Debug Tool for z/OS
Debug Tool Utilities and Advanced Functions for z/OS

Reference Summary

Version 8.1

SC19-1199-01

Debug Tool for z/OS
Debug Tool Utilities and Advanced Functions for z/OS

Reference Summary

Version 8.1

SC19-1199-01

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page 55.

| This edition applies to Debug Tool for z/OS, Version 8.1 (Program Number 5655-S17) with the PTF for APAR
| PK58192 applied, which supports the following compilers:

- AD/Cycle® C/370™ Version 1 Release 2 (Program Number 5688-216)
- C/C++ for MVS/ESA™ Version 3 (Program Number 5655-121)
- C/C++ feature of OS/390® (Program Number 5647-A01)
- C/C++ feature of z/OS (Program Number 5694-A01)
- OS/VS COBOL, Version 1 Release 2.4 (5740-CB1) - with limitations
- VS COBOL II Version 1 Release 3 and Version 1 Release 4 (Program Numbers 5668-958, 5688-023) - with limitations
- COBOL/370™ Version 1 Release 1 (Program Number 5688-197)
- COBOL for MVS & VM Version 1 Release 2 (Program Number 5688-197)
- COBOL for OS/390 & VM Version 2 (Program Number 5648-A25)
- Enterprise COBOL for z/OS and OS/390 Version 3 (Program Number 5655-G53)
- High Level Assembler for MVS & VM & VSE Version 1 Release 4 and Version 1 Release 5 (Program Number 5696-234)
- OS PL/I Version 2 Release 1, Version 2 Release 2, Version 2 Release 3 (Program Numbers 5668-909, 5668-910) - with limitations
- PL/I for MVS & VM Version 1 Release 1 (Program Number 5688-235)
- VisualAge® PL/I for OS/390 Version 2 Release 2 (Program Number 5655-B22)
- Enterprise PL/I for z/OS and OS/390 Version 3.6 or earlier (Program Number 5655-H31)

Parts of this edition apply to Debug Tool Utilities and Advanced Functions for z/OS, Version 8.1 (Program Number 5655-S16).

This edition also applies to all subsequent releases and modifications until otherwise indicated in new editions or technical newsletters.

You can order publications online at www.ibm.com/shop/publications/order, or order by phone or fax. IBM Software Manufacturing Solutions takes publication orders between 8:30 a.m. and 7:00 p.m. Eastern Standard Time (EST). The phone number is (800)879-2755. The fax number is (800)445-9269.

You can find out more about Debug Tool by visiting the IBM Web site for Debug Tool at: http://www.ibm.com/software/awdtools/debugtool

Contents

About this document

Debug Tool combines the richness of the z/OS® environment with the power of Language Environment® to provide a debugger for programmers to isolate and fix their program bugs and test their applications. Debug Tool gives you the capability of testing programs in batch, using a nonprogrammable terminal in full-screen mode, or using a workstation interface to remotely debug your programs.

This document contains a summary of commands, built-in functions, and EQAOPTS options provided by Debug Tool. Each topic contains the name of the command, built-in function, or EQAOPTS option and then a syntax diagram. Some of these commands require that you purchase and install Debug Tool Utilities and Advanced Functions and are so noted. For more information on each command or built-in function, refer to *Debug Tool Reference and Messages*. For more information on each EQAOPTS option, see *Debug Tool Customization Guide*.

The Debug Tool Utilities and Advanced Functions Coverage Utility is referred to throughout this document as the Debug Tool Coverage Utility or Coverage Utility.

Who might use this document

This document is intended for programmers using Debug Tool to debug high-level languages (HLLs) with Language Environment and assembler programs either with or without Language Environment. Throughout this document, the HLLs are referred to as C, C++, COBOL, and PL/I.

The following operating systems and subsystems are supported:

- z/OS
  - CICS®
  - DB2®
  - IMS™
  - JES batch
  - TSO
  - UNIX® System Services in remote debug mode or full-screen mode through a VTAM terminal only
  - WebSphere® in remote debug mode or full-screen mode through a VTAM terminal only

To use this document and debug a program written in one of the supported languages, you need to know how to write, compile, and run such a program.

Accessing z/OS licensed documents on the Internet

z/OS licensed documentation is available on the Internet in PDF format at the IBM® Resource Link™ Web site at:

`http://www.ibm.com/servers/resourcelink`

Licensed documents are available only to customers with a z/OS license. Access to these documents requires an IBM Resource Link user ID and password, and a key code. With your z/OS order you received a Memo to Licensees, (GI10-0671), that includes this key code.

To obtain your IBM Resource Link user ID and password, log on to:
`http://www.ibm.com/servers/resourcelink`

To register for access to the z/OS licensed documents:

1. Sign in to Resource Link using your Resource Link user ID and password.
2. Select **User Profiles** located on the left-hand navigation bar.

Note: You cannot access the z/OS licensed documents unless you have registered for access to them and received an e-mail confirmation informing you that your request has been processed.

Printed licensed documents are not available from IBM.

You can use the PDF format on either **z/OS Licensed Product Library CD-ROM** or IBM Resource Link to print licensed documents.

Using LookAt to look up message explanations

LookAt is an online facility that lets you look up explanations for most of the IBM messages you encounter, as well as for some system abends and codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can use LookAt from the following locations to find IBM message explanations for z/OS elements and features, z/VM®, VSE/ESA™, and Clusters for AIX® and Linux®:

- The Internet. You can access IBM message explanations directly from the LookAt Web site at http://www.ibm.com/eserver/zseries/zos/bkserv/lookat/.
- Your z/OS TSO/E host system. You can install code on your z/OS or z/OS.e systems to access IBM message explanations, using LookAt from a TSO/E command line (for example, TSO/E prompt, ISPF, or z/OS UNIX System Services running OMVS).
- Your Microsoft® Windows® workstation. You can install code to access IBM message explanations on the *z/OS Collection* (SK3T-4269), using LookAt from a Microsoft Windows command prompt (also known as the DOS command line).
- Your wireless handheld device. You can use the LookAt Mobile Edition with a handheld device that has wireless access and an Internet browser (for example, Internet Explorer for Pocket PCs, Blazer, or Eudora for Palm OS, or Opera for Linux handheld devices). Link to the LookAt Mobile Edition from the LookAt Web site.

