -----><
```

**Description:**

Position the focus column at the right zone column.

**Example:**

```
cla
Set focus column to column 50 (if right ZONE boundary is set to 50).
```

**See Also:**

SET/QUERY/EXTRACT Option: [ZONE](#)

---

## CLIPBOARD

---

**Syntax:**

```
>>-- CLIPboard +-+-----+-----><
                |         |
                +-----+-----+
                |         |
                +-+-----+-----+
                |         |
                +- APPEND +-+ CUT -----+
                |         |
                +- PUT -----+--- BOX ---+ string ---+
                |         |
```



```
clip append put box This is some text.
```

The string " This is some text." (1 preceding blank) is appended to a new line in the clipboard as a box block.

```
clip clear
```

Clear the contents of the clipboard.

### See Also:

**CUT MARK PASTE**

## CLOCATE

### Syntax:

```
>>-- CLocate --+-----+-----><
                |         |
                +- column-target -+
```

### Description:

Locate and then position the focus column at a column target. In some cases, the focus line will also be set.

The focus line is scanned for column-target and, if successful sets the focus column to be the target column. If no match is found in the focus line and STREAM is set ON, then subsequent lines are scanned until a match is found, in which case the line containing the match becomes the focus line.

Where column-target is not specified, CBLc repeats the last CLOCATE command issued.

### Parameters:

*column-target*  
Column-target condition.

### Example:

```
c1 3      Focus column is set 3 columns to the right of its current position.
c1:50    Focus column is set at column 50.
c1/Hello/ Focus column is set at the start of the first occurrence of the string "Hello".
```

## CMSG

### Syntax:

```
>>-- CMSG --+-----+-----><
                |         |
                +-- string --+
```

### Description:

Place a text string on the command line. CMSG is most often used in macros to place a command on the command line.

If no text string is specified, then the command line is cleared.

### Parameters:

*string*  
Text string to be placed on command line.

### Example:

```
cmsg 'ALL /test data/'
```

Place the string "ALL /test data/" on the command line.

---

## COMMAND

---

### Syntax:

```
>>-- COMMAND -- command -----><
```

### Description:

The COMMAND command temporarily disables CBL's synonym processing.

When SYNONYM ON is in effect, a command entered from a CBL command line is automatically checked to determine whether it is the name of a defined synonym. If so, the action taken will be that specified by the synonym definition.

Prefixing a command or macro name with COMMAND will bypass synonym checking for the command/macro being executed.

### Parameters:

```
command
```

Any CBL command or macro name followed by its parameters.

### Example:

```
command ALL /=/
```

The CBL command ALL is executed without checking whether a synonym for ALL exists.

### See Also:

**SYNEX** and the SET/QUERY/EXTRACT Option: **SYNONYM**

---

## COMPARE

---

### Syntax:

```
>>-- COMPare -- fileid1 --- fileid2 -----><
```

### Description:

Compare lines of text in two files that are displayed in existing edit views within the current CBL edit environment.

Text starting at the left **ZONE** boundary of the line following the current line in file 1 is compared with text starting at the left **ZONE** boundary of the line following the current line in file 2.

Where the zone widths of the two files differ, then, for the compare operation only, the lines belonging to the file with the shorter zone are assumed to be padded with blanks. The text compare is case sensitive unless **SET CASE MIXED|UPPER IGNORE** is in effect in **both** file edit views.

If the compared lines match, then the text in the next line in sequence following the current line on file 1 is compared with the equivalent line in file 2. This process is repeated until either the lines being compared do not match or the end of **RANGE** is encountered for one of the files.

Note that, where the **RANGE** is not explicitly set for an edited file, then the "End of File" line is the end of range. If end of range is encountered on one file before it is encountered on the other, then a difference is flagged.

If no differences are found, the current line is unchanged, otherwise the first line that contains a difference becomes the current line in both views.

### Parameters:

```
fileid1
```

The fileid of the first file to be compared. The file must exist in an edit view.

*fileid2*

The fileid of the second file to be compared. The file must exist in an edit view.

**Example:**

```
compare CBL.ACCT.X017993.T070125 CBL.ACCTLIB(X017993)
Compare text in a sequential file with that in a PDS member.
```

**See Also:**

SET/QUERY/EXTRACT Options: **RANGE ZONE**

## COPY

**Environments:**

COPY primary command exists in the following application environments.

<b>Text Edit (ISPF)</b>	Text Editor with INTERFACE=ISPF (default for z/OS).
<b>Text Edit (XEDIT)</b>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
<b>Data Edit</b>	SDE Data Editor.

### ISPF Interface

If **INTERFACE=XEDIT** is in effect, the ISPF version may be invoked by prefixing the command with **ICOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

**ISPF Syntax:**

```
>>-- Copy -----+-----+-----+-----+-----+----->
      |             |             |             |             |
      +-- fileid --+ +-- AFTER --+ .name --+
              |             |
              (1) +- BEFORE -+
      >-----+-----+-----+-----+-----<<
      |             |             |             |             |
      +-----+-- start_line -+ * -----+
      |             |             |             |             |
      +- FROM --+             +- end_line -+
```

**Note:**

1. If AFTER and BEFORE are omitted, then line (prefix) command "A" or "B" must have been specified in the the current text edit view.

**Description:**

Copy lines from an existing sequential or VSAM data set, PDS/PDSE library member or HFS file into the focus text edit view.

The line within the focus text edit view that identifies the target of the copy is specified via a line label name. If BEFORE or AFTER line label name is not specified, then the target line is the first line containing the line (prefix area) command "A" (AFTER) or "B" (BEFORE). If neither exists, error message ZZSE130I is returned.

If COPY is entered with no parameters, a popup window is displayed which prompts the user to enter a fileid from which all records will be copied.

**Parameters:**

*fileid*

Specifies the fileid of the source file from which records are to be copied.

If *fileid* is a less than 8 characters in length and is a valid member name, the target file will be a member of member name *fileid* belonging to the same PDS/PDSE library referenced in the current text editor view. If the focus edit view does not display a library member, the *fileid* will be treated as an HFS file within the current HFS working directory.



If *fileid* includes a volume id, then the source file may be a cataloged or uncataloged data set or PDS/PDSE library which exists in that volume's VTOC. e.g. VOLWKA:DEV.UNCATLG.FILE.

AFTER | BEFORE *.name*

Identifies whether lines are to be copied after or before the target line specified by label *.name* in the current text edit view.

[ FROM ] *start\_line*

Identifies the first record number in the group of records belonging to *fileid* to be copied into the current text edit view. Default is record 1.

*end\_line*

Identifies the last record number in the group of records belonging to *fileid* to be copied into the current text edit view.

If *end\_line* is greater than *start\_line*, the values are swapped so that a positive range of line numbers are defined.

Default is "\*" (asterisk), the last record belonging to *fileid*.

### Examples:

```
copy CBL.JCL(ALLOCX1) before .STEP1
```

Copy JCL member ALLOCX1 before labelled line .STEP1 in the current edit view.

### See Also:

CLIPBOARD CREATE CUT MOVE PASTE REPLACE

## XEDIT Interface

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

### XEDIT Syntax:

```
>>-- Copy -- group-target ---+-----+-----><
                        |               |
                        +- line-target -+
```

### Description:

Copy text from a target area to a target line.

Where BLOCK is specified, COPY supports copying text between files. Otherwise COPY can only operate on lines in the same edited file. The first line of the copied block becomes the focus line.

Where BLOCK is not specified as the group-target, the last line of the copied target group of lines becomes the focus line.

Where line-target is omitted, the current focus line is used.

### Parameters:

*group-target*

**Group-target** condition defining the end of the source target area. If the group-target is not satisfied then the COPY command will fail.

If BLOCK is specified, then the marked block will be copied as follows:

#### Line Block

Marked line(s) are copied to the line immediately following the line-target.

#### Box Block

Marked box is copied to the focus column of the target line.

**Note:** group-target key word ALL is not supported.

*line-target*

**Line-target** condition defining a destination target line. If the line-target is not satisfied then the COPY command will fail. Where the source target area is not a box block, lines are copied to the line immediately following the target line.

### Examples:

```
copy 8 :28
```

The focus line and the 7 lines following are copied to line below line 29.

```
co -4 6
```

The focus line and the 3 lines preceding are copied below the 5th line following the focus line.

- co 3 -/SELC/  
The focus line and the 2 lines following are copied to the line following the first line containing the string "SELC", scanning backwards from the focus line.
- co prefix /call/ word /state/  
The focus line and all lines up to, but not including, the first line to contain a word beginning "call" are copied to the line below the first line following the focus line that contains the word "state".
- co block  
The marked box block is copied to the focus column of the focus line and lines that follow up to the depth of the marked block.

**See Also:**

**MOVE**

## COUNT

**Syntax:**

```

>>-- COUnt -- /string/ -----><
                +----- 1 -----+
                |                   |
                +-----+-----+
                |                   |
                +- group-target -+
    
```

**Description:**

Count occurrences of string on the focus line and on lines up to, but not including, the line containing the first match for group-target.

The "/" (slash) character is normally used as the delimiter encompassing and separating the string arguments. However, any non-alphanumeric character that does not have a special meaning to CBL may be used as a string delimiter (e.g. "%", "#", "."). The delimiter character used must not appear in the either of string arguments.

Where the string arguments do not contain special or blank characters, blanks may be used as the delimiter character.

The STAY setting determines which line is the focus line following a successful COUNT command. If STAY is ON, the focus line remains unchanged. If STAY is OFF, the focus line is set to the last line of the target area (i.e. the line immediately preceding the group-target line.)

Where BLOCK is specified as the group-target, the COUNT command will operate within the a currently-defined block. For line blocks, the COUNT command operates on text within the current ZONE setting, whereas, for box blocks, the zone setting is ignored. The count is incremented only if string is contained entirely within the block boundaries.

When a COUNT command is used within an edit macro, statistical information relating to the last execution of COUNT may be obtained as REXX compound variables using **EXTRACT COUNT**.

**Parameters:**

*/string/*  
The search string.

**Note:** settings for ARBCHAR, CASE, HEX and ZONE affect the search for string.

*/string/* may be prefixed by one of the following special keywords:

<b>WORD</b>	String must match a complete word.
<b>PREFIX</b>	String must match the leading characters of a complete word.
<b>SUFFIX</b>	String must match the trailing characters of a complete word.

*group-target*  
**Group-target** condition defining the end of the target area for the command. If the group-target is not satisfied then the COUNT command will fail.

**Example:**

count esc  
Count occurrences of the string "esc" on the focus line.

count /abc/ ALL  
Count occurrences of the string "abc" on all lines.



- ◆ **C** or **CC** (Copy), if the group of lines in the current file is to be preserved following successful execution of CREATE.
- ◆ **M** or **MM** (Move), if the group of lines in the current file is to be deleted following successful execution of CREATE.

**Description:**

Create a new sequential or VSAM data set, PDS/PDSE library member or HFS file, containing a group of lines extracted from the current text edit view.

If the target file already exists, then the following message is returned:

```
ZZSE190E File fileid already exists.
```

If output is to a new member of an existing PDS/PDSE library or to an HFS file, the target file will be created automatically. When CREATE defines a new HFS file, the permission bits are set to 740 (rwxr----).

If output is to a non-existent data set or to a member of a non-existent PDS/PDSE library, the **Allocate Non-VSAM** panel is displayed in order to allocate the new data set.

Beware that the specified *fileid* character string must not be "BUTTON" (minimum abbreviation BUT) which conflicts with text edit primary command **CREATE BUTTON**.

If CREATE is entered with no parameters, a popup window is displayed which prompts the user to enter the fileid of the file to be created.

**Parameters:**

*fileid* Specifies the fileid of the target file to be created.

If *fileid* is a less than 8 characters in length and is a valid member name, the target file will be a member of member name *fileid* belonging to the same PDS/PDSE library referenced in the current text edit view. If the current text edit view does not display a library member, the *fileid* will be treated as an HFS file within the current HFS working directory.

*.name1* A label name identifying the first line of the group of lines to be copied to the target file.

If not specified, then the group of lines must marked using "C" or "M" line (prefix area) commands.

*.name2* A label name identifying the last line of the group of lines to be copied to the target file. If *.name1* has been specified, *.name2* is mandatory.

**Examples:**

```
create DEV.USER223.COB.COPY(QX95R001) .GRBEG .GREND
Create new library member "DEV.USER223.COB.COPY(QX95R001)" and copy records from the current text edit view which fall between defined line label names .GRBEG and .GREND inclusively.
```

**See Also:**

**CLIPBOARD COPY CUT PASTE REPLACE**

## CREATE BUTTON

**Syntax:**

```
>>-- CREate BUTton /name/command/ --<<
+-----+
v      |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
+--| OptA |--+ | +-- 0 --+ +-- 1 --+ |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
+-- row +-----+-----+
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
+-- col --+
```

**OptA:**

```
|-----+-----+-----+-----+-----+-----+-----+-----+-----+
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
+---- PRESsed ----+ | OptB |-----+
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
```

```
+---- CURSor -----+
| +- KEEP ---+      |
| |             |      |
+-----+-----+
| |             |      |
| +- PASS ---+      |
+-----+-----+
```

**OptB:**

```
+-- Blue (1) --+
+- White (2) --+ +- REVvideo --+
| |             | |             |
+-----+-----+-----+
| |             | |             |
+- Green -----+ +- Blink -----+
| |             | |             |
+- Pink -----+ +- NONE -----+
| |             | |             |
+- Red -----+ +- Uscore -----+
| |             | |             |
+- Turquoise --+
| |             | |             |
+- Yellow -----+
+-----+-----+-----+
```

**Notes:**

1. Default for PRESSED.
2. Default for COLOUR.

**Description:**

Create a selectable button within the MDI parent display which, when pressed, executes the associated command or macro.

The "/" (slash) character is normally used as the delimiter encompassing and button name and associated command. However, any non-alphanumeric character that does not have a special meaning to CBL may be used as a string delimiter (e.g. "%", "#", "."). The delimiter character used must not appear in the name or command strings.

CBL distributes the edit macro PROFIRST which uses the CREATE BUTTON command to add some useful buttons to the CBL and SELCOPY Interactive MDI windows when they are first opened.

**Parameters:**

COLOUR  
COLOR

Define the colour and extended highlighting to be applied to the button text when it is in an unselected state.

Default is WHITE REVVIDEO.

PRESSED

Define the colour and extended highlighting to be applied to the button text when it is in a selected (pressed) state.

The button is in a selected state for the duration of the command execution after which the button reverts back to being in the unselected state. Unless the screen is refreshed during execution of the command (e.g. if CURSOR specified), the user does not see the button in a selected state.

Default is BLUE REVVIDEO.

CURSOR

Delay execution of the command until the <Enter> key is hit, allowing the user to position the cursor in the display area within an edit view prior to executing the command. CURSOR should be used where the command is sensitive to the cursor position.

If the cursor is positioned on a command line when <Enter> is hit or if any other AID key is hit while the cursor is within an edit view, then the button reverts back to the unselected state and the command is discarded.

KEEP  
PASS

Prior to executing the command, either keep the cursor in its current position when the button is selected (i.e. positioned on the button) or PASS it back to the command line of the current edit view.

Default is KEEP.

row

Row within the MDI parent display at which the button is to be positioned. Rows are numbered from 0 (zero) to the depth of the MDI parent display area where row 0 is the row containing the menu bar.

Default is 0.

col

Column within the MDI parent display at which the button is to be positioned. Columns are numbered from 0 (zero) to the width of the MDI parent display area where column 1 is the column containing the system menu button.

Default is 1.

*name*

Defines the name assigned to the button and the text displayed on the button itself.

*command*

The command string to be executed. This may be any command string supported by the current edit view.

Where more than one command is to be executed, they should be invoked via a macro name (in storage or saved to disk) or via the IMMEDIATE command. The LINEND character is not respected in a CREATE BUTTON command.

**Example:**

```
create button 0 80 /HW/msg Hello World/
crea but col yell uscore press yell rev 1 1 /HD/macro HD/
crea but col gr non 1 5 /Ring/imm 'ext ring';do i=1 to ring.0;'msg' ring.i;end/
crea but col bl non press wh rev cursor 1 10 /ColN/imm 'ext cursor';'msg' cursor.2/
crea but press turq rev pass 0 85 #CL#cmsg imm 'stream on';'cl/xxx/';'stream off'#
```

**See Also:**

**DESTROY BUTTON**

## CREPLACE

**Syntax:**

```
>>-- CReplace --- string -----><
```

**Description:**

Replace text in the focus line with the specified text string starting at the focus column.

All characters supplied in the text string, including leading, imbedded and trailing blanks, replace characters in corresponding positions in the focus line.

**Parameters:**

*string*

Replace existing text with this text string.

Where more than 1 blank separates the text string from the command verb, the additional separating blanks are treated as part of the text string.

**Example:**

```
cr abc
Replace text at focus line and column with "abc".
```

```
cr ABC_X YZ
Original text at focus line and column is: 0123456789 After column replace command text is: ABC_X YZ9
```

## CURSOR

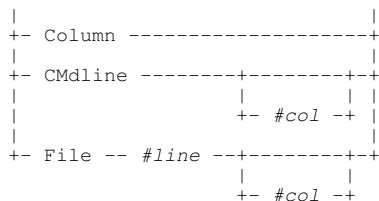
**Environments:**

CURSOR primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor.

**Syntax:**

```
>>-- CURSOR --+ HOME -----><
```



**Description:**

Position the cursor within the current edit window view. The CURSOR command is most often used in macros.

**Note:** The CURSOR command does not alter the window view itself. i.e. the current line and column remains unchanged.

**Parameters:**

HOME  
 If the cursor is on the command line, the cursor is positioned at the line and column number at which it was positioned when it was last in the file area. If this line is not in the current edit view, the cursor is positioned in column 1 of the current line.

If the cursor is in the file area, the cursor is positioned at column 1 of the command line.

COLUMN  
 The cursor is positioned at the focus column of the focus line.

CMDLINE #col  
 The cursor is positioned at column #col of the command line.  
 Default is column 1 of the command line.

FILE #line #col  
 The cursor is positioned at line number #line of the file. If #col is specified, the cursor is positioned at this column number within #line.  
 Default is column 1 of #line.  
 If the specified file line or column number is outside the current file display area, then error message **ZZSE044E** is returned. e.g.

```
ZZSE044E Invalid cursor line or column in command cursor file 138 20.
```

---

## CUT

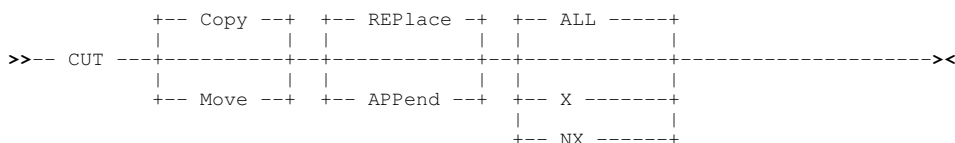
---

**Environments:**

CUT primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor.

**Syntax:**



**Description:**

CUT may be used instead of primary command **CLIPBOARD** to copy or move a group of lines or marked block of lines to the FileKit **clipboard**. Once in the clipboard, **PASTE** may be used to copy the text from the clipboard to another text edit view.

Before text can be copied to the clipboard, it must first be marked within the text display of an edit view. This may be done using prefix commands C(n), CC, M(n) or MM; or using <F17> or <F18> which are, by default, assigned to **MARK BOX** and **MARK LINE** respectively. An error message is displayed if there is no block marked in the current file.

If M(n) or MM is used to select a group of lines to be moved to the clipboard, then CUT COPY is equivalent to CUT MOVE.

Parameters COPY and REPLACE are default and so CUT with no parameters is equivalent to CLIPBOARD COPY.

### Parameters:

**COPY**  
Copy the group of lines or marked block of text to the clipboard. If M(n) or MM is used to select a group of lines, then COPY MOVE is performed. Otherwise the group of lines or marked block of text within the display area is preserved following the copy.

CUT COPY is equivalent to CLIPBOARD COPY.

**MOVE**  
Move the group of lines or marked block of text to the clipboard so that it no longer exists within the display area.

CUT MOVE is equivalent to CLIPBOARD CUT.

**REPLACE**  
Replaces any text that already exists in clipboard storage. This is default.

**APPEND**  
Append data to a new line following the existing clipboard text.

CUT COPY APPEND is equivalent to CLIPBOARD APPEND COPY. CUT MOVE APPEND is equivalent to CLIPBOARD APPEND CUT.

**ALL | X | NX**  
Indicates whether all (ALL) lines, only excluded lines (X) or only non-excluded lines (NX) that exist within the selected group of lines or marked block of text are to be included in the copy or move operation. ALL is default.

### Examples:

**cut**  
Take a copy of a group of lines in the current text edit view selected using CC pair of line commands.

### See Also:

[CLIPBOARD COPY](#) [CREATE](#) [PASTE](#) [REPLACE](#)

## DEFINE

### Syntax:

```
>>-- DEFine -----><
      |
      +- fileid -----+
      |
      +- macroname -+-----+
                |
                +-- macrodef --+
```

### Description:

DEFINE is used to load an existing CBL REXX macro from disk into storage or to assign a temporary macro definition in storage.

DEFINE, with no parameters, will display the macro names of macros that have been loaded into storage.

Keeping a copy of a macro in storage is beneficial when it is to be called frequently. The disk I/O involved in loading the macro is not repeated each time the macro is called and so performance is improved. However, keeping a macro in storage will incur storage overheads.

Use the **PURGE** command to remove macros from storage.

### Parameters:

*fileid*  
The full fileid of the macro to be loaded into storage.

*macroname*  
The macroname (filename) of the macro file on disk, or the name assigned to a new macro definition, to be loaded into storage.





If no range of lines is specified, an implicit range of .ZFIRST .ZLAST is used (i.e. all lines of text in the edit view.) If only the first line of a range of lines is specified (*.name1* or *line1*) then the default last line is the same as the first (i.e. defines a range of 1 line only.)

Also see the "D(n)" and "DD" delete line ([prefix area](#)) commands.

### Parameters:

ALL All lines within the specified or implied range of lines are eligible for deletion.

*.name1* | *line1*  
A label name or line number identifying the first or only line in a range of text edit lines eligible for deletion. The preceding "." (dot) in *.name1* is mandatory.

*.name2* | *line2*  
A label name or line number identifying the last line in a range of text edit lines eligible for deletion. The preceding "." (dot) in *.name2* is mandatory.

If *.name2* or *line2* references a text edit line number lower than that referenced by *.name1* or *line1*, then the order is reversed to define a positive number of lines. If *.name2* and *line2* are omitted, only the first line in the range is eligible for deletion.

EX | X | NX  
EX or X indicate that only excluded text edit lines that fall within the explicit or implicit range of lines are to be deleted. NX indicates that only non-excluded text edit lines that fall within this range are to be deleted.

Default is to deleted both excluded and non-excluded lines.

BLOCK All text marked by a box or line block is deleted.

Before DELETE BLOCK can be used, text must first be marked within the display of the edit view using line ([prefix area](#)) commands ML or MB; or using <F17> or <F18> which are, by default, assigned to [MARK BOX](#) and [MARK LINE](#) respectively. An error message is displayed if no marked block exists.

### Examples:

del 5 Delete line number 5.

delete nx Delete all non-excluded lines.

del all x .fix .fixe Delete all excluded lines that exist within the range of lines starting at label name .FIX, ending at label name .FIXE.

del all Delete all lines.

### See Also:

[INPUT](#) (synonym [INSERT](#))

## XEDIT Interface

---

If [INTERFACE=ISPF](#) is in effect, the XEDIT version may be invoked by prefixing the command with [ECOMMAND](#). See [ISPF/XEDIT Primary Command Precedence](#).

### XEDIT Syntax:

```

+----- 1 -----+
|                   |
>>-- DElete  ----->>
|                   |
+- group-target -+

```

### Description:

Delete one or more lines from the edited file starting at the focus line.

### Parameters:

*group-target*

**Group-target** condition defining the end of the source target area. If the group-target is not satisfied then the DELETE command will fail.

### Example:

```
delete      Delete the focus line only.
del :10     Delete the focus line and all lines up to, but not including, line 10.
del /abc/   Delete the focus line and all lines up to, but not including, the first line to contain the string "abc".
```

---

## DESTROY BUTTON

---

### Syntax:

```
>>-- DEStroy BUTton  ---/name/---><
                    |         |
                    +--- * ---+
```

### Description:

Destroy a specified button or all buttons generated by a previous CREATE BUTTON command.

As for CREATE BUTTON, any supported delimiter character may be used to encompass the name string so long as the delimiter character used does not appear in the name string.

### Parameters:

*name*      The name of the button to be destroyed.

\*

(asterisk) indicates that all buttons are to be destroyed.

### Example:

```
destroy button /HW/
destroy button *
```

### See Also:

**CREATE BUTTON**

## DIALOG

### Syntax:

```
>>-- DIALOG -- /prompt/ -----+----->
      |
      +- EDITfield +-+-----+
          |         +- /text/ +-+ PASSWORD +-
          |         |         |
          +----- OK -----+ +- DEFButton 1 -+
          |         |         |
      >-----+-----+-----+----->
      |         |         |         |
      +- TITLE /title/ -+ +- OKCANCEL -+ +- DEFButton n -+
          |         |         |
          +----- YESNO -----+
          |         |
          +- YESNOCANCEL -+

      >-----+-----><
          |
          +- ICONExclamation -+
          |
          +- ICONInformation -+
          |
          +- ICONQuestion ----+
          |
          +- ICONStop -----+
          |
```

### Description:

For use in CBLc macros only, DIALOG opens a dialog window prompting the user to enter text and/or select an action.

The "/" (slash) character is normally used as the delimiter encompassing the character string for **prompt**, **text** and **title**. However, any non-alphanumeric character that does not have a special meaning to CBLc may be used as a string delimiter (e.g. "%", "#", "."). The delimiter character used must not appear in the string argument.

Following a DIALOG operation, the value entered and button selected by the user may be obtained as REXX compound variables using [EXTRACT DIALOG](#).

### Parameters:

*prompt*

Character string, within delimiters, to be displayed in the dialog window prompting the user for a response. The delimiters are not included as part of the character string.

EDITFIELD

Display an editable field in the dialog window allowing the use to type a response string.

*text*

Character string, within delimiters, to be displayed in the editable field when the dialog window is opened. The delimiters are not included as part of the character string.

PASSWORD

Hide the character string typed in the edit field so as not to display potentially sensitive data.

TITLE /title/

Display a title at the top of the dialog window. Character string, within delimiters, defining text to be displayed in the title field. The delimiters are not included as part of the character string.

OK  
OKCANCEL  
YESNO  
YESNOCANCEL

Defines the action buttons to be displayed at the bottom of the dialog window. OK is the default unless EDITFIELD is specified in which case OKCANCEL is default.

DEFBUTTON *n*

Default button number. The action buttons at the bottom of the dialog window are numbered from left to right. DEFBUTTON defines which of these buttons are default. Where EDITFIELD is not specified, the cursor is positioned on the default button.  
Default is button number 1.

ICONExclamation  
ICONInformation  
ICONQuestion  
ICONStop

Display one of the standard CBLc icons in the dialog window. These are "!" (exclamation mark), "i", "?" (question mark) or "STOP".

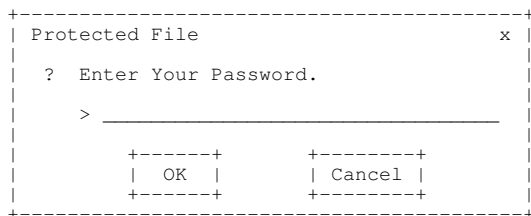
**Example:**

The following excerpt of CBL e REXX macro syntax opens a dialog window to prompt the user to enter a password before proceeding further:

```
'dialog /Enter Your Password./ editfield password',
'title /Protected File/ OKCANCEL iconq'

if dialog.2 = 'CANCEL' | dialog.1 <> 'open sesame' then
do; call AuthFailed; exit; end
```

The following dialog window will be displayed:



**See Also:**

[POPUP](#)

## DOWN

**Environments:**

DOWN primary command exists in the following application environments.

<a href="#">Text Edit (ISPF)</a>	Text Editor with INTERFACE=ISPF (default for z/OS).
<a href="#">Text Edit (XEDIT)</a>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
<a href="#">Data Edit</a>	SDE Data Editor.
<a href="#">System</a>	FileKit base windows system.

## ISPF Interface

If **INTERFACE=XEDIT** is in effect, the ISPF version may be invoked by prefixing the command with **ICOMMAND**. See [ISPF/XEDIT Primary Command Precedence](#).

**ISPF Syntax:**

```
>>- DOWN -----><
|
|--- Cursor -----|
|--- CSR -----|
|
|--- Data -----|
|
|--- Half -----|
|
|--- Max -----|
|
|--- Page -----|
|
|--- n_lines -----|
```

**Description:**

Scroll the view of the data within the text editor window down towards the bottom of the displayed text.

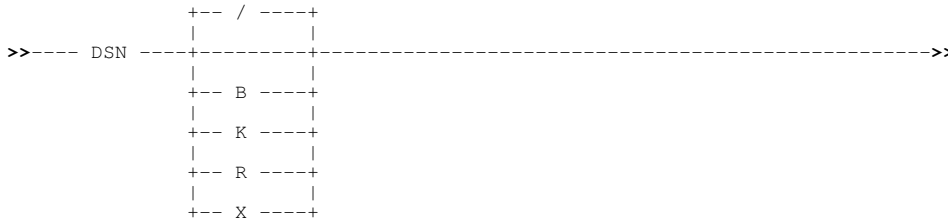
DOWN is assigned to function key **F8** by default. Note that any text entered at the command prompt when a PFKey is pressed will be treated as parameter input to the command associated with the PFKey (e.g. If MAX is at the command prompt when F8 is pressed, DOWN MAX will be executed.) Where no parameter is specified, the scroll amount will be the value specified in the **Scroll>** field in the top right corner of the window display.



UP

## DSN

### Syntax:



### Description:

DSN provides utility functions to be executed against the data set name on which the cursor is positioned within the CBLLe edit view window. This provides a point-and-shoot capability for data set names contained in edited files such as:

1. MVS JCL DD, IDCAMS DEFINE and TSO/CBLLe ALLOC statements.
2. SELCOPY DIR output files. (e.g. Output from DIR and SELCFILF macros.)
3. Any file where the cursor is positioned on a DSN.

If an invalid parameter is specified with a length greater than one, the SET DSN command will be invoked to set the DSN of the data being edited in the current edit view. If the cursor is not positioned within the edit view display area, no action is taken.

To execute, type DSN and optionally a valid parameter at the command prompt of the edit view, position the cursor on a DSN in the edited data and hit <Enter>. Alternatively, the DSN command may be assigned to a PFKey.

If the data set name in the edit view references CBLLe environment variables, JCL symbolic parameters or MVS system symbols, then they will be resolved by the DSN command before the requested action is taken. Note that, to resolve JCL symbolic parameters, the JCL SET statement must exist within the edited data.

### Parameters:

/ Display the DSN Cursor Fileid Utilities Menu. This allows the user to select the action to be taken against the data set name.

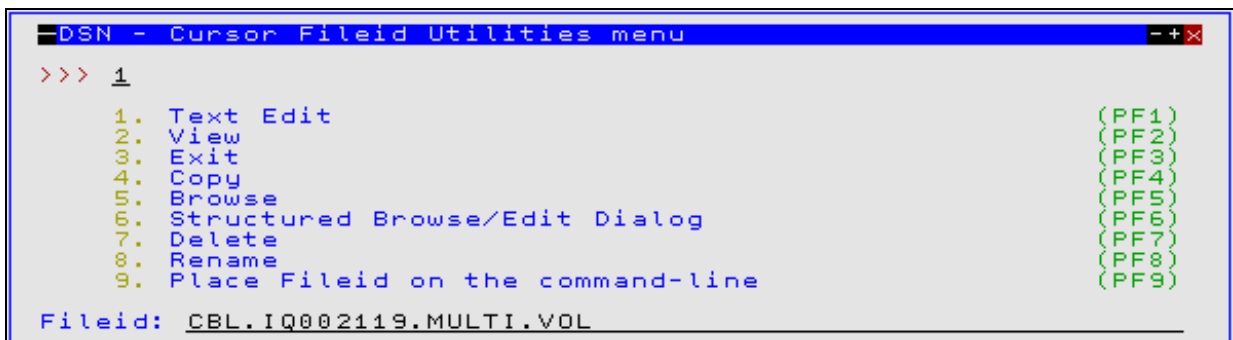


Figure 17. DSN Cursor Fileid Utilities Menu.

- B Open an SDE window view to browse the data set.
- K Generate an ERASE command for the data set name and place it on the command line. Note that the ERASE command is not executed immediately and the command verb at the command prompt will have the first character overtyped with "" to avoid unintentional execution of the ERASE by the user.
- R Generate a RENAME command for the data set name and place it on the command line. Note that the RENAME command is not executed immediately.
- X Open a CBLLe text edit window view to edit the data set.

## DUPLICATE

### Environments:

DUPLICATE primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor.

### Syntax:

```

>>-- DUPLICATE +--- 1 -----+ +----- 1 -----+
              |               | |               |
              +-----+-----+ +-----+-----+
              |               | |               |
              +--- ntimes ---+ +- group-target -+
    
```

### Description:

Duplicate lines in the file one or more times. Lines to be duplicated are defined by group-target.

Duplicated lines are inserted immediately following the defined group-target area. The first duplicated line becomes the focus line.

### Parameters:

- ntimes  
The number of times the line or group of lines are to be duplicated. Default is 1.
- group-target  
**Group-target** condition defining group of lines to be duplicated. If the group-target is not satisfied then the DUPLICATE command will fail.
- BLOCK is supported as a group-target for DUPLICATE for line blocks only.

### Examples:

- duplicate  
Duplicate the focus line once.
- dup 5  
Duplicate the focus line 5 times.
- dup 3 /SELC/  
Duplicate 3 times the focus line and lines up to, but not including, the first line to contain the string "SELC".
- dup 1 BLOCK  
Duplicate the marked line block once.

## ECOMMAND

### Syntax:

```

>>-- ECommand -- command -----><
    
```

### Description:

Where a command exists in both the XEDIT and ISPF edit command sets and INTERFACE=ISPF is active, ECOMMAND may be used as a prefix in order to force the use of the XEDIT version of the command.

If INTERFACE=XEDIT is active, the ECOMMAND prefix has no effect.

### Parameters:





Open a test edit view for a specific file or place focus on the next or previous **document** (child) view (e.g. text edit, data edit, list or panel view) within the Text Editor frame window. If, following execution of EDIT, the focus view is a text edit view, then that view becomes the **CBLe current window**.

If *fileid* is the same as that of a file which is already in the ring of edited files, then focus is placed on the edit view containing that file. Otherwise, a file with this fileid is read from disk and a new edit view is created to edit the data. If no matching file is found on disk, then a new, empty edit view is opened for this fileid.

Where a single token is specified for *fileid*, CBLe treats it as the FNAME portion of a fileid. The file edited will have a fileid with the specified FNAME with all other portions of the fileid (FPATH, FTYPE and FMODE) being equal to that of the file in the current edit view.

If no parameters are specified, default parameter "+" (plus) is used and the next view in the ring has focus. This is equivalent to the operation performed by the FileKit environment command, **MDINEXT**. If "-" (minus) is specified, the previous document view in the ring has focus. This is equivalent to the operation performed by the FileKit environment command, **MDIPREV**.

Where *fileid* is identified as being an HFS path (i.e. begins with "." or contains "/"), HFS options may be specified to determine the handling of data within the file.

For successful edit, the user must have read/write access to the requested file and the file file must not be locked (exclusively enqueued). If the file to be edited is already being used exclusively by another process or user, then a dialog window is opened which offers the user read-only edit (i.e. **VIEW**) of the file data instead.

The maximum number of files that may be edited is restricted by system resources only.

### Parameters:

-Q Same as **QUIET** option below, bypasses any popup dialogs that offer Data-Edit for VSAM or large datasets and warn that a load size threshold has been exceeded.

+ (plus) Place focus on the next document view within the Text Editor frame window. This is the default.

- (minus) Place focus on the previous document view within the Text Editor frame window.

*fileid* The fileid of the file to be edited and made the current file in the ring of edited files.

*fileid* The fileid of the file to be edited.

For MVS, *fileid* may be any of the following:

- ◇ The DSN of a physical sequential data set.
- ◇ The DSN of a VSAM (KSDS, ESDS, RRDS, VRDS or LDS) data set.
- ◇ The library DSN and parenthesised member name of a PDS or PDSE library member.
- ◇ The library DSN, parenthesised member name and absolute or relative number of a PDSE library member generation as described under **z/OS PDSE Library Member Generations**. (PDSE version 2 with MAXGENS only.)
- ◇ The name of a member to be edited from the same PDS or PDSE library as the member displayed in the **current text edit window** view.
- ◇ The name of an HFS/ZFS file system fileid.

For VSE, *fileid* is the member name of a LIBR library in lib.sublib.mn.mt format.

For CMS, *fileid* is a CMS fileid.

PROFILE *macroname* The REXX edit macro to be executed as the profile when editing the file.

The macro name must exist in a library within the CBLe macro path.

The PROFILE option only affects the profile for the file that is being added to the edit ring, and does not affect the profile to be used when additional files are added to the edit ring later in the edit session.

*macroname* overrides use of the default profile macro defined by the INI variable Edit.DefProfile and/or the CBLe command **SET DEFPROFILE** (default PROFILE.)  
Default is macro name, PROFILE.

NOPROFILE Suppress the use of a profile macro when editing the file.

The NOPROFILE option only affects the file currently being added to the ring, and does not affect the profile to be used when additional files are added to the ring later in the edit session.

QUIET Same as **-Q** option above, bypasses any popup dialogs that offer Data-Edit for VSAM or large datasets and warn that a load size threshold has been exceeded.

EOL=STD|NL|CR|LF|CRLF|LFCR|CRNL|*string*

Sets the **EOLIN** (input end-of-line) delimiter value used to determine the end of each record for non-RECFM F input. EOLIN delimiters are not included in the edited record data or record length. EOL parameter elements are as follow:

<b>STD</b>	-	Any standard line-end.
<b>NL</b>	X'15'	New Line.
<b>CR</b>	X'0D'	Carriage Return.
<b>LF</b>	X'0A'	Line Feed.
<b><i>string</i></b>	-	A 2-byte user specified character or hex string.

STD is default so that the EDIT operation scans the input data for any of the standard EOL combinations (not *string*), stopping when one is found. This EOL combination is used as EOLIN for the file.

RECFM F

Specifies that the data is to be treated as containing Fixed length format records as defined by the LRECL argument.

LRECL *lrecl*

Specifies the maximum record length of input records.

Records terminated by an EOL sequence will wrap onto the next line of data if the record length exceeds *lrecl*. Where a record has wrapped, the prefix area contains the "=="EOL>" flag. Furthermore, read-only edit is forced in order to suppress save of a wrapped record as multiple, individual records.

For RECFM F data, *lrecl* is the fixed length of the records in the edit view. If the file size is not a multiple of the *lrecl* value an error occurs and edit is cancelled.

Note: Use VIEW to display data as fixed format with the last record padded with blanks up to the *lrecl* length.

For EOL delimited records, default *lrecl* is 32752. Otherwise, for RECFM F, default *lrecl* is 80.

**Example:**

```
edit          Make the next file in the ring the current file.

edit-        Make the previous file in the ring the current file.

edit cbl.cmd(temp)
             Edit member "TEMP" of MVS PDS "CBL.CMD".

edit cbl.jcl(ssfind) (profile jclprof)
             Edit member "SSFIND" of MVS PDS "CBL.JCL" and override the default profile macro with macro "JCLPROF".
```

**See Also:**

SET/QUERY/EXTRACT Option: **FILEID**



GET retrieves global edit variables.  
GETF retrieves file edit variables.

DEL  
DELF

DEL/DELF is used in an edit macro or executed from a command line to delete one or more named variables.

DEL deletes global edit variables.  
GETF deletes file edit variables.

LIST  
LISTF

LIST/LISTF is used to list the specified edit variable names and their values to the CBL message lines.  
Default is all applicable variable names.

LIST will list global edit variables.  
LISTF will list file edit variables.

*varname*

Edit variable name.

*value*

Edit variable string value.  
Must be a single token for SET/SETF.

### Examples:

```
editv set tmp nbj.tmp work nbj.work.txt user guest
editv setl docs All of this is assigned to "docs"
editv list
```

### See Also:

[EQU](#)

---

## EMSG

---

### Syntax:

```
>>-- EMSG ---+-----+-----><
           |           |
           +-- string --+
```

### Description:

Display a message string on the message line as an error message. EMSG is most often used in CBL macros.

If BEEP is ON then EMSG will trigger the 3270 terminal alarm.

If no text string is specified, then the message line is cleared.

### Parameters:

*string*

Text string to be displayed on the message line.

### Example:

```
emsg 'Error.. Timestamp mismatch'
```

### See Also:

[MSG](#)

---



An environment variable is set to NULL if equals (=) is specified without *value*. A NULL environment variable will be substituted with nothing, as though the variable was not specified within the command string. To reset a variable "MyVar", so that no substitution occurs when it is enclosed within the ENVARS delimiter character, the following command should be used:

```
VIGNORE EQU MyVar %MyVar%
```

Percent symbol (%) is the default ENVARS delimiter character and prefixing the EQU command with **VIgnore** ensures that %MyVar% is not itself substituted when the EQU command is executed.

Since EQU is itself a command, it too may contain environment variable specifications. e.g. The following assigns environment variable HLQ to a *value* that includes standard environment variables for username and date. HLQ is then used to specify the high level qualifiers of a library in an EDIT command for library member, JOBCARD.