You can obtain code to install LookAt on your host system or Microsoft Windows workstation from a disk on your *z/OS Collection* (SK3T-4269), or from the LookAt Web site (click **Download**, and select the platform, release, collection, and location that suit your needs). More information is available in the LOOKAT.ME files available during the download process.

How to read syntax diagrams

This section describes how to read syntax diagrams. It defines syntax diagram symbols, items that may be contained within the diagrams (keywords, variables, delimiters, operators, fragment references, operands) and provides syntax examples that contain these items.

Syntax diagrams pictorially display the order and parts (options and arguments) that comprise a command statement. They are read from left to right and from top to bottom, following the main path of the horizontal line.

Symbols

The following symbols may be displayed in syntax diagrams:

Symbol	Definition
►►—	Indicates the beginning of the syntax diagram.
—►	Indicates that the syntax diagram is continued to the next line.
►—	Indicates that the syntax is continued from the previous line.
—►◄	Indicates the end of the syntax diagram.

Syntax items

Syntax diagrams contain many different items. Syntax items include:

- Keywords - a command name or any other literal information.
- Variables - variables are italicized, appear in lowercase and represent the name of values you can supply.
- Delimiters - delimiters indicate the start or end of keywords, variables, or operators. For example, a left parenthesis is a delimiter.
- Operators - operators include add (+), subtract (-), multiply (*), divide (/), equal (=), and other mathematical operations that may need to be performed.
- Fragment references - a part of a syntax diagram, separated from the diagram to show greater detail.
- Separators - a separator separates keywords, variables or operators. For example, a comma (,) is a separator.

Keywords, variables, and operators may be displayed as required, optional, or default. Fragments, separators, and delimiters may be displayed as required or optional.

Item type	Definition
Required	Required items are displayed on the main path of the horizontal line.
Optional	Optional items are displayed below the main path of the horizontal line.
Default	Default items are displayed above the main path of the horizontal line.

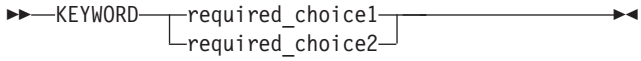
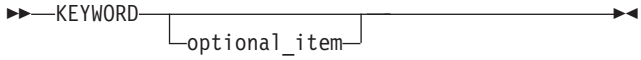
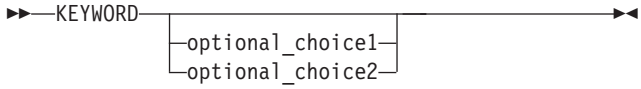
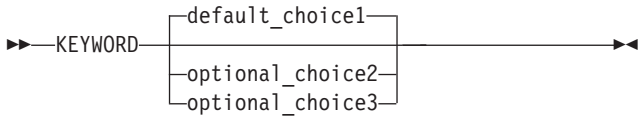
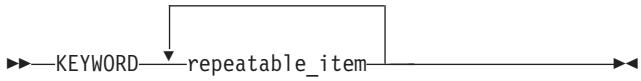
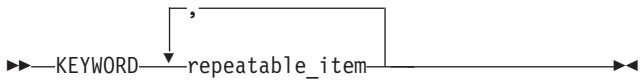
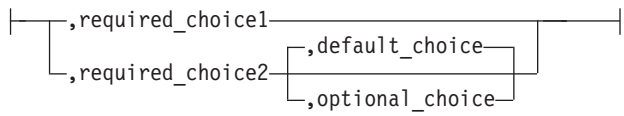
Syntax examples

The following table provides syntax examples.

Table 1. Syntax examples

Item	Syntax example
Required item. Required items appear on the main path of the horizontal line. You must specify these items.	►►—KEYWORD—required_item———————————————►◄

Table 1. Syntax examples (continued)

Item	Syntax example
Required choice. A required choice (two or more items) appears in a vertical stack on the main path of the horizontal line. You must choose one of the items in the stack.	
Optional item. Optional items appear below the main path of the horizontal line.	
Optional choice. An optional choice (two or more items) appears in a vertical stack below the main path of the horizontal line. You may choose one of the items in the stack.	
Default. Default items appear above the main path of the horizontal line. The remaining items (required or optional) appear on (required) or below (optional) the main path of the horizontal line. The following example displays a default with optional items.	
Variable. Variables appear in lowercase italics. They represent names or values.	
Repeatable item. An arrow returning to the left above the main path of the horizontal line indicates an item that can be repeated. A character within the arrow means you must separate repeated items with that character. An arrow returning to the left above a group of repeatable items indicates that one of the items can be selected, or a single item can be repeated.	 
Fragment. The ┤ fragment ├ symbol indicates that a labelled group is described below the main syntax diagram. Syntax is occasionally broken into fragments if the inclusion of the fragment would overly complicate the main syntax diagram.	 **fragment:** 

How to send your comments

Your feedback is important in helping us to provide accurate, high-quality information. If you have comments about this document or any other Debug Tool documentation, contact us in one of these ways:

- Use the Online Readers' Comment Form at www.ibm.com/software/awdtools/rcf/. Be sure to include the name of the document, the publication number of

the document, the version of Debug Tool, and, if applicable, the specific location (for example, page number) of the text that you are commenting on.

- Fill out the Readers' Comment Form at the back of this document, and return it by mail or give it to an IBM representative. If the form has been removed, address your comments to:

  IBM Corporation
  H150/090
  555 Bailey Avenue
  San Jose, CA 95141-1003
  USA

- Fax your comments to this U.S. number: (800)426-7773.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Summary of changes

This section lists the key changes made to Debug Tool for z/OS and Debug Tool Utilities and Advanced Functions for z/OS that affect this document.

Changes introduced with the PTF for APAR PK58192

You can now debug VS COBOL II programs that are compiled with the NOTEST compiler option and linked with a non-Language Environment library in the same manner that OS/VS COBOL programs are debugged. With the introduction of this support, a new term (*non-Language Environment COBOL*) has been introduced to refer to both OS/VS COBOL programs and these VS COBOL II programs. The following changes have been made to this document:

- Most instances of the term *OS/VS COBOL* have been changed to *non-Language Environment COBOL*. These changes are marked with revision bars.

The processing for the Debug Tool user exit (EQADBCXT, EQADDCXT, EQADICXT) is enhanced to include an optional program name token in the user exit data set name pattern. This provides a mechanism to have different TEST runtime options for different programs.

The processing for the Debug Tool user exit (EQADBCXT, EQADDCXT, EQADICXT) is enhanced with a recovery routine that handles the s913 exception, which occurs when you cannot access the data set for security reasons.

Changes introduced with the PTF for APAR PK53826

- You can now debug Enterprise PL/I DLL programs.
- The Monitor window has been changed to align the beginning of data along the same column.
- Debug Tool now supports the changes to the TEST compiler option introduced in Enterprise COBOL for z/OS, Version 4.1. See *Enterprise COBOL for z/OS Programming Guide* for more information about the changes made to the TEST compiler option, including the addition of the EJPD subooption.
- Debug Tool now supports the changes to the TEST compiler options introduced in Enterprise PL/I for z/OS, Version 3.7. See *Enterprise PL/I for z/OS Programming Guide* for a description of the changes.
- You can now specify the IPv6 format for TCP/IP addresses.
- You now have better control over how Debug Tool handles invalid comparisons.
- Miscellaneous improvements to text to improve clarity and accuracy.

Changes introduced with Debug Tool V8.1

- When a FINISH, CEE066 or CEE067 thread termination condition is raised by Language Environment, your system administrator can now prevent Debug Tool from prompting the user by specifying the THREADTERMCOND option in the EQAOPTS options file.
- Debug Tool now supports only single socket connection types with remote debuggers. The VADTCPIP& and TCPIP& suboptions of the TEST runtime option will both start single socket connections.

- The NAMES EXCLUDE command has been enhanced so that you can more easily indicate the specific types of compile units that you do not want to debug.
- Debug Tool now handles characters that cannot be displayed in their declared data type in a more consistent manner across all programming languages.

  Previously, Debug Tool did not allow you to modify COBOL characters, which could not be displayed in their declared data type, unless you display those characters in hexadecimal format. This behavior was different from the other programming languages. Changes have been made so that Debug Tool handles these characters in the same manner across all programming languages.
- The instructions on how to prepare a DB2 stored procedure have been improved.
- Debug Tool has made the following enhancements to help make it easier to use command sequences across different programming languages:
  - You can use the BEGIN and END command with all supported programming languages.
  - You can use a single syntax (X'xxxxxxxx') for hexadecimal addresses in all supported programming languages.
  - You can use quotation marks (") for strings in all supported programming languages.

  This improvement makes it easier to write command sequences that can be used in an environment where several programming languages are used. It also helps improve the ability to restore these command sequences into programs that are written in programming languages other than the one in which they were created.
- Debug Tool now supports the display of the 64-bit general purpose registers when running on hardware that supports this feature. New symbols, %GPRG*n* and RG*n*, are provided for referencing the 64-bit general purpose registers in disassembly expressions. No support is provided for 64-bit addressing.
- In previous releases, Debug Tool assumed that all floating point data items and registers were in hexadecimal floating point format. Now, for disassembly programs, Debug Tool also supports floating point register in binary (IEEE) and decimal floating point format.
- Debug Tool has a new window called the Memory window, which provides you with better navigation and display of memory in the full screen mode.

  In order to understand how to navigate through all the windows in a Debug Tool session panel, you need to understand the difference between a physical window and a logical window.
- A new topic has been added, called Chapter 3, “EQAOPTS options,” on page 51, which summarizes all of the options that can be specified in the EQAOPTS option file.
- The following enhancements have been made to prefix commands that you enter in the Monitor window:
  - You can use the DEF prefix command like the HEX prefix command so that it operates on individual array and structure elements.
  - You can use the HEX prefix command on Enterprise PL/I variables.
- For CICS programs, you can specify a non-Language Environment assembler program, which is loaded through an EXEC CICS LOAD command, in the Program Id(s) field of the main DTCN screen.
- The following enhancements have been made for remote debug mode:
  - The default port has been changed to 8001.

- You can now enter the SET DEFAULT LISTINGS command through the debug console.
- You can now use a preferences file, global preferences file, and commands file in remote debug mode.
- The remote debugger's interface has been enhanced so that you can filter variables. See the remote debugger's online help for a description of this improvement.

Changes introduced with Debug Tool Utilities and Advanced Functions V8.1

- In CICS, Debug Tool Utilities and Advanced Functions now enhances Debug Tool to provide a way to disable or re-enable DTCN and CADP pattern match breakpoints from within a debugging session. You can use DISABLE CADP, DISABLE DTCN, ENABLE CADP and ENABLE DTCN commands to control pattern-match breakpoints.
- Debug Tool Utilities and Advanced Functions has expanded the number of ways you can link in the CEEBXITA Language Environment user exit routine, which is provided by Debug Tool.
- Debug Tool Utilities and Advanced Functions provides a new CICS transaction (called DTST), which runs separately from Debug Tool. The DTST transaction helps you view and modify CICS storage.
- Debug Tool Utilities and Advanced Functions adds to Debug Tool the ability to check for specific types of storage violations in CICS.
- Debug Tool Utilities and Advanced Functions enhances Debug Tool so it can now display CICS channels and containers.