```
EQU hlq %user%.CBLINST.D%yy%mm%dd%
EDIT %hlq%.INIT.JCL(JOBCARD)
```

**Parameters:**

- name* The name of a new or existing user environment variable to be displayed or assigned to *value*.
- value* The string value to be assigned to the user environment variable *name*. A preceding equals symbol (=) is optional. If *value* is omitted in order to assign a value of NULL, then a preceding equals symbol (=) is mandatory. Leading and trailing blanks are always stripped from the *value* string. However, *value* may comprise a number of blank delimited words, in which case all intermediate blanks are preserved.

**See Also:**

[EDITV](#)

## EXCLUDE, X

**Environments:**

EXCLUDE primary command exists in the following application environments.

<a href="#">Text Edit</a>	Text Editor. (Interface ISPF and XEDIT)
<a href="#">Data Edit</a>	SDE Data Editor.

**Syntax:**

```
>>+-+ EXclude +-+ (1) string ----->
  |          |
  +- X -----+
                                     +- NEXT ---+ +- CHARs ---+
                                     |           |           |
                                     +- ALL ----+ +- PREFIX -+
                                     |           |           |
                                     +- FIRST --+ +- SUFFIX -+
                                     |           |           |
                                     +- LAST ---+ +- WORD ---+
                                     |           |           |
                                     +- PREV ---+
                                     |           |           |
                                     +- .ZFIRST -- .ZLAST --+
                                     |           |           |
>-----+-----+-----+-----<<
  |          |          |          |
  +- pos1 ---+-----+ +- .name1 ---+-----+
  |          |          |          |
  +- pos2 ---+          +- .name2 ---+
  |          |          |          |
```

**Notes:**

- Operands may be entered in any order.

**Description:**

Exclude lines that contain a match for a specified search string.

Lines that are already excluded are not included in the search and will remain excluded following execution of EXCLUDE. Therefore, execution of successive EXCLUDE commands has a cumulative effect.

EXCLUDE employs the same string search and field highlighting methods as the **FIND** command. Unless EXCLUDE ALL is executed, all occurrences of the search string that are not excluded by the EXCLUDE command, are highlighted in the text. **RESET FIND** may be used to reset search string highlighting and **RESET EXCLUDED** will redisplay excluded lines.

If the matching text falls outside the currently displayed page of text, the display is scrolled so that the excluded line becomes the **current line** of the display.

**RFIND**, which is assigned to function key **F5** by default, may be used to repeat an EXCLUDE command if it was the last EXCLUDE or FIND command to be executed.

The prevailing **BOUNDS** left and right column values define the area of the record within which the search occurs. i.e. the matched data must begin at or after the left bound and not exceed the right bound.

The BOUNDS columns may be overridden using *pos1* and *pos2* positional parameters.

**Parameters:**

*string*

The EXCLUDE search string. The search string may be any of the following:

- ◇ An unquoted character string containing no commas or blanks. The search for the character string will be case-insensitive so that uppercase and lowercase characters are treated as being the same. This type of search must not match an EXCLUDE parameter keyword.

- ◇ A character string enclosed in apostrophes (') or quotation marks ("). The search string may contain embedded commas and blanks and the character string will be case-insensitive.

Two adjacent quotation mark characters that are embedded in a search string which is enclosed by the same quotation mark characters, will be treated as a single occurrence of the character. e.g.

```
EXCLUDE "He said ""Go home!"""
```

Exclude the next line containing the string 'He said "Go home!"'.

- ◇ A character string enclosed in apostrophes (') or quotation marks (") with the prefix C. This is equivalent to specifying a quoted search string but that the string search will be case-sensitive. (e.g. C'Book')

- ◇ A hexadecimal string enclosed in apostrophes (') or quotation marks (") with the prefix X.

- ◇ A picture string enclosed in apostrophes (') or quotation marks (") with the prefix P.

Picture strings use special characters to represent a generic group of characters as described below. Any character in a picture string that is not one of these special characters is untranslated.

String	Description
P'_'	Any character.
P'~'	Any non-blank character.
P'.'	Any non-displayable character.
P'#'	Any numeric character, 0-9.
P'.'	Any non-numeric character.
P'@'	Any uppercase or lowercase alpha character.
P'<'	Any lowercase alpha character.
P'>'	Any uppercase alpha character.
P'\$'	Any non-alphanumeric special character.

- ◇ A regular expression string enclosed in apostrophes (') or quotation marks (") with the prefix R. For example:

```
R'C[0-9]^2'
```

is a regular expression which would match the upper case character **C** followed by 2 digits. This expression would match **C00 C91 C22** etc. but not **c99 C 99** etc.

Regular expressions enable powerful string pattern matching at the cost of rather complex syntax and potentially extended command processing time. For syntax and usage see **Regular Expressions** in the text editor documentation.

ALL Search forwards from the Top of File indicator and exclude all records containing an occurrence of the search string.

FIRST Search forwards from the Top of File indicator to exclude the first occurrence of the string.

LAST Search backwards from the End of File indicator to exclude the last occurrence of the string.

NEXT



Search forwards from the current cursor location to exclude the next occurrence of the string. If the cursor is not within the window's data display area, the search begins at the **current line** of the display.

**PREV** Search backwards from the current cursor location to exclude the previous occurrence of the string. If the cursor is not within the window's data display area, the backwards search begins at the current line of the display.

**CHARS** CHARS indicates that a successful match occurs if the search string is found anywhere within the text being searched.

**PREFIX** PREFIX indicates that a successful match only occurs if the search string is found at the start of a word within the text being searched. i.e. the matched text must precede an alphanumeric character and either be preceded by a non-alphanumeric character or be at the start of a text line.

**SUFFIX** SUFFIX indicates that a successful match only occurs if the search string is found at the end of a word within the data being searched. i.e. the matched text must be preceded by an alphanumeric character and either precede a non-alphanumeric character or be at the end of a text line.

**WORD** WORD indicates that a successful match only occurs if the search string is found to be a complete word within the data being searched. i.e. the matched text must either be preceded by a non-alphanumeric character or be at the start of a text line, and either precede a non-alphanumeric character or be at the end of a text line.

*pos1* The first position of a range of positions within the lines of text to be searched.

*pos1* must be a positive integer value (not zero) and must be a value that is less than or equal to the maximum length of the data records.

*pos2* The last position of a range of positions within the lines of text to be searched.

Like *pos1*, *pos2* must be a positive integer value. If *pos1* references a position which is higher than that referenced by *pos2*, then the *pos1* and *pos2* values are swapped.

If *pos2* is greater than the maximum record length then *pos2* is set equal to the maximum record length.

Default is *pos1* plus the length of the search string minus 1.

*.name1* A label name identifying the first record of a range of data records to be searched. The preceding "." (dot) is mandatory. Default is .ZFIRST.

*.name2* A label name identifying the last record of a range of data records to be searched. The preceding "." (dot) is mandatory. If *.name2* occurs before *.name1* in the display, then the order is reversed. If EXCLUDE PREV is executed and *.name1* is specified, the default is .ZFIRST. Otherwise the default is .ZLAST.

**Examples:**

`exclude faithful`  
Exclude a line containing the next occurrence of the string 'faithful' in any character case mixture.

`exclude all c'AND' word 5 30`  
Exclude all lines containing the uppercase string 'SELCOPY' as a complete word between text positions 5 and 30.

**See Also:**

**CHANGE FIND FLIP RFIND**

**EXTRACT**

**Environments:**

EXTRACT primary command exists in the following application environments.

<b>Text Edit</b>	Text Editor (both XEDIT and ISPF interfaces.)
<b>Data Edit</b>	SDE Data Editor.

**Description:**

See [SET/QUERY/EXTRACT Options](#).

## FILE, FFILE

### Environments:

FILE primary command exists in the following application environments.

<b>Text Edit</b>	Text Editor (both XEDIT and ISPF interfaces.)
<b>Data Edit</b>	SDE Data Editor.

### Syntax:

```
>>-- FILE -----><
      |           |           |           |
      |  fileid  |  +- (  --- NEWGEN  +-+
      |           |           |           |
      |           |           |  +- NOGEN  --+
      |           |           |           |
>>-- FFile -----><
      |           |           |           |
      |  fileid  |  +- (  --- NEWGEN  +-+
      |           |           |           |
      |           |           |  +- NOGEN  --+
      |           |           |           |
```

### Description:

Save the current file to disk with an associated fileid and, if successful, exit the edit view.

If a *fileid* is specified, the data is saved under the new fileid. File data that exists on disk under the original fileid is unchanged. If *fileid* is a member belonging to a PDSE library which supports member generations, the option NOGEN is redundant as a new member generation will always be created.

If *fileid* is not specified, FILE attempts to write the file to disk using the currently assigned fileid. By default, this fileid is that which was used to initially edit the file, unless subsequently updated by a SET FILEID, FNAME, FTYPE, FMODE, FPATH command.

If the fileid to be used by FILE does not already exist, a new file will be created. For MVS Sequential, PDS(E) and VSAM data sets, this will open the Allocate NonVSAM or Define VSAM dialog window prompting the user to provide the required new data set characteristics.

The file save will fail and return an error if either of the following conditions are true:

- The fileid used to save the data differs from that used on the initial edit of the file and this fileid already exists for a file on disk.
- For HFS files, the current "Modified" timestamp of the file is later than the time at which the file was last saved, or else first edited, in the current CBL session. i.e. the file's data has been changed by some other process or user edit.

FFILE will save the file regardless of the above error conditions.

Where *fileid* is identified as being an HFS path (i.e. begins with "." or contains "/"), DSORG is set to be HFS and the file is saved with the permission mode 740 and the following record delimitation:

- If the current RECFM is F, the HFS file will contain no EOL delimiters and each record will be padded to the current LRECL value.
- For RECFM V or U, HFS file records will be delimited with EOL (end-of-line) characters as defined by the current setting of EOLOUT.

### Parameters:

*fileid*

The fileid to be assigned to the file on writing it to disk.

For MVS data sets, *fileid* may be the DSN of a Sequential or VSAM data set, the DSN and member name of a PDS(E) library member or an HFS absolute or relative path name.

For VSE files, *fileid* may be the full LIBR member id, lib.sublib.mname.mtype.

For CMS files, *fileid* may be the full CMS file id, FNAME FTYPE FMODE (or FNAME.FTYPE.FMODE) If only two qualifiers are specified, FMODE defaults to the current FMODE, and if only one qualifier is specified, then FTYPE and FMODE

default to the current FTYPE and FMODE.

**NEWGEN**

Applicable only if *fileid* is not specified and output is to a version 2 PDSE library member that supports member generations. Data will be saved to a new primary member generation. This option overrides the default set by the **GENSAVE** option.

**NOGEN**

Applicable only if *fileid* is not specified and output is to a version 2 PDSE library member that supports member generations. Data will be saved to back to the same member generation referenced in the focus edit view. This option overrides the default set by the **GENSAVE** option.

**See Also:**

**SAVE QUIT** and SET/QUERY/EXTRACT Options: **FILEID GENSAVE**

---

## FILEKIT

---

**Syntax:**

```
>>--+ FILEKIT  --+-----  command  ----->><
      |         |
      +- CBLI  -----+
```

**Description:**

Execute a FileKit environment command.

The CBL edit command processor is bypassed and the specified command is passed directly to the FileKit environment.

**Parameters:**

*command*

Command syntax to be executed outside the CBL text edit environment.

**Example:**

```
FILEKIT QUIT
```

Execute the FileKit environment QUIT command to end FileKit, as opposed to the CBL text edit QUIT command to close the current child window view.

**FILEKIT QUIT** is equivalent to executing **FILEKITCANCEL**.

**See Also:**

**SYSCOMMAND**

---

## FILLBOX

---

**Syntax:**

```
>>-- FILLbox --+-----+----->><
              |         |
              +---- x ----+
              |         |
              +-- string --+
```

**Description:**

Fill a marked block with the specified single character or insert the specified text string in every line of the marked block beginning at the left column of the block.

The original contents of the block are overwritten.

Where the block is a line block, only positions between the left zone column and the right zone column are filled.

**Parameters:**

`x` A single fill character for the block. All positions in the block will contain this character. If no character is specified, the block is filled with blanks.

`string` Text string to be placed in the left column of every line in the block.

Where more than 1 blank separates the text string from the command verb, the additional separating blanks are treated as part of the text string.

The specified string is truncated or padded with blanks accordingly, in order to fit the marked block.

**Example:**

`fill` Fill a marked block with blanks.

`fill A` Fill a marked block with character "A".

`fill ABC` Fill each line of a block with "ABC" in the first marked column and pad with blanks.

`fill XY Z` Fill each line of a block with " XY Z" in the first marked column and pad with blanks.

## FILLDIALOG

**Syntax:**

>>-- FILLDialog -----<<

**Description:**

Open the [CBLLe Fill Dialog Window](#) to perform a [FILLBOX](#) command.

This dialog window may also be opened by selecting **Fill** from the **Actions** menu item in the [CBLLe Main Menu Bar](#).

**See Also:**

[CBLLe Fill Dialog Window](#)

## FIND, FINDUP

**Environments:**

FIND primary command exists in the following application environments.

<a href="#">Text Edit (ISPF)</a>	Text Editor with INTERFACE=ISPF (default for z/OS).
<a href="#">Text Edit (XEDIT)</a>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
<a href="#">Data Edit</a>	SDE Data Editor.
<a href="#">Lists</a>	List display windows.
<a href="#">FSU &amp; FCOPY</a>	The "Select Input Members" sub-panel of the File Search/Copy/Update/Remap and File Copy utility panels.

### ISPF Interface

If [INTERFACE=XEDIT](#) is in effect, the ISPF version may be invoked by prefixing the command with [ICOMMAND](#). See [ISPF/XEDIT Primary Command Precedence](#).



P'_'	Any non-blank character.
P'.'	Any non-displayable character.
P'#'	Any numeric character, 0-9.
P'.'	Any non-numeric character.
P'@'	Any uppercase or lowercase alpha character.
P'<'	Any lowercase alpha character.
P'>'	Any uppercase alpha character.
P'\$'	Any non-alphanumeric special character.

◇ A regular expression string enclosed in apostrophes (') or quotation marks (") with the prefix R. For example:

```
R'C[0-9]^2'
```

is a regular expression which would match the upper case character **C** followed by 2 digits. This expression would find **C00 C91 C22** etc. but not **c99 C 99** etc.

Regular expressions enable powerful string pattern matching at the cost of rather complex syntax and potentially extended command processing time. For syntax and usage see [Regular Expressions](#) in the text editor documentation.

ALL

If none of the records to be scanned are EXCLUDED or NX is specified, then FIND ALL is the same as FIND FIRST except that a message is displayed providing the number of matches found for the search string. Unlike FIND FIRST, **all** excluded records that contain an occurrence of the string are made visible if NX is not specified.

FIRST

Search forwards from the top of the file data (i.e. the first position of the first data record) to find the first occurrence of the string.

LAST

Search backwards from the bottom of the file data (i.e. the last position of the last data record) to find the last occurrence of the string.

NEXT

Search forwards from the current cursor location to find the next occurrence of the string. If the cursor is not within the window's data display area, the search begins at the first position of the first visible or excluded record within the display area.

PREV

Search backwards from the current cursor location to find the previous occurrence of the string. If the cursor is not within the window's data display area, the backwards search begins at the first position of the first visible or excluded record within the display area.

CHARS

CHARS indicates that a successful match occurs if the search string is found anywhere within the text being searched.

PREFIX

PREFIX indicates that a successful match only occurs if the search string is found at the start of a word within the text being searched. i.e. the matched text must precede an alphanumeric character and either be preceded by a non-alphanumeric character or be at the start of a text line.

SUFFIX

SUFFIX indicates that a successful match only occurs if the search string is found at the end of a word within the data being searched. i.e. the matched text must be preceded by an alphanumeric character and either precede a non-alphanumeric character or be at the end of a text line.

WORD

WORD indicates that a successful match only occurs if the search string is found to be a complete word within the data being searched. i.e. the matched text must either be preceded by a non-alphanumeric character or be at the start of a text line, and either precede a non-alphanumeric character or be at the end of a text line.

EX  
X

Search EXCLUDED data records only.

NX

Search only visible data records (i.e. not EXCLUDED).

*pos1*

The first position of a range of positions within the lines of text to be searched.

*pos1* must be a positive integer value (not zero) and must be a value that is less than or equal to the maximum length of the data records.

*pos2*

The last position of a range of positions within the lines of text to be searched.

Like *pos1*, *pos2* must be a positive integer value. If *pos1* references a position which is higher than that referenced by *pos2*, then the *pos1* and *pos2* values are swapped.

If *pos2* is greater than the maximum record length then *pos2* is set equal to the maximum record length.

Default is *pos1* plus the length of the search string minus 1.

*.name1*

A label name identifying the first record of a range of data records to be searched. The preceding "." (dot) is mandatory. Default is .ZFIRST.

*.name2*

A label name identifying the last record of a range of data records to be searched. The preceding "." (dot) is mandatory. If *.name2* occurs before *.name1* in the display, then the order is reversed. If FIND PREV is executed and *.name1* is specified, the default is .ZFIRST. Otherwise the default is .ZLAST.

**Examples:**

`find sincere`  
Find the next occurrence of the string 'sincere' in any character case mixture.

`find c'DISP=(' prefix last 16 71`  
Find the last occurrence of uppercase 'DISP=(' between positions 16 and 71 and at the start of a word.

**See Also:**

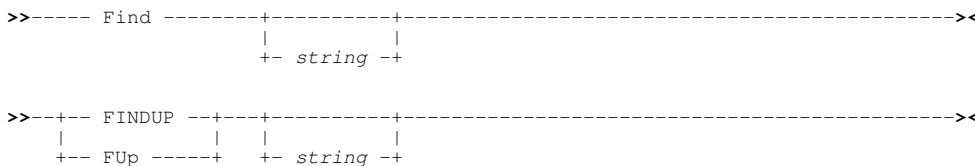
**CHANGE EXCLUDE RFIND**

**XEDIT Interface**

---

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

**XEDIT Syntax:**



**Description:**

Find the first line following (FIND) or previous to (FINDUP) the focus line that contains the specified string in column 1 and make this line the new focus line.

Where more than 1 blank separates the text string from the command verb, the additional separating blanks are treated as being part of the text string.

If WRAP is set ON and string is not found before bottom-of-file or top-of-file is encountered, then the scan continues from the opposite extreme of the file up to and including the focus line.

**Parameters:**

*string*  
Search text string.  
No delimiting characters should be specified.

Blanks in the search text are each treated as a wildcard representing a single character in the file data. In order to find a blank space, the "\_" (underscore) character should be specified in the corresponding position of the find string.

If string is not specified, then the string argument specified on the last FIND/FINDUP command executed, is used. If no previous FIND/FINDUP command has been executed in this instance of CBL, then the following message is displayed:

```
ZZSE060E No remembered operand for FIND command.
```

**Example:**

`find Hello`  
Focus line is set at the first line below the focus line to contain the string "Hello" starting in column 1.

`f ABC DEF`  
Focus line is set at the first line below the focus line to contain any character in column 1, the string "ABC" starting in column 2, any two characters in columns 5 and 6, and the string "DEF" starting in column 7.





ALL EXCLUDE HIDE LESS MORE

## FORWARD

### Environments:

FORWARD primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Lists	List display windows.
Help	HTML format Help windows.

### Syntax:

```
>>-- FOrward -----<<
```

### Description:

The FORWARD command scrolls the focus window forwards 1 page towards the bottom of the file.

### See Also:

[BACKWARD](#)

## FREE

### Syntax:

```
>>-- FREE -- freeparms -----<<
```

### Description:

The FREE command may be used to:

1. Dynamically unallocate a single data set.
2. Override the disposition of allocated data sets.
3. Override the output class.

FREE allows users to unallocate files whether or not a TSO environment is available. The syntax of the command closely matches that of the TSO FREE command and so most FREE commands, executed without TSO as a prefix, will give the same results.

FREE is identical to the [ALLOCATE](#) -FREE command and is supported for MVS only.

### Parameters:

*freeparms*

Parameters supported by the TSO FREE command as follow:

CATALOG	KEEP
DAtaset( <i>dsn</i> ) DSName( <i>dsn</i> )	SPIN(UNALLOC)
DDname( <i>ddname</i> ) File( <i>ddname</i> )	SYSOUT[ <i>(class)</i> ]
DELETE	UNCATALOG

### Examples:

```
free f(indd)
    Unallocate an existing data set.
```

```
free f(ixnbj) keep
    Unallocate a ddname overriding the disposition with KEEP.
```

**See Also:**

**ALLOCATE**

# GEN

**Syntax:**

```

      (1)      (2)      (3)
      +- *-1 -----+ +- View -----+
      |              | |              |
>>-- GEN -----+-----+----->
      |              | |              |
      +- 0 -----+ +- Browse ---+
      |              | |              |
      +- -nnn ---+ +- Compare---+
      |              | |              |
      +- *-nnn ---+ +- Edit      |
      |              | |              |
      |              | +- List ----+
      |              | |              |
      |              | +- Orphan ---+
      |              | |              |
      |              | +- Recover -+
      |              | |              |
      |              | +- Sde -----+
      |              | |              |
      |              | +- Upd -----+
      |              | |              |
      |              | +- View -----+
      |              | |              |
      +- ? -----+
    
```

**Notes:**

1. Parameters may be entered in any order.
2. Default for RECOVER operation is the currently edited/browsed generation.
3. Default operation is VIEW if current session is Text-Edit and BROWSE if current session is Data-Edit.

**Description:**

For use only when the current Text Editor or Data Editor view contains data belonging to a member of a PDSE version 2 library for which member generations is supported, or a previous generation of a member.

GEN will perform utility operations based on the library DSN, member name or generation number of the member generation in the current Text Editor or Data Editor view.

Parameters may be supplied in any order, but if none are supplied then the default is to open the immediately earlier generation. e.g. if currently browsing the 0 (zero) generation then the -1 generation will be opened.

**Parameters:**

*nnn* | *-nnn* | *\*-nnn*  
 Identifies a generation of the member displayed in the current Text or Data Editor view.

This is the generation against which the requested BROWSE, COMPARE, EDIT, RECOVER, SDE or UPD operation will be performed. This value is ignored for operations LIST and ORPHAN.

The generation may be identified using its absolute generation number (*nnn*), relative generation number (*-nnn*) or its number relative to the generation in the current view (*\*-nnn*). For example, if currently browsing the **-3** generation of a member then **"\*-2"** will refer to the **-5** generation.

The default for this parameter is **"\*-1"**.

**?**  
 Displays this help document.

**BROWSE**  
 A new Data Editor browse view will be opened for the generation specified.

For example, if currently editing generation **0** of member **"MY.PDSE2.LIB(ABC)"** in the Text Editor, then **GEN B** will browse generation **-1** of the same member.

This is equivalent to typing:

```
BROWSE MY.PDSE2.LIB(ABC.-1)
```

## COMPARE

The currently browsed/edited member generation will be compared with the generation specified.

For example, if currently editing generation **-2** of member **"MY.PDSE2.LIB(ABC)"** then **GEN -3 C** (or just **GEN C**) will compare generation **-2** with generation **-3**.

This is equivalent to typing:

```
COMPFILE MY.PDSE2.LIB(ABC.-2) MY.PDSE2.LIB(ABC.-3) CONTEXT 10
```

It is also equivalent to the **GENCOMP** command with no parameters.

## EDIT

A new edit view will be opened to edit the generation specified. The new edit session will be opened using the Text Editor or Data Editor depending on the current session.

For example, if currently browsing generation **-4** of member **"MY.PDSE2.LIB(ABC)"** in the Data Editor, then **GEN 0 E** will edit the (current) version of the same member in a Data Editor view.

This is equivalent to typing:

```
EDIT MY.PDSE2.LIB(ABC.0)
```

Beware that if you are already editing any generation of a member then an **enqueue violation** will occur and a popup will be displayed. e.g.

```
+-----+
| Enqueue Failed                                     x |
| ? File MY.PDSE2.LIB(ABC) is being edited by user |
| MYUSER1.                                         |
| Do you want to edit in read-only mode?          |
| Yes                                             No  |
+-----+
```

This is a system restriction since an exclusive ENQ cannot be obtained for more than one generation of the same member. The conflict may be avoided by typing **GO Browse** or **GO View** prior to issuing the **GEN E** command.

## LIST | \*

All generations of the currently browsed/edited member will be listed.

From the list-window line-commands may be entered against one or more member generation(s).

For example, enter **"D"** to delete, **"B"** to browse or **"RC"** to recover a generation.

## ORPHAN

Lists all orphaned members in the library to which the currently browsed/edited member belongs. An orphaned generation is one where the member base generation (generation 0) has been deleted or renamed.

For example, if currently editing member **"MY.PDSE2.LIB(ABC)"** in a Text Editor view, then **GEN ORPH** will list all orphaned members of **"MY.PDSE2.LIB"**.

This is equivalent to typing:

```
GENORPH MY.PDSE2.LIB
```

See also the **"List/Delete PDSE v2 Orphaned Member Generations"** tool which is available from the **"Utilities"** menu or by executing command **GENORPH** with no parameters.

## RECOVER

Recover a previous generation so that it becomes the prime member copy.

For example, if currently browsing generation **-4** of member **"MY.PDSE2.LIB(ABC)"** in a Data Editor view, then **GEN R** will recover relative generation **-4** as it as the prime (0) generation.

This is equivalent to typing:

```
RECOVER MY.PDSE2.LIB(ABC.-4)
```

Unlike other operations, for **RECOVER** the default when the generation number is not explicitly supplied, is the generation being browsed/edited (not the **\*-1** previous generation).

An error is given in this case if this results in you attempting to recover a member which is already the prime copy.

## SDE

A new Data Editor edit view will be opened for the generation specified. The session will allow insert and delete of records as well as modification of existing data. You may use the **UPD** option instead if record insert and delete are not required.

For example, if currently browsing generation **-4** of member **"MY.PDSE2.LIB(ABC)"** then **GEN S** will open a Data Editor view to perform "Full-Edit" of the **-5** generation of the same member.

This is equivalent to typing:

```
SD EDIT MY.PDSE2.LIB(ABC.-5)
```

Beware that if you are already editing any generation of a member then an **enqueue violation** will occur and a popup will be displayed as described for the **EDIT** operation.

UPD

A new Data Editor edit view will be opened for the generation specified. The session will only allow modification of existing data. You must use the **SDE** option instead if record insert and delete are required.

For example, if currently browsing generation **-4** of member **"MY.PDSE2.LIB(ABC)"** then **GEN U** will open a Data Editor view to perform "Update-in-Place" of the **-5** generation of the same member.

This is equivalent to typing:

```
SD EDIT MY.PDSE2.LIB(ABC.-5) UPDATE
```

Beware that if you are already editing any generation of a member then an **enqueue violation** will occur and a popup will be displayed as described for the **EDIT** operation.

## GENCOMP

### Syntax:

```
>>--+ GENComp  +--+-----+-----+-----+-----+-----+-----+-----+-----+
      |         |         |         |         |         |         |         |         |
      +- COMPGen -+ +- 0 -----+ +- compare-options -+
      |         |         |         |         |         |         |         |         |
      +- -nnn  --+
      |         |         |         |         |         |         |         |         |
      +- *-nnn --+
```

### Description:

For use only when the current Text Editor or Data Editor view contains data belonging to a member of a PDSE version 2 library for which member generations is supported, or a previous generation of a member.

**GENCOMP** may be used to compare the currently displayed member generation with another (earlier or later) generation of the same member.

If no parameters are supplied, then the default is to compare with text belonging to the member generation in the focus view with that belonging to the previous generation. For example, if currently browsing the prime (0) member generation, then it will be compared with the -1 generation.

**GENCOMP** generates and executes a call to the **COMPFILE** utility which in turn will detect differences between the two versions and display a readable report highlighting the changed data.

### Parameters:

0 | -nnn | \*-nnn  
 Identifies the generation of the member displayed in the current Text or Data Editor view, against which the compare operation will take place.

The generation may be identified using its absolute generation number (*nnn*), relative generation number (*-nnn*) or its number relative to the generation in the current view (*\*-nnn*). For example, if currently browsing the **-3** generation of a member then **"\*-2"** will refer to the **-5** generation.

The default for this parameter is **"\*-1"**.

#### *compare-options*

These are options that may be passed on the **COMPFILE** command that **GENCOMP** generates and executes.

The default options are **"CONTEXT 10"** which means that, around each group of one or more inserted/deleted record(s), up to ten matching records will be displayed in the output report, providing potentially useful context.

An example of when this default is usefully overridden might be where you anticipate the only differences between the two versions of the member will be changes to existing records (no inserts and/or deletes), and you would like to see individual byte changes highlighted in the comparison report.

For example, if dates/times have been refreshed then type:

```
GENC *-1 1TO1
```





# GO

## Environments:

GO primary command exists in the following application environments.

<b>Text Edit</b>	Text Editor (both XEDIT and ISPF interfaces.)
<b>Data Edit</b>	SDE Data Editor.

## Syntax:

```
>>-- GO -----+-- Browse ----+-----><
      |         |         |         |
      +-- Edit  +-----+
      |         |         |         |
      +-- SE   +-----+
      |         |         |         |
      +-- SU   +-----+
      |         |         |         |
      +-- View +-----+
```

## Description:

GO closes the current view of the file data and redisplay it in either a text edit or data edit view using the specified edit/browse type.

If specified with no parameter, the **Text/Data Edit GO Options** panel is displayed.

If the current view contains unsaved alterations to the text, the GO operation will stop and an error message displayed.

## Parameters:

- BROWSE Browse the text in a SDE data edit window view.
- EDIT Display the text in a CBLLe text edit window view for edit.
- SE Display the text in a SDE data edit window view with full edit capabilities.
- SU Display the text in a SDE data edit window view with update-in-place edit capabilities.
- VIEW Display the text in a CBLLe text edit window view for read-only edit (view).

# HELP

## Environments:

The HELP primary command exists in the following application environments.

<b>Text Edit</b>	Text Editor (both XEDIT and ISPF interfaces.)
<b>Data Edit</b>	SDE Data Editor.
<b>System</b>	FileKit base windows system.

## Syntax:

```
>>-- Help -----+-----><
      |         |         |
      +-- topic +-----+
```

## Description:

Use the HELP command to open the Help Window and optionally link directly to help on a specific help topic.

Where *topic* is not specified or not found, the relevant table of contents is displayed.

The Help window may also be opened via the Help item of the window's menu bar.

**Parameters:**

*topic* Display help on a specific FileKit topic.

If *topic* is enclosed in single or double quotes, the string is treated as the fileid of an HTML data set to be browsed. This may be the fully qualified fileid of an HTML document or the name of a PDS member that exists in the default HELP library.

This allows use of the FileKit help browser to display any cataloged HTML document.

If the help topic is not found, the **Help Topic Index List Window** is opened using the given topic as a search string.

**Example:**

help fill  
Display the help page for the FILL command.

H "OEM.CBL.HTML(TEST)"  
Open a specific HTML document library member.

## HEX

**Environments:**

HEX primary command exists in the following application environments.

<b>Text Edit (ISPF)</b>	Text Editor with INTERFACE=ISPF (default for z/OS).
<b>Text Edit (XEDIT)</b>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
<b>Data Edit</b>	SDE Data Editor.

**Syntax:**

```
>>-- HEX ---+--- ON ----+----->>
           |   |   |
           |   |   |
           +--- OFF ----+----->>
```

**Description:**

Displays the contents of the current text editor window as character or as both character and hexadecimal. HEX ON/OFF is equivalent to SET option primary command **SET VIEW HEX/CHAR** and may be interrogated using QUERY/EXTRACT VIEW

**Parameters:**

ON  
OFF  
HEX display format is set ON or OFF.

## HIDE

**Environments:**

HIDE primary command exists in the following application environments.

<b>Text Edit</b>	Text Editor (both XEDIT and ISPF interfaces.)
<b>Data Edit</b>	SDE Data Editor.



**Syntax:**

```
>>-- HIDE -----><
```

**Description:**

Hide all shadow lines that represent groups of one or more excluded lines in the text edit display. HIDE is equivalent to SET option primary command **SET SHADOW OFF** and may be interrogated using QUERY/EXTRACT SHADOW.

**RESET HIDE** will redisplay all shadow lines. (SET SHADOW ON).

## ICOMMAND

**Syntax:**

```
>>-- ICommand -- command -----><
```

**Description:**

Where a command exists in both the XEDIT and ISPF edit command sets and INTERFACE=XEDIT is active, ICOMMAND may be used as a prefix in order to force the use of the ISPF version of the command.

If INTERFACE=ISPF is active, the ICOMMAND prefix has no effect.

**Parameters:**

*command*  
Any text editor primary command followed by its parameters.

**Example:**

```
ICommand Change abc 'DEF' all
```

The ISPF version of the CHANGE command is used regardless of INTERFACE=XEDIT.

**See Also:**

**ECOMMAND**

## IMMEDIATE

**Syntax:**

```
>>-- IMMediate -- macrodef -----><
```

**Description:**

IMMEDIATE allows the user to write and execute a short CBLLe REXX macro from the command line. IMMEDIATE is especially useful for executing one-off functions and for testing excerpts of logic when writing CBLLe macros.

**Parameters:**

*macrodef*  
A short CBLLe REXX macro that may be defined on a single line.

**Example:**

```
immediate 'msg' x2d(1AE)
```

Convert x'1AE' to decimal and display the result on the message line.

```
imm 'wrap off'; ':1'; do forever; 'nomsg /=START='; if rc<>0 then leave; 'cl:12'; 'cov **'; '-1'; 'a5'; '5'; end; ':1'
```

Locate all lines containing the string "=START=", overlay column 12 with "\*\*\*" and insert 5 blank lines prior.

**See Also:****DEFINE PURGE**

---

**INPUT**

---

**Environments:**

INSERT primary command exists in the following application environments.

<b>Text Edit</b>	Text Editor (both XEDIT and ISPF interfaces.)
<b>Data Edit</b>	SDE Data Editor.

**Syntax:**

```
>>--- Input -----><
      |             |             |
      +-- Insert ---+ +-- string ---+
```

**Description:**

Insert a new line following the focus line containing the specified text string. The new line becomes the focus line and the cursor placed in column 1 of the new line.

Where no string is specified, the inserted line is blank. This is the same as ADD.

**Parameters:**

*string*  
Text string to be inserted in column 1 of the new line.

Where more than 1 blank separates the text string from the command verb, the additional separating blanks are treated as part of the text string.

**Example:**

```
input/Hello/
  Insert a line containing the text string "/Hello/" following the focus line.
```

```
ins ABC
  Insert a line containing the text string " ABC" following the focus line.
```

---

**JOIN**

---

**Syntax:**

```
>>-- Join -----><
      |             |
      +-- Aligned ---+
```

**Description:**

Join text from the line below the focus line to the focus line itself starting at the focus column. Text at, and to the right of, the focus column is overlaid.

The LRECL setting defines the last column of the text to be joined.

Where ALIGNED is not specified, text starting in column 1 of the line below the focus line is joined to the focus line.

**Parameters:**

ALIGNED  
A number of leading blanks, not exceeding the number of leading blanks in the focus line, are stripped from the start of the line below the focus line before the join is actioned.

**Example:**

In the following, focus column is column 13.

```
<...+...1..|.+.2...+...3...+...4...+
      Hello Dave
      World
```

Command JOIN would give:

```
<...+...1..|.+.2...+...3...+...4...+
      Hello      World
```

Whereas JOIN ALIGNED would give:

```
<...+...1..|.+.2...+...3...+...4...+
      Hello World
```

**See Also:**

[SPLIT](#) [SPLTJOIN](#)

---

## LABEL

---

**Syntax:**

```
>>-- LABEL --+-- line_num --+---- = --- .label_2 ----+-----+-----><
          |                |                |                |
          +-- .label_1 --+                +-- level --+
```

**Description:**

Primarily for ISPF Edit macro compatibility, LABEL may also be executed as primary command to set a line label at a specific line in the text edit view.

Note that LABEL performs similar operation to the [SET POINT](#) option. Furthermore, labels set by the LABEL command are included in QUERY/EXTRACT POINT output.

**Parameters:**

*line\_num* Set the label at the specified line number *line\_num* in the current text edit view.

*.label\_1* Set the label at a line identified by existing line label *.label\_1* within the current text edit view. The preceding "." (dot) in *.label\_1* is mandatory.

*.label\_2* The name, *.label\_2*, of the label to be set. within the current text edit view. The preceding "." (dot) in *.label\_2* is mandatory.

*level* The highest nesting level (0-255) at which this label is visible to the user or to an ISPF macro.

**See Also:**

SET/QUERY/EXTRACT Option: [POINT](#)

## LEFT

### Environments:

LEFT primary command exists in the following application environments.

Text Edit (ISPF)	Text Editor with INTERFACE=ISPF (default for z/OS).
Text Edit (XEDIT)	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
Data Edit	SDE Data Editor.
System	FileKit base windows system.

### ISPF Interface

If **INTERFACE=XEDIT** is in effect, the ISPF version may be invoked by prefixing the command with **ICOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

### ISPF Syntax:

```
>>- LEFT -----><
      |-----|
      |  Cursor -----|
      |  CSR -----|
      |  Data -----|
      |  Half -----|
      |  Max -----|
      |  Page -----|
      |  n_cols -----|
```

### Description:

Scroll the view of the data within the text editor window left towards the first column of edited text.

LEFT is assigned to function key **F10** by default. Note that any text entered at the command prompt when a PFKey is pressed will be treated as parameter input to the command associated with the PFKey (e.g. If MAX is at the command prompt when F10 is pressed, LEFT MAX will be executed.) Where no parameter is specified, the scroll amount will be the value specified in the **Scroll** field in the top right corner of the window display.

The first column of text becomes the first column displayed when an attempt is made to scroll the display left beyond the first column of text.

### Parameters:

CURSOR  
CSR

The column of text on which the cursor is positioned (i.e. the **focus column**) becomes the last column of the scrolled display. If the cursor is positioned outside the display area or on the last column of text in the current display, then LEFT PAGE is executed instead.

DATA

Scroll left to display the page (i.e. the display window width) less one column of text to the left of the first column in the current display.  
i.e. The first column of text in the current display becomes the last column of the scrolled display.

HALF

Scroll left half a page of data.  
The column of text that is half way across the page of data in the current display becomes the last column of the scrolled display.

MAX

Scroll left so that the first text column of the edited data becomes the first column in the scrolled display.

PAGE

Scroll left to display the page (i.e. the display window width) of text to the left of the first column in the current display.  
i.e. The column of text to the left of the first column in the current display becomes the last column of the scrolled display.

n\_cols

Scroll left a specified number of columns.  
The column of text that is *n\_cols* columns to the left of the first column in the current display becomes the first column of the scrolled display.



**Description:**

Remove from the window view or un-tag all lines that match line-target.

If the current DISPLAY setting is DISPLAY 0 1 or DISPLAY 1 1, CBLe assumes that the ALL command has previously been issued, thus allocating a subset of lines within the file with a selection level of 1. In this case the selection level of all lines matching the line-target are set back to 0.

If any other DISPLAY setting is in effect, CBLe sets DISPLAY 1 1, sets the selection level of all lines that match the line-target to 0 and sets the selection level of all other lines to 1.

If the TAG or MORE TAG command has been issued, the tag bit may be set on for specific lines in the display. Lines with the tag bit set on are automatically highlighted.

The TAG parameter may be inserted before the line-target of a LESS command to remove the tag bit from lines in the display that satisfy the specified line-target.

**Parameters:**

`TAG` Indicates un-tagging required as opposed to excluding lines from the display.

`line-target`  
`line-target` condition.

**Examples:**

```
less /XYZ/
All lines containing string "XYZ" are excluded from the window view.
```

```
less tag /###/
Remove the tag bit and highlight from lines containing the string "###".
```

**See Also:**

[ALL](#) [TAG](#) [MORE](#)

## LINE

**Syntax:**

```
>>-- LINE  +-+-- line_num  +-+---- =  -- data  -----><
           |                |
           +-+ .label  ----+
```

**Description:**

Primarily for ISPF Edit macro compatibility, LINE may also be executed as primary command to replace text in a specified text edit line.

**Parameters:**

`line_num` Replace text in the line assigned the specified line number `line_num`. In almost all cases, this will be the record number of an edited file.

`.label` Replace text in the line assigned the specified label, `.label`. The preceding "." (dot) in `.label` is mandatory.

`data` The text that will replace existing text in the specified line.

**See Also:**

[CREPLACE](#) [LINE\\_AFTER](#) [LINE\\_BEFORE](#) [REPLACE](#)

---

## LINE\_AFTER

---

**Syntax:**

```

                                     +- DATALINE +-
                                     |               |
>>-- LINE_AFTER  ---+-- line_num ---+----- = ---+-----+--- data -----><
               |               |
               +--- .label -----+

```

**Description:**

Primarily for ISPF Edit macro compatibility, LINE\_AFTER may also be executed as primary command to insert a line of text following a specified line in the edited data.

**Parameters:**

*line\_num*  
Insert a line of text after the line assigned the specified line number *line\_num*. In almost all cases, this will be the record number of an edited file.

*.label*  
Insert a line of text after the line assigned the specified label, *.label*. The preceding "." (dot) in *.label* is mandatory.

[DATALINE] *data*  
The text that will be inserted after the specified line.

**See Also:**

INPUT (INSERT) LINE LINE\_BEFORE

---

## LINE\_BEFORE

---

**Syntax:**

```

                                     +- DATALINE +-
                                     |               |
>>-- LINE_BEFORE  ---+-- line_num ---+----- = ---+-----+--- data -----><
               |               |
               +--- .label -----+

```

**Description:**

Primarily for ISPF Edit macro compatibility, LINE\_BEFORE may also be executed as primary command to insert a line of text before a specified line in the edited data.

**Parameters:**

*line\_num*  
Insert a line of text before the line assigned the specified line number *line\_num*. In almost all cases, this will be the record number of an edited file.

*.label*  
Insert a line of text before the line assigned the specified label, *.label*. The preceding "." (dot) in *.label* is mandatory.

[DATALINE] *data*  
The text that will be inserted before the specified line.

**See Also:**

INPUT (INSERT) LINE LINE\_AFTER

# LIST

## Environments:

LIST primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor. (List <b>MODULES</b> , <b>RPOS</b> , <b>SDO</b> , <b>STORAGE</b> and <b>UKRS</b> )

## Syntax:

```
>>-- List -- listtype -- /listparms/ -----+-----+-----+----->
                                     |         |         |
                                     +- STEM rexx_stemvar +- +- STRIP +-
                                     |         |
                                     +- FILE filename -----+

+- Lines ----+
>-----+-----+-----+-----+-----+-----+-----+----->>
|         |         |         |         |         |         |
+- Columns +- +- SUBset /select_clause/ +-
```

## Description:

Extract rows of data returned from various FileKit LIST type commands and either place the output in a temporary edit view or assign the fields to REXX stem variables for use in a CBL e REXX macro.

## Parameters:

- listtype*
- The FileKit list type function to extract. This must be one of the following:
- AMS  
List **IDCAMS command** output.
  - APE  
List **Assembler Program Environment**. This is the list of modules in the current version of FileKit.
  - CLD  
List details of the first occurrence of a member name (including the library in which it was found) within a concatenated library search path. The *listparms* parameter is the DDname allocated to the concatenation, and optionally a parenthesised list of member masks.
  - FS | FSEARCH  
List records that match a **File Search** utility search string.
  - HFS | PATH  
List **HFS** files.
  - LA | ALLOCated  
List **Allocated** datasets.
  - LAS | ASSOCIATIONS  
List **Associations** for cataloged entries.
  - LAYOUT  
**Record Layout** of record type definitions in a nominated SDE structure (SDO), COBOL/PL1 copybook or COBOL/PL1 ADATA output file.
  - LC | CATalog  
List **Catalog** entries.
  - LCA  
List Catalog type=X (ALIAS) entries.
  - LD | DATasets | DSNs | FILEs  
List **Dataset** attributes.
  - LJQ | JOBENQueues  
List **MVS Job Enqueues** for a given job.
  - LL | LIBrary  
List **Library Members**.
  - LLS  
List **Loaded Structures**.
  - LQ | ENQueues



**List MVS Enqueues** for a given queue.

LSG | LISTSTORAGEGROUPS

**List MVS SMS Storage Groups.**

LSGV | LISTSTORAGEGROUPVOLS

**List MVS SMS Pool Storage Group Volumes** for a given storage group name.

LV | VTOC

**List VTOC file.** entries.

LVOL | VOLumes

**List DASD volumes.**

LVR

**List CBLVCAT Raw** output.

LX | EXTents

**List VTOC extents.**

POWER

VSE only: **POWER Queue** list output.

RETRIEVE

The list of commands available via the **RETRIEVE** and **CRETRIEV** commands. CRETRIEV is assigned to F12 by default.

SDO

**List SDE structure (SDO)** library members.

SQL

MVS only: List output from a **DB2 Dynamic SQL** operation. This may be DB2 table row data obtained by a DB2 SQL SELECT query. Note that a connection will be made to the DB2 sub-system to which the user last connected.

STRUCTure

MVS only: Structured Data Environment **DISPLAY STRUCTURE** list output.

SWA

List **Window Attributes** of all open windows.

SYSAPF

MVS only: List **APF authorised libraries.**

SYSLL

MVS only: List libraries in the active **Link List.**

See the relevant List type commands for details of the column names and data returned for each.

*listparm*

A parameter string passed to the list function. This string must always be present. If it contains no blanks or special characters it can be blank-delimited, otherwise it must be delimited by a pair of special characters. e.g. If a null string is required it can be provided as `"/"`.

For file lists, suitable trailing wild cards will be appended to the *listparm* file name mask provided, as documented by the relevant list operation.

FILE *filename*

A keyword parameter which requests that the LIST command displays the listed data in an in-storage edit view with the specified fileid, *filename*. The list output may be saved to disk using this *filename*.

This option is the default if the Llist command is issued from the command line or via a function key in which case the *filename* will be `"%user%.listtype.LIST"`.

STEM *rexx\_stemvar*

A keyword parameter which requests that the Llist command places the listed data in an array of REXX variables with this stem name.

This option is the default if the Llist command is issued from a macro.

If the stem name is not given then the list type name is used.

If the stem name does not end with a period then one is added.

STRIP

A keyword parameter which requests that the data is stripped of leading and trailing blanks before being placed in REXX variables.

Lines

A keyword parameter which requests that the Llist command returns complete list lines.

If the FILE option is used then the list column header and list lines are placed in an edit window in the current ring with the file name specified (but not actually written to disk).

If the STEM option is used then the following variables are set:

```
rexex_stemvar.0
    The number of list lines.

rexex_stemvar.i
    The ith list line.

rexex_stemvar.columns
    The list column header line.
```

#### Columns

A keyword parameter which requests that the List command returns the list data in column variables.

If the FILE option is used, or the List command is not issued from a macro, this option is invalid. An error message is issued and no data is returned.

If the STEM option is used then the following variables are set:

```
rexex_stemvar.0
    The number of list lines.

rexex_stemvar.i.colname
    The value of column colname for the ith list line.

rexex_stemvar.columns.0
    The number of list columns.

rexex_stemvar.columns.j
    The name of the jth column.
```

#### SUBset /clause/

A keyword parameter which must be followed by a list subset command delimited by a special character.

The subset *clause* can contain one or more of the following (in any order):

```
select clause
    Use the select clause to define the list of column names to return. The default is "*" which means all columns.

where clause
    Use the where clause to apply a filter which restricts the lines returned based on a condition defined in terms of the list's column names and their values.

sort (order by) clause
    values in the columns of the list. Use the sort or order by clause to determine the order in which the list lines are returned based on column values.
```

See [Selecting, Sorting and Filtering](#) for details of the syntax of these clauses.

#### Examples:

```
list volumes/CBL*/ file lac.vollist subset/select unit,vol,freecyl/
li alloc /SYSEXEC/
li alloc SYSPROC
li library/cbl.cbli120.asm(edt*)/ stem ll.
li library/cbl.cbli120.asm(edt*)/ file lac.liblist
li catalog/cbl.cbli*.**/ file lac.catlist
li dataset/cbl.cbli*.**/ file lac.dsnlist
li vtoc /cblmct/ file lac.vtoclist subset / select vol,dsn,org /
li extent /cblmct/ file lac.extlist
li enqueue/sysdsn LAC./ file lac.enqlist
li enqueue/spfedit LAC./ file lac.enqlist
```

---

## LOCATE

---

#### Environments:

LOCATE primary command exists in the following application environments.

<b>Text Edit (ISPF)</b>	Text Editor with INTERFACE=ISPF (default for z/OS).
<b>Text Edit (XEDIT)</b>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
<b>Data Edit</b>	SDE Data Editor.
<b>Lists</b>	List display windows.





order of line number, "-" indicates that lines are processed in descending order of line number.

*command*

Any primary command or macro specification.

### Example:

```
l 3      Focus line is set 3 lines down from the current focus line.

:50     Focus line is set at line 50.

/Hello/ Focus line is set at the first line below the focus line to contain the string "Hello".

-*     Focus line is set to the top-of-file line.

:10 ADD 3 Focus line is set at line 10 then 3 lines are added.

:6 /CBL/ 1 DEL
Focus line is set at line 6, then to the first line below line 6 to contain the text "CBL", then to 1 line below the line
containing "CBL" before eventually deleting the focus line.

locate - r /\-^100/
A regular expression is used to set the new focus line to be the first line above the focus line to contain 100 hyphon sybols
(-).
```

---

## LOWERCASE

---

### Syntax:

```
>>-- LOWercase -----+-----+-----><
                        |         |
                        +- group-target -+
```

### Description:

Lower case all alpha characters in the target area.

Only alpha characters that are within the current ZONE columns will be lower cased.

Where BLOCK is specified as the group-target and the target area is a box block, then the ZONE setting is ignored and all alpha characters within the block are lower cased.

The focus line is not changed by a LOWERCASE command.

### Parameters:

*group-target*

**Group-target** condition defining the end of the source target area. If the group-target is not satisfied then the LOWERCASE command will fail.

### Example:

```
lowercase 6
Alpha characters in the focus line and the 5 lines following are lower cased.

low -8
Alpha characters in the focus line and the 7 lines preceding are lower cased.

lower /SELC/
Alpha characters in the focus line and all lines up to, but not including, the first line following the focus line to contain the
string "SELC", are lower cased.
```

### See Also:

**UPPERCASE**

---

## MACRO

---

**Environments:**

MACRO primary command exists in the following application environments.

<b>Text Edit</b>	Text Editor (both XEDIT and ISPF interfaces.)
<b>Data Edit</b>	SDE Data Editor.

**Syntax:**

```
>>-- MACRO -- macroname ---+-----+-----><
                        |         |
                        +- text -+
```

**Description:**

Execute the REXX language macro specified by *macroname*. Any text specified following the *macroname* is passed to the macro as an argument. The specified macro is read into memory from disk, executed and removed from memory on completion.

Where IMPMACRO is set ON (default), CBL e first checks a user supplied token for a recognised CBL e command. If it is not recognised as a command, CBL e attempts to find a macro with the same name. Therefore, when invoking a macro, the command verb MACRO may not be necessary.

If the macro cannot be located, an error message is issued and the MACRO command fails.

**Parameters:**

*macroname*

Name of the macro to be executed.

If *macroname* is a full fileid containing file name, path, etc., then CBL e reads the macro from the specified location. If only a file name is specified, CBL e searches each directory in the macro path for a matching *macroname*.

*text*

Text to be passed to the macro as arguments.

**Example:**

*macro open*

Execute the user macro OPEN, not the CBL e command OPEN.

*ldiff*

Execute the user macro LDIFF. (CBL e command LDIFF does not exist.)

---

## MARK

---

**Syntax:**

```
>>-- MARK ---+--- Line ---+-----><
                        |         |
                        +--- Box ---+
```

**Description:**

Mark the boundaries of a line or box. MARK LINE marks the focus line as one edge of a line block and MARK BOX marks the focus column in the focus line as one corner of a box block. MARK is most often used in CBL e macros prior to a FILL, DUPLICATE, DELETE, etc. command.

A marked block in a file other than the current file, will be reset when the MARK command is issued.

The marked blocks remain marked until explicitly unmarked. When defining a new boundary for an existing block, the block will resize from the focus line or column to the most extreme boundary of the current block.

**Parameters:**

LINE Mark the focus line as a boundary for a line block.

BOX Mark the focus column in the focus line as a boundary for a box block.

**See Also:**

RESET

---

**MORE**

---

**Environments:**

MORE primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor.

**Syntax:**

```
>>-- MORE ---+-----+--- line-target ----->><
          |         |
          +--- TAG ---+
```

**Description:**

Add to the current window view or tag all lines that match line-target. The MORE command is usually used following an ALL or LESS command.

MORE sets the selection level of all lines matching line-target to selection level 1.

Where the ALL command has not previously been issued, The MORE command also sets DISPLAY 1 1. This is the same as executing the ALL line-target command.

The TAG parameter may be inserted before the line-target of a MORE command to set the tag bit on lines in the display that satisfy the specified line-target. Unlike the TAG command, MORE TAG does not remove the tag bit for lines that do not satisfy the line target.

**Parameters:**

*line-target*  
line-target condition.

TAG Indicates tagging required as opposed to adding extra lines to the display.

**Examples:**

```
more tag /###/
Tag (highlight) lines that contain the string "###". (Lines that are already have the tag bit on are unaffected.)
```

```
more /XYZ/
All lines containing string "XYZ", which have been excluded from the window view by a previous ALL or LESS command, are included in the current window view.
```

**See Also:**

ALL TAG LESS

## MOVE

### Environments:

MOVE primary command exists in the following application environments.

Text Edit (ISPF)	Text Editor with INTERFACE=ISPF (default for z/OS).
Text Edit (XEDIT)	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).

### ISPF Interface

If **INTERFACE=XEDIT** is in effect, the ISPF version may be invoked by prefixing the command with **ICOMMAND**. See [ISPF/XEDIT Primary Command Precedence](#).

### ISPF Syntax:

```
>>-- MOVE -----><
      |-----|-----|-----|-----|
      |  fileid  |  AFTER  |  .name  |
      |          |  (1)  |  |
      |  BEFORE  |  |
```

### Note:

1. If AFTER and BEFORE are omitted, then line (prefix) command "A" or "B" must have been specified in the the current text edit view.

### Description:

Move all lines from an existing sequential or VSAM data set, PDS/PDSE library member or HFS file into the focus text edit view.

The line within the focus text edit view that identifies the target of the copy is specified via a line label name. If BEFORE or AFTER line label name is not specified, then the target line is the first line containing the line (prefix area) command "A" (AFTER) or "B" (BEFORE). If neither exists, error message ZZSE130I is returned.

If MOVE is entered with no parameters, a popup window is displayed which prompts the user to enter a fileid from which all records will be moved.

Following the move operation, a popup prompt is displayed to confirm deletion of the source file.

### Parameters:

*fileid*

Specifies the fileid of the source file from which records are to be moved.

If *fileid* is a less than 8 characters in length and is a valid member name, the target file will be a member of member name *fileid* belonging to the same PDS/PDSE library referenced in the current text editor view. If the focus edit view does not display a library member, the *fileid* will be treated as an HFS file within the current HFS working directory.

If *fileid* includes a volume id, then the source file may be a cataloged or uncataloged data set or PDS/PDSE library which exists in that volume's VTOC. e.g. VOLWKA:DEV.UNCATLG.FILE.

AFTER | BEFORE *.name*

Identifies whether lines are to be moved after or before the target line specified by label *.name* in the current text edit view.

### Examples:

```
move CBL.TEST.KSDS after .LAST
```

Move VSAM KSDS data set records after labelled line .LAST in the current edit view.

### See Also:

CLIPBOARD COPY CREATE CUT PASTE REPLACE

### XEDIT Interface

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See [ISPF/XEDIT Primary Command Precedence](#).



**XEDIT Syntax:**

```
>>-- Move -- group-target ---+-----+-----><
                        |           |
                        +- line-target -+
```

**Description:**

**MOVE** is also an ISPF primary command.

If **INTERFACE=ISPF** is in effect (default for FileKit running in an MVS environment), then the ISPF version of the command is executed instead. See section **ISPF/CBLe CLI Command Precedence** and IBM publication "ISPF: Edit and Edit Macros". Use the CBLe CLI command **ECOMMAND** to override.

Move text from a target area to a target line.

Where **BLOCK** is specified, **MOVE** supports moving text between files. Otherwise **MOVE** can only operate on lines in the same edited file. The block remains marked in its new position and the first line of the moved block becomes the focus line.

Where **BLOCK** is not specified as the group-target, the last line of the moved target group of lines becomes the focus line.

Where line-target is omitted, the current focus line is used.

**Parameters:**

*group-target*

**Group-target** condition defining the end of the source target area. If the group-target is not satisfied then the **MOVE** command will fail.

If **BLOCK** is specified, then the marked block will be moved as follows:

Line Block Marked line(s) are moved to the line immediately following the line-target.

Box Block Marked box is moved to the focus column of the target line.

**Note:** group-target key word **ALL** is not supported.

*line-target*

**Line-target** condition defining a destination target line. If the line-target is not satisfied then the **MOVE** command will fail.

Where the source target area is not a box block, lines are moved to the line immediately following the target line.

**Example:**

move 6 :16

The focus line and the 5 lines following are moved to line below line 16.

m -8 22

The focus line and the 7 lines preceding are moved after 22nd line following the focus line.

mo 3 -/SELC/

The focus line and the 2 lines following are moved to the line below the first line containing the string "SELC", scanning backwards from the focus line.

mov block

The marked box block is moved to the focus column of the focus line and lines that follow up to the depth of the marked block.

**See Also:**

**COPY**

---

**MSG**

---

**Syntax:**

```
>>-- MSG ---+-----+-----><
                        |           |
                        +-- string ---+
```

**Description:**

Display a message string on the message line. MSG is most often used in CBL macros.

If no text string is specified, then the message line is cleared.

**Parameters:**

*string*  
Text string to be displayed on the message line.

**Example:**

```
msg 'Hello World'
```

Output "Hello World" to the message line.

**See Also:**

[EMSG](#) [NOMSG](#)

## NFIND, NFINDUP

**Syntax:**

```
>>----- NFind -----+-----+----->>
                        |         |
                        +- string -+

>>+--- NFINDUP ---+---+-----+----->>
  |         |         |         |
+--- NFUP  -----+ +- string -+
```

**Description:**

Find the first line following (NFIND) or previous to (NFINDUP) the focus line that does **not** contain the specified string in column 1 and make this line the new focus line.

Where more than 1 blank separates the text string from the command verb, the additional separating blanks are treated as being part of the text string.

If WRAP is set ON and string is found in column 1 of every line before bottom-of-file or top-of-file is encountered, then the scan continues from the opposite extreme of the file up to and including the focus line.

**Parameters:**

*string*  
Search text string.

No delimiting characters should be specified.

Blanks in the search text are each treated as a wildcard representing a single character in the file data. In order to find a blank space, the "\_" (underscore) character should be specified in the corresponding position of the find string.

If string is not specified, then the string argument specified on the last NFIND/NFINDUP command executed, is used. If no previous NFIND/NFINDUP command has been executed in this instance of CBL, then the following message is displayed:

```
ZZSE060E No remembered operand for NFIND command.
```

**Example:**

```
nfind Hello
```

Focus line is set at the first line below the focus line that does **not** contain the string "Hello" in column 1.

```
nf ABC DEF
```

Focus line is set at the first line below the focus line **not** to contain the string "ABC" in column 2 and the string "DEF" in column 7.

```
nfup _XYZ
```

Focus line is set at the first line above the focus line **not** to contain a blank in column 1 followed by the string "XYZ" in column 2.

**See Also:**

**FIND FINDUP** and the SET/QUERY/EXTRACT Options: **WRAP CASE**

## NOMSG

**Syntax:**

```
>>-- NOMsg --- command -----><
```

**Description:**

Execute the specified CBL or SDE command, suppressing the display of any messages that may be returned. NOMSG is most often used in CBL and SDE edit macros.

Execution of NOMSG stores the current MSGMODE setting, sets MSGMODE OFF, executes the specified command and finally restores the MSGMODE setting.

**Note:** QUERY and EXTRACT LASTMSG recalls the last message that would have been displayed if the command had been executed without NOMSG.

**Parameters:**

*command*  
A CBL or SDE primary command.

**Example:**

```
nomsg ecommand change /ABC/DEF/* *  
Any messages generated by the CHANGE command are suppressed.
```

**See Also:**

**MSG EMSG**

## NOND

**Syntax:**

```
>>-- NOND -----><
```

**Description:**

Supplied as a text edit macro, NOND uses the **SET COLOUR NONDISPLAY** option to toggle the display of underscores on printable text in lines of data which include unprintable characters.

## NORECALL

**Syntax:**

```
>>-- NORecall --- command -----><
```

**Description:**

Execute the specified Text Editor primary command but exclude it from the Text Editor stack of commands executed. NORecall is most often used in macros.

The specified command will not be recalled when the FileKit command "RETRIEVE -" or CRETRIEV is executed. (Default action for F12).

**Parameters:**

*command*  
A Text Editor primary command.

**Example:**

```
nor change /ABC/DEF/* *
    Exclude this execution of CHANGE from the recallable command stack.
```

**ONLY**

**Syntax:**

```
>>--- Only +-----+<<
           |         |
           +---| Find Spec |---+
```

**Find Spec:**

```
(1)
>- string +-----+>
          |         |
          +- CHARs --+
          |         |
          +- PREFIX -+ +- pos1 +-----+ +- .name1 +-----+
          |         |         |         |         |         |
          +- SUFFIX -+         +- pos2 -+         +- .name2 -+
          |         |
          +- WORD ---+
```

**Notes:**

1. Operands may be entered in any order.

**Description:**

The ONLY command uses an ISPF style FIND specification to display only those lines which match the specification.

ONLY is equivalent to the ISPF primary commands EXCLUDE ALL followed by FIND ALL EX find\_spec.

The ONLY command issued with no find specification is equivalent to the ISPF primary command RESET EXCLUDED and redisplayes all excluded lines.

**Parameters:**

The parameters of the ONLY command are the same as those of FIND except that FIND ALL EX is implied so that the NEXT|ALL|FIRST|LAST|PREV and EX|NX|X parameters are invalid. See ISPF FIND.

**Example:**

```
only c'Name' .a .b
    Exclude all lines from the display which are not between line names .a and .b and do not contain the case sensitive string Name.
```

# OPEN

## Syntax:

```
>>--- OPEN -----+-----+-----><
                |           |
                +-- filter --+
```

## Description:

For z/OS, open the Text Edit Entry panel or, for CMS and VSE, open the OPEN dialog window. This command is equivalent to selecting "Open" from the "File" menu item on the main window menu bar.

For CMS and VSE, if *filter* is not specified, the Open window displays the list of fileids for the filter mask last specified in an Open window.

## Parameters:

*filter*

For z/OS, *filter* will populate the appropriate fields of the input sequential, GDG or VSAM dataset, PDS/PDSE member or HFS path.

For CMS and VSE, *filter* will populate the Filter field of the Open window. If filter is not specified, then the filter mask used is that specified in the last Open window opened. If an Open window has not already been opened for the current instance of the text editor, the fileid of the current text edit view is used.

For both the Text Editor entry panel and the Open dialog, *filter* may include the following wildcard characters so that a file entry may be selected from a filtered list.

- \* A single asterisk indicates that either a qualifier or one or more characters within a qualifier can occupy that position. An asterisk can precede or follow a set of characters.
- \*\* A double asterisk indicates that zero or more qualifiers can occupy that position. A double asterisk cannot precede or follow any characters; it must be preceded or followed by either a dot or a blank.
- % A single percent sign indicates that exactly one character can occupy that position. (Up to 8 percent signs can be specified in each qualifier.)

**Note:** for CMS, valid qualifiers may be considered to be File Name, Type or Mode.

On MVS or VSE, an entry field that does not contain an \* (asterisk) wild card will be appended with \*.\* to list those cataloged data sets whose names begin with the entry string.

## Example:

open CBL.S\*.\*  
For z/OS, open the Text Edit entry panel, listing all files whose fileids match the filter mask "CBL.S\*.\*".

open SS\* CTL \*  
For CMS, open the OPEN dialog window, listing all files whose fileids match the FN FT FM mask "SS\* CTL \*".

# OPTIONS

## Environments:

OPTIONS primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor.

## Syntax:

```
>>--- OPTions -----+-----+-----><
                |           |
                +-- ALL -----+
```

+-- Edit ----+

**Description:**

Execute the OPTIONS command to perform one of the following:

- Execute the **QUERY** command for all possible query parameters with the exception of QUERY MACRO, to display all current settings.
- Generate a CMX file containing **SET** commands for all current SET option values. The CMX command file also contains references to HELP pages for each SET option.

The OPTIONS command is also executed via the Options CBLe Menu bar item.

**Parameters:**

ALL	Display current query option values.
EDIT	Generate temporary OPTIONS CMX file containing all available SET commands with prevailing option settings.

## OVERLAY

**Syntax:**

```
>>-- Overlay --- string -----><
```

**Description:**

Overlay text in the focus line with the specified text string starting at column 1. The text string begins immediately after the single separating blank that follows the OVERLAY command verb.

Characters in the focus line that correspond to blanks in the supplied overlay string are unchanged. In order to overlay a character with a blank, the "\_" (underscore) character should be specified in the corresponding position of the overlay string.

All other characters supplied in the overlay string replace characters in corresponding positions in the focus line.

**Parameters:**

*string* Overlay text string.

Where more than 1 blank separates the text string from the command verb, the additional separating blanks are treated as part of the text string.

**Examples:**

```
overlay abc
Overlay text at column 1 of the focus line with "abc".
```

```
O ABC_X YZ
Where original text at column 1 of the focus line is: 0123456789
After the OVERLAY command text is: 0ABC X6YZ9
```

## OVERLAYBOX

**Syntax:**

```
>>-- OVERLAYBOX -----><
```

**Description:**

Overlay text on and, if necessary, below the focus line with text from a marked line block or box block.

Where a box block has been marked, text at, and to the right of the focus column is overlaid.

## PASTE

### Environments:

PASTE primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor.

### Syntax:

```

      +--- Keep ----+
      |               |
>>-- PASTE -----><
      |               |
      +--- Delete --+
    
```

### Description:

PASTE may be used instead of primary command **CLIPBOARD PASTE** to paste and optionally clear lines of text from the FileKit **clipboard** to the current text edit view.

Text from the clipboard is copied to a target line determined by the first occurrence in the text edit view of the line (prefix area) command "A" (after) or "B" (before). Otherwise, the text is copied following the **focus line**.

### Parameters:

**KEEP** Text in the clipboard is preserved following the PASTE operation. This is default.

**DELETE** Text in the clipboard is deleted following the PASTE operation.

PASTE DELETE is equivalent to executing CLIPBOARD PASTE followed by CLIPBOARD CLEAR.

### See Also:

**CLIPBOARD COPY CREATE CUT REPLACE**

## POPUP

### Syntax:

```

      +-----+
      |               |
      | v         |
>>-- POPUP ----- / --- string -- / ---><
    
```

### Description:

For use in CBL macros only, opens a pop-up menu containing any number of items specified by separate string arguments.

The "/" (slash) character is normally used as the delimiter encompassing and separating each item string in the pop-up menu. However, any non-alphanumeric character that does not have a special meaning to CBL may be used as a string delimiter (e.g. "%", "#", "."). The delimiter character used must not appear in any of the string arguments.

The following REXX stem variables, which do not require an **EXTRACT** command, are set on completion of the POPUP command:

Following a POPUP operation, the text of the menu item selected by the user may be obtained as REXX compound variables using **EXTRACT POPUP**.

### Parameters:

**string** Character string of maximum length 64 and enclosed by delimiters to be displayed as a menu item in the pop-up window. Any number of menu item character strings may be specified, each having a maximum length of 64 characters.

**string** may be prefixed by "~" (tilde) to represent an item that may not be selected by the user. This item will be displayed in a different colour (default is blue) to the live menu items (default is white).

If **string** consists only of "-" (minus), the menu item is displayed as a separator line within the menu.

**Example:**

```
'popup /John Smith/Robert Jones/~Susan Fisher/~Jane Dough/',
  'Rachel Meredith~/New contact/'

select
  when popup.1 = ''           then 'msg No contact name selected.'
  when popup.1 = 'New contact' then call GetNewName
  otherwise                  'msg 'popup.1 'selected...'
end
```

**See Also:**

**DIALOG**

## PRESERVE

**Syntax:**

```
>>-- PREServe -----><
```

**Description:**

Save current values of SET options so that they may be temporarily changed and ultimately restored using the RESTORE command. PRESERVE is most often used in macros.

SET options are saved for the following:

ARBCHAR	LINEND	RECFM	STREAM
CASE	LRECL	SCALE	UNDOING
COLOUR	MSGLINE	SCOPE	VARBLANK
DISPLAY	MSGMODE	SHADOW	WRAP
IMPMACRO	PREFIX	STAY	ZONE

**See Also:**

**RESTORE**

## PURGE

**Syntax:**

```

      +-----+
      v       |
>>-- PURge ---+-- macroname ---><

```

**Description:**

PURGE is used to unload macro definitions that have been loaded into storage via the **DEFINE** command.

Multiple macros may be unload in a single PURGE execution by specifying a blank separated list of macro names.

**Parameters:**

*macroname*  
The name of the macro in storage.



**Example:**

```
purge puttime delsame addnext
  Unload macros "puttime", "delsame" and "addnext" from storage.
```

**See Also:**

**DEFINE IMMEDIATE**

---

## QUERY

---

**Environments:**

QUERY primary command exists in the following application environments.

<b>Text Edit</b>	Text Editor (both XEDIT and ISPF interfaces.)
<b>Data Edit</b>	SDE Data Editor.

**Description:**

See **SET/QUERY/EXTRACT Options**.

---

## QUEUE

---

**Syntax:**

```
>>-- QUEUE --- cmd_stream -----><
```

**Description:**

Primarily for use in CBLLe REXX macros, QUEUE stacks a command stream that gets executed the next time the screen is refreshed. This is an effective method of piping commands to a non-CBLLe edit view, e.g. List windows.

A screen refresh occurs when the macro execution completes or when the **REFRESH** command is executed during macro execution.

Any number of command streams may be stacked for subsequent execution in the sequence first in, first out. The command stream is executed from the window view within the CBLLe MDI environment that last received focus and may consist of several commands separated by the command line end separator character.

**Parameters:**

*cmd\_stream*  
Any valid executable command stream appropriate to the focus window.

**Examples:**

```
'queue where recfm=VB ; sizewindow d=30 w=60'
'listdataset NBJ.TEST*.*'
'refresh'
```

LISTDATASET opens a new MDI child window within CBLLe. When the REFRESH command is executed, the LISTDATASET window (the focus window) receives and executes the queued WHERE and SIZEWINDOW commands before the screen is repainted by FileKit.

---

## QUIT, QQUIT

---

**Environments:**

QQUIT primary command exists in the following application environments.

<a href="#">Text Edit</a>	Text Editor (both XEDIT and ISPF interfaces.)
<a href="#">Data Edit</a>	SDE Data Editor.

**Syntax:**

```
>>----- Quit -----><
```

```
>>----- QQuit -----><
```

**Description:**

QUIT is a synonym for primary command [END](#).

QQuit performs the same functionality as QUIT except that no warning or attempt to save data will occur for unsaved changes to the edited text.

**See Also:**

[CANCEL](#) [END](#) [FILE](#)

---

## RCHANGE

---

**Environments:**

RCHANGE primary command exists in the following application environments.

<a href="#">Text Edit</a>	Text Editor. (Interface ISPF and XEDIT)
<a href="#">Data Edit</a>	SDE Data Editor.

**Syntax:**

```
>>--- RChange ----->
```

**Description:**

Repeat the find and replace performed by the last ISPF format [CHANGE](#) command.

Where the last [CHANGE](#) issued was [CHANGE LAST](#) or [CHANGE PREV](#), [RCHANGE](#) will perform a [CHANGE PREV](#) operation, otherwise [RCHANGE](#) performs a [CHANGE NEXT](#) operation. i.e. Search forwards ([NEXT](#)) or backwards ([PREV](#)) from the current cursor location to find the next or previous occurrence of the string/value respectively.

If the cursor is not within the window's data display area, the forwards or backwards search begins at the first position of the first visible or excluded line within the display area.

[RCHANGE](#) is assigned to function key **F6** by default.

**See Also:**

[CHANGE](#) [EXCLUDE](#) [FIND](#) [RFIND](#)

---

## REDO

---

**Environments:**

REDO primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor.

**Syntax:**

```
>>-- REDO -----><
```

**Description:**

REDO re-applies a change made to the current file that has been undone by a previous UNDO command. Where UNDO has not been issued, executing REDO has no affect.

This command is equivalent to selecting "Redo" from the "Edit" menu on the CBL e window menu bar.

Multiple levels of changes undone by repeated UNDO commands may be re-applied by same number of repeated REDO commands.

Following one or more executions of UNDO, REDO will recover the changes even after the following have occurred:

- The cursor position changes.
- The file is saved.
- Changes are made to other files in the file ring.

However, REDO is not able to recover changes following an UNDO if subsequently:

- Further changes are made to the file (lines updated, deleted or added.)
- Changes are made to the selection level of any line (e.g. on an ALL command.)
- Changes are made to the line flags of any line (e.g. on a TAG command.)
- Changes are made to the line name of any line (e.g. on a SET POINT command.)

The third number following "Alt=" on the status line indicates the number of change levels. If this number is followed by an "\*" (asterisk), then it is possible to REDO previous UNDO commands.

**See Also:**

[UNDO](#)

---

## REFRESH

---

**Environments:**

REFRESH primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor.
Interactive Panels	Panels containing embedded tables.

**Syntax:**

```
>>-- REFRESH -----><
```

**Description:**

For use in a CBL e/SDE REXX macro, the REFRESH command will refresh the edit display during execution of the macro.

The display is not normally updated until a macro is completed or unless keyboard input is required.

**Note:** Frequent refresh of the display within a macro can increase the macro's run time.

**Examples:**

```
imm do 10; 'add1'; 'refresh'; end
Add 10 blank lines, one at a time, and refresh the view after each line added.
```

## REPLACE

**Environments:**

REPLACE primary command exists in the following application environments.

<b>Text Edit (ISPF)</b>	Text Editor with INTERFACE=ISPF (default for z/OS).
<b>Text Edit (XEDIT)</b>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
<b>Data Edit</b>	SDE Data Editor. (See <a href="#">REPLACE</a> and <a href="#">REPLACELINE</a> )

### ISPF Interface

If **INTERFACE=XEDIT** is in effect, the ISPF version may be invoked by prefixing the command with **ICOMMAND**. See [ISPF/XEDIT Primary Command Precedence](#).

**ISPF Syntax:**