  By using the DESCRIBE CHANNEL and LIST CONTAINER commands, you can now display the contents of CICS channels and containers. For more information about CICS channels and containers, see the section "Enhanced inter-program data transfer: channels as modern-day COMMAREAs" in the *CICS Application Programming Guide*.
- Debug Tool Utilities and Advanced Functions has enhanced the information Debug Tool displays in the automonitor section of the Monitor window for assembler programs.

  Whenever possible, Debug Tool displays user variable and register names. In other cases, it appends comments so you can easily see how _STORAGE operands are associated with the user-coded operands.
- The following enhancements have been made to the automonitor section of the Monitor window:
  - You can use the HEX and DEF prefix commands in the prefix area of the automonitor section of the Monitor window.
  - You can use the MONITOR *n* HEX command to display the value of a variable in hexadecimal format.
  - You can use the MONITOR *n* DEF command to display the value of a variable in the variable's declared data type.
  - For COBOL characters displayed in the automonitor section of the Monitor window, Debug Tool displays the values in character format, regardless of whether the string contains characters that cannot be displayed in their declared format. You can modify these values by typing over the existing values.
- Debug Tool Utilities and Advanced Functions adds new symbols to Debug Tool, %GPRG*n* and RG*n*, to reference the 64-bit general purpose registers in assembler

expressions. You can now display and modify 64-bit arithmetic data items and (on hardware that supports 64-bit) the 64-bit general purpose registers. These 64-bit items can also be used in assembler arithmetic expressions. No support is provided for 64-bit addressing.

- In previous releases, Debug Tool assumed that all floating point data items and registers were in hexadecimal floating point format. Now, for assembler and disassembly programs, Debug Tool Utilities and Advanced Functions enhances Debug Tool so that it also supports floating point data items in binary (IEEE) and decimal floating point format. You can reference and display the floating point registers in any of the three formats. Debug Tool correctly displays all floating-point data items for all three data types (hexadecimal, binary, or decimal). You can use the assembler assignment commands to assign constant values to binary and decimal floating point data items and registers.
- Support for MasterCraft Q++ has been added. For information on MasterCraft Q++, contact Tata Consultancy Services Ltd.

Chapter 1. Debug Tool commands

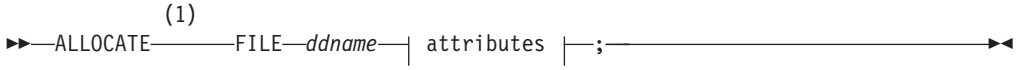
Debug Tool provides the following commands:

? command

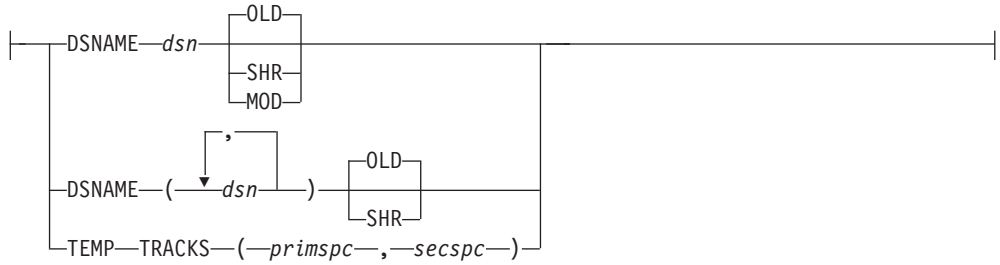
Displays a list of all commands or, if used in combination with a command, displays a list of options that you can specify for that command.

ALLOCATE command

The ALLOCATE command allocates a file (*ddname*) to an existing data set, a concatenation of existing data sets, or a temporary data set.



attributes:

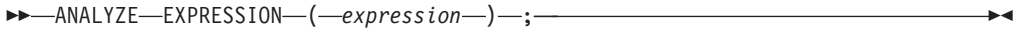


Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

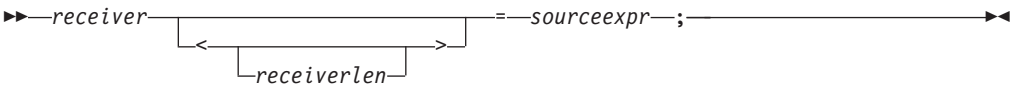
ANALYZE command (PL/I)

The ANALYZE command displays the process of evaluating an expression and the data attributes of any intermediate results.



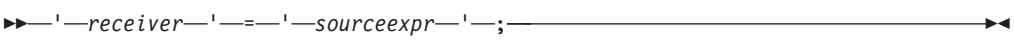
Assignment command (assembler and disassembly)

The Assignment command copies the value of an expression to a specified memory location or register.



Assignment command (non-Language Environment COBOL)

The Assignment command assigns the value of an expression to a specified reference. It is the equivalent of the COBOL COMPUTE statement.

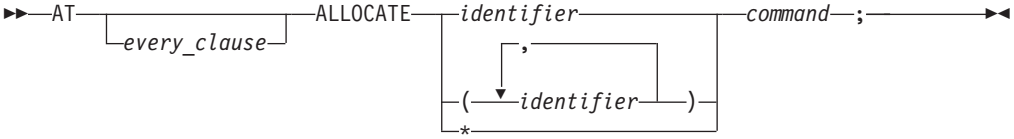


Assignment command (PL/I)

The Assignment command copies the value of an expression to a specified reference.