```
>>-- REPlace -----><
      |
      +--- fileid -----+
          |
          +--- .name1 ---- .name2 -----+
              (1)
```

**Note:**

1. If a group of lines are not specified using line label names, then a group of lines must be specified using one of the following line (prefix area) commands:
  - ◆ **C** or **CC** (Copy), if the group of lines in the current file is to be preserved following successful execution of REPLACE.
  - ◆ **M** or **MM** (Move), if the group of lines in the current file is to be deleted following successful execution of REPLACE.

**Description:**

Replace the contents of a sequential or VSAM data set, PDS/PDSE library member or HFS file, with a group of lines extracted from the current text edit view. The target sequential or library data set may be uncataloged.

If the target file does not already exist, then one of the following will occur:

1. If output is to a new member of an existing PDS/PDSE library or to an HFS file, the target file will be automatically created. When REPLACE defines a new HFS file, the permission bits are set to 740 (rwxr-----).
2. If output is to a data set or to a member of a non-existent PDS/PDSE library, the **Allocate Non-VSAM** panel is displayed in order to allocate the new data set or library.

If REPLACE is entered with no parameters, a popup window is displayed which prompts the user to enter a fileid in which records will be replaced.

**Parameters:**

*fileid*

Specifies the fileid of the target file in which records are to be replaced.

If *fileid* is a less than 8 characters in length and is a valid member name, the target file will be a member of member name *fileid* belonging to the same PDS/PDSE library referenced in the current text edit view. If the current text edit view does not display a library member, the *fileid* will be treated as an HFS file within the current HFS working directory.

If *fileid* includes a volume id, then the target file may be a cataloged or uncataloged data set or PDS/PDSE library which exists in that volume's VTOC. e.g. VOLWKA:DEV.UNCATLG.FILE.

*.name1*

A label name identifying the first line of the group of lines to be copied to the target file.

If not specified, then the group of lines must be marked using "C" or "M" line (prefix area) commands.

*.name2*

A label name identifying the last line of the group of lines to be copied to the target file. If *.name1* has been specified, *.name2* is mandatory.

**Examples:**

```
replace DEV.USER223.TESTDATA.D2012324 .CDBEG .CDEND
Replace the contents of data set "DEV.USER223.TESTDATA.D2012324" with records in the current text edit view which fall between defined line label names .CDBEG and .CDEND inclusively.
```

**See Also:**

[CLIPBOARD COPY CREATE CUT PASTE](#)

**XEDIT Interface**

---

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See [ISPF/XEDIT Primary Command Precedence](#).

**XEDIT Syntax:**

```
>>-- Replace --+-----+-----><
                |         |
                +-- string --+
```

**Description:**

Replace the focus line with the specified text string. The text string begins immediately after the single separating blank that follows the REPLACE command verb.

**Parameters:**

*string*

Replace text string.

Where more than 1 blank separates the text string from the command verb, the additional separating blanks are treated as part of the text string.

If string is omitted, the focus line is replaced with a blank line.

**Examples:**

```
r Hello World
Focus line is replaced with "Hello World starting in column 1.
```

```
replace
Focus line is replaced with a blank line.
```

---

**RESET**

---

**Environments:**

RESET primary command exists in the following application environments.

<b>Text Edit (ISPF)</b>	Text Editor with INTERFACE=ISPF (default for z/OS).
<b>Text Edit (XEDIT)</b>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
<b>Data Edit</b>	SDE Data Editor.





---

## RFIND

---

**Environments:**

RFIND primary command exists in the following application environments.

Text Edit	Text Editor. (Interface ISPF and XEDIT)
Data Edit	SDE Data Editor.

**Syntax:**

```
>>--- RFind ----->
```

**Description:**

Repeat the search performed by the last ISPF format **FIND** command. If RFIND is executed following an **EXCLUDE** or ISPF format **CHANGE** command, then the search string will be that specified on the EXCLUDE or CHANGE command.

Where the last FIND issued was FIND LAST or FIND PREV, RFIND will perform a FIND PREV operation, otherwise RFIND performs a FIND NEXT operation. i.e. Search forwards (NEXT) or backwards (PREV) from the current cursor location to find the next or previous occurrence of the string respectively.

If the cursor is not within the window's data display area, the forwards or backwards search begins at the first position of the first visible or excluded record within the display area.

RFIND is assigned to function key **F5** by default.

**See Also:**

**CHANGE EXCLUDE FIND RCHANGE**

---

## RIGHT

---

**Environments:**

RIGHT primary command exists in the following application environments.

Text Edit (ISPF)	Text Editor with INTERFACE=ISPF (default for z/OS).
Text Edit (XEDIT)	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
Data Edit	SDE Data Editor.
System	FileKit base windows system.

---

## ISPF Interface

---

If **INTERFACE=XEDIT** is in effect, the ISPF version may be invoked by prefixing the command with **ICOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

**ISPF Syntax:**

```
>>- RIGHT -----><
|
+-- Cursor -----+
+-- CSR -----+
|
+-- Data -----+
|
+-- Half -----+
|
+-- Max -----+
|
+-- Page -----+
|
+-- n_cols -----+
```



**Description:**

Scroll the view of the data within the text editor window right towards the last column of edited text.

RIGHT is assigned to function key **F11** by default. Note that any text entered at the command prompt when a PFKey is pressed will be treated as parameter input to the command associated with the PFKey (e.g. If MAX is at the command prompt when F11 is pressed, RIGHT MAX will be executed.) Where no parameter is specified, the scroll amount will be the value specified in the **Scroll>** field in the top right corner of the window display.

The last column of text, as identified by the maximum record length of the data (Recl), becomes the last column displayed when an attempt is made to scroll the display right beyond this column of text.

**Parameters:**

CURSOR  
CSR

The column of text on which the cursor is positioned (i.e. the **focus column**) becomes the first column of the scrolled display. If the cursor is positioned outside the display area or on the first column of text in the current display, then RIGHT PAGE is executed instead.

DATA

Scroll right to display the page (i.e. the display window width) less one column of text to the right of the last column in the current display.  
i.e. The last column of text in the current display becomes the first column of the scrolled display.

HALF

Scroll right half a page of data.  
The column of text that is half way across the page of data in the current display becomes the first column of the scrolled display.

MAX

Scroll right so that the last text column of the edited data (Recl) becomes the last column in the scrolled display.

PAGE

Scroll right to display the page (i.e. the display window width) of text to the right of the last column in the current display.  
i.e. The column of text to the right of the last column in the current display becomes the first column of the scrolled display.

*n\_cols*

Scroll right a specified number of columns.  
The column of text that is *n\_cols* columns to the right of the first column in the current display becomes the first column of the scrolled display.

**See Also:**

**BOTTOM DOWN LEFT TOP UP**

**XEDIT Interface**

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

**XEDIT Syntax:**

```

      +- 1 ----+
      |         |
>>-- R ight -+-----+-----><
      |         |
      +- n ----+
      |         |
      +- HALF --+
  
```

**Description:**

Position the focus column one or more columns to the right of the current focus column. Alternatively, position the focus column to the right of the current focus column, for a number of columns equal to one half the width of the edit view.

Where the specified number is greater than the number of columns to the right of the focus column (i.e. beyond the LRECL value for the file) then the last column becomes the focus column.

**Parameters:**

*n*

The number of columns to the right of the current focus column at which the new focus column is to be set. If omitted this parameter defaults to 1.

**HALF**  
Position the focus column a number of columns equal to half the width of the edit view, from the current focus column position.

**Example:**

`right`  
Set the focus column to be 1 column to the right of the current focus column.

`ri10`  
Set the focus column to be 10 columns to the right of the current focus column.

**See Also:**

[LEFT](#)

## RUNSELCOPY

**Syntax:**

```

+--- -PGM SELCOPY ---+
|                       |
>>-- RUNSELCopy -----+-----><
|                       | |
+--- -PGM progname --+  +--- parmstring -----+
    
```

**Description:**

RUNSELCOPY will execute the SELCOPY program using SYSIN control statements taken from the contents of the focus edit view. The statements in this edit view do not necessarily need to have been saved to disk.

SYSPRINT output is captured and displayed in a separate edit view which is assigned a temporary DSN of RECFM=VBA and LRECL=1024, and is not saved to disk.

**Parameters:**

`-PGM progname`  
Identifies the load module name, *progname*, to be used to execute the input control statements. Default for *progname* is SELCOPY.

`parmstring`  
Specified last on the command input, *parmstring* specifies a parameter string to be passed, without changes, to the SELCOPY program.

## RUNSLC

**Syntax:**

```

+--- -PGM SLC -----+
|                       |
>>-- RUNSLC -----+-----><
|                       | |
+--- -PGM progname --+  +--- parmstring -----+
    
```

**Description:**

RUNSLC will execute the SLC program (the C++ source version of SELCOPY) using SYSIN control statements taken from the contents of the focus edit view. The statements in this edit view do not necessarily need to have been saved to disk.

SYSPRINT output is captured and displayed in a separate edit view which is assigned a temporary DSN of RECFM=VBA and LRECL=1024, and is not saved to disk.

**Parameters:**

-PGM *programe*

Identifies the load module name, *programe*, to be used to execute the input control statements. Default for *programe* is SLC.

*parmstring*

Specified last on the command input, *parmstring* specifies a parameter string to be passed without change to the SLC program.

## SAVE, SSAVE

### Environments:

SAVE primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor. (See <b>SAVE</b> and <b>SAVEAS</b> )

### Syntax:

```
>>-- SAVE -----><
      |         |         |         |
      +-- fileid ---+   +- ( ---+ NEWGEN ---+
                        |         |
                        +- NOGEN ---+

>>-- Ssave -----><
      |         |         |         |
      +-- fileid ---+   +- ( ---+ NEWGEN ---+
                        |         |
                        +- NOGEN ---+
```

### Description:

Save the current edited file to disk with an associated fileid.

If a *fileid* is specified, the fileid in the title bar of the CBL edit view is updated and the data is saved under the new fileid. File data that exists on disk under the original fileid is unchanged. If *fileid* is a member belonging to a PDSE library which supports member generations, the option NOGEN is redundant as a new member generation will always be created. SAVE with *fileid* is equivalent to performing the "Save As" item from the File drop down menu.

If *fileid* is not specified, SAVE attempts to write the file to disk using the currently assigned fileid. By default, this fileid is that which was used to initially edit the file, unless subsequently updated by a SET FILEID, FNAME, FTYPE, FMODE, FPATH command. This is equivalent to performing the "Save" item from the File drop down menu.

If the fileid to be used by SAVE does not already exist, a new file will be created. For MVS Sequential, PDS(E) and VSAM data sets, this will open the Allocate NonVSAM or Define VSAM dialog window prompting the user to provide the required new data set characteristics.

SAVE will fail and return an error if either of the following conditions are true:

- The fileid used to save the data differs from that used on the initial edit of the file and this fileid already exists for a file on disk.
- For HFS files, the current "Modified" timestamp of the file is later than the time at which the file was last saved, or else first edited, in the current CBL session. i.e. the file's data has been changed by some other process or user edit.

SSAVE will save the file regardless of the above error conditions.

Where *fileid* is identified as being an HFS path (i.e. begins with "." or contains "/"), DSORG is set to be HFS and the file is saved with the permission mode 740 and the following record delimitation:

- If the current RECFM is F, the HFS file will contain no EOL delimiters and each record will be padded to the current LRECL value.
- For RECFM V or U, HFS file records will be delimited with EOL (end-of-line) characters as defined by the current setting of **EOLOUT**.

### Parameters:

*fileid*

The fileid to be assigned to the file on writing it to disk.

For MVS data sets, *fileid* may be the DSN of a Sequential or VSAM data set, the DSN and member name of a PDS(E) library member or an HFS absolute or relative path name.

For VSE files, *fileid* may be the full LIBR member id, lib.sublib.mname.mtype.

For CMS files, *fileid* may be the full CMS file id, FNAME FTYPE FMODE (or FNAME.FTYPE.FMODE) If only two qualifiers are specified, FMODE defaults to the current FMODE, and if only one qualifier is specified, then FTYPE and FMODE default to the current FTYPE and FMODE.

NEWGEN

Applicable only if *fileid* is not specified and output is to a version 2 PDSE library member that supports member generations. Data will be saved to a new primary member generation. This option overrides the default set by the **GENSAVE** option.

NOGEN

Applicable only if *fileid* is not specified and output is to a version 2 PDSE library member that supports member generations. Data will be saved back to the same member generation referenced in the focus edit view. This option overrides the default set by the **GENSAVE** option.

**See Also:**

**FILE** and SET/QUERY/EXTRACT Options: **FILEID GENSAVE**

## SDATA

**Syntax:**

```
>>-- SData -- sde_command -----><
```

**Description:**

Direct a command to the FileKit Structured Data Environment ( **SDE** ).

The SDATA command allows SDE primary commands to be issued from a CBLe text-edit window, typically using F16 to point-and-shoot at commands stored in a command-centre (CMX) file, such as your HOME-file.

**Parameters:**

*sde\_command*  
Any **SDE** command.

**Examples:**

```
<sdata create structure      CBL.FILEKIT.STRUCT (COMPSTR)  \
    from cobol CBL.COPYBOOK.COBOL (COMPDEF)

<sd edit  CBL.SDE.EMP  using CBL.FILEKIT.STRUCT (COMPSTR)
<sd select Key,InvNumb,DeliveryDate  from Orders  in CBL.FILEKIT.STRUCT (COMPSTR)
```

## SET

SET primary command exists in the following application environments.

<b>Text Edit</b>	Text Editor (both XEDIT and ISPF interfaces.)
<b>Data Edit</b>	SDE Data Editor.

**Description:**

See **SET/QUERY/EXTRACT** Options.

# SETPT

## Syntax:

```

+---- 70 ----+ +--- 132 ---+
|             |             |
>>-- SETPT ---+-----+-----+-----+-----><
|             |             |             |
+- start_col -+ +- end_col -+ +- Q -+
|             |             |             |
+- V -+
    
```

## Description:

A single execution of SETPT assigns a label name to multiple lines within a file based on the line data found between the specified column boundary limits. Compare with SET POINT, which unconditionally sets an individual label name on the focus line, or setting a label by simply typing the label name in a line's prefix area.

If a blank delimited word of no more than 9 characters beginning with "." (dot/period) **starts** within the column limits defined by *start\_col* and *end\_col*, then a label name equal to the word string is assigned to that line. Multiple labels will be assigned to the same line if more than one word matching this criteria is found within the column limits.

Note that a word within the data that extends beyond the *end\_col* column limit may still be eligible as a label name so long as the preceding "." is within the column limits.

If a single label is set, then the assigned label name is displayed in the line's prefix area. Multiple labels assigned to the same line are not displayed in the line's prefix area.

The same label name may not be assigned to more than one line in the current file. Therefore, the last occurrence of a label name within the file will be assigned by SETPT. All previous occurrences of the same label name will be unassigned.

Like SET POINT, SETPT takes effect at the File level.

## Parameters:

*start\_col*  
 Defines the left column limit in a range of columns in which to search for blank delimited, "." prefixed words.  
 Default is column 70.

*end\_col*  
 Defines the right column limit in a range of columns in which to search for blank delimited, "." prefixed words.  
 Default is column 132.

Q  
 Suppress the default report message, "n points have been set by SETPT."

V  
 List the label names that have been set by SETPT before reporting the total number of points set by SETPT.

## Examples:

SETPT 56 63 V  
 For the following lines of data will set the label ".SMS" on line 238 and label names ".RACF" and ".SEC" on line 239. These label names will be included in the list of label names that have been set on completion of the SETPT command.

```

|...+...1...+...2...+...3...+...4...+...5...+...6...+...
000238 <edit NBJ.CONFIG.CMX(SMS) | ** SMS Configuration ** .SMS
000239 <edit NBJ.CONFIG.CMX(RACF) | ** RACF Configuration ** .RACF .SEC .X
    
```

# SHIFT

## Environments:

SHIFT primary command exists in the following application environments.

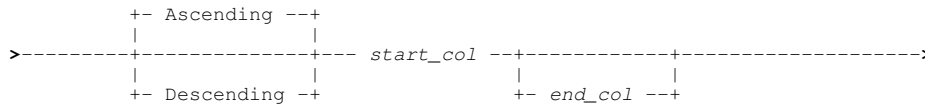
Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor.

## Syntax:

```

+- 1 -+ +----- 1 -----+
|     | |             |
>>-- SHift ---+-----+-----+-----+-----><
|     | |             |
|     | |             |
    
```



**Sort Field:****Description:**

**SORT** primary command sorts lines of text based on the contents of one or more key fields.

Only text between the left and right boundary (zone) columns is sorted. Text belonging to sorted lines that is located before the left boundary column and following the right boundary column is unchanged. An error occurs if any of the specified sort key fields include text columns that fall outside the boundary columns.

Sorting may be restricted to operate on a range of lines that fall between two labelled text edit lines. If no range is specified, an implicit range of `.ZFIRST` `.ZLAST` is used (i.e. all lines of text in the edit view.) If only the first line of a range is specified (`.name1`) then the default last line is `.ZLAST`.

Where **BLOCK** is specified, the current boundary columns are temporarily updated to be the left and right columns of the marked block. For line blocks, this means that the left boundary column is 1 and the right boundary column is the maximum record length (Recl). Similarly, the range of lines affected by the sort operation is defined by the first and last lines of the marked block.

Multiple, non-overlapping key fields, defined by start and end column numbers (`start_col` and `end_col`), may be specified each with their own sort order (ascending or descending.) The order in which the key fields occur define their order of precedence in the sort operation. Sorting is initially performed for the first key field, then, for lines containing matching key values, by the second key field, etc.

If no key fields are specified, sorting is performed on an implied key field starting at the left boundary column and ending at the right boundary column, with text sorted in ascending order.

**Parameters:**

`.name1`  
A label name identifying the first line in the range of text edit lines to be included in the sort operation. The preceding "." (dot) in `.name1` is mandatory.

`.name2`  
A label name identifying the last line in the range of text edit lines to be included in the sort operation. The preceding "." (dot) in `.name1` is mandatory.

If `.name2` references a line number which is lower than that referenced by `.name1`, then the order is reversed to define a positive number of lines. Default for `.name2` is `.ZLAST`.

**BLOCK**  
A marked line or box block which defines both the boundary columns and range of text edit lines to be included in the sort operation.

Before **SORT BLOCK** can be used, text must first be marked within the display of the edit view using line (prefix area) commands **ML** or **MB**; or using `<F17>` or `<F18>` which are, by default, assigned to **MARK BOX** and **MARK LINE** respectively. An error message is displayed if no marked block exists.

**EX | X | NX**  
**EX** or **X** indicate that only excluded text edit lines that fall within the explicit or implicit range of lines are to be sorted. **NX** indicates that only non-excluded text edit lines that fall within this range are to be sorted. The line numbers of excluded or non-excluded lines that do not form part of the sort operation are unchanged.

Default is to sort both excluded and non-excluded lines.

**ASCENDING | DESCENDING**  
Specifies the sort order of character text in the sort key field definition that follows.

Ascending will arrange the lines so that the lowest key field values are displayed first and the highest last. Descending will arrange the lines so that the highest key field values are displayed first and the lowest last.

Default is **ASCENDING**.

`start_col`  
The column number of the first column of a sort key field.

The specified column number must be within the left and right boundary columns and must not be a column within an already specified key field.

`end_col`  
The column number of the last column of a sort key field.

The specified column number must be within the left and right boundary columns and must not define a key field that overlaps an already specified key field.

If *end\_col* references a column number which is lower than that referenced by *start\_col*, then the order is reversed to define a positive number of columns. Default for *end\_col* is the right boundary column.

### Examples:

```
sort .S1 .S2 nx asc 1 10
```

Sort a range of non-excluded text lines starting at line .S1 and ending at .S2 (inclusively) in ascending order of key text defined as starting at column 1, ending at column 10. The line numbers of excluded text lines that fall within this range are unchanged.

```
sort block a 5 15 d 18 20
```

Sort lines of text marked by a block in ascending order of key text at columns 5 through 10, and, where key text matches, in descending order of key text at columns 18 through 20.

### See Also:

**BOUNDS**

## XEDIT Interface

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

### XEDIT Syntax:

```
>>-- SORT -- group-target -----><
|
| +-----+
| | (1)
| | +- Ascending +-
| v |
| +-----+ coll -----+
| +- Descending +         +- col2 -+
|
```

### Notes:

1. Default for all sort fields until Descending is specified, in which case the default is reversed until Ascending is specified.

### Description:

Sort lines in the target area specified by *group-target* in ascending or descending order.

Characters between the specified start and end columns are compared against those characters in equivalent columns on all lines in the target area.

The SORT command processes all lines in the target area, regardless of line selection level.

Were STAY is set ON, the focus line is unchanged by the SORT command. Otherwise, the line that becomes the first line in the target area following the SORT command, also becomes the focus line.

### Parameters:

*group-target*

**Group-target** condition defining the end of the source target area. If the group-target is not satisfied then the SORT command will fail.

ASCENDING  
DESCENDING

Sort the field that follows in ascending or descending order. ASCENDING is default.

The specified sort order prevails for all sort fields that follow until otherwise specified.

*coll col2..*

Pair of column numbers defining the leftmost and rightmost columns of a sort field. Any number of sort field column pairs may be specified. Precedence is given to those sort fields specified first in the list.

"" (asterisk) may be specified for *col2* indicating that the current right zone column should be used.

Where *col2* is not specified for the last sort field column pair, then "" (asterisk) is implied.

Where *col1* and *col2* are not specified, the current ZONE columns are used. If, however, the target area is a box block, the left and right boundaries of the block are used.



**Example:**

```
sort 10 10 20
Sort the focus line and the following 9 lines in ascending order based on the single sort field specified as columns 10 to 20.
```

```
sort :20
Sort the focus line and all lines up to, but not including, line 20 in ascending order based on the implied sort field defined by the right and left ZONE columns.
```

```
sort /abc/ d 55 58 10 12 a 3 8
Sort the focus line and all lines up to, but not including, the next line below the focus line that contains the string "abc". These lines are sorted firstly in descending order based on sort field specified as columns 55 to 58, in descending order based on sort field specified as columns 50 to 12, and finally in ascending order based on sort field specified as columns 3 to 8.
```

## SOS

**Syntax:**

```
>>-- SOS --- action -----><
```

**Description:**

Perform Screen Operation Simulation to action cursor placement and text editing. in CBL e macros.

**Parameters:**

```
action
```

Perform the explicit SOS action.

Currently supported SOS actions are:

```
ADDline
LINEAdd
```

Add a blank line following the focus line. The cursor is positioned in column 1 of the new line.

```
DELLine
LINEDel
```

Delete the focus line. The line following becomes the focus line.

```
MAKECURR
```

If the cursor is not on the command line, make the cursor line the current line.

```
REFresh
```

Refresh the 3270 display. Use SOS REFresh in macros which invoke external commands which might alter the 3270 display in ways which cannot be detected by FileKit (such as ISPF commands).

The next time FileKit displays the 3270 screen it will rebuild it completely thus clearing any screen formatting done by the external function.

**See Also:**

**ADD DELETE REFRESH**

## SPLIT

**Environments:**

SPLIT primary command exists in the following application environments.

<b>Text Edit (ISPF)</b>	Text Editor with INTERFACE=ISPF (default for z/OS).
<b>Text Edit (XEDIT)</b>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).

## ISPF Interface

If **INTERFACE=XEDIT** is in effect, the ISPF version may be invoked by prefixing the command with **ICOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

### ISPF Syntax:

```
>>-- SPLIT ---+-----+-----><
          |           |           |
          +- | ISPF SWAP Operands | -+
```

### Description:

When FileKit has been started in an ISPF environment and **INTERFACE=ISPF** is active (the default), the **SPLIT** command and its operands is simply passed to ISPF to split the screen horizontally.

The existence of an ISPF format **SPLIT** command is necessary only to distinguish it from the XEDIT format of the command. In all other FileKit applications and the FileKit system itself, **SPLIT** is an unrecognised command and so it is automatically passed to the controlling environment (i.e. ISPF).

### Parameters:

ISPF SWAP Operands  
Any operand supported by ISPF **SPLIT**.

## XEDIT Interface

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

### XEDIT Syntax:

```
>>-- SPliT ---+-----+-----><
          |           |           |
          +-- ALigned --+
```

### Description:

Split the focus line into two lines starting at the focus column. Text at, and to the right of, the focus column is moved to column 1 of a new line following the focus line.

The focus line pointer and focus column pointer are unchanged following **SPLIT**.

### Parameters:

ALIGNED  
A number of leading blanks, equal to the number of leading blanks in the focus line, are inserted before the text split onto the new line.

### Example:

In the following, focus column is column 18.

```
Command>
<...+...1...+...2|...+...3...+...4...+
    Hello Jane,    Goodbye John

Command> split
<...+...1...+...2|...+...3...+...4...+
    Hello Jane,
Goodbye John
```

Or using parameter **ALIGNED**...

```
Command> split al
<...+...1...+...2|...+...3...+...4...+
```

```
Hello Jane,
Goodbye John
```

**See Also:**

[JOIN](#) [SPLTJOIN](#)

**SPLTJOIN, SJOIN****Syntax:**

```
>>--- SPLITJOIN ---<<
      |             |
      +--- SJoin ----+
```

**Description:**

Perform a SPLIT ALIGNED or JOIN ALIGNED on the focus line.

If there exists text at, or to the right of, the focus column in the focus line, then SPLIT ALIGNED is executed, otherwise JOIN ALIGNED is executed.

**Parameters:**

None.

**See Also:**

[JOIN](#) [SPLTJOIN](#)

**STEMINSERT****Syntax:**

```
>>-- STEMInsert --- rexx_stemvar -----<<
```

**Description:**

For use in CBL e REXX macros, STEMINSERT is a record mass insert command based on a REXX compound (stem) variable.

STEMINSERT determines the number of lines to insert from the *rexx\_stemvar.0* value and inserts new lines with text obtained from the value *rexx\_stemvar.n\_line*, where *n\_line* is the insert line index (*n\_line*=1,2,3,...,*rexx\_stemvar.0*). The new lines are inserted following the focus line.

STEMINSERT is a fast method of insert and should be used in place of the following REXX syntax:

```
do i = 1 to linetext.0
  'insert' linetext.i
end
```

Replace this with:

```
'steminsert' linetext
```

**Parameters:**

*rexx\_stemvar*

The stem portion of an assigned REXX compound variable.

---

## SUBMIT

---

**Syntax:**

```
>>-- SUBmit --+-----+-----><
          |         |
          +-- fileid --+
```

**Description:**

Submit the specified batch job file to the MVS or VSE batch system. The file should contain a valid Job Card and Job Control.

If fileid is not specified, the file in the current CBLed edit view is submitted.

The ZZSE032I job submitted message is returned containing the job name and job number.

```
ZZSE032I Job xxxxxxxx(JOBnnnnn) submitted.
```

The job name and job number may subsequently be used in the FileKit command **EO** to edit (read only) the job's output listing.

**Parameters:**

*fileid*  
Full fileid of the file to be submitted to batch.

**Example:**

```
submit          Submit the currently edited job to batch.

submit cbl.jcl(scanpds)
Submit MVS job CBL.JCL(SCANPDS) to batch.
```

---

## SYNEX

---

**Syntax:**

```
>>-- SYNEX -- command -----><
```

The SYNEX (SYNONym EXecute) command is for use in CBLed REXX macros.

When SYNONYM ON is in effect, a command entered from a CBLed command line is automatically checked to determine whether it is the name of a defined synonym. If so, the action taken will be that specified by the synonym definition.

By default, CBLed macros do not perform any synonym processing. Prefixing the command with SYNEX when SYNONYM ON is in effect, will force CBLed to check whether the command to be executed has had its behaviour defined via a SET SYNONYM command. If not, the command is executed as normal.

**Parameters:**

*command*  
Any synonym name, CBLed command or macro name followed by its parameters.

**Examples:**

```
synex CC red
Check existence of "CC" as a synonym name. Failing that, treat it as a CBLed command or macro name.
```

**See Also:**

**COMMAND** and the SET/QUERY/EXTRACT Option: **SYNONYM**

---

## SYSCOMMAND, TSO, CMS, DOS

---

**Syntax:**

```
>>--+ SYScommand -+--- command -----><
      |           |
      +- TSO -----+
      |           |
      +- CMS -----+
      +- DOS -----+
```

**Description:**

Pass the command directly to the local CMS or TSO environment for execution.

When a command is issued to CBLLe, the following occurs:

1. If the command is recognised as a CBLLe command it is executed by CBLLe.
2. If the command is not recognised as a CBLLe command and IMPMACRO is set ON, then CBLLe checks for a matching macro name and, if found, executes the macro.
3. If the command is not a macro name or IMPMACRO is set OFF, CBLLe passes the command and control to the CMS or TSO environment.

**Parameters:**

*command*  
Valid CMS or TSO command or expression.

**Example:**

```
cms query dasd
      Pass the command "query dasd" to CMS.
```

**See Also:**

[FILEKIT](#) [MACRO](#)

---

## SYSEDIT

---

**Syntax:**

```
>>-- SYSEdit ---+-----+-----><
              |           |
              +- fileid ---+
```

**Description:**

For CMS or ISPF environments only, open the system text editor, XEDIT or ISPF Edit, to edit the specified file.

SYSEDIT does not support HFS files.

System edit of the file in the current CBLLe edit view may also be achieved by selecting **System Edit** from the **Edit** menu item in the **CBLLe Main Menu Bar**.

If the file is already open in a CBLLe edit view and has alterations, SYSEDIT fails with the following error message:

```
ZZSE150E You must save the changes to this file before using SYSEDIT.
```

Having made changes to the file in the system editor, saving the changes and quitting from that editor will refresh any existing CBLLe edit views displaying the file. The following edit warning message is returned:

```
ZZSE151W This file has been refreshed because you made changes to it in SYSEDIT.
```

**Parameters:**

*fileid*

The full fileid of the file to be edited.  
Default is the current fileid.

**See Also:****EDIT**

---

**TAG**

---

**Syntax:**

```
>>-- TAG ---+-----+----->>
           |         |
           +- line-target -+
```

**Description:**

Tag (set the tag bit on) and highlight all lines matching the specified line-target.

Where line-target is not specified, highlighting is removed and the tag bit is set off for all lines in the file.

**Parameters:***line-target*

**Line-target** condition.

**Example:**

tag /#\*/

Highlight and tag all lines containing the string "#\*".

---

**TFIND**

---

**Syntax:**

```
>>-- TFind ---+-----+----->>
           |         |
           +- string-target -+
```

**Description:**

Locate the first line from the focus line to contain the specified line-target in the 1st position of the current ZONE (BOUNDS) and make this line the new focus line.

If WRAP is set ON and the line-target is not found before bottom-of-file or top-of-file is encountered, then the scan continues from the opposite extreme of the file up to and including the focus line.

Where string-target is not specified, CBL e repeats the last TFIND command issued.

**Parameters:***string-target*

**Line-target** condition.

**Example:**

TF "///} DD"

Find next DD card in MVS JCL, where 'ARBchar ON }' and default 'ZONE 1 \*\*' are in effect.

TFIND ~/ /

Find next line not beginning with 3 blanks.

TF /++/ | /###/

Find next line beginning with either 2 plus or 3 hash signs.

**See Also:**

**LOCATE** and the SET/QUERY/EXTRACT Options: **ARBCHAR WRAP ZONE**

## TFLOW

**Syntax:**

```
>>-- TFlow -----+-----+-----><
                |         |
                +-- col_num --+
```

**Description:**

Flow text in the **focus line** and all text lines that follow until the End of File indicator or a line containing only blank characters between the boundary (zone) columns is encountered.

Text flows between the left boundary (zone) column and the specified column number *col\_num*, which has a default of the right boundary column. Text belonging to lines involved in the TFLOW operation which is located before the left boundary column and following the right boundary column is unchanged.

To flow text, the text editor joins and splits text between the TFLOW columns on consecutive lines as required. Blanks between words are preserved. However, since text is split at blank delimited word boundaries, a split may occur in a column before *col\_num* so inserting additional blank characters at the end of the text. A word that is split from a line is always positioned at the left boundary column of the line that follows.

Similarly, when text is joined to the preceding line, the first word of the joined text will be separated from any existing text by a single blank unless the existing text end with "." (full stop/period) in which case two separating blanks are used.

TFLOW performs the similar functionality to line **prefix area** command "TF(n)".

**Parameters:**

*col\_num* Column number of the right most column margin at which the text will flow. If *col\_num* is not specified or is greater than the right boundary column number, then the right boundary column is used.

**See Also:**

**BOUNDS JOIN SPLIT SPLTJOIN TSPLIT**

## TOP

**Environments:**

TOP primary command exists in the following application environments.

<b>Text Edit (ISPF)</b>	Text Editor with INTERFACE=ISPF (default for z/OS).
<b>Text Edit (XEDIT)</b>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
<b>Data Edit</b>	SDE Data Editor.
<b>System</b>	FileKit base windows system.

## ISPF Interface

If **INTERFACE=XEDIT** is in effect, the ISPF version may be invoked by prefixing the command with **ICOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

**ISPF Syntax:**

>>-- TOP -----><

**Description:**

TOP is an alias for **UP MAX** which scrolls the display of the data upwards to display the first page of edited data.

**See Also:**

**BOTTOM UP**

---

**XEDIT Interface**

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

**XEDIT Syntax:**

>>-- TOP -----><

**Description:**

Make the Top of File line the focus line.

**See Also:**

**BOTTOM**

---

**TSPLIT**

**Syntax:**

```

>>-- TSplit  +-- 1 -----+
             |             |
             +-----+-----><
             |             |
             +-- n_lines --+
    
```

**Description:**

Split the **focus line** at the **focus column** and insert a number of blank lines between the split text.

Text before the focus column and starting at the right boundary (zone) column is unchanged by TSPLIT. Text between the focus column and the right boundary column is moved to the left boundary column of a new line inserted immediately following the focus line with *n\_lines* intervening blank lines.

TSPLIT performs the same functionality as line **prefix area** command "TS(n)".

**Parameters:**

*n\_lines* Number of intervening blank lines to be inserted between the focus line and the new line containing the split text. Default (and minimum) value is 1.

**See Also:**

**BOUNDS JOIN SPLIT SPLITJOIN TFLOW**



## UNDO

### Environments:

UNDO primary command exists in the following application environments.

Text Edit	Text Editor (both XEDIT and ISPF interfaces.)
Data Edit	SDE Data Editor.

### Syntax:

```
>>-- UNDO -----><
```

### Description:

Undo one level of changes made to the current file.

This command is equivalent to selecting "Undo" from the "Edit" menu on the CBL window menu bar.

Multiple levels of changes may be undone by repeated UNDO commands.

The third number following "Alt=" on the status line indicates the number of change levels. If this number is not zero, then changes may be undone using UNDO.

### See Also:

REDO and the SET/QUERY/EXTRACT Options: [ALT UNDO UNDOING](#)

## UP

### Environments:

UP primary command exists in the following application environments.

Text Edit (ISPF)	Text Editor with INTERFACE=ISPF (default for z/OS).
Text Edit (XEDIT)	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
Data Edit	SDE Data Editor.
System	FileKit base windows system.

### ISPF Interface

If **INTERFACE=XEDIT** is in effect, the ISPF version may be invoked by prefixing the command with **ICOMMAND**. See [ISPF/XEDIT Primary Command Precedence](#).

### ISPF Syntax:

```
>>-- UP -----><
      |-----+
      |  Cursor -----+
      |  CSR -----+
      |  Data -----+
      |  Half -----+
      |  Max -----+
      |  Page -----+
      |  n_lines -----+
```

### Description:

Scroll the view of the data within the text editor window upwards towards the top of the displayed text.

UP is assigned to function key **F7** by default. Note that any text entered at the command prompt when a PFKey is pressed will be treated as parameter input to the command associated with the PFKey (e.g. If MAX is at the command prompt when F7 is pressed, UP MAX will be executed.) Where no parameter is specified, the scroll amount will be the value specified in the **Scroll>** field in the top right corner of the window display.

The Top of File indicator becomes the **current line** when the display is scrolled up beyond the first line of text.

### Parameters:

CURSOR  
CSR

The line of text on which the cursor is positioned (i.e. the **focus line**) becomes the last line of the scrolled display. If the cursor is positioned outside the display area or on the last line within the display area, then UP PAGE is executed instead.

DATA

Scroll up to display one page (i.e. the display window depth), less one line of text.  
The first line of text (current line) of the current display area becomes the last line of the scrolled display.

HALF

Scroll up half a page of data.  
The line of text that is half way down the page of data in the current display area becomes the last line of the scrolled display.

MAX

Scroll up to display the first page of data.  
The "Top of File" indicator line becomes the first line of the scrolled display.  
Equivalent to the **TOP** command.

PAGE

Scroll up to display the next whole page of data.  
The text line before the first line of text (current line) in the current display area becomes the last line of the scrolled display.

*n\_lines*

Scroll up a specified number of lines.  
The text line that is *n\_lines* lines above the first line of text (current line) in the current display becomes the new first line (current line) of the scrolled display.

### See Also:

**BOTTOM DOWN LEFT RIGHT TOP**

## XEDIT Interface

---

If **INTERFACE=ISPF** is in effect, the XEDIT version may be invoked by prefixing the command with **ECOMMAND**. See **ISPF/XEDIT Primary Command Precedence**.

### XEDIT Syntax:

```

      +- 1 -----+
      |           |
>>-- Up ---+----->>
      |           |
      +- n_lines -+

```

### Description:

Position the focus line one or more lines above the current focus line.

Where the specified number is greater than the number of lines above the focus line, the Top of File line becomes the focus line.

### Parameters:

*n\_lines*

The number of lines above the current focus line at which the new focus line is to be set. If omitted this parameter defaults to 1.

### Example:

```
up      Set the focus line to be 1 line above the current focus line.
```

```
u10
```

Set the focus line to be 10 lines above the current focus line.

**See Also:**

[DOWN](#)

## UPPERCASE

**Syntax:**

```
>>-- UPPercase -----+-----+-----><
                        |         |
                        +- group-target -+
```

**Description:**

Upper case all alpha characters in the target area.

Only alpha characters that are within the current ZONE columns will be upper cased.

Where BLOCK is specified as the group-target and the target area is a box block, then the ZONE setting is ignored and all alpha characters within the block are upper cased.

The focus line is not changed by a UPPERCASE command.

**Parameters:**

group-target  
**Group-target** condition defining the end of the source target area. If the group-target is not satisfied then the UPPERCASE command will fail.

**Example:**

uppercase 6  
Alpha characters in the focus line and the 5 lines following are upper cased.

upp -8  
Alpha characters in the focus line and the 7 lines preceding are upper cased.

upper /SELC/  
Alpha characters in the focus line and all lines up to, but not including, the first line following the focus line to contain the string "SELC", are upper cased.

**See Also:**

[LOWERCASE](#)

## VIEW

**Environments:**

VIEW primary command exists in the following application environments.

<a href="#">Text Edit</a>	Text Editor. (Interface ISPF and XEDIT)
<a href="#">Text Edit Option</a>	Text Editor view level option.
<a href="#">Data Edit</a>	SDE Data Editor.
<a href="#">System</a>	FileKit base windows system.

**Syntax:**

```
>>--+ View -----+-----+ (plus) -----+><
      |         |         |         |         |
      |         |         |         |         |
```



**Parameters:**

command  
Any command followed by its parameters.

**Example:**

VIgnore Change /%user%/system% 1 \*  
All occurrences of the literal string "%user%" are changed to the string "system" on the focus line.

VIgnore EQU LOADLIB %PREFIX%.LOAD  
Macro EQU sets user environment variable "LOADLIB" to the literal string "%PREFIX%.LOAD".

VRespect LL %LOADLIB%  
List library members belonging to "%PREFIX%.LOAD" where "%PREFIX%" may be set to different values for separate invocations of this LL command.

**See Also:**

[EDITV](#) [VRESPECT](#) [SET ENVVARS](#)

## VRESPECT

**Syntax:**

```
>>-- VRespect -- command -----><
```

**Description:**

When ENVVARS is set OFF, VRESPECT may be used as a prefix to a command string to temporarily set ENVVARS ON for the duration of the command execution.

Note that if the command executed is a CBL REXX macro, then variable translation will be obeyed for all commands within the macro unless they are prefixed with VIGNORE or ENVVARS is explicitly set back OFF.

If ENVVARS is set ON, the VRESPECT prefix has no effect.

**Parameters:**

command  
Any command followed by its parameters.

**Example:**

VRespect Change /%user%/system% 1 \*  
Both variables "user" and "system" are substituted and all occurrences on the focus line of the value substituted for "user" are changed to the the value substituted for "system".

VRespect EQU LOADLIB %PREFIX%.LOAD  
The current value of variable "prefix" is replaces "%PREFIX%" in "%PREFIX%.LOAD" and macro EQU sets user environment variable "LOADLIB" to the resolved string.

**See Also:**

[EDITV](#) [VIGNORE](#) and the SET/QUERY/EXTRACT Option: [ENVVARS](#)

# WINDOW

## Environments:

WINDOW primary command exists in the following application environments.

<b>Text Edit</b>	Text Editor (both XEDIT and ISPF interfaces.)
<b>Data Edit</b>	SDE Data Editor.
<b>System</b>	FileKit base windows system.

## Syntax:

```

+----- + (plus) -----+
>>-- Window -----><
+----- - (minus) -----+
+----- NEXTwindow -----+
+----- PREVwindow -----+
+----- CAScade -----+
+----- TITLE -----+
+----- |             +--- HOr -----+
+----- |             +--- Vert -----+
+----- ARRANGEicons -----+
+----- NEWindow -----+
+----- HEX -----+
+----- |             +--- DOcument ---+
+----- CLOse -----+
+----- |             +--- FRame -----+
+----- |             +--- FILE -----+
+----- MENUmode -----+
+----- |             +--- DOcument ---+
+----- |             +--- FRame -----+
+----- |             +--- FILE -----+
+----- |             +--- Edit -----+
+----- |             +--- ACTions ---+
+----- |             +--- OPTions ---+
+----- |             +--- WINDow ---+
+----- |             +--- Help -----+
+----- |             +--- DOcument ---+
+----- RESTore -----+
+----- |             +--- FRame -----+
+----- MINimise ---+
+----- MINimize ---+
+----- MAXimise ---+
+----- MAXimize ---+
    
```

## Description:

Perform window focusing, positioning and sizing operations on the current edit (document) view or MDI parent (frame) window.

## Parameters:

- + (plus) Place focus on the next MDI child window.
- (minus) Place focus on the previous MDI child window.
- NEXTWINDOW Place focus on the next edit view in the ring.
- PREVWINDOW Place focus on the previous edit view in the ring.

## CASCADE

Cascade all MDI child windows within the MDI parent window.

## TILE

Tile all MDI child windows within the MDI parent window.

HOR	horizontally (default.)
VERT	vertically.

## ARRANGEICONS

Arrange all minimised MDI child windows so that they are lined up along the bottom of the MDI parent display area.

## NEWWINDOW

Open a new edit view for the data in the current edit view.

## HEX

Open a hex view of the focus line.

A hex view is a **storage display window** that contains a hexadecimal and character representation of the line of edited data. The contents of the line may be updated by the user in either of the data representations, simply by overtyping the existing data and hitting <Enter>.

F7 and F8 scroll up and down respectively through the displayed data, whereas F10 and F11 scroll backwards and forwards through the lines of edited data.

## CLOSE

Close the specified window type.

DOCUMENT	Current document window (edit view) (default.)
FRAME	Frame window (MDI parent) and all its child windows.
FILE	All edit views that contain the file currently being edited.

## MENUMODE

Places the cursor at the specified menu bar item and, where possible, opens the drop down menu. If no item is specified the cursor is simply placed at the first item of the menu bar.

DOCUMENT	Document window (edit view) system menu.
FRAME	Frame window (MDI parent) system menu.
FILE	File drop down menu.
EDIT	Edit drop down menu.
ACTIONS	Actions drop down menu.
OPTIONS	Options drop down menu.
WINDOW	Window drop down menu.
HELP	Help item.

## MAXIMISE

## MAXIMIZE

Maximise the current edit view (DOCUMENT), the default, or the MDI parent (FRAME) window.

## MINIMISE

## MINIMIZE

Minimise the current edit view (DOCUMENT), the default, or the MDI parent (FRAME) window.

## RESTORE

Restore the current edit view (DOCUMENT), the default, or the MDI parent (FRAME) window back to its original state, prior to being maximised or minimised.

**Examples:**

`win max`

Maximise the current edit view.

`win new`

Open a new edit view for the data in the current edit view.

`win clo file`

Close all edit views containing the file currently being edited.

**See Also:**

SET/QUERY/EXTRACT Options: **WINPOS** **WINNAME** **WINSIZE**

---

## WINDOWCOMMAND

---

**Syntax:**

```
>>-+- WINDOWCOMMAND -+-- windowname ----- command -----><
  |                   |
  +- WINCMD -----+
```

**Description:**

WINDOWCOMMAND may be used to direct a command to a specifically named window. i.e. as if the command were typed in on the named window's command line.

This especially useful for:

1. Storing commands in your HOME file (to be executed with the ACTION key) that are intended for other windows e.g. a "WHERE" command to select rows based on certain criteria from a ListDataset window.
2. Directing commands executed from a REXX edit macro.

Window names are initially assigned automatically by FileKit, but may be overridden using the **SET WINNAME** command. However, the **SHOWATTR** (SWA) command may be used to determine the window name.

From a REXX edit macro you may use the following command to determine the focus window name.

```
'list SWA // subset /select NAME where FOCUS="Y"/ stem MYSWA columns'
```

This will set the REXX variable "MYSWA.1.NAME" to the window name of the focus window, meaning that commands may be directed to that window using the **WINDOWCOMMAND** (WINCMD) command. e.g.

```
'wincmd' MYSWA.1.NAME 'where LRECL=80 and RECFM="F"'
```

For an example of its usage please see the distributed REXX macro "LISTSELD" (type "EM LISTSELD" to edit the macro) which provides a simplistic "dialog" interface for selecting and ordering the columns visible in such windows as Dataset, Library and VTOC lists.

**Parameters:**

*windowname*  
The name of the window to which the command will be directed,

*command*  
Any command followed by its parameters.

**Example:**

```
WinCmd LISTFILE1 mw x=3 y=26
WinCmd LISTFILE2 szw w=72 d=31
WinCmd LISTFILE3 select entry,org,lrecl,recfm,blkosz,referenced,vol,pdse,created
```

**See Also:**

**FILEKIT SYSCOMMAND**



# SET/QUERY/EXTRACT Options

## Syntax:

```

>>+-----+----- option_name ----- value -----><
  |         |         |
  +- SET -----+

>>--- Query ----- option_name -----><

          +-----+
          |         |
          v         |
>>--- EXtract ---+--- /option_name ---+--- / -----><

```

## Description:

Text Editor Environment options may set, and their current values queried or extracted into stem-variables for use in REXX macros, using the SET, QUERY and EXTRACT commands respectively. Additional operands supported by QUERY and EXTRACT, allow the user to obtain information about the working environment and its current status.

Options for a new text editor document view are initialised as follows:

1. Via SET commands executed by the text edit **profile REXX macro** (default name PROFILE) or any macro that it may invoke. Note that if use of a profile macro is not suppressed on the EDIT or VIEW operation, the macro will be executed by default on open of a new text edit document window for a fileid that is not already open for text or data edit.
2. If the new document view displays the same data as the current text edit document view, a profile macro is **not** executed and the options set in the current text edit document view are inherited by the new view. Note that opening a new text edit view of the same data is achieved using the **WINDOW NEW** primary command or on selecting "New Window" from the drop down menu displayed by the "Window" menu bar item.
3. If an option value is not initialised via one of these sources, it will be set to a value saved in the FileKit User INI file and which was established on startup of FileKit. The INI file contains the values of most text edit options as they were last assigned in the previous FileKit session. (See the **SAVEOPTIONS** option which is set on by default.)

Thereafter, options may be set via the SET primary command executed at the text edit command prompt, via the ACTION facility, from within a text edit macro or via the Options item of the menu bar.

## Option Scope:

Each text editor option is designed to take effect at one of the following different levels of operation.

Level	Scope
Global	The option affects all text editor document views.
File	The option affects all text editor document views that display the same data and so may be set differently for text editor document views that display different data.
View	The option affects only the current text editor document view and so may be set differently for each text editor document view. Other text editor views that display the same data are not affected.

## Parameters:

*option\_name*

The name, synonym or abbreviated name of a valid text editor environment option. See **Option List** below for valid *option\_name* options.

Multiple option values maybe extracted in a single EXTRACT command execution by separating each *option\_name* with a blank or "/" (forward slash).

*value*

For SET, the new value(s) to be assigned for *option\_name*.

**Option List:**

ACTION	EOLIN	LCOLOUR	SAVEOPTIONS
ACTIONCOMMENT	EOLOUT	LENGTH	SCALE
ACTIONCURSOR	FIDCHANGED	LINE	SCOLOR
ACTIONDELIM	FILEID	LINEFLAG	SCOLOUR
ALT	FLSCREEN	LINEND	SCOPE
ARBCHAR	FMODE	LISTFILEACTION	SCREEN
AUTOSAVE	FNAME	LOADWARNING	SELECT
BEEP	FPATH	LRECL	SHADOW
BLOCK	FTYPE	LSCREEN	SIZE
CASE	GENSAVE	MACRO	SIZEWARNING
CHANGE	HCOLOR	MACROPATH	STAY
CLIPBOARD	HCOLOUR	MBR	STREAM
CMDDEF	HEXSTRING	MSGLINE	SYNONYM
CMDLINE	HILITE	MSGMODE	THIGHLIGHT
COLOR	HILIGHT	NBWINDOW	UNDO
COLOUR	HSCROLLCURSOR	OPSYS	UNDOING
COLUMN	IMPMACRO	PFKEY	USERNAME
COUNT	INIFILE	POINT	VARBLANK
CURLINE	INIVAR	PREFIX	VERSION
CURSOR	INSTANCE	PSCOPE	VIEW
DEFPROFILE	INTERFACE	RANGE	WINNAME
DISPLAY	ISPFMODE	RECFM	WINPOS
DSN	KEY	REDO	WINSIZE
DSORG	LASTMSG	RESERVED	WRAP
ENVVARS	LCOLOR	RING	ZONE

---

## ACTION - QUERY/EXTRACT

---

**Syntax:**

```
>>--- Query -----+- ACTION -----+----->>
                |         |
                +- CMDTEXT -----+
```

```
>>--- EXTRACT --- / +- ACTION -----+ / ----->>
                |         |
                +- CMDTEXT -----+
```

**Description:**

QUERY and EXTRACT ACTION reports the command that would be executed by the ACTION facility based on the current cursor position. i.e. The command text at the **focus line** and **focus column** location within the displayed data.

**QUERY Response:**

The command text at the focus line and column as interpreted by the ACTION facility.

**EXTRACT Rexx variables:**

action.0 (cmdtext.0)	3
action.1 cmdtext.1	The text of the command at the focus line and column that would be executed or placed on the command line had the <b>ACTION</b> (CMDTEXT) command been issued.
action.2 cmdtext.2	'<' indicating that the command should be executed immediately, or '>' indicating that the command should be placed on the command-line.
action.3 cmdtext.3	The placement position of the cursor within the resulting command had it been put on the command line, as defined by the presence of the 1st '_' (underscore) character in the original source field. Note that the first underscore is removed from the resulting command. If no underscore is present then action.3 will be 0.

**See Also:**

SET/QUERY/EXTRACT Options: **ACTIONCOMMENT ACTIONCURSOR ACTIONDELIM**

---

## ACTIONCOMMENT - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+-- ACTIONComment ---- char_string -----><
  |         |
  +- SET ----+

>>--- Query ----- ACTIONComment -----><

>>--- EXtract --- /ACTIONComment/ -----><
```

**Description:**

Identifies the character (or string of characters) to represent the start of comment data in a line of text processed by the **ACTION** facility.

The ACTIONCOMMENT option corresponds to the ACTION Key Options Comment value set in the **Text Edit Settings (=0.3)** panel.

SET ACTIONCOMMENT takes effect at the Global level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**SET Value:**

*char\_string*  
A character string, with a maximum length of 4, which defines the start of comment data when processed by the ACTION utility.

**QUERY Response:**

The current setting of the ACTIONCOMMENT option.

**EXTRACT REXX variables:**

actioncomment.0	1
actioncomment.1	A 1-4 character string representing the start of comment data in a line of text processed by the ACTION facility.

**See Also:**

SET/QUERY/EXTRACT Options: **ACTION ACTIONCURSOR ACTIONDELIM**

---

## ACTIONCURSOR - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+-- ACTIONCURSOR ---+-- ON --+-----><
  |         |                 |   |
  +- SET ----+               +-- OFF -+

>>--- Query ----- ACTIONCURSOR -----><

>>--- EXtract --- /ACTIONCURSOR/ -----><
```

**Description:**

Controls the significance of the first '\_' (underscore) as a special character in a line of text processed by the **ACTION** facility. If set on, the first '\_' is excluded from the command string and identifies the location at which the cursor is to be positioned if the command string is placed at the command prompt.

The ACTIONCURSOR option corresponds to the ACTION Key Options Place Cursor value set in the **Text Edit Settings (=0.3)** panel.

SET ACTIONCURSOR takes effect at the Global level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

ON | OFF

Set ACTION facility cursor positioning on or off.

**QUERY Response:**

The current setting of the ACTIONCURSOR option.

**EXTRACT Rexx variables:**

actioncursor.0	1
actioncursor.1	ON   OFF

**See Also:**SET/QUERY/EXTRACT Options: [ACTION](#) [ACTIONDELIM](#) [ACTIONCOMMENT](#)

---

**ACTIONDELIM - SET/QUERY/EXTRACT**

---

**Syntax:**

```
>>+-----+--- ACTIONDELIM ----+--- ON --+----->>
  |         |         |         |         |
  +- SET ----+         +- OFF --+
>>--- Query ----- ACTIONDELIM ----->>
>>--- EXtract --- /ACTIONDELIM/ ----->>
```

**Description:**

Controls the significance of the '|' (OR symbol) as a special character in a line of text processed by the [ACTION](#) facility. If set on, '|' is excluded from the command string and is treated as a command string delimiter.

If '||' (2 consecutive OR symbols) follow '<' or '>' at the start of the line of text then the setting of ACTIONDELIM is temporarily reversed for that line of text only. i.e. If ACTIONDELIM ON is set, occurrences of '|' will not partition the line of text but will be treated as part of the executable command text. Likewise, if ACTIONDELIM OFF is set, occurrences of '|' will partition the line of text.

The ACTIONDELIM option corresponds to the ACTION Key Options Multiple value set in the [Text Edit Settings \(=0.3\)](#) panel.

SET ACTIONDELIM takes effect at the Global level and and its setting is saved if [SAVEOPTIONS](#) ON is in effect.

**Set Options:**

ON | OFF

Set ACTION facility command string delimitation on or off.

**QUERY Response:**

The current setting of the ACTIONDELIM option.

**EXTRACT Rexx variables:**

actiondelim.0	1
actiondelim.1	ON   OFF

**See Also:**SET/QUERY/EXTRACT Options: [ACTION](#) [ACTIONCURSOR](#) [ACTIONCOMMENT](#)

## ALT - SET/QUERY/EXTRACT

### Syntax:

```
>>+-----+ ALT --- n1 -----><
|         |         |         |
+- SET ----+         +- n2 --+

>>--- Query ----- ALT -----><

>>--- EXtract --- /ALT/ -----><
```

### Description:

Update the CBL or SDE edit window alteration count which is reported on the status line.

SET ALT takes effect at the File level.

"Alt=*n1,n2;x*" on the status line indicates the number of alterations made to the file since it was last saved or autosaved, the number of alterations made to the file since it was last saved (regardless of intervening autosaves) and the number of change levels that may be undone using UNDO. The "\*" (asterisk) indicates that change levels may be re-applied with REDO.

**Note:** AUTOSAVE is not currently supported and so, unless values are set specifically using SET ALT, *n1* and *n2* will always be the same.

### Set Options:

*n1*            Number of changes since last save or autosave.  
*n2*            Number of changes since last save.

### QUERY Response:

The current values for *n1* and *n2*. Values which are also displayed in the status line.

### EXTRACT REXX variables:

alt.0	2
alt.1	Number of alterations since last AUTOSAVE or SAVE.
alt.2	Number of alterations since last SAVE.

### Example:

```
set alt 10
Set alteration count since last SAVE or AUTOSAVE to 10.
```

## ARBCHAR - SET/QUERY/EXTRACT

### Syntax:

```
>>+-----+ ARBchar --- ON -----><
|         |         |         |         |         |
+- SET ----+         +- OFF -+ +- c1 -+ +- c2 -+

>>--- Query ----- ARBchar -----><

>>--- EXtract --- /ARBchar/ -----><
```

### Description:

Activate and optionally allocate arbitrary characters (ARBCHAR characters). ARBCHAR characters are treated as wildcard characters when used in strings for **line-target**, **column-target** and the **CHANGE** command.

SET ARBCHAR takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

ON | OFF  
Set ACTION facility cursor positioning on or off.

c1  
ARBCHAR character representing zero or more characters in a string. Default is "\$" (dollar).

c2  
ARBCHAR character representing a single character in a string. Default is "?" (Question Mark).

**QUERY Response:**

Displays the current setting of ARBCHAR (ON or OFF) followed by the first and second ARBCHAR characters.

**EXTRACT REXX variables:**

arbchar.0	3
arbchar.1	ON   OFF
arbchar.2	First ARBCHAR character.
arbchar.3	Second ARBCHAR character.

**Example:**

arbch on % #  
Activate ARBCHARs "%" (percent) and "#" (hash). The command CLOCATE /1%2#3/ would be successful for:

12a3 1112a3 12222b3 133334442h3

However, it would fail for:

123 1112ab3

arbch off  
Deactivate use of ARBCHARs.

---

## AUTOSAVE - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+ AUTOSave ---+ ON -----+<<
|         |           |         |         |
+- SET ----+         +- PROMPT ----+ |
|         |           |         |         |
+--- OFF ---+         +- NOPROMPT ---+ |
|         |           |         |         |

>>--- Query ----- AUTOSave -----<<

>>--- EXtract --- /AUTOSave/ -----<<
```

**Description:**

This option controls the action taken if unsaved changes exist in edited data and the **END** (or **QUIT**) command is executed.

The following table identifies the action taken for each value of AUTOSAVE when END is executed in an edit view for which changes exist. (i.e. When the alteration count is not zero or the fileid has been updated.)

AUTOSAVE	Action Taken
ON	All edited text is automatically saved to the fileid associated with the text edit view without prompting the user.
OFF PROMPT	The END command is terminated with the following message:  ZZSD465I Data Changed - Use SAVE/CANCEL (AUTOSAVE OFF PROMPT is set).
OFF NOPROMPT	The CBLEDIT Close popup window is displayed prompting the user to reply "Yes" to save the text, "No" to close without save and so discard the changes or "Cancel" to terminate the END command and return to the edit view.

SET AUTOSAVE takes effect at the Global level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**SET Value:**

ON | OFF

Controls whether changes to data will be automatically saved (ON) or not (OFF).

PROMPT | NOPROMPT

For AUTOSAVE OFF only, controls whether the message prompt (PROMPT) or the CBLEDIT Close popup window (NOPROMPT) will be displayed if changes exist when END is executed.

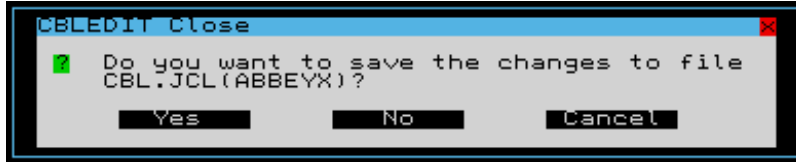


Figure 18. CBLEDIT Close Window.

**QUERY Response:**

The current setting of the AUTOSAVE option, ON, OFF PROMPT or OFF NOPROMPT.

**EXTRACT Rexx variables:**

autosave.0	1 if AUTOSAVE is ON. 2 if AUTOSAVE is OFF.
autosave.1	The current setting of automatic SAVE. <b>ON</b> or <b>OFF</b> .
autosave.2	Only for AUTOSAVE OFF, the current setting of the prompted message, <b>PROMPT</b> or <b>NOPROMPT</b> .

---

## BEEP - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+ BEEP -----+ ON ---+-----><
  |         |             |         |
+- SET -----+         +- OFF -+
>>--- Query ----- BEEP -----><
>>--- EXtract --- /BEEP/ -----><
```

**Description:**

Determines whether the speaker beeps when an error message is returned.

SET BEEP takes effect at the Global level.

**Example:**

beep on  
Beep on all subsequent error messages.

**Set Options:**

ON | OFF

Set ACTION facility cursor positioning on or off.

**QUERY Response:**

Displays the current setting of BEEP (ON or OFF).

**EXTRACT Rexx variables:**

beep.0	1
beep.1	ON   OFF

## BLOCK - QUERY/EXTRACT

**Syntax:**

```
>>--- Query ----- BLOCK -----><
>>--- EXtract --- /BLOCK/ -----><
```

**Description:**

QUERY and EXTRACT BLOCK reports the attributes of the current marked block.

**QUERY Response:**

For a marked block, displays the block type, first marked line number, first marked column number, last marked line number, last marked column number, fileid containing the marked block and the string PERSISTANT. If no marked block exists, NONE is displayed.

**EXTRACT Rexx variables:**

block.0	8
block.1	LINE   BOX   NONE
block.2	Line number of start of block. (Null if block.1=NONE).
block.3	Column number of start of block. (Null if block.1=NONE).
block.4	Line number of end of block. (Null if block.1=NONE).
block.5	Column number of end of block. (Null if block.1=NONE).
block.6	Fileid of file containing marked block. (Null if block.1=NONE).
block.7	PERSISTENT (Null if block.1=NONE).
block.8	Contents of a block that spans 1 line only. (Null if block.1=NONE or marked block spans multiple lines).

**See Also:**

**MARK**

## CASE - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+--- CASE ---+ Mixed +-----+-----><
  |         |         |         |         |         |
+- SET -----+         +- Upper +- Respect +- Respect +-
                                     +- Ignore --+ +- Ignore --+

>>--- Query ----- CASE -----><
>>--- EXtract --- /CASE/ -----><
```

**Description:**

Determine the way in which the case of character strings are interpreted.

SET CASE takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.



**Set Options:**

MIXED  
UPPER

For all new text typed in the CBL window view, alpha characters appear in upper or lower case (MIXED), as controlled via the user's keyboard, or entirely in upper case (UPPER). Default is MIXED.

RESPECT  
IGNORE

For string target searches and SORT fields, either IGNORE or RESPECT the case of each alpha character in the search string or sort field. Default is IGNORE.

RESPECT  
IGNORE

For the CHANGE command, either IGNORE or RESPECT the case of each alpha character in string1. i.e. with IGNORE in effect, all strings that match string1, regardless of capitalisation, will be changed to string2. Default is RESPECT.

**QUERY Response:**

Displays the case in which text is entered (MIXED or UPPER case), the case setting for string searches (RESPECT or IGNORE case) and the case setting for comparisons made during the CHANGE command (RESPECT or IGNORE case.)

**EXTRACT Rexx variables:**

case.0	3
case.1	MIXED   UPPER
case.2	RESPECT   IGNORE
case.3	RESPECT   IGNORE

**Example:**

case u i i

New text appears in upper case, case is ignored for string searches and for the CHANGE command string1 parameter.

◇ INSERT abcABC would input text as "ABCABC".

◇ /abc/ would locate the next line containing one of "abc", "Abc", "ABc", "ABC", "aBc", "aBC" or "abC".

◇ CHANGE/abc/def/\* \* would change all occurrences of "abc", "Abc", "ABc", "ABC", "aBc", "aBC" and "abC" to "def".

**See Also:**

**CHANGE SORT**

---

**CHANGE - EXTRACT**

---

**Syntax:**

```
>>--- EXtract --- /CHANGE/ -----><
```

**Description:**

For use in macros, EXTRACT CHANGE obtains statistics relating to the last execution of the **CHANGE** primary command within the same macro. CHANGE command statistical information does not persist beyond the life of the macro.

EXTRACT CHANGE is supported for both the XEDIT and ISPF versions of the CHANGE command.

**EXTRACT Rexx variables:**

CHANGE.0	3
CHANGE.1	Number of occurrences changed.
CHANGE.2	Number of lines changed.
CHANGE.3	Number of lines truncated. (Not yet supported.)

**See Also:**

**CHANGE**

---

## CLIPBOARD - QUERY/EXTRACT

---

**Syntax:**

```
>>--- Query ----- CLIPboard -----><
>>--- EXTRACT --- /CLIPboard/ -----><
```

**Description:**

QUERY and EXTRACT CLIPBOARD reports the current content of the FileKit [clipboard](#).

**QUERY Response:**

Displays the number of elements in the clipboard and, if there are elements, the type of elements, either whole lines (type *LINE*) or contiguous sets of columns (type *COLUMN*).

For example:

```
ZZSE010I CLIPBOARD ELEMENTS 5 TYPE COLUMN
```

**EXTRACT Rexx variables:**

clipboard.0	2
clipboard.1	The number of elements in the clipboard.
clipboard.2	The type of elements in the clipboard. <i>LINE</i> or <i>COLUMN</i> . (Null if clipboard.1=0).

**See Also:**

[CLIPBOARD](#)

---

## CMDDEF - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+--- CMDDEF ---+ ALPHA -----><
|         |         |         |         |
+- SET ----+         +- ALPHANumeric -+
>>--- Query ----- CMDDEF -----><
>>--- EXTRACT --- /CMDDEF/ -----><
```

**Description:**

Determines whether numeric values are allowed in macro or command synonym names.

By default, the CBL interface allows specification of numeric parameters or relative line targets to be adjoined to the preceding or following command verb with no intervening blanks (i.e. CMDDEF ALPHA). e.g. "3add6" will locate the 3rd line following the focus line and then add 6 new lines.

"z1 80" will set the left and right zone (bound) columns to columns 1 and 80 respectively.

This means that, by default, numerics contained within macro names and command synonyms are interpreted by the command processor and not considered part of the command verb. (This may be bypassed for macro execution by preceding the macro name with the [MACRO](#) command.)

If CMDDEF ALPHANUMERIC is in effect (default for the CBL interface), then numerics within command verbs, macro names and synonyms are not interpreted by the command processor. e.g. "3add6" will attempt to execute a command synonym or macro with name "3add6".

"z1 80" will attempt to execute a command synonym or macro with name "z1" and parameter "80".

If CMDDEF ALPHA is in effect "3add6" will place focus on the line 3 lines below the current focus line, then insert 6 blank lines; "z1 80" will set the left and right zone (BOUNDS) columns to be 1 and 80 respectively. See [SET ZONE](#).

SET CMDDEF takes effect at the Global level and its setting is saved if [SAVEOPTIONS ON](#) is in effect.

**Set Options:**

APLHA  
APLHANUMERIC

Specifies whether command verbs contain alphanumeric characters or only alpha characters.  
Default for INTERFACE=ISPF is ALPHANUMERIC. Default for INTERFACE=CBLE is ALPHA.

**QUERY Response:**

Displays the current setting of CMDDEF (ALPHA or ALPHANUMERIC).

**EXTRACT Rexx variables:**

cmddef.0	1
cmddef.1	ALPHA   ALPHANUMERIC

---

## CMDLINE - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+--- CMDline ---+--- TOP -----><
   |         |           |         |
   +- SET ----+         +- BOTtom ---+
>>--- Query ----- CMDline -----><
>>--- EXtract --- /CMDline/ -----><
```

**Description:**

Position the command line at the top or bottom of the display window. SET CMDLINE takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

TOP  
BOTTOM

Location of command line. TOP positions the command line at the first line following the title bar while BOTTOM positions the command line immediately following the horizontal scroll bar.  
Default is BOTTOM.

**QUERY Response:**

Displays the current command line location (TOP or BOTTOM).

**EXTRACT Rexx variables:**

cmdline.0	1
cmdline.1	TOP   BOTTOM

**Example:**

```
cmdline top
Position the command line at the top of the display window.
```

**See Also:**

**COMMANDLINE** and the Command Line item of the **System Menu**



**PREFIX**

Prefix area (with PREFIX ON). Default colour is GREEN.

**SCALE**

Scale line (with SCALE ON). Default colour is WHITE.

**SHADOW**

Shadow lines (for excluded lines when SHADOW ON). Default colour is WHITE.

**THIGHLIGHT**

Highlighted targets. Default colour is GREEN.

**TOFEOF**

Top of File and End of File lines. Default colour is YELLOW.

**BLUE | RED | PINK | GREEN | TURQUOISE | YELLOW | WHITE | DEFAULT**

Supported colours on each field. If DEFAULT is specified, the default colour for the field is set.

**BLINK | REVVIDEO | USCORE | NONE**

Extended highlighting of the specified field. The colour may blink, be displayed in reverse video or be underlined. Default is NONE.

**QUERY Response:**

Displays all the current colour settings for fields in the focus text edit view.

**EXTRACT Rexx variables:**

color.0 colour.0	Number of items reflecting the SET COLOUR commands in effect for the current file.
color.i colour.i	Each SET COLOUR item, field name in uppercase, followed by its colour and extended highlighting code. See SET COLOUR.

**Examples:**

```
set colour prompt green uscore
```

```
set colour hi g bli
    Text in tagged lines will be green and blinking.
```

```
col com turq rev
```

**See Also:**

SET/QUERY/EXTRACT Options: **HCOLOUR LCOLOUR SCOLOUR**

## COLUMN - QUERY/EXTRACT

**Syntax:**

```
>>--- Query ----- COlumn ----->>
```

```
>>--- EXtract --- /COlumn/ ----->>
```

**Description:**

QUERY and EXTRACT COLUMN reports the column number of the focus column.

**QUERY Response:**

Displays the column number of the focus column.

**EXTRACT Rexx variables:**

column.0	1
column.1	Column number of the focus column.

**See Also:**

CFIRST CLAST CLOCATE CURLINE

---

**COUNT - EXTRACT**

---

**Syntax:**

&gt;&gt;--- EXTRACT --- /COUNT/ -----&lt;&lt;

**Description:**

For use in macros, EXTRACT COUNT obtains statistics relating to the last execution of the **COUNT** primary command within the same macro. COUNT command statistical information does not persist beyond the life of the macro.

**EXTRACT Rexx variables:**

COUNT.0	2
COUNT.1	Number of occurrences of string.
COUNT.2	Number of lines containing at least one occurrence of string.

**See Also:****COUNT**

---

**CURLINE - EXTRACT**

---

**Syntax:**

&gt;&gt;--- EXTRACT --- /CURLINE/ -----&lt;&lt;

**Description:**

For use in macros, EXTRACT CURLINE returns information relating to the **current line** and **focus line** within the current text edit view.

**EXTRACT Rexx variables:**

curline.0	6
curline.1	Value of CURLINE option. Since SET CURLINE is not yet supported, this value is always 0.
curline.2	Line offset from the top of the edit view document window client area at which the current line is located. Since SET CURLINE is not yet supported, this value is always 0.
curline.3	Contents of the focus line in mixed case.
curline.4	Status of both the new and change settings for the focus line in the current edit session. "ON" if the line has been altered (new or changed). "OFF" if the focus line has not been altered.
curline.5	Status of the new and change setting for the focus line in the current edit session. "NEW CHANGED" if the line has been added. "OLD CHANGED" if the line has been changed. "OLD" if the line is not new and unchanged.
curline.6	Selection level of the focus line.

**See Also:**SET/QUERY/EXTRACT Option: **COLUMN**

---

## CURSOR - QUERY/EXTRACT

---

**Syntax:**

```
>>--- Query ----- CURsor -----><
>>--- EXtract --- /CURsor/ -----><
```

**Description:**

QUERY and EXTRACT CURSOR report information relating to the current cursor position.

**QUERY Response:**

QUERY CURSOR displays the current cursor location as 4 numeric values: the line and column number within the client area of document window followed by the absolute line and column number within the edited data.

The absolute line value within the edited data is 0 if the cursor is located in either the Top of File or scale lines (SCALE ON), otherwise -1 if the cursor is located outside the display of edited data. Similarly, the absolute column value within the edited data is 0 if the cursor is located in the column before column 1 of the edited data, otherwise, -1 if the cursor is located outside the display of edited data.

**EXTRACT REXX variables:**

cursor.0	4
cursor.1	Line number of the cursor within the client area of the text edit document window.
cursor.2	Column number of the cursor within the client area of the text edit document window.
cursor.3	Absolute line number of the cursor within the edited data. 0 if in the Top of File or scale lines (SCALE ON). -1 in all other locations outside the display of edited data.
cursor.4	Absolute column number of the cursor within the edited data. 0 if in the column immediately to the left of edited data column 1. -1 in all other locations outside the display of edited data.

**See Also:**

**CURSOR**

---

## DEFPROFILE - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+ DEFPROFile --- macroname -----><
  |         |
  +- SET -----+
>>--- Query ----- DEFPROFile -----><
>>--- EXtract --- /DEFPROFile/ -----><
```

**Description:**

Define the name of the default profile macro that gets executed every time a new file is added to the ring of edited files.

SET DEFPROFILE takes effect at the Global level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

*macroname*  
Name of a CBL e REXX macro.

This may be the full data set name and member name or simply the name of a member found in the macropath libraries.

**QUERY Response:**

Displays the name of the default profile macro.

**EXTRACT Rexx variables:**

defprofile.0	1
defprofile.1	Name of default profile macro.

**Examples:**

```
defprof cbleprof
defprof cbl.test.cble(profile)
```

**See Also:**

The CBL Text Editor [Profile](#) macro.

## DIALOG - EXTRACT

**Syntax:**

```
>>--- EXtract --- /DIALOG/ -----><
```

**Description:**

For use in macros, EXTRACT DIALOG obtains the value entered and/or button selected by the user following the last execution of the [DIALOG](#) primary command within the same macro. DIALOG information does not persist beyond the life of the macro.

**EXTRACT Rexx variables:**

dialog.0	2
dialog.1	The contents of the edit field when the dialog window is closed. If no edit string is entered or if no edit field is present, then this variable is set to the null string.
dialog.2	The upper case text string of the action selected when the dialog window is closed. i.e. "OK", "CANCEL", "YES", "NO" or "PF3". ("PF3" indicates the dialog window was closed without selecting an action.)

**See Also:**

[DIALOG](#)

## DISPLAY - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+--- DISPlay --- n1 +-----+><
|   SET   |           |   n2   |
+-+-----+           +-+-----+

>>--- Query ----- DISPlay -----><

>>--- EXtract --- /DISPlay/ -----><
```

**Description:**

Display only lines with the specified selection level or lines whose selection level falls within the specified range.

SET DISPLAY takes effect at the View level.



Every line in the edited file is initially assigned a selection level of 0. The selection level of a line may be changed to any number in the range 0 to 255 via the SET SELECT command. Also, the selection level of a line may be automatically changed by the ALL and EXCLUDE primary commands and the X and S line (prefix area) commands.

All line selected by an ALL or MORE command are assigned a selection level of 1 and DISPLAY 1 1 is set. All lines selected by the ISPF style EXCLUDE command are assigned selection level 255 and the DISPLAY setting is unchanged.

Initially, DISPLAY 0 0 is set and so all lines in the file are displayed.

### Set Options:

*n1* The selection level of the lines to be displayed or the low value in a range of selection levels for which lines will be displayed.

*n2* The high value in a range of selection levels for which lines will be displayed. "\*" (asterisk) may be specified to indicate 255. If not specified, *n2* is equal to *n1*.

### QUERY Response:

Displays the low and high values that define the range of line selection levels for which lines may be displayed.

### EXTRACT Rexx variables:

display.0	2
display.1	Minimum displayable selection level.
display.2	Maximum displayable selection level.

### Example:

```
DISP 1
    Display lines with a selection level of 1.
```

```
DISPLAY 0 1
    Display all lines with a selection level between 0 and 1.
```

```
DISPLAY 10 20
    Display all lines with a selection level between 10 and 20.
```

### See Also:

**ALL SELECT**

---

## DSN - SET/QUERY/EXTRACT

---

### Syntax:

```
>>+-----+ DSN --- dsname -----<<
   |         |
   +- SET -----+
```

```
>>--- Query ----- DSN -----<<
```

```
>>--- EXtract --- /DSN/ -----<<
```

### Description:

For HFS paths only, DSN is equivalent to **FILEID**.

Change the DSN (Data Set Name) portion of the fileid assigned to the currently edited in-storage data. The fileid in the title bar of the window view is updated to reflect the change and the FIDCHANGED flag is set on. Following a save operation, the edited data will be saved to disk using the changed fileid.

SET DSN takes effect at the File level.

**Set Options:***dsname*

For MVS data sets, *dsname* replaces the full data set name (excluding any PDS member name) of the current file and must contain at least two qualifiers. For HFS paths, *dsname* is the full fileid of the HFS file.

For VSE files, *dsname* replaces the full lib.sublib.mname.mtype LIBR member id of the current file. i.e. the full fileid.

For CMS files, *dsname* replaces the full FNAME FTYPE FMODE of the current file. *dsname* is specified as FNAME.FTYPE.FMODE where each qualifier must be a valid CMS fileid token. If only two qualifiers are specified, then only the FNAME and FTYPE are replaced. Similarly, if only one qualifier is specified, then only the FNAME is replaced.

**QUERY Response:**

Displays the data set name of the file in the current text or data edit view.

**EXTRACT Rexx variables:**

dsn.0	1
dsn.1	Data Set Name portion of the fileid belonging to the current file.

**Example:**

```
DSN CBL.VVC.LST.V210
```

Change the MVS PDS name of the current file.

```
DSN PRD2.CBLINST.VNF210.TXT
```

Change the VSE LIBR member id of the current file.

```
DSN PROFILE.CBL
```

Change the CMS FNAME and FTYPE of the current file. **Note:** FMODE is unchanged.

**See Also:**

SET/QUERY/EXTRACT Option: **FIDCHANGED FILEID FMODE FNAME FPATH FTYPE**

---

**DSORG - SET/QUERY/EXTRACT**

---

**Syntax:**

```
>>+-----+ DSORG ---+ PDS -----+<<
|         |         |         |         |
+- SET ----+         | SEQ -----+
|         |         |         |         |
+- KSDS -- kp1 --- kp2 ---+
|         |         |         |         |
+- ESDS -----+
|         |         |         |         |
+- RRDS -----+
|         |         |         |         |
```

```
>>--- Query ----- DSORG -----<<
```

```
>>--- EXtract --- /DSORG/ -----<<
```

**Description:**

Change the data set organisation for the currently edited data. DSORG may only be set for data sets that have not yet been allocated or defined.

SET DSORG takes effect at the File level.

When changing the DSORG to PDS, the FNAME (penultimate) qualifier is removed from the DSN and used as the member name. If only 2 qualifiers exist in the original DSN, then the command will fail.

Coversely, when changing DSORG from PDS to any other supported organisation, the member name is inserted into the DSN as the penultimate qualifier.

**Note:** SET FNAME is synonymous with SET MBR.

**Set Options:**

PDS | SEQ | KSDS | ESDS | RRDS  
Supported data set organisations.

kp1 kp2  
Start and end positions of the key field. These are required if defining the data to be a VSAM KSDS data sets.