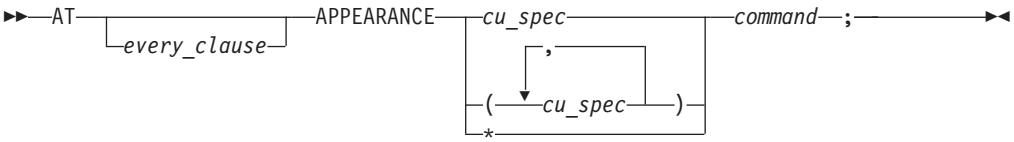
AT ALLOCATE (PL/I)

AT ALLOCATE gives Debug Tool control when storage for a named controlled variable or aggregate is dynamically allocated by PL/I.



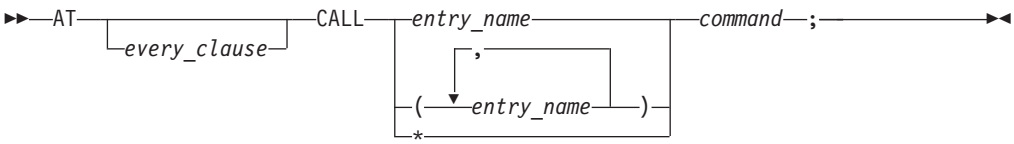
AT APPEARANCE

Gives Debug Tool control when the specified compile unit is found in storage.



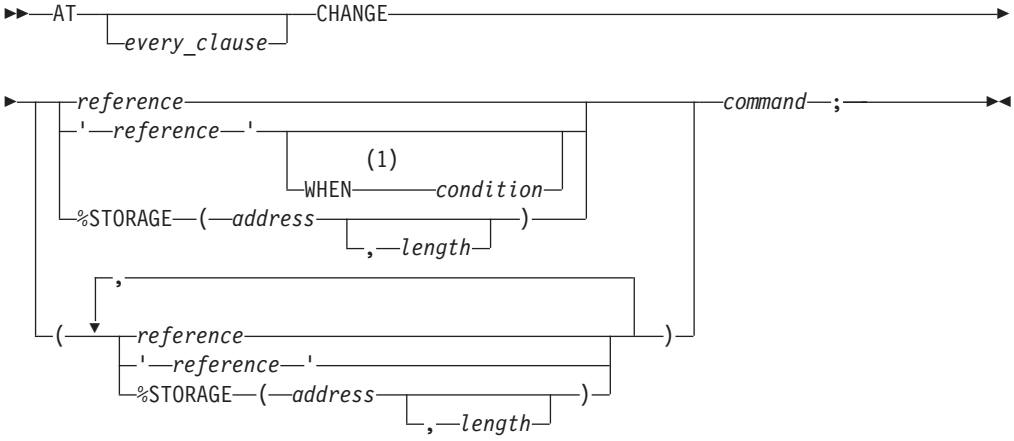
AT CALL

Gives Debug Tool control when the application code attempts to call the specified entry point.



AT CHANGE

Gives Debug Tool control when either the program or Debug Tool command changes the specified variable value or storage location.



Notes:

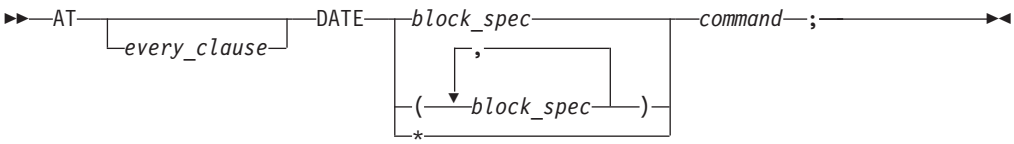
1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

AT CURSOR (full-screen mode)

Provides a cursor controlled method for setting a statement breakpoint.

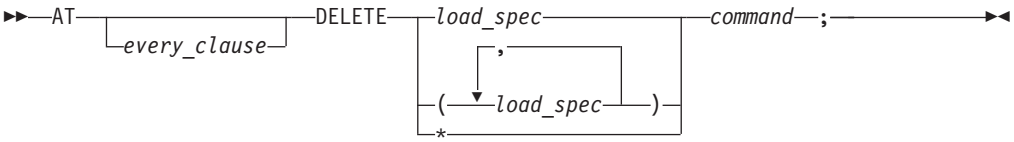
AT DATE (COBOL)

Gives Debug Tool control for each date processing statement within the specified block.



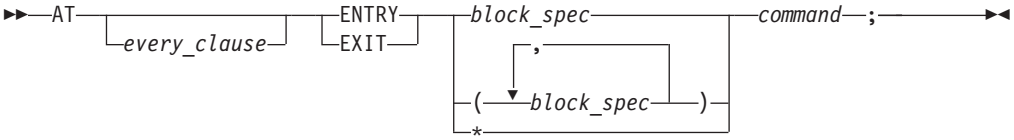
AT DELETE

Gives Debug Tool control when a load module is removed from storage by a Language Environment delete service, such as on completion of a successful C `release()`, COBOL `CANCEL`, or PL/I `RELEASE`.



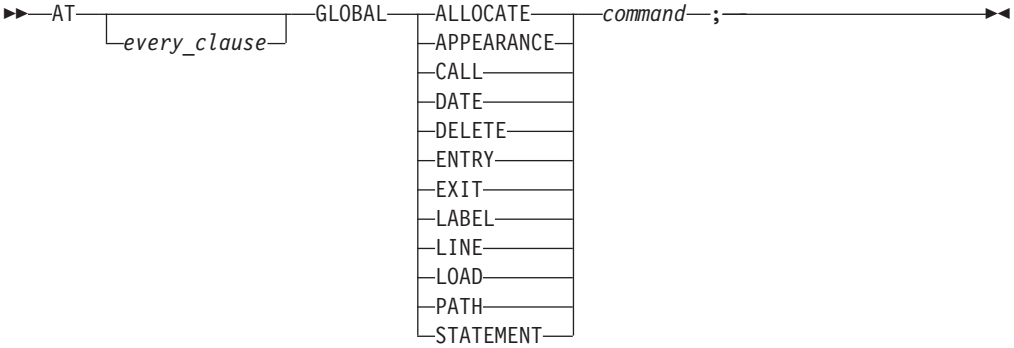
AT ENTRY and AT EXIT

Defines a breakpoint at the specified entry point or exit in the specified block.