**QUERY Response:**

Displays the Data Set Organisation of the file in the current text or data edit view. (PDS, SEQ, HFS, KSDS, ESDS, RRDS, LDS, CMS or LIBR)

**EXTRACT Rexx variables:**

dsorg.0	1
dsorg.1	PDS   SEQ   HFS   KSDS   ESDS   RRDS   LDS   CMS   LIBR

**Examples:**

dsorg ksd s 1 16  
Make the currently edited data a VSAM KSDS with KEY at position 1 (offset 0) for length 16.

dsorg pds  
Make the currently edited data a PDS library member.

**See Also:**

SET/QUERY/EXTRACT Option: **KEY**

## ENVVARS - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+ ENVVars ---+ ON ---+-----+<<
  |         |         |         |         |
+- SET -----+         +- OFF -+ +- c1 -+
>>--- Query ----- ENVVars -----<<
>>--- EXtract --- /ENVVars/ -----<<
```

**Description:**

ENVVARS (**Environment Variables**) determines whether translation of environment variables occurs in commands issued from the focus text or data edit view. This includes commands executed from the command prompt or from within an edited file (typically a CMX file) via the **ACTION** facility.

ENVVARS also defines the character used to delimit variable names.

Environment variables include standard system determined variables, MVS System symbolics, EDITV variables and INI variables.

SET ENVVARS takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

ON  
OFF  
ON and OFF switches translation on and off respectively. If ENVVARS is OFF, command string will be passed without translating variables. Default is ON.

c1  
The delimiter character. If c1 is not specified, the delimiter character last defined in the current CBLLe view, is unchanged. Default is "%" (percent - X'6C'.)

**QUERY Response:**

Displays the current setting for environment variable translation (ON or OFF) and the current environment variable delimiter character.

**EXTRACT Rexx variables:**

envvars.0	2
envvars.1	ON   OFF
envvars.2	ENVVAR delimiter character.

**Example:**

```
envv on #
Set the variable delimiter character to "#" (hash).
```

**See Also:**

**EDITV VIGNORE VRESPECT**

---

## EOLIN - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+ EOLIn +-----><
|         |         |         |
+- SET ----+         +-- CR ----+
|         |         |         |
+-- LF ----+         +-- NL ----+
|         |         |         |
+-- CRLF ----+        +-- LFCR ----+
|         |         |         |
+-- CRNL ----+        +-- string --+
|         |         |         |

>>--- Query ----- EOLIn -----><

>>--- EXtract --- /EOLIn/ -----><
```

**Description:**

EOLIN alters the current input EOL (end-of-line) delimiter string used to interpret variable length records obtained from an HFS file via the **GET** or ISPF style **COPY** commands.

An EOLIN value is set for all text and data edit views of the same data, including those containing non-HFS files. In text edit views, this allows use of the GET and ISPF style **COPY** commands to retrieve records from an HFS file into a Sequential file, VSAM file or PDS(E) member.

When an edit view is opened and before the edit data is read, the default EOLIN is automatically set to be one of the following values, in the order of precedence:

1. The EOL parameter argument specified on the EDIT or BROWSE command.
2. For data edit (SDE) EDIT/BROWSE only, the EOLIN value set in the SDE profile macro (using SET EOLIN).
3. The EOL format value defined in the directory entry.
4. EOLIN=NL (new line).

SET EOLIN takes effect at the File level.

**Set Options:**

CR|LF|NL|CRLF|LFCR|CRNL|*string*

Identifies the end-of-line delimiter. Delimiter elements are as follow:

<b>NL</b>	X'15'	New Line.
<b>CR</b>	X'0D'	Carriage Return.
<b>LF</b>	X'0A'	Line Feed.
<b>string</b>	-	A 2-byte user specified character or hex string.

**QUERY Response:**

Displays the current EOL (end-of-line) setting for EOLIN which is used for the GET or ISPF format COPY *fileid* command when *fileid* is an HFS path name.

**EXTRACT Rexx variables:**

eolin.0	1
eolin.1	CR   LF   NL   CRLF   LFCR   CRNL   X' <i>string</i> '

**Example:**

DSN CBL.VVC.LST.V210

Change the MVS PDS name of the current file.

DSN PRD2.CBLINST.VNF210.TXT

Change the VSE LIBR member id of the current file.

DSN PROFILE.CBL<sub>e</sub>

Change the CMS FNAME and FTYPE of the current file. **Note:** FMODE is unchanged.

**See Also:**

**EOLOUT GET**

---

## EOLOUT - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+ EOLOut +-----+<<
|         |         |         |
+- SET -+-+         +--- CR -+-+
|         |         |         |
+--- LF -+-+         +--- NL -+-+
|         |         |         |
+--- NL -+-+         +--- CRLF -+-+
|         |         |         |
+--- LFCR -+-+        +--- CRNL -+-+
|         |         |         |
+--- CRNL -+-+        +--- string -+-+
|         |         |         |

>>--- Query ----- EOLOut -----<<

>>--- EXtract --- /EOLOut/ -----<<
```

**Description:**

EOLOUT determines the output HFS file EOL (end-of-line) delimiter string to be used when saving edited data to an HFS fileid which is not fixed format. i.e RECFM F is **not** the current setting for the edited data. By default, the EOLOUT value is equal to the **EOLIN** value established when the text or data edit view is initially opened.

An EOLOUT value is also set for non-HFS files allowing the user to subsequently save the data in the view to a new HFS file simply by using the SAVE *fileid* command where *fileid* is an HFS path name.

SET EOLOUT takes effect at the File level.

**Set Options:**

CR | LF | NL | CRLF | LFCR | CRNL | *string*

Identifies the end-of-line delimiter. Delimiter elements are as follow:

<b>NL</b>	X'15'	New Line.
<b>CR</b>	X'0D'	Carriage Return.
<b>LF</b>	X'0A'	Line Feed.
<b>string</b>	-	A 2-byte user specified character or hex string.

**QUERY Response:**

Displays the current EOL (end-of-line) setting for EOLOUT which is used for save output to an HFS file.

**EXTRACT REXX variables:**

eolout.0	1
eolout.1	CR   LF   NL   CRLF   LFCR   CRNL   X' <i>string</i> '

**See Also:**

**SAVE** and the SET/QUERY/EXTRACT Option: **EOLIN**

---

## FIDCHANGED - SET/QUERY/EXTRACT

---

**Syntax:**

```
>> +-----+ FIDCHanged  +- ON  +-----><
    |         |         |         |
    +- SET  +-----+         +- OFF  +-><
```

```
>> --- Query  ----- FIDCHanged  -----><
```

```
>> --- EXtract  --- /FIDCHanged/ -----><
```

**Description:**

This option allows the user to manually control the internal CBLc flag that indicates that the fileid of the current file has been changed by the user. (i.e. via the SET options DSN, FMODE, FNAME, MBR, FPATH, FTYPE or FILEID.)

If the FIDCHANGED flag is on, then CBLc QUIT or END commands will prompt the user to save the file before the window is closed.

The SET FIDCHANGED functionality allows the user to override CBLc's axiom that any change to the fileid assigned to the data edited in storage will prompt for a save operation. e.g. Consider the following steps:

1. Assign a temporary fileid (e.g. via SET DSN) to the data being edited, so releasing the exclusive MVS SPFEDIT ENQ or VSE LIBR LOCK on the original DSN. This automatically sets the FIDCHANGED flag on.
2. Perform operations requiring an exclusive ENQ or LOCK on the original fileid. (e.g. DELETE)
3. Restore the original fileid to the edited data.
4. Set FIDCHANGED off so that, when executing QUIT or END, the user is not prompted to save the file.

This is the technique employed in the distributed ERA and RENAME CBLc REXX macros.

SET FIDCHANGED takes effect at the File level.

**Set Options:**

ON | OFF  
FIDCHANGED flag is set ON or OFF.

**QUERY Response:**

Displays the current setting of the FIDCHANGED flag (ON or OFF).

**EXTRACT REXX variables:**

fidchanged.0	1
fidchanged.1	The current setting of the FIDCHANGED flag. <b>ON</b> or <b>OFF</b> .

**See Also:**

SET/QUERY/EXTRACT Options: **DSN FILEID FMODE FNAME FPATH FTYPE**

## FILEID - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+----+ Fileid --+--- fileid ----->>
    |         |    |         |
    +- SET -----+  +- FID -----+

>>--- Query ----- Fileid ----->>

>>--- EXtract ----- /Fileid/ ----->>
```

**Description:**

Change the full fileid assigned to the currently edited in-storage data. The fileid in the title bar of the window view is updated to reflect the change and the FIDCHANGED flag is set on. Following a save operation, the edited data will be saved to disk using the changed fileid.

SET Fileid takes effect at the File level.

**Set Options:**

*fileid*

For MVS data sets, *fileid* replaces the full DSN of the current data set name. For PDSs, this includes the member name. For HFS path names, *fileid* sets the absolute or relative fileid for the currently edited data. Changing the fileid from an MVS DSN to an HFS file will change the DSORG to HFS.

For VSE files, *fileid* replaces the full lib.sublib.mname.mtype LIBR member id of the current file.

For CMS files, *fileid* replaces the full FNAME FTYPE FMODE of the current file. *fileid* is specified as FNAME.FTYPE.FMODE where each qualifier must be a valid CMS fileid token. If only two qualifiers are specified, then only the FNAME and FTYPE are replaced. Similarly, if only one qualifier is specified, then only the FNAME is replaced.

**QUERY Response:**

Displays the full fileid of the file in the current text or data edit view.

**EXTRACT Rexx variables:**

fileid.0	1
fileid.1	Full fileid of the current file.

**Example:**

```
FI CBL.VVC.LST.V210 (TEST003)
    Change the current fileid to an MVS PDS name and member id.

FILE /u/cbl/nbj/oem.cbl.selcnam
    Change the current fileid to an HFS path name.

FI PRD2.CBLINST.VNF210.TXT
    Change the full VSE LIBR member id of the current file.

FI PROFILE.CBLe
    Change the CMS FNAME and FTYPE of the current file. Note: FMODE is unchanged.
```

**See Also:**

SET/QUERY/EXTRACT Options: **DSN FIDCHANGED FMODE FNAME FPATH FTYPE**

---

## FLSCREEN - QUERY/EXTRACT

---

**Syntax:**

```
>>--- Query ----- FLSCReen -----><
```

```
>>--- EXTract --- /FLSCReen/ -----><
```

**Description:**

QUERY and EXTRACT FLScreen report the numbers of the first and last lines of data visible in the current text edit document view.

**QUERY Response:**

Displays the line number of the first and last text lines visible in the current text edit window view.

**EXTRACT Rexx variables:**

flscreen.0	2
flscreen.1	File line number of first text line visible in the window view.
flscreen.2	File line number of last text line visible in the window view.

---

## FMODE - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+--- FMode --- qualifier -----><
   |         |
   +- SET -----+
```

```
>>--- Query ----- FMode -----><
```

```
>>--- EXTract --- /FMode/ -----><
```

**Description:**

Change the FMODE (File Mode) portion of the fileid assigned to the currently edited in-storage data. The fileid in the title bar of the window view is updated to reflect the change and the FIDCHANGED flag is set on. Following a save operation, the edited data will be saved to disk using the changed fileid.

SET FMODE takes effect at the File level.

**Set Options:***qualifier*

For MVS data sets, *qualifier* replaces the high level qualifier (HLQ) of the current data set name. For HFS files, *qualifier* sets the absolute HFS path name's first level directory name above the root directory for the currently edited data. (e.g. "usr" in "/usr/include/arpa/inet.h")

For VSE files, *qualifier* replaces the lib LIBR library name.

For CMS files, *qualifier* replaces the FMODE of the current file.

**QUERY Response:**

Displays the file mode (HLQ, HFS directory, LIBR library name) of the file in the current text or data edit view.

**EXTRACT Rexx variables:**

fmode.0	1
fmode.1	File Mode portion of the fileid belonging to the current file.



**Example:**

```
FM CBL110
    Change the MVS High Level qualifier of the current file.

FM PRD1
    Change the VSE LIBR library name of the current file.

FM G1
    Change the CMS FMODE of the current file.
```

**See Also:**

SET/QUERY/EXTRACT Options: [DSN](#) [FIDCHANGED](#) [FILEID](#) [FNAME](#) [FPATH](#) [FTYPE](#)

## FNAME, MBR - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+-----+ FName --+--- qualifier -----><
    |           |           |           |
    +- SET -----+   +- MBR -----+

>>--- Query -----+--- FName ---+-----><
                |           |
                +- MBR -----+

>>--- EXtract --- / +- FName ---+ / -----><
                |           |
                +- MBR -----+
```

**Description:**

Change the Member/File Name portion of the fileid assigned to the currently edited in-storage data. The fileid in the title bar of the window view is updated to reflect the change and the FIDCHANGED flag is set on. Following a save operation, the edited data will be saved to disk using the changed fileid.

SET FNAME and SET MBR take effect at the File level.

**Set Options:**

*qualifier*  
 For MVS sequential data sets, *qualifier* replaces the penultimate qualifier of the current data set name. For MVS PDS data sets, *qualifier* replaces the member name. For HFS files, *qualifier* sets the file name portion of the HFS path name for the currently edited data. (e.g. "inet" in "/usr/include/arpa/inet.h")

For VSE files, *qualifier* replaces the mname LIBR member name.

For CMS files, *qualifier* replaces the FNAME of the current file.

**QUERY Response:**

Displays the file name of the file in the current text or data edit view.

**EXTRACT Rexx variables:**

fname.0 mbr.0	1
fname.1 mbr.1	File Name portion of the fileid belonging to the current file.

**Example:**

```
FN TEST001
    Change the MVS PDS member name.

FN V210INST
    Change the VSE LIBR member name of the current file.
```

FN CBLVJ27

Change the CMS FNAME of the current file.

**See Also:**SET/QUERY/EXTRACT Options: **DSN FIDCHANGED FILEID FMODE FPATH FTYPE**

---

**FPATH - SET/QUERY/EXTRACT**

---

**Syntax:**

```
>>+-----+-- FPath --- qualifier -----><
   |           |
   +- SET -----+
```

```
>>--- Query ----- FPath -----><
```

```
>>--- EXtract --- /FPath/ -----><
```

**Description:**

Change the FPATH (File Path) portion of the fileid assigned to the currently edited in-storage data. The fileid in the title bar of the window view is updated to reflect the change and the FIDCHANGED flag is set on. Following a save operation, the edited data will be saved to disk using the changed fileid.

SET FPATH takes effect at the File level.

**Set Options:***qualifier*

For MVS data sets, *qualifier* replaces all qualifiers except the low level qualifier of the current data set name. For HFS files, *qualifier* sets the file path portion of the HFS path name for the currently edited data. (e.g. "/usr/include/arpa" in "/usr/include/arpa/inet.h")

Changing the file path from an MVS DSN to an HFS file path will change the DSORG to HFS.

For VSE files, *qualifier* replaces the lib.sublib LIBR library and sublibrary names.

For CMS files, *qualifier* replaces the file mode (FMODE) of the current file.

**QUERY Response:**

Displays the file path of the file in the current text or data edit view.

**EXTRACT REXX variables:**

fpath.0	1
fpath.1	File Path portion of the fileid belonging to the current file.

**Example:**

FP CBL.CMD

Change all but the low level qualifier of the current MVS data set.

FP PRD2.CBL

Change the VSE LIBR lib.sublib name of the current file.

FP A1

Change the CMS file mode (FMODE) of the current file.

**See Also:**SET/QUERY/EXTRACT Options: **DSN FIDCHANGED FILEID FMODE FNAME FTYPE**

## FTYPE - SET/QUERY/EXTRACT

### Syntax:

```
>>+-----+ FType --- qualifier -----><
  |         |         |
  +- SET -----+

>>--- Query ----- FType -----><

>>--- EXtract --- /FType/ -----><
```

### Description:

Change the FTYPE (File Type) portion of the fileid assigned to the currently edited in-storage data. The fileid in the title bar of the window view is updated to reflect the change and the FIDCHANGED flag is set on. Following a save operation, the edited data will be saved to disk using the changed fileid.

SET FTYPE takes effect at the File level.

### Set Options:

*qual*

For MVS data sets, *qual* replaces the low level qualifier of the current data set name. For HFS files, *qual* sets the file type portion of the HFS path name for the currently edited data. (e.g. "h" in "/usr/include/arpa/inet.h")

For VSE files, *qual* replaces the LIBR member type.

For CMS files, *qual* replaces the FTYPE of the current file.

### QUERY Response:

Displays the file type of the file in the current text or data edit view.

### EXTRACT REXX variables:

fctype.0	1
fctype.1	File Type portion of the fileid belonging to the current file.

### Example:

```
FT S210      Change the low level qualifier of the current MVS data set.
FT OBJ      Change the VSE LIBR member type of the current file.
FT EXEC     Change the CMS FTYPE of the current file.
```

### See Also:

SET/QUERY/EXTRACT Options: [DSN](#) [FIDCHANGED](#) [FILEID](#) [FMODE](#) [FNAME](#) [FPATH](#)

## GENSAVE - SET/QUERY/EXTRACT

### Syntax:

```
>>+-----+ GENSave ---+ AUTO -----><
  |         |         |         |
  +- SET -----+         +- NEWGEN -----+
                               |
                               +- NOGEN -----+
                               |
                               +- PROMPT -----+
```

>>--- Query ----- GENSave ----->><<

>>--- EXTRACT --- /GENSave/ ----->><<

**Description:**

This option controls the default action taken when a **SAVE**, **SSAVE**, **FILE** or **FFILE** command is executed for a member generation in a PDSE version 2 library allocated with a MAXGENS value.

The default action set by GENSAVE may be temporarily overridden by specifying the **NEWGEN** or **NOGEN** option on the **SAVE**, **SSAVE**, **FILE** and **FFILE** primary commands.

The following table identifies the action taken for each of the GENSAVE values.

GENSAVE	Action Taken
<b>AUTO</b>	If the edited member text is for the primary generation (generation 0) then <b>NEWGEN</b> is used. Otherwise, <b>NOGEN</b> is used.  This is the FileKit system default and matches ISPF Edit.
<b>NEWGEN</b>	The data is automatically saved to a new primary generation of the member. e.g. When editing generation -3 of a member, data will be saved as a new generation (generation 0) of the member and the data in generation -3 becomes generation -4.  The absolute and relative generation values in the text edit window view title bar is updated to be that of the new generation 0.
<b>NOGEN</b>	The data is automatically saved to the same generation of the member. e.g. When editing generation -3 of a member, data will be saved back to generation -3. The generation numbers of all other generations of the same member are unchanged.
<b>PROMPT</b>	The CBLEDIT Save Member Generation popup window is displayed prompting the user to reply "Yes" to save a new generation, "No" to save to the same generation or "Cancel" to terminate the SAVE command and return to the edit view.

SET GENSAVE takes effect at the File level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**SET Value:**

- AUTO Same as **NEWGEN** for the primary generation, otherwise **NOGEN**.
- NEWGEN Forces the save of member generation data automatically creates a new primary generation.
- NOGEN Forces the save of member generation data automatically replaces data in the generation referenced in the current text edit view.
- PROMPT Opens the CBLEDIT Save Member Generation popup window when a save operation is executed for any member of a PDSE version 2 library which supports multiple generations.

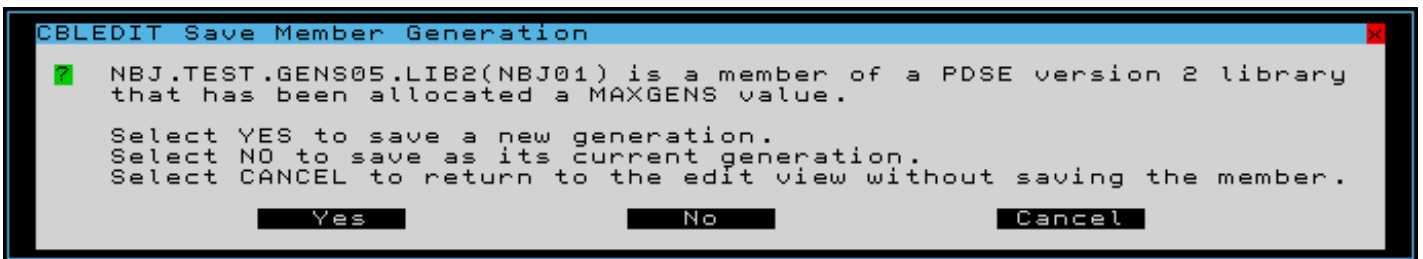


Figure 19. CBLEDIT Save Member Generation Window.

**QUERY Response:**

The current setting of the GENSAVE option: **AUTO**, **NEWGEN**, **NOGEN** or **PROMPT**.

**EXTRACT Rexx variables:**

autosave.0	1
autosave.1	The current setting of member generation save: <b>AUTO</b> , <b>NEWGEN</b> , <b>NOGEN</b> or <b>PROMPT</b> .

## HCOLOR, HCOLOUR - SET/QUERY/EXTRACT

### Syntax:

```

>>+-----+---+ HCOLOr  --+---+ DEFault  -----+---+ Blue  -----+---+ NONE  -----+
|         |   |         |   |         |   |         |   |         |   |         |   |
+- SET  -+  +- HCOLOur -+  +- COMment -----+ +- Red  -----+ +- BLInk  ----+
|         |   |         |   |         |   |         |   |         |   |         |   |
+- ERRor  -----+ +- Pink  -----+ +- REVvideo +
|         |   |         |   |         |   |         |   |         |   |         |   |
+- KEYword -----+ +- Green -----+ +- Uscore  ---+
|         |   |         |   |         |   |         |   |         |   |         |   |
+- NUMber  -----+ +- Turquoise +
|         |   |         |   |         |   |         |   |         |   |         |   |
+- OPERator -----+ +- Yellow  ---+
|         |   |         |   |         |   |         |   |         |   |         |   |
+- PARentthesis -+ +- White  ----+
|         |   |         |   |         |   |         |   |         |   |         |   |
+- STRing  -----+ +- Default  ---+
|         |   |         |   |         |   |         |   |         |   |         |   |
+- VARiable -----+

```

```

>>--- Query  ---+---+ HCOLOr  --+-----+-----+-----+-----+-----+-----+-----+
|         |   |         |   |         |   |         |   |         |   |         |   |
+- HCOLOur -+

```

```

>>--- EXtract  --- /  +- HCOLOr  +- /  -----+-----+-----+-----+-----+-----+-----+
|         |   |         |   |         |   |         |   |         |   |         |   |
+- HCOLOur -+

```

### Description:

Assign colour to individual syntax elements when syntax colouring is set on by the **SET HILITE** option.

SET HCOLOUR takes effect at the File level and its setting is saved if **SAVEOPTIONS ON** is in effect.

### Set Options:

**DEFAULT** Default colour for text that is not one of the other recognised syntax elements. Default colour is BLUE.

**COMMENT** Comment text. Default colour is TURQUOISE.

**ERROR** Syntax in error. Default colour is PINK REVVIDEO.

**KEYWORD** Syntax language keywords. Default colour is RED.

**NUMBER** Numeric constants. Default colour is BLUE.

**OPERATOR** Operator special characters (e.g. "=", ">", "<"). Default colour is YELLOW.

**PARENTHESIS** Parenthesis special characters (i.e. "(" and ")"). Default colour is PINK.

**STRING** Quoted string constants. Default colour is WHITE.

**VARIABLE** Variable names. Default colour is GREEN.

**BLUE | RED | PINK | GREEN | TURQUOISE | YELLOW | WHITE | DEFAULT**  
Supported colours on each field. If **DEFAULT** is specified, the default colour for the field is set.

**BLINK | REVVIDEO | USCORE | NONE**  
Extended highlighting of the specified field. The colour may blink, be displayed in reverse video or be underlined. Default is NONE.

### QUERY Response:

Displays all the current syntax colouring settings for fields in the focus text edit view.

**EXTRACT Rexx variables:**

hcolor.0 hcolour.0	10
hcolor.i hcolour.1	Each SET HCOLOUR syntax element name in uppercase, followed by its currently assigned colour and extended highlighting code.

**See Also:**

SET/QUERY/EXTRACT Options: [COLOUR](#) [HILITE](#) [LCOLOUR](#) [SCOLOUR](#)

**HEXSTRING - SET/QUERY/EXTRACT**

HEX primary command exists in the following application environments.

<a href="#">Text Edit (ISPF)</a>	Text Editor with INTERFACE=ISPF (default for z/OS).
<a href="#">Text Edit (XEDIT)</a>	Text Editor with INTERFACE=XEDIT (default for z/VM CMS and z/VSE).
<a href="#">Data Edit</a>	SDE Data Editor.

**Syntax:**

```
>>+-----+--- HEXstring ---+ ON ---+----->>
|         |         |         |         |
+- SET ----+         +- OFF --+

>>--- Query ----- HEXstring ----->>

>>--- EXtract --- /HEXstring/ ----->>
```

**Description:**

Controls interpretation of hexadecimal strings.

A valid hexadecimal string consists of 1 or more pairs of hexadecimal digits (0-9, A-F), enclosed in "" (apostrophes) or "" (quotes) and immediately preceded by an upper or lower case "X". A pair of hexadecimal digits represent a single value in the range x'00' to x'FF' each corresponding to a single character. e.g. The following are equivalent:

```
'ABCDE'
x'C1C2C3C4C5'
```

SET HEXSTRING takes effect at the View level and its setting is saved if [SAVEOPTIONS](#) ON is in effect.

**Set Options:**

ON | OFF

With HEXSTRING ON, CBL will interpret any string argument enclosed in "" (apostrophes) or "" (quotes) and immediately preceded by an upper or lower case "X", as being hexadecimal. With HEXSTRING OFF, strings of this type are not interpreted.

Default is OFF.

**QUERY Response:**

Displays the current setting of HEXSTRING (ON or OFF).

**EXTRACT Rexx variables:**

hexstring.0	1
hexstring.1	ON   OFF

**See Also:**

[CHANGE](#) and discussion on [Targets](#)

## HILITE, HILIGHT - SET/QUERY/EXTRACT

### Syntax:

```

                                     +- ON ---+
>>+-----+-----+ Hilite ---+-----+-----+><
    |         |         |         |         |
    +- SET -----+ +- HIlight -+ +- OFF -+ +- ASM -----+
                                     |
                                     +- AUTO -----+
                                     |
                                     +- C -----+
                                     |
                                     +- CMX -----+
                                     |
                                     +- COBo1 -----+
                                     |
                                     +- HTML -----+
                                     |
                                     +- JCL -----+
                                     |
                                     +- PL1 -----+
                                     |
                                     +- REXx -----+
                                     |
                                     +- SELCopy ----+
                                     |
                                     +- SQL -----+
                                     |
                                     +- XML -----+

>>--- Query -----+ Hilite ---+-----+-----+><
                |         |
                +- HIlight -+

>>--- EXtract -- / +- Hilite ---+ / -----+-----+><
                |         |
                +- HIlight -+

```

### Set Options:

Control text edit syntax colour highlighting which is sensitive to the language or format of the edited source.

Selected colours of recognised syntax elements may be altered using [SET HCOLOUR](#).

SET HILITE takes effect at the File level, the setting being saved/restored across sessions based on the edited file's file-type (low-level qualifier).

### Set Options:

ON  
OFF

Set syntax sensitive colour highlighting on or off.

ASM | ASSEMBLER

Edited text is Assembler language source.

AUTO

Language will be determined automatically by examining the contents of the edited text

C

Edited text is C/C++ language source.

CMX

Edited text is a FileKit Command Centre file.

COBOL

Edited text is COBOL language source.

HTML

Edited text is HTML source.

JCL

Edited text is MVS Job Control Language.

PL1 | PLI

Edited text is PL/1 language source.

REXX

Edited text is REXX language source.

## SELCOPY

Edited text is SELCOPY language source.

## SQL

Edited text is SQL (Structured Query Language) source.

## XML

Edited text is XML (Extended Markup Language) source.

**QUERY Response:**

Displays the current setting of HILITE (ON or OFF) followed by the text syntax type.

**EXTRACT Rexx variables:**

hilite.0 highlight.0	2
hilite.1 highlight.1	ON   OFF
hilite.2 highlight.2	Text source type.

hilite.0 highlight.0	3
hilite.1 highlight.1	ON   OFF
hilite.2 highlight.2	AUTO
hilite.3 highlight.3	Automatically selected source type.

**See Also:**

SET/QUERY/EXTRACT Option: **HCOLOUR**

**HSCROLLCURSOR - SET/QUERY/EXTRACT****Syntax:**

```
>>+-----+--- HSCROLLCursor ---+ ON ---+----->>
  |           |           |           |
+- SET -----+           +- OFF -+

>>--- Query ----- HSCROLLCursor ----->>

>>--- EXtract ---- /HSCROLLCursor/ ----->>
```

**Description:**

Controls horizontal scrolling when the target of a successful CLOCATE command is outside the displayed width of the edit view.

When HSCROLLCURSOR is in effect and the target column is outside the edit view, the view is scrolled horizontally so that the target (new focus) column becomes the first column displayed in the edit view.

SET HSCROLLCURSOR takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

ON | OFF

Set HSCROLLCURSOR ON or OFF.  
Default is OFF.

**QUERY Response:**

Displays the current setting for HSCROLLCURSOR (ON or OFF).

**EXTRACT Rexx variables:**

hscrollcursor.0	1
hscrollcursor.1	ON   OFF



**Examples:**

```
hscrollc on
```

**See Also:**

[CLOCATE](#)

---

## IMPMACRO - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+--- IMPMACro ---+- ON --+-----><
   |         |           |   |   |
   +- SET ----+         +- OFF -+
>>--- Query ----- IMPMACro -----><
>>--- EXTract ---- /IMPMACro/ -----><
```

**Description:**

IMPMACRO (Implied Macro) determines whether CBLc will search for a macro when a command is issued that does not match a CBLc command name. If IMPMACRO is ON and a macro exists with the same name as the command issued, then the macro is executed.

If a matching macro name is not found or IMPMACRO is OFF, then an error message is issued.

SET IMPMACRO takes effect at the Global level and its setting is saved if [SAVEOPTIONS ON](#) is in effect.

**Set Options:**

ON | OFF  
Set IMPMACRO ON or OFF. Default is ON.

**QUERY Response:**

Displays the current setting of IMPMACRO (ON or OFF).

**EXTRACT REXX variables:**

impmacro.0	1
impmacro.1	ON   OFF

**See Also:**

[MACRO](#)

---

## INIFILE - QUERY/EXTRACT

---

**Syntax:**

```
>>--- Query ----- INIFile -----><
>>--- EXTract --- /INIFile/ -----><
```

**Description:**

QUERY and EXTRACT INFILE report the fileid (data set name) of the System and User INI files.

**QUERY Response:**

Displays the names of the active System and User INI files.

**EXTRACT Rexx variables:**

inifile.0	2
inifile.1	The FileKit System INI fileid <i>system.inifile</i>
inifile.2	The User INI fileid <i>user.inifile</i>

**See Also:**

SET/QUERY/EXTRACT Option: **INIVAR**

---

## INIVAR - SET/UNSET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+----- INIVar --- var_name +-----+-----<<
|         |         |         |         |         |
+- SET  +-----+         +- var_value +-----+
|         |         |         |         |         |

>>--- UNSET  ----- INIVar --- var_name -----<<

>>--- Query  ----- INIVar -----<<

>>--- EXtract --- / +- INIVar ---+ / -----<<
|         |         |         |         |         |
+- INIVARS ---+         +- INIVARS ---+
```

**Description:**

SET INIVAR adds or updates a User INI file variable in storage. This may be one of FileKit's standard INI variables or one that is user defined.

UNSET INIVAR removes a User INI file variable from storage.

Note that the values of most standard FileKit INI variables are established only when FileKit is started. Therefore, a change to a value assigned to these types of variables will not take effect until FileKit is re-started.

When the FileKit session is closed, the new or updated INI variable and its value is written to the FileKit User INI file which is personal to the user. This variable and its assigned value will then be established in the user's subsequent FileKit sessions. New variables are introduced at the end of the FileKit User INI file following a comment line reporting the number of new variables added and their date of insert. New values assigned to variables that already have an entry in the User INI file, will replace the existing value at the same location within the data.

INI variable names are **not** case sensitive and are comprised of three qualifiers, each separated by an intermediate dot (".").

1. The first qualifier is either "USER" or "SYSTEM" and represents the source of the variable, i.e. the User of System INI file. Variables assigned using SET INIVAR always have a first qualifier of "USER".
2. The second qualifier represents the category of the variable. Associated INI variables are grouped together using the same category name. A category name is represented in parentheses "(") on a line of its own within the INI file.  
e.g. (DEVELOPMENT)
3. The third qualifier is a descriptive name given to the variable. This name, followed by equals ("=") and its assigned value, occupies a line in the INI file immediately following the category name to which it belongs or following another variable belonging to the same category.  
e.g. TESTHLQ=*var\_value*

The value of any INI variable may be referenced directly within CBL CLI command syntax by simply specifying the variable name between "%" (percent) characters.

e.g. LD %USER.DEVELOPMENT.PREFIX%

SET INIVAR takes effect at the Global level.

**Set/Unset Options:**

*var\_name*

A two-level INI variable name, in the format *nnnnn.mmmmm*, representing the second and third qualifiers of the User INI file variable to be added or updated. e.g. DEVELOPMENT.TESTHLQ  
 Since INI variable names are not case sensitive, *var\_name* will automatically be upper cased prior to being written to the User INI file.

*var\_value*

The value to be assigned to the INI variable name.  
 All tokens following *var\_name* are parsed, as though being read from the INI file, and the resultant value assigned to *var\_name*. If *var\_value* contains an unquoted "\*" (asterisk), it is treated as the start of comment data and so the end of *var\_value*. The "\*" and any text following is ignored and not written to the INI file. The significant *var\_value* text is passed, unaltered, as the value assigned to *var\_name*.  
 Default is a null string.

**QUERY Response:**

Displays the active System and User INI file variables and their values.

**EXTRACT Rexx variables:**

inivar.0 inivars.0	Total number of variables set in both the System and User INI files.
inivar.i inivars.i	<p>All specified system and user INI file variables and their values in alphabetical order. (i=1 to inivar.0) The first token of inivar.i is "SYSTEM" or "USER" indicating the INI file from which the variable has been extracted. System INI file variables are extracted before User INI file variables.</p> <p>The remainder of the extracted value is a 2 level variable name (vlev1.vlev2) immediately followed by equals ("=") and the assigned value in mixed case, as defined in the INI file.</p> <ul style="list-style-type: none"> <li>vlev1 identifies the category of the variable as specified in parentheses "(" in the INI file.</li> <li>vlev2 is the descriptive variable name.</li> </ul> <p>INI file variable names are not case sensitive and are in upper case. e.g.</p> <pre>SYSTEM    CBLVCAT.ESR=222 SYSTEM    EDIT.INSTANCE=Single USER      EDIT.INITIALSIZE=185,70</pre>

**Example:**

```
set inivar Edit.UseDSNPrefix YES
    Update the value of the existing FileKit standard INI variable USER.Edit.UseDSNPrefix.

set inivar Prefix.SysVol Z19
    Add a variable to the User INI file to identify the system volumes prefix. e.g. To subsequently list all system volumes (Z19*):

    LVOL    %USER.PREFIX.SYSVOL%*

set inivar %user%.LogDSN %user%.LOG.D%yy%mm%dd%
    Add a variable to the User INI file to identify the name of a user defined log data set. Both the variable name and its value include standard environment variables which are translated before being passed to the SET INIVAR operation.
```

---

**INSTANCE - SET/QUERY/EXTRACT**

---

**Syntax:**

```
>>+-----+----- INSTANCE ---+ SINGLE -----<<
    |         |         |         |         |
    +- SET  +-----+         +- MULTIPLE -+

>>--- Query ----- INSTANCE -----<<

>>--- EXTRACT ---- /INSTANCE/ -----<<
```

**Description:**

INSTANCE determines whether multiple instances or only a single instance of CBLe may be active at any one time.

SET INSTANCE takes effect at the Global level.

**Set Options:**

SINGLE  
MULTIPLE

If a copy of CBLe is already loaded and a file is edited via a List window prefix command or a menu bar item, then one of the following occurs:

1. For INSTANCE SINGLE, the file will be edited in the existing CBLe window.
2. For INSTANCE MULTIPLE, the file will be edited in a new CBLe window.

**QUERY Response:**

Displays the current setting of INSTANCE (SINGLE or MULTIPLE).

**EXTRACT REXX variables:**

instance.0	1
instance.1	SINGLE   MULTIPLE

## INTERFACE - SET/QUERY/EXTRACT

**Syntax:**

```

                +- VIEW ----+
                |           |
>>+-----+--- INTERFace ---+ ISPF ---+-----+-----+<<
    |           |           |           |           |
    +- SET ----+           +- XEDit -+ +- FILE ----+ +- INitialise -+
                               +- CBLe --+ |           |
                                   +- GLObal -+

```

```

>>--- Query ----- INTERFace -----<<
>>--- EXTRact ---- /INTERFace/ -----<<

```

**Description:**

Sets the text editor interface to be either CBLE (XEDIT) or ISPF mode.

The CBLE interface is compatible with IBM's CMS editor XEDIT and Mansfield Software's PC adaptation KEDIT.

The ISPF interface is compatible with IBM's TSO ISPF editor and includes support for the most commonly used primary and line (prefix) commands, scrolling and PF key/command line concatenation.

INTERFACE ISPF is the default on z/OS systems.  
INTERFACE CBLE (XEDIT) is the default on VM/CMS and VSE systems.

The interface can be set globally, at the file or view level, allowing different interfaces for different files and even different views of the same file.

The INTERFACE option setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

ISPF | XEDIT | CBLE

Set the current edit interface to be either CBLE (synonym XEDIT) or ISPF.

VIEW | FILE | GLOBAL

Set for the level at which the SET INTERFACE command will take effect.

VIEW	The current edit view only.
FILE	All edit views of the current file.
GLOBAL	All current and any new edit views.

INITIALISE

Initialise the default display area fields that correspond to the particular edit environment. This uses system defined defaults and any INI variable overrides. e.g. For ISPF, displays a SCROLL field in the command line. For CBLe, displays a reserved scale line as the first line of the edit display area.

**QUERY Response:**

Displays the current setting for INTERFACE (CBLe or ISPF).

**EXTRACT Rexx variables:**

interface.0	1
interface.1	CBLe   ISPF

**Example:**

```
INTERFace CBLe
```

Set the CBLe edit mode for the current file edit view.

```
INTERFACE ISPF FILE INI
```

Set the ISPF edit mode for all edit views of the current file and initialise their edit display areas to defaults associated with the ISPF interface.

**See Also:**

[ECOMMAND](#) [ICOMMAND](#)

---

## ISPFMODE - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+--- ISPFMode ---+- ON ---+----->>
  |         |         |         |         |
+- SET -----+         +- OFF -+-

>>--- Query ----- ISPFMode ----->>

>>--- EXtract ---- /ISPFMode/ ----->>
```

**Description:**

When running in an ISPF environment, SET ISPFMODE controls whether 3270 screen I/O is managed by ISPF (ON) or CBL3270 (OFF). This is the same as executing the FileKit command ISPF with no parameters which toggles ISPF screen management on and off.

If SET ISPFMODE is issued in a non-ISPF environment, the following error message is returned:

```
ZZSE095E ISPF is not available in the current environment.
```

With ISPFMODE ON, the menu bar item **Swap** is added to the FileKit Main Menu which, if selected, will execute **ISPF SWAP** to display an ISPF split screen.

SET ISPFMODE takes effect at the Global level.

**Set Options:**

```
ON
OFF
```

Set ISPFMODE ON or OFF. In an ISPF environment, the default is ON. Otherwise ISPF is OFF.

**QUERY Response:**

Displays the current setting for ISPFMODE (ON or OFF).

**EXTRACT Rexx variables:**

ispfmode.0	1
ispfmode.1	ON   OFF

---

## KEY - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+----- KEY --- kp1 -- kp2 -----><
    |         |
    +- SET -----+
```

```
>>--- Query ----- KEY -----><
```

```
>>--- EXtract --- /KEY/ -----><
```

**Description:**

Change the key position and key length for the current KSDS file. KEY may only be set for KSDS data sets that have not yet been defined.

SET KEY takes effect at the File level.

**Set Options:**

*kp1 kp2*  
Start and end positions of the key field. Both are mandatory for SET KEY.

**QUERY Response:**

Displays the start and end columns containing the key in the current KSDS file.

**EXTRACT Rexx variables:**

key.0	2
key.1	Start column of the primary key in the edited KSDS data set.
key.2	End column of the primary key in the edited KSDS data set.

**Examples:**

```
key 101 116
```

**See Also:**

SET/QUERY/EXTRACT Option: **DSORG**

---

## LASTMSG - QUERY/EXTRACT

---

**Syntax:**

```
>>--- Query ----- LASTmsg ---+-----+-----><
                                |         |
                                +- ASIS ---+
```

```
>>--- EXtract --- /LASTmsg/ -----><
```

**Description:**

QUERY and EXTRACT LASTMSG report the text of the last message displayed.

**QUERY Response:**

Displays the last message or error message generated for the current file view. This message may not have been displayed if NOMSG was used or MSGMODE was OFF.

The ASIS sub-parameter suppresses the string "ZZSE010I LASTMSG" which, by default, is prefixed to the last message string.

**EXTRACT REXX variables:**

lastmsg.0	1
lastmsg.1	Text of last message displayed (mixed case).

---

## LCOLOR, LCOLOUR - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>-----+-----+-----+ LCOLor  +-----+----->
      |         |         |         |
      +- SET  +-         +--- LCOLour +-
                                     +- Green  +---+ +- NONE  +---+
                                     |         |         |
>-----+ line-target +-----+-----+-----+-----+<<
      |         |         |         |         |         |
      +- Blue  +---+ +- BLINK  +---+ +- name  +-+
      +- Default ---+ +- REVvideo +-
      +- Pink  +---+ +- UScore  +---+
      +- Red   +---+
      +- Turquoise -+
      +- White  +---+
      +- Yellow +---+
      |         |         |         |
      +-----+-----+ name  +---+ OFF +-----+
                                     |         |
                                     +--- *  +---+
```

```
>>--- Query  +-----+ LCOLor  +-----+-----+<<
              |         |
              +--- LCOLour +-
```

```
>>--- EXtract --- / +--- LCOLor  +--- / -----+-----+<<
              |         |
              +--- LCOLour +-
```

**Description:**

Assign and unassign colours to text on all lines that contain the specified line-target string. The SET LCOLOUR command is influenced by the current setting for CASE and ZONE.

All SET LCOLOUR commands that have been issued for the file, are stored and applied in order to any new text entered in the file.

SET LCOLOUR takes effect at the File level.

**Set Options:**

**line-target**  
 Colour only those lines which satisfy the line-target condition. The line-target must be a string target.

**BLUE | DEFAULT | GREEN | PINK | RED | TURQUOISE | WHITE | YELLOW**  
 Supported colours. If DEFAULT is specified the 3270 hardware default is used.

**BLINK | NONE | REVVIDEO | USCORE**  
 Extended highlighting. The colour may blink, be displayed in reverse video or be underlined.  
 Default is NONE.

**name**  
 Unique name by which the SET LCOLOUR entry is referenced.  
 Name may be enclosed within delimiter characters which are included in QUERY and EXTRACT LCOLOUR output but do not form part of the name itself.

Name may be used to reference a specific SET LCOLOUR entry to set it off.

If the name used already exists for a prior SET LCOLOUR command, then the original SET LCOLOUR entry is set off and replaced by the new definition.

If name is not specified, it defaults to being the line-target string.

**Note:** name does not include any special keywords used in the line-target (i.e. WORD, PREFIX or SUFFIX.)

**\***  
 (Asterisk) references all SET LCOLOUR entries.

OFF

Remove a SET LCOLOUR entry and undo any colouring caused by that entry.

**QUERY Response:**

Displays the active LCOLOR definitions in the order that they were entered. The output also displays the SET CASE and SET ZONE options that were active when each LCOLOR command was issued.

**EXTRACT Rexx variables:**

lcolor.0 lcolour.0	Number of SET LCOLOR commands in effect for the current file.
lcolor.i lcolour.i	<p>All SET LCOLOR definitions and their associated parameters in the order in which they were entered (i=1 to lcolor.0).</p> <p>Each lcolor.i variable contains the following information:</p> <ol style="list-style-type: none"> <li>1. The string line-target (as entered on SET LCOLOR.) Note that the line-target may contain more than one token. e.g. "word /to/", "/string containing blanks/", etc.</li> <li>2. The string line-target (as entered on SET LCOLOR.)</li> <li>3. Colour in lower case.</li> <li>4. Extended highlighting in lower case.</li> <li>5. Name associated with the SET LCOLOR command in upper case. If no name was specified, the target string is used.</li> <li>6. "case" in lower case.</li> <li>7. Either "respect" or "ignore" in lower case. (The prevailing SET CASE value for string target search when SET LCOLOR was issued.)</li> <li>8. "zone" in lower case.</li> <li>9. Left zone value. (The prevailing SET ZONE left zone value when SET LCOLOR was issued.)</li> <li>10. Right zone value. (The prevailing SET ZONE right zone value when SET LCOLOR was issued.)</li> </ol> <p>e.g.</p> <pre> /=AB/ pink default /=AB/ case respect zone 1 1 /xYZ/ red revvideo sll case ignore zone 8 20                     </pre>

**Examples:**

```

set lcolour /==/ green uscore equal2
lcol /alloc/ yel rev "tso allocs"
lcol word level blue blink
                    
```

**See Also:**

SET/QUERY/EXTRACT Options: [CASE](#) [COLOUR](#) [HCOLOUR](#) [SCOLOUR](#) [ZONE](#)

**LENGTH - QUERY/EXTRACT**

**Syntax:**

```

>>--- Query ----- LENgth -----><
>>--- EXtract --- /LENgth/ -----><
                    
```

**Description:**

QUERY and EXTRACT LENGTH report the length of the focus line in the current text or data edit view.

**QUERY Response:**

Displays the length of data in the focus line. Trailing blanks are not included in this value.





**Set Options:**

CHANGE | NOCHANGE

Set the change flag bit on/off. Default is NOCHANGE.

NEW | NONEW

Set the new flag bit on/off. Default is NONEW.

TAG | NOTAG

Set the tag flag bit on/off. Default is NOTAG.

*group-target*

Group-target condition defining the end of the source target area. If the group-target is not satisfied then the DELETE command will fail.

**QUERY Response:**

Displays line flag settings for the focus line (NEW or NONEW, CHANGE or NOCHANGE and TAG or NOTAG).

**EXTRACT REXX variables:**

lineflag.0	3
lineflag.1	NEW   NONEW
lineflag.2	CHANGE   NOCHANGE
lineflag.3	TAG   NOTAG

**Examples:**

set lineflag nonew

Set the new flag bit off for the focus line only.

set lineflag new nochange /##/

Set the new flag bit on and the change flag bit off for the focus line and all lines up to, but not including, the first line to contain the string "##".

set lineflag tag 20

Set the tag flag bit on for the focus line and the 19 lines that follow.

**See Also:****TAG**

---

**LINEND - SET/QUERY/EXTRACT**

---

**Syntax:**

```
>>+-----+----- LINEND ---+ ON ---+-----+----->>
  |         |         |         |         |         |
+- SET -----+         +- OFF -+ +- cI -+

>>--- Query ----- LINEND ----->>

>>--- EXTRACT ----- /LINEND/ ----->>
```

**Description:**

LINEND (Line End) determines whether multiple commands may be issued from the command line of the current edit view by separating each command with the line end character. LINEND may also be used to define the line end character.

The LINEND option is obeyed for all text edit window views. For all other FileKit application windows, command separation is always active and obeys the command delimiter assigned in the **System Settings (=0.2)** panel.

Note that active command separation may be temporarily suspended for an entered command by preceding it with the currently assigned line end character. e.g. Where LINEND ON ; is active...

```
;imm 'extract color' ;do i=1 to color.0; 'msg' color.i; end
```

SET LINEND takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.



**EXTRACT Rexx variables:**

listfileaction.0	1
listfileaction.1	BROWSE   ERASE   NONE

**Example:**

```
listfileact browse
    Change default action to Edit read/only.
```

## LOADWARNING - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+----- LOADWarning +-----+----->>
  |         |         |         |         |
  +- SET -----+         +- OFF ---+   +- nnn  ---+
                                     |         |
                                     +- nnnK ---+
                                     |         |
                                     +- nnnM ---+

>>--- Query ----- LOADWarning ----->>

>>--- EXtract ---- /LOADWarning/ ----->>
```

**Description:**

When a file is opened for edit, the contents of the file are first loaded into storage. LOADWARNING defines the warning threshold value for the current number of bytes loaded for the edited file. When the count of the number of bytes loaded exceeds a value that is a multiple of the loadwarning value, a warning popup message window is displayed.

The purpose of the file load warning threshold is to stop a user from accidentally editing a very large file. If a LoadWarning value has never been set, the default value is 1M (one megabyte).

SET LOADWARNING takes effect at the Global level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

ON | OFF  
 Enable or Disable file size checking during load. Default is ON.

nnn | nnnK | nnnM  
 Number of bytes loaded after which a warning is issued. This value may be specified as a number of bytes (*nnn*), number of Kilobytes (*nnn\*1024*) or a number of Megabytes (*nnn\*1024\*1024*).

**QUERY Response:**

Displays the current status of LOADWARNING (ON or OFF) and the text editor file load warning threshold in number of bytes.

**EXTRACT Rexx variables:**

loadwarning.0	2
loadwarning.1	ON   OFF
loadwarning.2	Text editor file load warning threshold in number of bytes.

**Example:**

```
loadw on 8M
    When a new file is edited, open a warning popup window after every 8 megabytes of data is loaded.
```

**See Also:**

SET/QUERY/EXTRACT Option: **SIZEWARNING**

## LRECL - SET/QUERY/EXTRACT

### Syntax:

```
>>+-----+----- LRecl --- n_bytes -----><
    |         |
    +- SET -----+

>>--- Query ----- LRecl -----><

>>--- EXTract --- /LRecl/ -----><
```

### Description:

Change the logical record length value of the in-storage data in the current text or data edit view.

The **Recl=** value in the status bar and LRECL value in the title bar of the text or data edit window view are updated to reflect the change.

For SDE data edit window views, SET LRECL fails and returns an error if the LRECL value specified is less than the maximum potential record length that may be mapped by any record type that is currently associated with records in the file.

e.g. A record type "Statistics" may contain variable length and variable numbers of fields so that, potentially, a record with a length in the range 125 to 350 may be mapped by this record type. If, when editing a structured file, record type "Statistics" is used to map at least one record, then the LRECL may not be set to a value lower than 350. If further record types are used that are capable of mapping longer records, then this LRECL minimum value will be higher.

Following a save operation, the edited data will be saved to disk using the changed lrecl value. For z/OS, LRECL may only be changed where the fileid of the currently edited data is not that of an existing data set. (i.e. a new data set will be allocated when the data is saved.)

SET LRECL takes effect at the File level.

### Set Options:

*n\_bytes*

The logical record length.

For a RECFM F (fixed record format) file, *n\_bytes* is the length of each record. CBL will pad short lines with blanks and truncate long lines.

For a RECFM V (variable record format) file, *n\_bytes* is the maximum record length. Short lines will not be padded, but long lines will be truncated.

### QUERY Response:

Displays the defined LRECL value of the file in the current edit view and the longest record encountered when the file was loaded.

### EXTRACT REXX variables:

lrecl.0	2
lrecl.1	Defined LRECL value of the current file.
lrecl.2	Length of longest record on load of the current file.

## LSCREEN - QUERY/EXTRACT

### Syntax:

```
>>--- Query ----- LSCReen -----><

>>--- EXTract --- /LSCReen/ -----><
```

**Description:**

QUERY and EXTRACT LSCREEN report information relating to the size and location of the focus document (MDI child) window and its client area.

**QUERY Response:**

Displays size and location information about the current logical screen (MDI child window and MDI client area). These values are what would need to be specified on the text and data edit options, WINSIZE and WINPOS, in order to achieve the size and location display characteristics of the focus edit view.

Values are displayed in the following order:

1. MDI child window depth (number of rows).
2. MDI child window width (number of columns).
3. MDI child window vertical position within the client area (row number).
4. MDI child window horizontal position within the client area (column number).
5. MDI client area depth (number of rows).
6. MDI client area width (number of columns).

**EXTRACT Rexx variables:**

lscreen.0	6
lscreen.1	MDI child window depth (number of rows).
lscreen.2	MDI child window width (number of columns).
lscreen.3	MDI child window vertical position within the client area (row number).
lscreen.4	MDI child window horizontal position within the client area (column number).
lscreen.5	MDI client area depth (number of rows).
lscreen.6	MDI client area width (number of columns).

**See Also:**

SET/QUERY/EXTRACT Options: [WINPOS](#) [WINSIZE](#)

---

## MACRO - QUERY

---

**Syntax:**

```
>>--- Query ----- MACRO ----+-----+-----+-----><
                                |         |
                                +- macroname ---+
```

**Description:**

QUERY MACRO displays the name and contents of macros that have been permanently loaded into storage by primary command DEFINE.

**QUERY Response:**

Displays the name and all lines belonging to all macros loaded into storage by a DEFINE command. If *macroname* is specified, then output is restricted to the name and contents of the specified, in-storage macro only.

The following message denotes the start of a defined macro:

```
ZZSE010I MACRO Name=<macroname> Text=Loaded from <location> ...
```

**See Also:**

[DEFINE](#)



**See Also:**

MACRO IMMEDIATE

---

**MBR - SET/QUERY/EXTRACT**

---

SET/QUERY/EXTRACT Option: **FNAME**

---

**MSGLINE - SET/QUERY/EXTRACT**

---

**Syntax:**

```

>>+-----+--- MSGLine --- ON -- line --+-----+-----><
    |           |           |           |           |
+- SET -----+           +- n -----+
                               |           |
                               +- OVERLAY -+

```

```

>>--- Query ----- MSGLine -----><

```

```

>>--- EXtract ---- /MSGLine/ -----><

```

**Description:**

MSGLINE defines the line number and number of lines CBL e uses for its output message lines. The line number is specified as being relative to the top, middle or bottom of the document window.

SET MSGLINE takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

*line*  
Line number of first message line relative to the top, middle or bottom of the document window.

Line numbers relative to the top of the document window are specified as positive integers. e.g. 2, 8

Line numbers relative to the middle of the document window are specified as offsets from M. e.g. M+1, M-3

Line numbers relative to the bottom of the document window are specified as negative integers. e.g. -6, -10

Initially line is set to 1.

*n*  
The number of message lines. CBL e messages may need to occupy more than one message line. If the number of message lines are exceeded or a message overlaps the command line, a pop-up message window entitled "CBL Text Edit Messages" is opened and all the CBL e messages displayed.

If *n* is not specified, the number of message lines last defined in the current CBL e view, is unchanged. Initially *n* is set to 20.

OVERLAY  
Specifies that the first message line should overlay a line normally used to display a line of the file. If omitted, the first message line will be reserved for message display only.

Initially OVERLAY is set on.

**QUERY Response:**

Displays the current setting for MSGLINE (ON or OFF), the line number of the first line used for messages, the number of lines that can be used for messages and OVERLAY if the first message line can overlay a file line.



**EXTRACT Rexx variables:**

msgline.0	4
msgline.1	Always ON
msgline.2	Line number within the client area of the text or data editor document window at which the first message line is located.
msgline.3	The number of message lines.
msgline.4	OVERLAY if the first message line is to overlay a line of displayed data.

**Example:**

```
msgline on 5 20
    Allocate 20 message lines beginning 5 lines down from the top of document window.

msgline on M+3 2
    Allocate 2 message lines beginning 3 lines down from the middle of document window.

msgline on -6 6
    Allocate 6 message lines beginning 6 lines up from the bottom of document window.
```

**See Also:**

SET/QUERY/EXTRACT Option: **MSGMODE**

---

## MSGMODE - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+----- MSGMode ---+- ON ---+-+-----+-----><
    |         |         |         |         |         |
    +- SET ---+-----+-----+-----+-----+-----+
    |         |         |         |         |         |
    +- OFF -+-+-----+-----+-----+-----+-----><
    |         |         |         |         |         |
    +- WRAP ---+-----+-----+-----+-----+-----><
    |         |         |         |         |         |
    +- NOWRAP -+-+-----+-----+-----+-----+-----><

>>--- Query ----- MSGMode -----><

>>--- EXtract ----- /MSGMode/ -----><
```

**Description:**

MSGMODE defines whether messages and error messages are displayed.  
 Parameters WRAP/NOWRAP are not supported for SDE window views.  
 SET MSGMODE takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

```
ON
OFF
```

MSGMODE ON will allow display of messages and error messages at the defined message lines.

**Note:** QUERY and EXTRACT LASTMSG will retrieve the text of last message issued regardless of the current setting for MSGMODE.

```
WRAP
NOWRAP
```

Not supported in SDE window views.  
 Determines whether message text with length greater than the width of the edit view window, wraps onto the next message line.  
 Default is WRAP.

**QUERY Response:**

Displays the current setting for MSGMODE (ON or OFF).

**EXTRACT Rexx variables:**

msgmode.0	1
msgmode.1	ON   OFF
msgmode.2	WRAP   NOWRAP

**See Also:**SET/QUERY/EXTRACT Option: **MSGLINE**

---

**NBWINDOW - QUERY/EXTRACT**

---

**Syntax:**

&gt;&gt;--- Query ----- NBWindow -----&lt;&lt;

&gt;&gt;--- EXtract --- /NBWindow/ -----&lt;&lt;

**Description:**

QUERY and EXTRACT NBWINDOW report information relating to the number of edit views.

**QUERY Response:**

Displays the following:

1. The total number of text edit views (MDI child windows) in the current text editor or SELCOPY Debug application main window (MDI parent).
2. The total number of text edit views displaying the same text (i.e have the same fileid) as the current edit view.
3. The current view's reference number in relation to other views displaying the same text.

**EXTRACT Rexx variables:**

nbwindow.0	3
nbwindow.1	Number of MDI child window text edit views.
nbwindow.2	Number of edit views that display the same text as the current text edit view.
nbwindow.3	Reference number of the current edit view in relation to other text edit views displaying the same text.

---

**OPSYS - QUERY/EXTRACT**

---

**Syntax:**

&gt;&gt;--- Query ----- OPSys -----&lt;&lt;

&gt;&gt;--- EXtract --- /OPSys/ -----&lt;&lt;

**Description:**

QUERY and EXTRACT OPSYS report information relating to operating environment in which FileKit is executing.

**QUERY Response:**

Displays operating system name and release.

**EXTRACT Rexx variables:**

opsys.0	3
opsys.1	Operating System name.
opsys.2	Operating System release.
opsys.3	Reserved (Null string).

---

## PFKEY - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+-----+ PFkey -- n +-----+-----+<<
  |         |         |         |         |         |
+- SET -----+ --- PFn -----+ +- BEFORE -+ +- string -+

>>--- Query -----+--- PFkey -- n +-----+-----+<<
          |         |         |         |
          +--- PFn -----+

>>--- EXtract - / +--- PFkey --+-----+--- / -----+<<
          |         |         |         |
          |         |         |         |
          +--- PFn -----+
          +- n ---+
          |         |
          +-----+-----+
          |         |
          +-----+-----+
```

**Description:**

Defines the action of a Program Function (PF) Key for the current text or data editor window view.

No separating blanks need be specified between the PFKEY option keyword and the pfkey number *n*. e.g. SET PFK13 HELP

When KEYLISTs are active (default for z/OS), PFKEY sets a key definition in a temporary keylist which, if not already assigned as a result of a previous SET PFKEY command, gets assigned to the current edit view. The temporary keylist name is @TMP*nnnn* where *nnnn* is a value in the range 0000-9999. The name of the first temporary keylist created within a FileKit session, is assigned the value 0000. The names of all temporary keylists created thereafter are assigned a name with a value incremented by 1.

**Note:** PFKEY does not affect the PFKey settings for named keylists (or, if KEYLIST OFF is in effect, the Class, Default, TitleBars and Borders key values) as set via the FileKit **KEYS** primary command and dialog panel.

SET PFKEY takes effect at the View level.

**Set Options:**

*n*  
Program Function Key number (1 to 24).

BEFORE  
Applicable only when KEYLIST OFF is in effect, the BEFORE keyword indicates that the action assigned to the Program Function Key will be executed **before** the contents of the command line and also before any outstanding changes to the file are committed to the in-storage copy of the edited data.  
Default is to execute the PFKey action **after** interpreting all other actions (command line contents, text changes, etc.)

*string*  
The command string to be assigned to the PFKey. If *string* is not specified, the PFKey is unassigned. If KEYLIST OFF is in effect, since PFKEY unassigns a PFKey value at the Window level, the PFKey will potentially action a value assigned at a lower level (e.g. the Class or Default level). This hierarchy of PFKey values is not applicable when KEYLIST usage is active.

**QUERY Response:**

Displays the currently assigned definition for the specific PFKey.

**EXTRACT Rexx variables:**

Where a specific PFKey number *n* is included.

pfkey.0	1
pfkey.1	Current PFKey assignment for specific PFKey.

Where **no** specific PFKey is included.

pfkey.0	24
pfkey.i	Current PFKey assignments for PFKeys 1-24. (i=1,24).

### Example:

```
pfkey 3 undo
Assign UNDO command to F3.
```

```
pf8 macro hs
Assign MACRO HS to F8.
```

### See Also:

FileKit environment primary commands: [KEYLIST](#) [KEYS](#)

---

## POINT - SET/QUERY/EXTRACT

---

### Syntax:

```
>>+-----+--- Point --- .name ---+ ON ---+-----+<<
|         |         |         |         |         |
+- SET ----+         +- OFF --+
>>--- Query ----- Point ---+-----+<<
|         |         |
+- * --+
>>--- EXtract ---- /Point ---+-----+ / ---+-----+<<
|         |         |
+- * --+
```

### Description:

Assign or unassign a label name to the focus line for subsequent use as a line target.

A line may be assigned more than one name, however, the same name may not be assigned to more than one line in the current file. Where the name is already assigned to a line in the current file, it is unassigned from that line and reassigned to the focus line.

A line name remains assigned to a line following changes to the text or a MOVE command.

SET POINT takes effect at the File level.

### Set Options:

*.name*  
A label name to be assigned to or unassigned from the focus line. The specified name may be of any length, may contain and begin with any alphanumeric or special character, but must be preceded by a "." (dot/period).

OFF  
Unassign the specified name from the focus line.

### QUERY Response:

If "\*" (asterisk) is not specified, QUERY POINT displays the line number and all label names currently allocated to the focus line. If no label names are allocated to the focus line, the following message is returned:

```
ZZSE010I POINT No points assigned to line n
```

If **QUERY POINT \*** is executed, a message line is displayed containing the line number and label names, one for each line within the current edit view that has been assigned a label name.

### EXTRACT REXX variables:

Where **no** "\*" asterisk is specified.

point.0	0 if focus line is not assigned a label name, otherwise 1.
point.1	Line number and label names assigned to the focus line.

Where "\*" asterisk is specified.

point.0	Number of lines within the current edit view for which a label name is assigned.
point.i	Line number and label names assigned to the ith line for which a label name is assigned.

**Example:**

```
point .joe
    Assign the name ".joe" to the focus line.

p .sale off
    Unassign the name ".sale" from the focus line.
```

**See Also:**

SET/QUERY/EXTRACT Option: **SETPT**

## POPUP - EXTRACT

**Syntax:**

```
>>--- EXtract --- /POPUP/ -----<<
```

**Description:**

For use in macros, EXTRACT POPUP obtains the text of the menu item selected by the user following the last execution of the POPUP primary command within the same macro. POPUP information does not persist beyond the life of the macro.

**EXTRACT Rexx variables:**

popup.0	1
popup.1	The contents of the menu item selected by the user. This value is returned exactly as specified on the POPUP command, and includes any blanks present in the original string. If no item was selected (because a 3270 AID other than <Enter> was received or because no valid entry was selected), this variable is set to the null string.

**See Also:**

**POPUP**

## PREFIX - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+--- PREFIX ---+ ON ---+-----+-----<<
  |         |         |         |         |         |
  +- SET ---+         +- OFF -+ +- Right -+ +- n_cols -+
                               |         |
                               +- Left --+

>>--- Query ----- PREFIX -----<<

>>--- EXtract ---- /PREFIX/ -----<<
```

**Description:**

PREFIX defines whether the prefix area is displayed and whether it is displayed on the left of the window view or on the right.

The prefix area displays the line number of lines in the window view.

**Prefix area commands** are entered in the prefix area.

SET PREFIX takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

### Set Options:

ON  
OFF

ON displays the prefix area, OFF suppresses display of the prefix area. Initially, PREFIX is ON.

RIGHT  
LEFT

Display the prefix area at the right or left edge of the window view. If neither RIGHT nor LEFT are specified, the position of the prefix area last defined in the current CBL view, is unchanged. Initially, the prefix area is on the RIGHT.

*n\_cols*

Width of the prefix area in number of columns. Default is 6.

### QUERY Response:

Displays the current setting for PREFIX (ON or OFF) including the relative position of the prefix area in the text edit window (LEFT or RIGHT) and its width in number of bytes.

### EXTRACT Rexx variables:

prefix.0	3
prefix.1	ON   OFF   NULLS
prefix.2	LEFT   RIGHT
prefix.3	Width of prefix area in number of bytes.

### Example:

```
pre on left
```

Display the prefix area on the left of the current window view.

### See Also:

[Prefix Commands](#)

---

## PSCOPE - SET/QUERY/EXTRACT

---

### Syntax:

```
>>+-----+--- PSCOpe ---+ All -----><
  |         |         |         |         |
  +- SET ----+         +- Display +-><

>>--- Query ----- PSCOpe -----><

>>--- EXTract ---- /PSCOpe/ -----><
```

### Description:

PSCOPE defines whether lines with selection levels that exclude them from the current display, are respected or ignored by text edit prefix area and ISPF edit line commands.

SET PSCOPE ALL is default for interface ISPF. SET PSCOPE DISPLAY is default for interface CBL.

SET PSCOPE takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

### Set Options:

ALL

Respect excluded lines for all CBL prefix area and ISPF line commands.

The only exception to this is '/' (slash - x'61') which makes the line the current (top) line of the display.

**DISPLAY**

Execute CBL prefix area and ISPF line commands against displayed lines only. i.e. Ignore excluded lines.

**QUERY Response:**

Displays the current setting for PSCOPE (ALL or DISPLAY) which determines whether excluded lines are respected or ignored by CBL prefix or ISPF Edit style line commands.

**EXTRACT Rexx variables:**

pscope.0	1
pscope.1	ALL   DISPLAY

**See Also:**

SET/QUERY/EXTRACT Option: **SCOPE**

---

## RANGE - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>+-----+--- RANge ---+-- line-target1 -- line-target2 --+-----<<
  |         |         |         |         |         |         |
  +- SET ----+         +----- BLock -----+
>>--- Query ----- RANge -----<<
>>--- EXtract ---- /RANge/ -----<<
```

**Description:**

Restricts the edit view to a range of consecutive lines within which CBL commands are to operate.

The top and bottom lines of the range are equal to the lines referenced by line-target1 and line-target2 respectively, or the top and bottom lines of a marked block.

SET RANGE does not change line numbering but those lines whose line numbers fall outside the range, are removed from the edit view and are not affected by subsequent edit commands. However, CBL commands that write data to disk (e.g. SAVE, FILE) will write all the file data regardless of the range of lines in view.

Where the top line of the range is not the first line of the file, the "Top of File" line becomes "Top of Range (Line=nnn)". Similarly, where the bottom line of the range is not the last line of the file, the "End of File" line becomes "End of Range (Line=mmm)".

SET RANGE takes effect at the View level.

**Set Options:**

```
line-target1
line-target2
```

Line targets identifying the first and last lines of the range.

Relative line number targets, string targets and line class targets are always determined relative to the current focus line. Unlike the LOCATE command, the focus line is not changed to be the line found by line-target1. Therefore line-target2 will be determined from the same focus line as target-line1. This may result in a non-ascending range of lines which would return the following error message:

```
ZZSE064E Range is invalid: end nn is less than top mm.
```

**BLOCK**

Use the top and bottom lines of a marked block as the limits of the range.

**QUERY Response:**

Displays the line number of the first and last lines in the current range.

**EXTRACT Rexx variables:**

range.0	2
range.1	Line number of first line in range
range.2	Line number of last line in range. (Equal to one less than RANGE.1 if no lines in current range)

**Examples:**

```
set range :10 :20
range 10 /==SELCEnd==/
range -* *
ran -/Section 1/ /Section 2/
ran :22 word /idle/
ran block
```

**See Also:**

SET/QUERY/EXTRACT Option: **ZONE**

## RECFM - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+--- RECFM ---+ V +-----><
|         |         |         |         |
+- SET ----+         +- F ----+

>>--- Query ----- RECFM -----><

>>--- EXtract ----- /RECFM/ -----><
```

**Description:**

Change the record format of the currently edited in-storage data.

The **Fmt=** value in the status bar and RECFM value in the title bar of the CBL or SDE edit window view are updated to reflect the change.

Following a save operation, the edited data will be saved to disk using the changed record format value. For MVS, RECFM may only be changed where the fileid of the currently edited data is not that of an existing data set. (i.e. a new data set will be allocated when the data is saved.)

**Note:** SET RECFM is not yet implemented for VSE.

SET RECFM takes effect at the File level.

**Set Options:**

- V Record format is variable.
- F Record format is fixed.

**QUERY Response:**

Displays the current file's Record Format.

**EXTRACT Rexx variables:**

recfm.0	1
recfm.1	FIXED   VARIABLE

**See Also:**

SET/QUERY/EXTRACT Option: **LRECL**



## REDO - QUERY/EXTRACT

See information on the SET/QUERY/EXTRACT Option: [UNDO](#)

## RESERVED - SET/QUERY/EXTRACT

### Syntax:

```
>>+-----+--- RESERved --- line --+ OFF -----+-----><
    |         |         |         |         |         |
    +- SET -----+         +- colour +-----+
                                     |         |
                                     +- string -+
```

```
>>--- Query ----- RESERved -----><
```

```
>>--- EXtract ---- /RESERved/ -----><
```

### Description:

Reserve a line and optionally insert a coloured text string in the CBL or SDE window view for the current file. The line number is specified as being relative to the top, middle or bottom of the document window.

A reserved line may be implemented to display useful information such as the structure of text fields in the current file.

SET RESERVED takes effect at the File level.

### Set Options:

- line* Line number of the reserved line relative to the top, middle or bottom of the document window.  
Line numbers relative to the top of the document window are specified as positive integers. e.g. 2, 8  
Line numbers relative to the middle of the document window are specified as offsets from M. e.g. M+1, M-3  
Line numbers relative to the bottom of the document window are specified as negative integers. e.g. -6, -10
- OFF* Switch off RESERVE for the specified line number.
- colour* The colour assigned to the reserved line. Supported colours and extended highlighting are described in SET COLOUR.
- string* Text string to be inserted in the reserved line. Where no string is specified, the reserved line will be blank.

### QUERY Response:

Displays the line numbers of all reserved lines in the current edit view.

### EXTRACT Rexx variables:

reserved.0	0 if no reserved lines, otherwise 1.
reserved.1	List of reserved line numbers, if any.

### Example:

```
reserve m-2 yellow rev |-Key-| |---Name---| |-----email-----|
Define a reserved line 2 lines up from the middle of the document window. The reserved line will contain a text string describing columns of data coloured in reverse video yellow.
```

```
reserve 3 blue F1=Top F2=Bottom F3=Quit
Define a reserved line 3 lines down from the top of the document window. The reserved line will contain a text string describing PF Keys coloured in blue.
```

**See Also:**

SET/QUERY/EXTRACT Option: **COLOUR**

## RING - QUERY/EXTRACT

**Syntax:**

```
>>--- Query ----- RING -----><
>>--- EXtract --- /RING/ -----><
```

**Description:**

QUERY and EXTRACT RING reports information on each file currently open for edit by the text editor.

**QUERY Response:**

Displays the number of files currently edited in the ring of text editor views, followed by a line for each edited file containing that file's alteration count, total number of records and fileid.

**EXTRACT Rexx variables:**

ring.0	Number of files being edited in the text editor ring.
ring.i	Information on each file being edited in the ring (i=1 to ring.0).  The tokens returned in ring.i are:  <ol style="list-style-type: none"> <li>1. Fileid.</li> <li>2. Line= Line number of current line.</li> <li>3. Col= Column number of current column.</li> <li>4. Size= File Size (number of lines.)</li> <li>5. Alt= Alteration count.</li> </ol> e.g.  CBL.CMX(NBJ) Line=23 Col=1 Size=429 Alt=0,0 NBJ.TEST Line=0 Col=67 Size=1 Alt=7,7

**See Also:**

SET/QUERY/EXTRACT Options: **ALT FILEID**

## SAVEOPTIONS - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+--- SAVEOPTions ---+ ON ---+><
    |         |         |         |         |
+- SET -----+         +- OFF --+><

>>--- Query ----- SAVEOPTions -----><

>>--- EXtract ----- /SAVEOPTions/ -----><
```

**Description:**

This option controls whether or not the following text edit options are saved for use in subsequent text edit views opened in the current and subsequent FileKit sessions:

ARBCHAR AUTOSAVE CASE	HSCROLLCURSOR IMPACRO INTERFACE	SCOPE SHADOW SIZEWARNING
-----------------------------	---------------------------------------	--------------------------------

```

CMDDEF
CMDLINE
COLOR, COLOUR
DEFPROFILE
ENVVARS
GENSAVE
HCOLOR, HCOLOUR
HEXSTRING

```

```

LINEND
LISTFILEACTION
LOADWARNING
MSGLINE
MSGMODE
PREFIX
PSCOPE
SCALE

```

```

STAY
STREAM
SYNONYM
THIGHLIGHT
UNDOING
VARBLANK
VIEW
WRAP

```

SET SAVEOPTIONS takes effect at the Global level.

#### QUERY Response:

Displays the current setting for SAVEOPTIONS (ON or OFF).

#### EXTRACT Rexx variables:

saveoptions.0	1
saveoptions.1	ON   OFF

---

## SCALE - SET/QUERY/EXTRACT

---

#### Syntax:

```

>>+-----+--- SCALE ---+ ON ---+-----+<<
  |         |         |   |   |         |
+- SET ----+         +- OFF -+ +- line -+
>>--- Query ----- SCALE -----<<
>>--- EXtract ---- /SCALE/ -----<<

```

#### Description:

SCALE defines whether the scale line is displayed in the CBL document window and optionally the line number at which the scale line is displayed.

The line number is specified as being relative to the top, middle or bottom of the document window.

The scale line not only identifies the column numbers, but displays the current left and right ZONE columns as "<" (less than) and ">" (greater than) respectively, and the current column as "|" (vertical bar).

SET SCALE takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

#### Set Options:

ON  
OFF

ON displays the scale line, OFF suppresses display of the scale line. Initially, SCALE is ON.

*line*

Line number of the scale line relative to the top, middle or bottom of the document window.

Line numbers relative to the top of the document window are specified as positive integers. e.g. 2, 8

Line numbers relative to the middle of the document window are specified as offsets from M. e.g. M+1, M-3

Line numbers relative to the bottom of the document window are specified as negative integers. e.g. -6, -10

#### QUERY Response:

Displays the current setting for SCALE (ON or OFF) including the location of the scale line in the text edit window.

#### EXTRACT Rexx variables:

scale.0	3
scale.1	ON   OFF
scale.2	As defined, the position of the scale line relative to the top, middle or bottom of the text edit view.
scale.2	Line number of the scale line relative to the top of the edit view window.