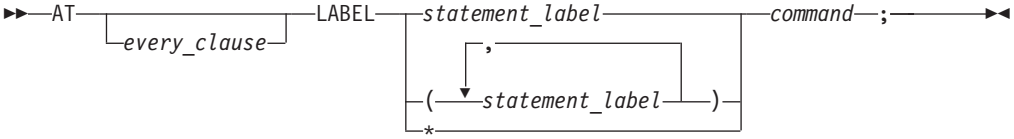
AT GLOBAL

Gives Debug Tool control for every instance of the specified `AT`-condition.



AT LABEL

Gives Debug Tool control when execution has reached the specified statement label or group of labels.

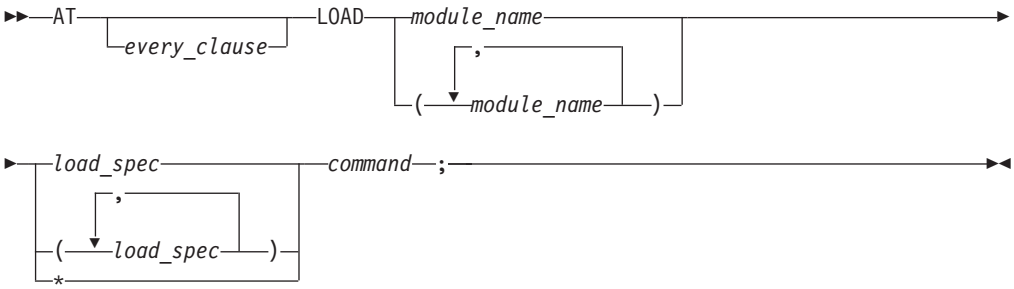


AT LINE

Gives Debug Tool control at the specified line. See “AT STATEMENT” on page 6.

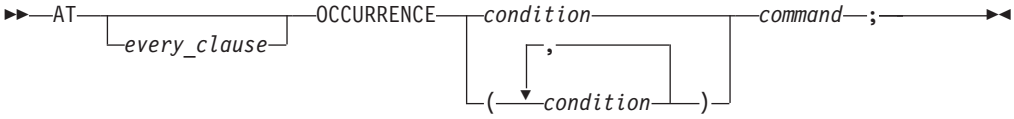
AT LOAD

Gives Debug Tool control when the specified load module is brought into storage.



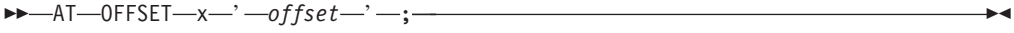
AT OCCURRENCE

Gives Debug Tool control on a language or Language Environment condition or exception.



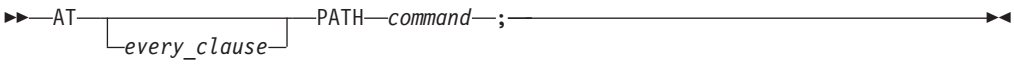
AT OFFSET (disassembly)

Gives Debug Tool control at the specified offset in the disassembly view.



AT PATH

Gives Debug Tool control when the flow of control changes (at a path point). `AT PATH` is identical to `AT GLOBAL PATH`.

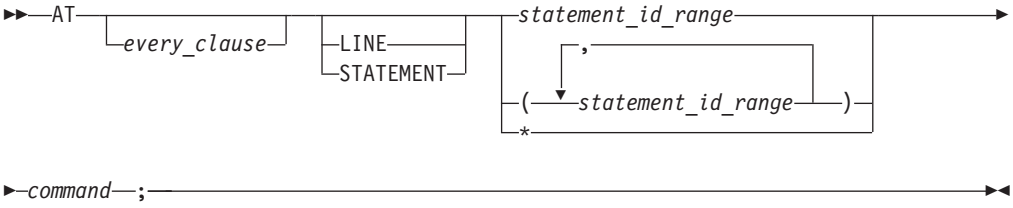


AT Prefix (full-screen mode)

Sets a statement breakpoint when you issue this command through the Source window prefix area.

AT STATEMENT

Gives Debug Tool control at each specified statement or line within the given set of ranges.



AT TERMINATION

Gives Debug Tool control when the application program is terminated.

BEGIN command

`BEGIN` and `END` delimit a sequence of one or more commands to form one longer command.

block command (C and C++)

The `block` command allows you to group any number of Debug Tool commands into one command.

break command (C and C++)

The `break` command allows you to terminate and exit a loop (that is, `do`, `for`, and `while`) or `switch` command from any point other than the logical end.

CALL %CEBR

Starts the CICS Temporary Storage Browser Program.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

CALL %CECI

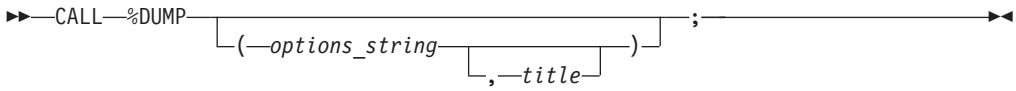
Starts the CICS Command Level Interpreter Program.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

CALL %DUMP

Calls the Language Environment dump service to obtain a formatted dump.



CALL %FA

Starts and instructs IBM Fault Analyzer to provide a formatted dump of the current machine state.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

CALL %HOGAN

Starts Computer Sciences Corporation's KORE-HOGAN application, also known as SMART (System Memory Access Retrieval Tool).

Notes:

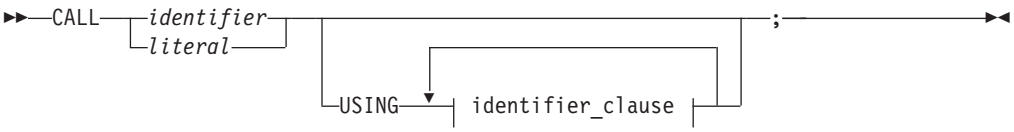
1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16) and can be used only in CICS.