**Example:**

```
set scal on 2
    Display the scale line 2 lines down from the top of the document window.
```

```
scale off
    Suppress display of the scale line.
```

---

## SCOLOR, SCOLOUR - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>-----+--- SCOLor ----->
  |         |         |
+- SET -+ +- SCOLour -+
  |         |         |
  +- Green -----+ +- NONE -----+
  |         |         |
>-----+ line-target -----<<
  |         |         |         |         |
  +- Blue -----+ +- BLInk -----+ +- name ---+
  +- Default ---+ +- REVvideo -+
  +- Pink -----+ +- Uscore ---+
  +- Red -----+
  +- Turquoise -+
  +- White -----+
  +- Yellow -----+
  |         |         |         |
  +-----+ name ---+ OFF -----+
  |         |         |
  +--- * ---+

>>--- Query ---+ SCOLor -----<<
  |         |         |
  +- SCOLour -+

>>--- EXtract --- / ---+ SCOLor ---+ / -----<<
  |         |         |
  +- SCOLour -+
```

**Description:**

Assign and unassign colours to all occurrences of the specified line-target string. The SET SCOLOR command is influenced by the current setting for CASE and ZONE.

All SET SCOLOR commands that have been issued for the file, are stored and applied in order to any new text entered in the file.

SET SCOLOR takes effect at the File level.

**Set Options:**

*line-target*  
The line-target string to be coloured.

BLUE | DEFAULT | GREEN | PINK | RED | TURQUOISE | WHITE | YELLOW  
Supported colours. If DEFAULT is specified the 3270 hardware default is used.

BLINK | NONE | REVVIDEO | USCORE  
Extended highlighting. The colour may blink, be displayed in reverse video or be underlined.  
Default is NONE.

*name*  
Unique name by which the SET SCOLOR entry is referenced.

Name may be enclosed within delimiter characters which are included in QUERY and EXTRACT SCOLOR output but do not form part of the name itself.

Name may be used to reference a specific SET SCOLOR entry to set it off.

If the name used already exists for a prior SET SCOLOR command, then the original SET SCOLOR entry is set off and replaced by the new definition.

If name is not specified, it defaults to being the line-target string.

**Note:** name does not include any special keywords used in the line-target (i.e. WORD, PREFIX or SUFFIX.)

\* (Asterisk) references all SET SCOLOR entries.

OFF Remove a SET SCOLOR entry and undo any colouring caused by that entry.

**QUERY Response:**

Displays all the current syntax colouring settings for fields in the focus text edit view.

**EXTRACT REXX variables:**

scolor.0 scolour.i	Number of SET SCOLOR commands in effect for the current file.
scolor.i scolour.i	<p>All SET SCOLOR definitions and their associated parameters in the order in which they were entered (i=1 to scolor.0).</p> <p>Each scolor.i variable contains the following information:</p> <ol style="list-style-type: none"> <li>1. The string line-target (as entered on SET SCOLOR.) Note that the line-target may contain more than one token. e.g. "word /to/", "/string containing blanks/", etc.</li> <li>2. Colour in lower case.</li> <li>3. Extended highlighting in lower case.</li> <li>4. Name associated with the SET SCOLOR command in upper case. If no name was specified, the target string is used.</li> <li>5. "case" in lower case.</li> <li>6. Either "respect" or "ignore" in lower case. (The prevailing SET CASE value for string target search when SET SCOLOR was issued.)</li> <li>7. "zone" in lower case.</li> <li>8. Left zone value. (The prevailing SET ZONE left zone value when SET SCOLOR was issued.)</li> <li>9. Right zone value. (The prevailing SET ZONE right zone value when SET SCOLOR was issued.)</li> </ol> <p>e.g.</p> <pre> /&lt;/ red default /&lt;/ case respect zone 1 * 'IQ00' yellow uscore Q1 case ignore zone 20 40                     </pre>

**Examples:**

```

set sc colour /*/ green none asterisk
scol help yel uscore "help items"
scol pre 'sup' pink blink
    
```

**See Also:**

SET/QUERY/EXTRACT Options: [CASE](#) [COLOR](#) [HCOLOR](#) [LCOLOR](#) [ZONE](#)

**SCOPE - SET/QUERY/EXTRACT**

**Syntax:**

```

>>+-----+----- SCOPE ---+ All ----->>
  |         |         |         |         |
+- SET -----+         +- Display -+
>>--- Query ----- SCOPE ----->>
>>--- EXtract ----- /SCOPE/ ----->>
    
```

**Description:**

SCOPE defines whether lines with selection levels that excludes them from the current display, are respected or ignored by most CBL commands.

On successful execution of an ALL **line-target** command, SCOPE DISPLAY is set automatically.

SET SCOPE takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

### Set Options:

**ALL**  
Respect excluded lines for all CBL commands.

**DISPLAY**  
Execute text edit primary commands against displayed line only. i.e. Ignore excluded lines.  
Exceptions to this are SAVE, FILE and SORT which operate on all lines in the display, regardless of SCOPE setting.  
Initially, SCOPE is DISPLAY

### QUERY Response:

Displays the current setting for SCOPE (DISPLAY or ALL).

### EXTRACT Rexx variables:

scope.0	1
scope.1	DISPLAY   ALL

### See Also:

**ALL** and the SET/QUERY/EXTRACT Options: **DISPLAY SELECT SHADOW**

## SCREEN - QUERY/EXTRACT

### Syntax:

```
>>--- Query ----- SCReen -----><
>>--- EXTract --- /SCReen/ -----><
```

### Description:

QUERY and EXTRACT SCREEN report the dimensions of the physical 3270 terminal in which FileKit is executing.

### QUERY Response:

Displays the number of rows and columns available in the 3270 terminal screen.

### EXTRACT Rexx variables:

screen.0	2
screen.1	Depth of 3270 terminal screen.
screen.2	Width of 3270 terminal screen.

### See Also:

SET/QUERY/EXTRACT Options: **FLSCREEN LSCREEN**

## SELECT - SET/QUERY/EXTRACT

### Environments:

SELECT primary command exists in the following application environments.



**Example:**

```
sel 8 /SELC/
    Set selection level 8 to the focus line and lines up to, but not including, the first line to contain the string "SELC".

select +1
    Add 1 to the selection level of the focus line.
```

**See Also:**

**ALL** and the SET/QUERY/EXTRACT Options: **DISPLAY SCOPE**

## SHADOW - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+----- SHADow ---+ ON ---+-----><
    |         |         |         |         |
    +- SET -----+         +- OFF --+
>>--- Query ----- SHADow -----><
>>--- EXtract ---- /SHADow/ -----><
```

**Description:**

SHADOW defines whether a shadow line is displayed to represent a line or group of lines that are excluded from the current CBLe view.

Each shadow line indicates the number of consecutive lines that have been excluded from that position in the text editor view.

SET SHADOW takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

```
ON
OFF
    Set SHADOW ON or OFF. Default is ON.
```

**QUERY Response:**

Displays the current setting of SHADOW (ON or OFF).

**EXTRACT REXX variables:**

shadow.0	1
shadow.1	ON   OFF

**See Also:**

**ALL** and the SET/QUERY/EXTRACT Options: **SELECT DISPLAY**

## SIZE - QUERY/EXTRACT

**Syntax:**

```
>>--- Query ----- SIZe -----><
>>--- EXtract --- /SIZe/ -----><
```





Enable (ON) or Disable (OFF) file size checking prior to load. Default is ON.

*nnn*  
*nnnK*  
*nnnM*

Number of bytes defining the maximum size of any file that can be loaded before the file size warning popup window is opened. This value may be specified as a number of bytes (*nnn*), number of Kilobytes (*nnn\*1024*) or a number of Megabytes (*nnn\*1024\*1024*).

### QUERY Response:

Displays the current file edit file size warning threshold in number of bytes.

### EXTRACT Rexx variables:

sizewarning.0	2
sizewarning.1	ON   OFF
sizewarning.2	File edit file size warning threshold in number of bytes.

### Example:

```
SIZEW ON 10K
```

When a new file is edited, open a warning popup window if its size exceeds 10 kilobytes.

### See Also:

SET/QUERY/EXTRACT Option: **LOADWARNING**

---

## STAY - SET/QUERY/EXTRACT

---

### Syntax:

```
>>+-----+--- STAY ---+-- ON ---+-----><
  |         |         |         |         |
  +- SET ----+         +- OFF --+
>>--- Query ----- STAY -----><
>>--- EXtract ---- /STAY/ -----><
```

### Description:

STAY defines whether the focus line is changed for the following:

1. An unsuccessful XEDIT style LOCATE or CLOCATE command with WRAP OFF. With STAY ON the focus line is unchanged. With STAY OFF the End of File line (or Top of File line for backward searches) becomes the focus line.
2. A successful XEDIT style CHANGE, SORT or SET SELECT command. With STAY ON the focus line is unchanged. With STAY OFF, the last line scanned or affected by the command becomes the new focus line.

SET STAY takes effect at the View level and its setting is saved if **SAVEOPTIONS** ON is in effect.

### Set Options:

ON | OFF

Set STAY ON or OFF. Default is ON.

### QUERY Response:

Displays the current setting for STAY (ON or OFF).

### EXTRACT Rexx variables:

stay.0	1
stay.1	ON   OFF

**See Also:**

**CHANGE CLOCATE LOCATE** and the SET/QUERY/EXTRACT Options: **SELECT SORT WRAP**

## STREAM - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+----- Stream ---+ ON ---+-----<<
|         |         |         |         |
+- SET -----+         +- OFF --+
>>--- Query ----- Stream -----<<
>>--- EXtract ----- /Stream/ -----<<
```

**Description:**

STREAM defines whether a search for a column target streams over multiple lines or is restricted to the focus line only.

**Note:** SET STREAM only affects CLOCATE and CDELETE commands as these are the only commands to use **column-target**.

SET STREAM takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

ON | OFF  
Set STREAM ON or OFF. Default is ON.

**QUERY Response:**

Displays the current setting for STREAM (ON or OFF).

**EXTRACT Rexx variables:**

stream.0	1
stream.1	ON   OFF

**See Also:**

**CDELETE CLOCATE**

## SYNONYM - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+----- SYNonym ---+ name ---+-----+----- action ---+-----<<
|         |         |         |         |         |         |         |
+- SET -----+         +- n_abbrev ---+         |         |
|         |         |         |         |         |         |         |
+-----+ ON +-----+ NOSPECIAL +-----+
|         |         |         |         |         |         |         |
+- OFF ---+ +- SPECIAL ---+
>>--- Query ----- SYNonym ---+-----+-----<<
|         |         |
+- * ---+
>>--- EXtract ----- /SYNonym ---+-----+ / -----<<
|         |         |
+- * ---+
```

**Description:**

SET SYNONYM has the following functions:

1. Controls text editor synonym processing.

With SYNONYM ON, the text editor checks any command verb to be executed as being a defined synonym name. The exception to this is when the command issued is done so via an edit macro. In this case, the SYNEX command must prefix the command to be executed in order to force the text editor's synonym processing.

With SYNONYM OFF, no synonym checking occurs.

The text editor's synonym checking may be bypassed by prefixing the command with the COMMAND command.

2. Defines new synonym names and associated actions.

The name token is assigned the specified action. The name may be defined with a minimal truncation of *n\_abbr* characters. Thereafter, only the first *n\_abbr* characters of the name need be entered to execute the associated action.

A blank is appended to the *action* character string unless it ends with a special character and SYNONYM ON SPECIAL is in effect.

Each name/action definition is stored until the MDI application is closed.

SET SYNONYM ON|OFF takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect. SET SYNONYM *name* takes effect at the Global level.

**Set Options:**

ON | OFF

Set SYNONYM processing ON or OFF. Default is ON.

NOSPECIAL | SPECIAL

Indicates whether or not a non-alphanumeric (special) character, as the last character of an *action* character string, is to be treated as SPECIAL in which case, a blank will **not** be automatically appended to the defined action string. Default is NOSPECIAL.

*name*

Synonym name which may be equal to an existing CBL command or macro name.

*n\_abbr*

Number of characters that CBL will accept as the minimal truncation for name.

*action*

CBL command stream.

**QUERY Response:**

If "\*" (asterisk) is not specified, **QUERY SYNONYM** displays the current setting for SYNONYM (ON or OFF, SPECIAL or NOSPECIAL).

If **QUERY SYNONYM \*** is executed, one message line is displayed for each defined synonym in the order in which they were entered. Each message line contains the synonym name followed by its and definition.

**EXTRACT REXX variables:**

Where "\*" (asterisk) **is not** specified.

synonym.0	1
synonym.1	ON   OFF
synonym.2	SPECIAL   NOSPECIAL

Where "\*" (asterisk) **is** specified.

synonym.0	Number of synonyms defined.
synonym.i	Extracts all defined synonyms in the order in which they were entered. (i=1 to synonym.0).  The format of the tokens returned in synonym.i are:  1. Synonym name. 2. Synonym definition.

**Examples:**

```
set synonym off
synonym find 1 locate
syn delblank imm 'ext/flscr/';'nomsg all blank';'del *';'all';':'flscreen.1
```

**See Also:**

[SYNEX](#)

## THIGHLIGHT - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+----- THIGHlight -----+-----+-----><
|         |         |         |         |         |
+- SET ----+         +- OFF -+ +-- FIRST -+
|         |         |         |         |         |
+-- ALL ---+         +-- ALL ---+
|         |         |         |         |         |
>>--- Query ----- THIGHlight -----><
>>--- EXTRACT ----- /THIGHlight/ -----><
```

**Description:**

Controls whether the target of a LOCATE or CLOCATE command is highlighted. The target remains highlighted until another LOCATE or CLOCATE command is executed or until text within the file is changed in any way. The colour used to highlight the target is determined by SET COLOUR THIGHLIGHT. SET THIGHLIGHT takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

- ON | OFF      Set THIGHLIGHT ON or OFF. Default is ON.
- FIRST | ALL    Highlight all occurrences of the target string or just the first. Default is ALL.

**QUERY Response:**

Displays the current setting for THIGHLIGHT (ON or OFF followed by ALL or FIRST).

**EXTRACT Rexx variables:**

thighlight.0	2
thighlight.1	ON   OFF
thighlight.2	ALL   FIRST

**Examples:**

```
thighlight on first
thighlight off
```

**See Also:**

[CLOCATE](#) [LOCATE](#) and the SET/QUERY/EXTRACT Option: [COLOUR](#)

## UNDO - QUERY/EXTRACT

### Syntax:

```
>>--- Query -----+-- UNDO --+-----+-----><
      |              |              | |
      +- REDO -+    +- * -+
>>--- EXtract --- / -+- UNDO -+- / -----><
      |              |
      +- REDO -+

```

### Description:

QUERY and EXTRACT UNDO and REDO report information relating to the undo chain of updates that have occurred for the file in the current text edit view.

### QUERY Response:

For the file in the current text edit view, QUERY UNDO and QUERY REDO without parameter "\*" (asterisk) display the number of change levels that may be undone and redone, and the amount of storage (KB) currently used to store undo information. The format of the message is:

```
ZZSE010I UNDO Undos=u Redos=r Storage=nK
```

QUERY UNDO \* and QUERY REDO \* display all change levels (Edit TRansactions - ETRs) and a numbered list of edit functions (Edit Transaction Elements - ETes) performed for each ETR as recorded in UNDO storage and which may be undone using the UNDO command.

### EXTRACT Rexx variables:

undo.0 redo.0	3
undo.1 redo.1	Number of levels of changes in the current file that may be undone.
undo.2 redo.2	Number of levels of changes in the current file that may be redone.
undo.3 redo.3	Amount of memory, in kilobytes, containing undo information for current file.

### See Also:

REDO UNDO and the SET/QUERY/EXTRACT Options: ALT UNDOING

## UNDOING - SET/QUERY/EXTRACT

### Syntax:

```
>>+-----+-- UNDOING --+ ON --+-----+-----><
      |              |              | |
      +- SET -----+              +- OFF -+ +- n -+-----+
                                      |      |
                                      +- k -+
>>--- Query ----- UNDOING -----><
>>--- EXtract --- /UNDOING/ -----><

```

### Description:

UNDOING defines whether the UNDO (and REDO) facility is enabled, the number of change levels that text editor will attempt to maintain and the maximum amount of storage it can allocate in order to store this information.

The third number following "Alt=" on the status line displays the current number of stored change levels.

The change level count is incremented by any undoable command. An undoable command is any command that changes the data in the document area or the flag bits of a line. Flag bits include a line's selection level and its tagged, changed and new indicators.

**Note:** typing text on a line in the document window is effectively treated as a CINSERT or CREPLACE command. Therefore, changing or adding text to multiple lines is equivalent to multiple CINSERT or CREPLACE commands each resulting in a different change level.

Multiple changes made to a file as a result of a macro execution are considered to be one change level only.

The text editor is informed of any changes to the 3270 terminal when an Attention ID (AID) is generated (e.g. on hitting the Enter key or any of the PF Keys). It is only then that changes to the file are committed to memory and the change level is updated. Therefore, where changes have been made to text on multiple lines, the text editor has no indication as to the order in which the lines were changed and so assigns a change level to each updated line in ascending order of line number.

SET UNDOING takes effect at the File level and its setting is saved if **SAVEOPTIONS ON** is in effect.

### Set Options:

ON | OFF

Set UNDOING ON or OFF. Default is ON.

*n*

Number of change levels maintained by the text editor for the file. If this value is exceeded, the oldest undoable change level is dropped.

*k*

Maximum amount of storage (KB) that may be obtained by the text editor for storing an individual file's undo information.

### QUERY Response:

Displays the current setting for UNDOING (ON or OFF), the maximum number of change levels and the maximum amount of memory (KB) that will be allocated by the text editor to store UNDO information.

### EXTRACT Rexx variables:

undoing.0	3
undoing.1	ON   OFF
undoing.2	The maximum number of undo levels the text editor will attempt to store in memory.
undoing.3	The maximum amount of memory, in kilobytes, that will be used to store undo information.

### See Also:

**REDO UNDO** and the SET/QUERY/EXTRACT Options: **ALT UNDO**

---

## USERNAME - QUERY/EXTRACT

---

### Syntax:

```
>>--- Query ----- USERNAME -----><
```

```
>>--- EXTRACT --- /USERNAME/ -----><
```

### Description:

QUERY and EXTRACT USERNAME report the RACF (or equivalent) logon id of the current user. This is the value returned by FileKit text editor environment variable %USER%.

### QUERY Response:

Displays the current user's RACF (or equivalent) logon userid.

### EXTRACT Rexx variables:

username.0	1
username.1	The current user's logon userid.

### See Also:

Environment Variables

## VARBLANK - SET/QUERY/EXTRACT

**Syntax:**

```
>>+-----+--- VARblank ---+- ON ---+-----><
    |         |         |         |         |
    +- SET -----+         +- OFF -+
>>--- Query ----- VARblank -----><
>>--- EXtract ---- /VARblank/ -----><
```

**Description:**

VARBLANK defines whether a single blank, specified as part of a **line-target** or **column-target** search string, represents one or more blanks.

SET VARBLANK takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

ON | OFF  
Set VARBLANK ON or OFF. Default is OFF.

**QUERY Response:**

Displays the current setting of VARBLANK (ON or OFF).

**EXTRACT REXX variables:**

varblank.0	1
varblank.1	ON   OFF

**Example:**

```
varblank on
A single blank in a line or column target search string represents one or more blanks. CLOCATE /ab c/ will successfully
match the following:
ab c
ab c
ab c
```

## VERSION - QUERY/EXTRACT

**Syntax:**

```
>>--- Query ----- VERSION -----><
>>--- EXtract --- /VERSION/ -----><
```

**Description:**

QUERY and EXTRACT VERSION report version information for the Text Editor application within the executing FileKit product.

**QUERY Response:**

Displays the version of the text editor. The message format is the text editor REXX environment name (CBLEDIT) followed by the version number (e.g. 3.20) and the build timestamp.



**EXTRACT Rexx variables:**

version.0	4
version.1	CBLEDIT
version.2	Version number. (x.xx)
version.3	Date of build. (yyyy-mm-dd)
version.4	Time of build. (hh:mm)

---

## VIEW - SET/QUERY/EXTRACT

---

**Syntax:**

```
>>--- SET ----- VIEW ---+- CHAR -+-----><
                               |         |
                               +- HEX  ---+
>>--- Query ----- VIEW -----><
>>--- EXtract ----- /VIEW/ -----><
```

**Description:**

VIEW the contents of the current text editor window in character or hexadecimal. SET VIEW is equivalent to ISPF interface primary command HEX ON/OFF.

Unlike other text edit options, specification of SET is mandatory to avoid conflict with primary command VIEW.

For CHAR display, each line displayed occupies 1 line, the line text in character format.

For HEX display, each line displayed occupies 4 lines as follows:

Line 1	The line text in character format.
Line 2	The zone digit of each character. (Hex line 1)
Line 3	The numeric digit of each character. (Hex line 2)
Line 4	A separator line of "-" (minus) signs.

Both the character and hex lines may be edited normally. Changes made to the character line are reflected in the hex lines when the changes are committed. Similarly for changes made to the hex lines.

SET VIEW takes effect at the View level and its setting is saved if **SAVEOPTIONS ON** is in effect.

**Set Options:**

CHAR | HEX  
Set VIEW HEX or CHAR. (Default is CHAR.)

**QUERY Response:**

Displays the current setting for the VIEW option (CHAR or HEX).

**EXTRACT Rexx variables:**

view.0	1
view.1	CHAR   HEX

**See Also:**

**VIEW** primary command.

## WINNAME - SET/QUERY/EXTRACT

### Syntax:

```
>>-+-----+--- WINNAME ---+-- FRame +-+ name -----><
    |         |         |         |         |
+- SET -----+         +- View ---+

>>--- Query ----- WINNAME -----><

>>--- EXtract ---- /WINNAME/ -----><
```

### Description:

Every window in the FileKit environment is allocated a unique window name which is displayed in the window title bar if "Show window names" is in effect (FileKit environment command WINDOWNAMES.) Similarly, the Window List (FileKit environment command WINDOWLIST) also displays each window's window name.

SET WINNAME is primarily used in CBLLe/SDE macros, allowing the user to assign a new window name to the current CBLLe or SDE edit window view or frame (MDI parent window.) The new name will replace the existing window name.

The window name may be used in certain FileKit environment window management commands (e.g. **WINDOWCOMMAND**, **MAXIMISE**, **MINIMISE**, **KEYS**, **COMMANDLINE**, etc.) to reference the window to which the command should apply.

### Set Options:

FRAME | VIEW  
Update the frame or edit view window name.

*name*  
Name to assign to the specified window. This may be of length up to 256 characters.

### QUERY Response:

Displays the internal name for the current text editor document window view and the Text Editor frame window (MDI parent window).

### EXTRACT Rexx variables:

winname.0	4
winname.1	FileKit window name for the current MDI child (document) text edit view.
winname.2	Title bar contents for the current MDI child (document) text edit view.
winname.3	FileKit window name for the current MDI parent (frame) window.
winname.4	Title bar contents for the current MDI parent (frame) window.

### Examples:

```
winname view PROJECT_VIEW1
Update the edit view window name.
```

```
winname fra PROJECT_FILES
Update the CBLLe frame window name.
```

### See Also:

**WINDOW** the SET/QUERY/EXTRACT Options: **WINPOS** **WINSIZE** and the FileKit environment primary command: **WINDOWNAMES**

## WINPOS - SET/QUERY/EXTRACT

### Syntax:

```

      +- View -+
      |       |
>>+-----+----- WINPOS +-----+----- row -- col -----<<
      |       |             |       |
      +- SET -----+     +- FRaMe -+
>>--- Query ----- WINPOS -----<<
>>--- EXtract ---- /WINPOS/ -----<<

```

### Description:

Position the current edit view at the specified row and column within the frame (MDI parent) client area, or position the current frame (MDI parent) window at the specified row and column of the screen.

A window's positional coordinates refer to the row x column position of the window's system menu button (found on the extreme left of the title bar.) It is this coordinate that is positioned at the new row x column position.

The lowest position within an MDI client area is row 1, column 1, which corresponds to the top left position below the window's menu bar.

The lowest position within the screen is row 1, column 2, which corresponds to the top left position occupied by the FileKit main window's system menu button.

A window cannot be positioned entirely outside its parent (MDI or FileKit main) window. If large row x column values are used, CBLc will ensure that at least 5 characters of the window's title bar remains in view.

### Set Options:

FRAME | VIEW

Position the frame or edit view window.  
Default is VIEW.

row

A positive, non-zero number specifying the row number at which the window is to be positioned.

col

A positive, non-zero number specifying the column number at which the window is to be positioned.

### QUERY Response:

Displays the row and column position of the current text edit document (MDI child) view within the main window client area and the current frame (MDI parent) window within the screen display.

### EXTRACT Rexx variables:

winpos.0	4
winpos.1	Row number in the client area of the current edit view.
winpos.2	Column number in the client area of the current edit view.
winpos.3	Row number in the screen display of the current frame window.
winpos.4	Column number in the screen display of the current frame window.

### Examples:

```
winpos view 5 10
```

Position the edit view at row 5, column 10 of the current frame's client area.

```
winpos 10 15
```

Position the edit view at row 10, column 15 of the current frame's client area.

```
winpos frame 1 2
```

Position the frame window at row 1, column 2 of the screen.

### See Also:

**WINDOW** the SET/QUERY/EXTRACT Options: **NBWINDOW** **WINNAME** **WINSIZE**

## WINSIZE - SET/QUERY/EXTRACT

### Syntax:

```

                +- View +-+
                |         |
>>+-----+----- WINSIZE ---+-----+----- rows -- cols -----><
    |         |         |         |         |
    +- SET ----+         +- FRAME -+

```

```

>>--- Query ----- WINSIZE -----><

```

```

>>--- EXTRACT ---- /WINSIZE/ -----><

```

### Description:

Resize the current edit view or frame (MDI parent) window to the specified number of rows and columns. The position of the window (i.e. top left corner) is unchanged.

CBLe does not allow an edit view display to be resized to a window less than 5 rows deep by 10 columns wide or to a window size greater than that of the MDI parent window.

Similarly, FileKit does not allow an MDI parent window display to be resized to a window of less than 10 rows deep by 10 columns wide or to a window size greater than that of the FileKit main window.

If WINPOS rows/cols values are used that exceed these limits, the window is resized to the allowable limit.

### Set Options:

FRAME | VIEW  
 Resize the frame or edit view window.

rows  
 A positive, non-zero number specifying the number of rows to which the window is to be resized.

cols  
 A positive, non-zero number specifying the number of columns to which the window is to be resized.

### QUERY Response:

Displays the rows x columns size of the current edit view, the current edit view's display area, the MDI parent window, the MDI parent window's display area and the 3270 terminal screen.

### EXTRACT Rexx variables:

winsize.0	10
winsize.1	Number of rows in the current edit view.
winsize.2	Number of columns in the current edit view.
winsize.3	Number of rows in the current edit view's display area.
winsize.4	Number of columns in the current edit view's display area.
winsize.5	Number of rows in the parent window.
winsize.6	Number of columns in the parent window.
winsize.7	Number of rows in the parent window's display area.
winsize.8	Number of columns in the parent window's display area.
winsize.9	Number of rows in the 3270 terminal.
winsize.10	Number of columns in the 3270 terminal.

### Examples:

```
winsiz view 28 80
```

Resize the current CBLe edit (MDI document) view to 28 rows by 80 columns.

```
winsize fra 40 100
```

Resize the current CBLe (MDI frame) window to 40 rows by 100 columns.

### See Also:

**WINDOW** the SET/QUERY/EXTRACT Options: **WINNAME** **WINPOS** and the FileKit environment primary command: **SIZEWINDOW**

---

## WRAP - SET/QUERY/EXTRACT

---

### Syntax:

```
>>+-----+--- WRAP ---+- ON ---+-----><
  |         |         |         |
+- SET -----+         +- OFF -+-
>>--- Query ----- WRAP -----><
>>--- EXtract ---- /WRAP/ -----><
```

### Description:

WRAP defines whether a scan for a string **line-target** continues from the Top of File after End of File has been reached. Similarly for backward searches where the scan may continue from the End of File when Top of File has been reached.

WRAP affects the LOCATE command and the CLOCATE command when STREAM is ON.

Where WRAP is OFF and the End of File or Top of File is reached, then the following message is returned:

```
ZZSE005E Target not Found
```

Where WRAP is ON and the target is found after a wrap has occurred, then the following message is returned:

```
ZZSE018I Wrapped ...
```

SET WRAP takes effect at the View level and its setting is saved if **SAVEOPTIONS** ON is in effect.

### Set Options:

ON | OFF  
Set WRAP ON or OFF. Default is OFF.

### QUERY Response:

Displays the current setting of WRAP (ON or OFF).

### EXTRACT Rexx variables:

wrap.0	1
wrap.1	ON   OFF

### See Also:

SET/QUERY/EXTRACT Options: **STREAM**

---

## ZONE - SET/QUERY/EXTRACT

---

### Syntax:

```
>>+-----+--- Zone --- n1 -- n2 -----><
  |         |
+- SET -----+
>>--- Query ----- Zone -----><
>>--- EXtract ---- /Zone/ -----><
```

**Description:**

ZONE defines the leftmost and rightmost column between which all string searches operate. i.e. **line-target**, **column-target**, **group-target** and **CHANGE** strings.

SET ZONE the same functionality as the ISPF interface **BOUNDS** primary command.

The left and right zone columns are represented in the scale line as "<" (less than) and ">" (greater than) respectively.

SET ZONE takes effect at the View level.

**Set Options:**

*n1* Left zone column. Initially, *n1* is 1.

*n2* Right zone column. "\*" (asterisk) may be specified to indicate the truncation column which, by default, is equal to the LRECL. Initially, *n2* is "\*" (asterisk).

**QUERY Response:**

Displays the current left zone column and right zone column.

**EXTRACT Rexx variables:**

zone.0	2
zone.1	Left zone column.
zone.2	Right zone column.

**Example:**

z 20 26 Set the left zone column to column 20, the right zone column to column 26.

z 10 \* Set the left zone column to column 10, the right zone column to the truncation column.

# Prefix Commands

The following commands can be entered in the prefix area of an edit window:

<code>.name</code>	Set a line pointer (line name).
<code>/</code>	Make this line the current line.
<code>([n] (([n]</code>	Column shift a line or block of lines n columns to the left. Characters shifted past the current BOUNDS setting are deleted.
<code>) [n] )) [n]</code>	Column shift a line or block of lines n columns to the right. Characters shifted past the current BOUNDS setting are deleted.
<code>&lt;[n] &lt;&lt;[n]</code>	Data shift a line or block of lines n columns to the left while attempting to prevent loss of data.
<code>&gt;[n] &gt;&gt;[n]</code>	Data shift a line or block of lines n columns to the right while attempting to prevent loss of data.
<code>A AK</code>	<b>Interface ISPF:</b> Make this line the target for a move or copy (move or copy lines After this line). AK is an intermediate line target, allowing for additional line targets to follow for the same copied or moved block of lines.
<code>A [n]</code>	<b>Interface XEDIT:</b> Insert (Add) a blank line or a block of n blank lines.
<code>B BK</code>	Make this line the target for a move or copy (move or copy lines Before this line). BK is an intermediate line target, allowing for additional line targets to follow for the same copied or moved block of lines.
<code>BOUNDs</code> <code>BOU</code> <code>BNDs</code>	Display the boundary definition line.
<code>C [n] CC</code>	Mark a line or a block of lines for copying. Lines may be copied or cut to the clipboard (using the CUT command) or copied to another position within the same edited data using prefix commands, A or B.
<code>COLs</code>	Displays a column identification line.
<code>D [n] DD</code>	Delete a line or a block of lines.
<code>F [n]</code>	<b>Interface ISPF:</b> Show the first n records of an excluded record group.
<code>F</code>	<b>Interface XEDIT:</b> Make this line the target for a move or copy (move or copy lines Following this line).
<code>HEX</code>	Opens a hex dump view of the line.
<code>I [n]</code>	Insert a new blank line or a block of n new blank lines.
<code>L [n]</code>	Show the last n records of an excluded record group.
<code>LC [n] LCC LCLC</code>	Mark a line or a block of lines for lower casing.
<code>M [n] MM</code>	Mark a line or a block of lines for move. Lines may be moved to the clipboard (using the CUT command) or moved to another position within the same edited data using prefix commands, A or B.
<code>MB</code>	Mark a corner of a box block at the cursor column position.
<code>ML</code>	Mark a limit of a line block.
<code>O [n] OK [n] OO OOK</code>	Mark a line or a block of lines to be the target of a move or copy (overlay this line or block of lines.) OK and OOK are intermediate line targets, allowing for additional line targets to follow for the same copied or moved block of lines.
<code>P</code>	Make this line the target for a move or copy (move or copy lines Previous to this line).
<code>R [n] RR (n)</code> <code>" [n] "" [n]</code>	Repeat (duplicate) a line or a block of lines n times.
<code>S [n]</code>	Show a number of excluded lines.
<code>SJ</code>	Split a line at the focus column if non-blank characters follow, otherwise join the line that follows to the focus column.
<code>T</code>	Tag a single line.
<code>TF [n]</code>	Text flow the current line and following lines up to the next blank line, wrapping text at the specified column, n, or at the display width.
<code>TS [n]</code>	Text split the current line at the cursor position inserting, n blank lines between the split text. (Default 1 blank line.)
<code>UC [n] UCC UCUC</code>	Mark a line or a block of lines for upper casing.
<code>X [n] XX</code>	Mark a line or a block of lines for exclusion from the display.

# Function Keys

The default Text-Edit KeyList is **TEXTEDIT** for INTERFACE=ISPF, or **XEDIT** for INTERFACE=XEDIT.

Individual Function Keys may be assigned new definitions using the **KEYS** command or the Text-Edit **SET PFKEY** command.

Note that SET PFKEY creates a temporary KeyList (@TMPnnnnn), modelled on the current keylist, and only affects keys at the edit view level (Window) but may be used in the **PROFILE** macro to selectively tailor PFKeys based on specific criteria.

The **KEYS** command may also be used to open the **Function Keys window** to display, and optionally update, the current function key settings. Alternatively, the Text-Edit commands, **QUERY PFnn** and **EXTRACT PFKEY**, may be used to obtain individual PFKey settings.

The FileKit Text-Editor maintains two sets of PFKey definitions, one for each of the ISPF and XEDIT interfaces.

The supplied default function keys for INTERFACE=ISPF (KEYLIST=TEXTEDIT) are:

F1	HELP	The standard HELP key.
F2	SPLIT	The ISPF SPLIT command.
F3	END	Quit the file. (Prompt to SAVE depends on the current setting of AUTOSAVE.)
F4	WINDOW	Navigate open FileKit windows.
F5	RFIND	Locate search string defined by last FIND or CHANGE command.
F6	RCHANGE	Repeat the change requested by the last CHANGE command.
F7	UP	Scroll the window display upwards by an amount determined by the scroll field or specified on the command line.
F8	DOWN	Scroll the window display downwards by an amount determined by the scroll field or specified on the command line.
F9	SWAP	The ISPF SWAP command.
F10	LEFT	Scroll the window display to the left by an amount determined by the scroll field or specified on the command line.
F11	RIGHT	Scroll the window display to the right by an amount determined by the scroll field or specified on the command line.
F12	CRETRIEV	Retrieve the last command to the command line.
F13	SOS LINEADD	Add a line following the cursor line.
F14	SOS LINEDEL	Delete the line containing the cursor.
F15	DUPLICATE	Duplicate the cursor line.
F16	ACTION	Execute FileKit <b>ACTION</b> (CMDTEXT) command on the focus line.
F17	MARK BOX	Mark the limit of a box block at the cursor.
F18	MARK LINE	Mark the limit of a line block. currently marked block.
F19	SPLTJOIN	If the cursor is followed by non-blank data on the line then split the line at the cursor. Otherwise join the line following to the focus line.
F20	BOX	Line-/Box block manipulation options menu.
F21	SWAP LIST	The ISPF SWAP LIST comamnd.
F22	UNDO	Undo the most recent change.
F23	REDO	Redo the most recently undone change.
F24	RESET BLOCK	Unmark the block.

The supplied default function keys for INTERFACE=XEDIT (KEYLIST=XEDIT) are:

F1	HELP	The standard HELP key.
F2	SPLIT	The ISPF SPLIT command.
F3	END	Quit the file. (Prompt to SAVE depends on the current setting of AUTOSAVE.)
F4	WINDOW	Navigate open FileKit windows.
F5	MACRO BLOCK UP MAJOR	Scroll up the file so that the first line previous to the current line that contains '****' or '===', becomes the current line.
F6	MACRO BLOCK DOWN MAJOR	Scroll down the file so that the first line following the current line that contains '****' or '===', becomes the current line.
F7	BACKWARD	Scroll the window display backwards 1 page towards the top of the file.
F8	FORWARD	Scroll the window display forwards 1 page towards the bottom of the file.



F9	SWAP	The ISPF SWAP command.
F10	LEFT HALF	Scroll the window display half the width of the edit view to the left.
F11	RIGHT HALF	Scroll the window display half the width of the edit view to the right.
F12	CRETRIEV	Retrieve the last command to the command line.
F13	SOS LINEADD	Add a line following the cursor line.
F14	SOS LINEDEL	Delete the line containing the cursor.
F15	DUPLICATE	Duplicate the cursor line.
F16	ACTION	Execute FileKit <b>ACTION</b> (CMDTEXT) command on the focus line.
F17	MARK BOX	Mark the limit of a box block at the cursor.
F18	MARK LINE	Mark the limit of a line block. currently marked block.
F19	SPLTJOIN	If the cursor is followed by non-blank data on the line then split the line at the cursor. Otherwise join the line following to the focus line.
F20	BOX	Line-/Box block manipulation options menu.
F21	SWAP LIST	The ISPF SWAP LIST comamnd.
F22	UNDO	Undo the most recent change.
F23	REDO	Redo the most recently undone change.
F24	RESET BLOCK	Unmark the block.

Note that the contents of the command line is concatenated to the definition of the function key and the result executed as a single command.

---

# Glossary

---

The following is a glossary of terms used in this document.

## **3270 Emulator**

Third party software that emulates Mainframe 3270 hardware terminals on PC and UNIX based platforms.

## **CLI**

A Command Line Interface is a text based method by which users can execute functions supported by the application.

## **CBLe**

A powerful text editor that runs as an MDI application under FileKit. CBLe supports its own CLI and has been developed based on specifications found in Mansfield Software's KEDIT for Windows.

## **CBLi**

The original brand name for FileKit. See FileKit.

## **CBLiVTAM**

Name of the multi-user version of the FileKit application that executes under VTAM.

## **CBLVCAT**

CBL licensable product that supports VSAM file tuning and VTOC, ICF/VSAM catalog and VSE LABEL reporting. Executes as a batch facility or interactively as a FileKit application.

## **Current Column**

The first visible column of text within the display area of the current text edit view.

## **Current Line**

The first visible line of text within the display area of the current text edit view.

## **Edit View or Text Edit View**

A CBLe MDI document window that contains a display of text edited data. If the same file is displayed in multiple windows, then the user has multiple edit views of the file. Each edit view can have a different current line, ARBCHAR setting, ZONE columns, etc.

## **FileKit**

The Interactive environment developed by CBL and supplied as part of SELCOPY and CBLVCAT licensable software products. Previously branded as SELCOPYi, and before that CBLi.

## **Focus Column**

The column of text on which the cursor is positioned within the focus line of the current text edit view. If the focus line is a shadow line representing an excluded group of lines, or if the cursor is positioned outside the display area (e.g. the command line) or within the prefix area, the focus column is defined as being the **current column**.

## **Focus Line**

The line of text within the current text edit view on which the cursor is positioned. If the cursor is positioned outside the display area (e.g. the command line), then the focus line is defined as being the **current line**.

## **INI File**

File containing configuration options for FileKit. The System INI file is processed on startup of FileKit and contains options that apply to all users. The User INI file contains options specific to each user that may, where appropriate, override options set in the System INI file.

## **List Window**

A class of FileKit window containing rows of associated information. List windows support point-and-shoot column sorting; select, sort and filter CLI commands; and prefix area commands.

## **MDI**

Multiple Document Interface is a Microsoft specification for PC applications that enable the user to work with multiple documents at the same time. Each document is displayed in a separate child window within the client area of the application's main (frame) window. Typical MDI applications on PCs include word-processing and spread sheet applications.

## **MDI Client Area window**

The MDI client area window is the display area within an MDI application's frame window. The MDI client area serves as the background for MDI child windows.

## **MDI Child/Document Window**

An MDI child or document window is opened in an application's client area window each time a document is opened. Each child window has a sizing border, title bar, window menu, minimise, maximise, restore and close buttons. A child window is clipped so that it is confined to the client window and cannot appear outside it.

When a child window is maximized, its client area completely fills the MDI client area window. In addition, the system automatically hides the child window's title bar, and adds the child window's window menu icon and Restore button to the MDI application's menu bar.

## **MDI Frame Window**

An MDI frame window may be considered the main window of an MDI application. It is the parent window of the MDI client area window in which MDI child windows are opened. It has a sizing border, title bar, window menu, minimise, maximise restore and close buttons.

**Ring**

The set of all **files** being edited within CBLe. It is not the set of all windows opened. e.g. The contents of one file may be displayed in more than one edit view (window.)

**SDB**

See SELCOPY Interactive.

**SELCOPY Interactive (SDB)**

An Intergrated Development Environment for SELCOPY (SELCOPY DeBug) that runs as an MDI application under FileKit.

**Storage Display Window**

A class of FileKit window containing hexadecimal and character display of areas of storage.