CALL %VER

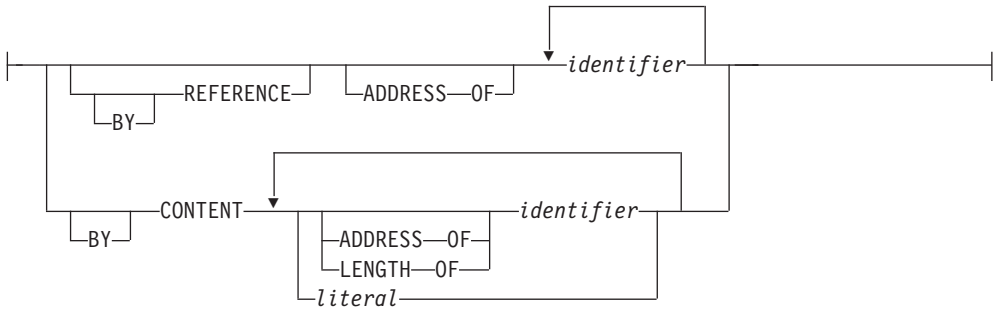
Adds a line to the log describing the maintenance level of Debug Tool that you have installed on your system..

CALL entry_name (COBOL)

Calls an entry name in the application program.



identifier_clause:



CALL procedure

Calls a procedure that has been defined with the PROCEDURE command.

CHKSTGV

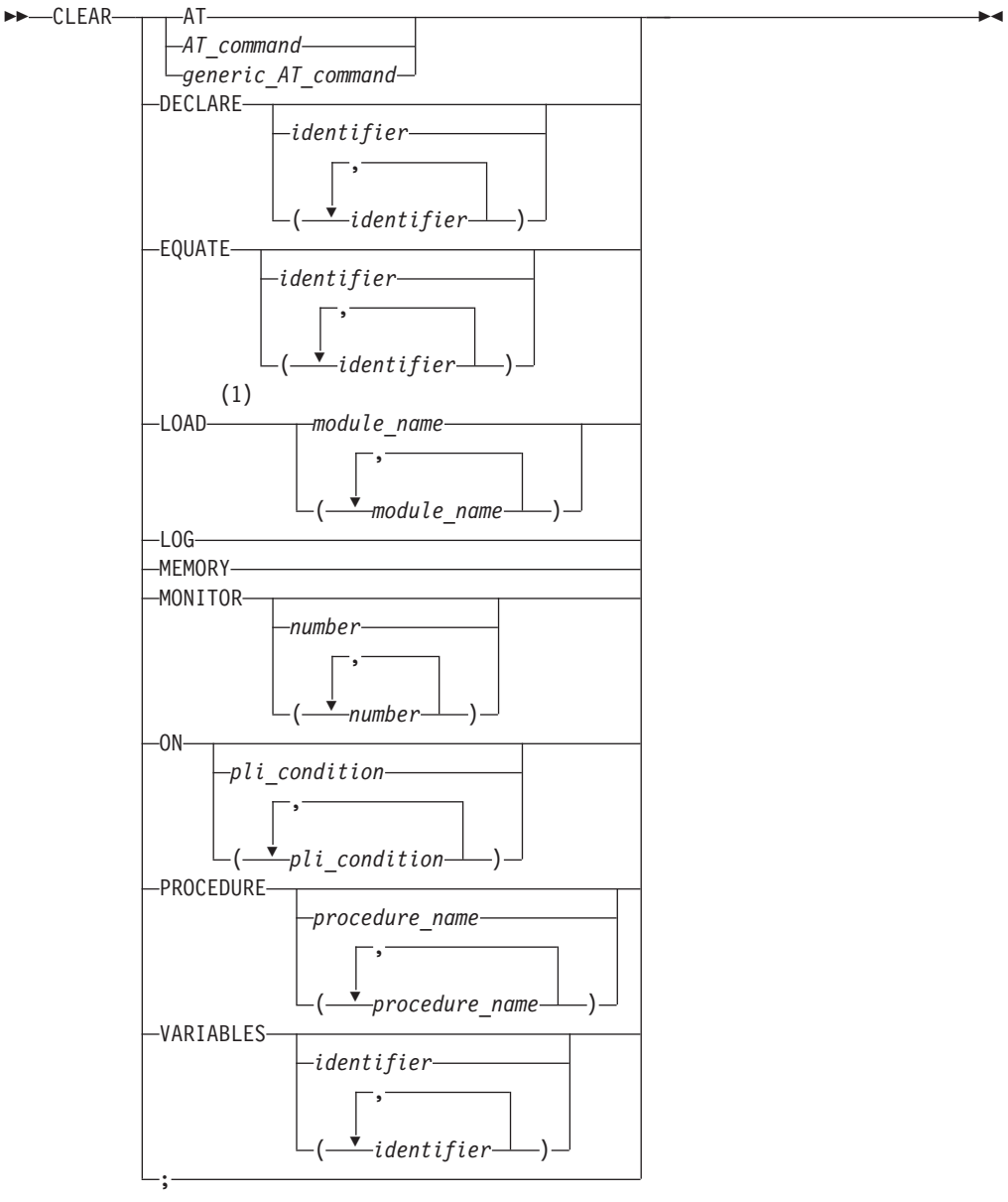
Check for a specific type of storage violation in the task you are currently debugging.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

CLEAR command

The CLEAR command removes the actions of previously issued Debug Tool commands.



Notes:

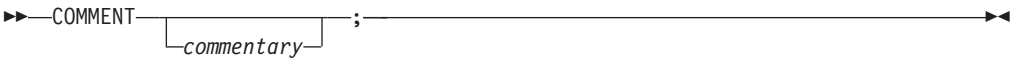
1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

CLEAR prefix (full-screen mode)

Clears a breakpoint when you issue this command through the Source window prefix area.

COMMENT command

The COMMENT command can be used to insert commentary in to the session log.

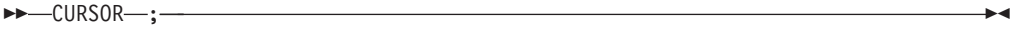


COMPUTE command (COBOL)

The COMPUTE command assigns the value of an arithmetic expression to a specified reference.

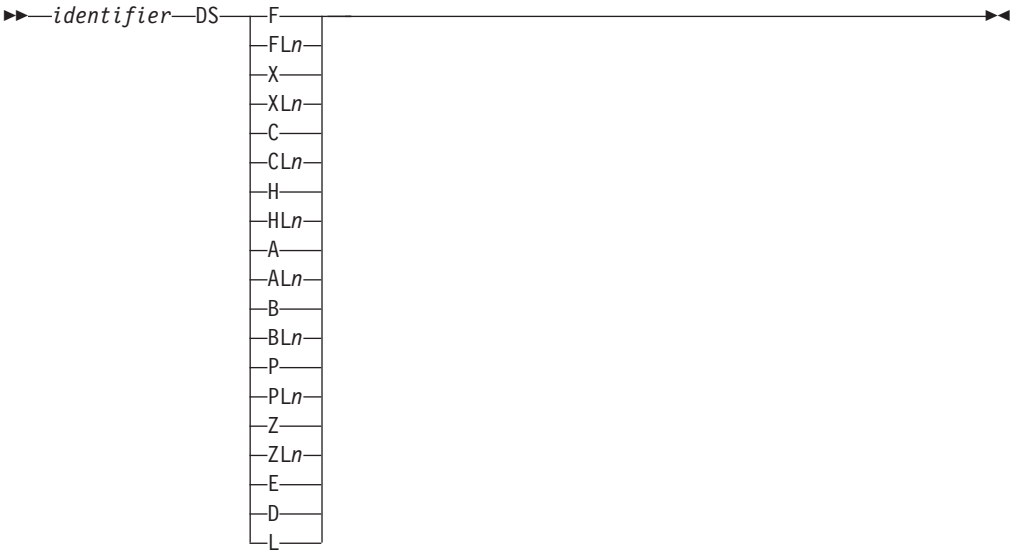
CURSOR command (full-screen mode)

The CURSOR command moves the cursor between the last saved position on the Debug Tool session panel (excluding the header fields) and the command line.



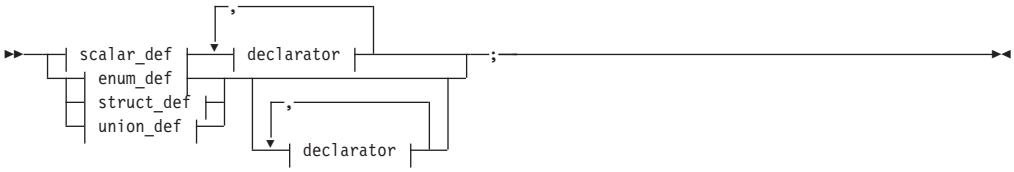
Declarations (assembler, disassembly, and non-Language Environment COBOL)

Use declarations to create session variables and tags effective during a Debug Tool session.

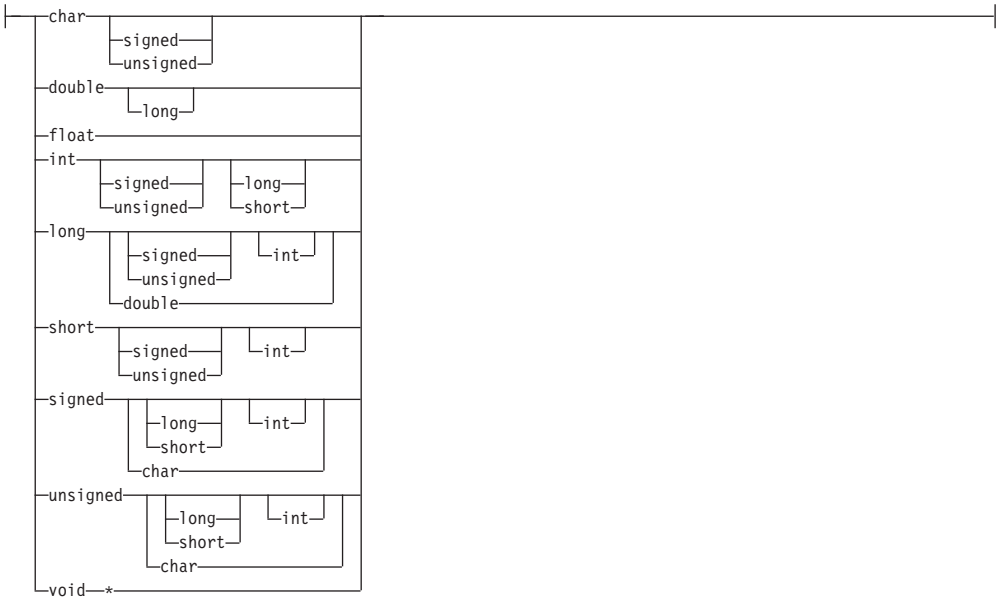


Declarations (C and C++)

Use `declarations` to create session variables and tags effective during a Debug Tool session.

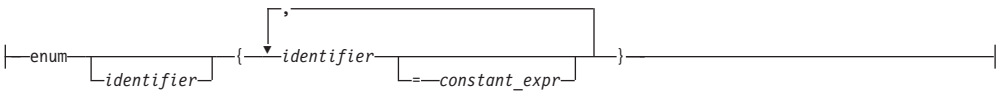


scalar_def:

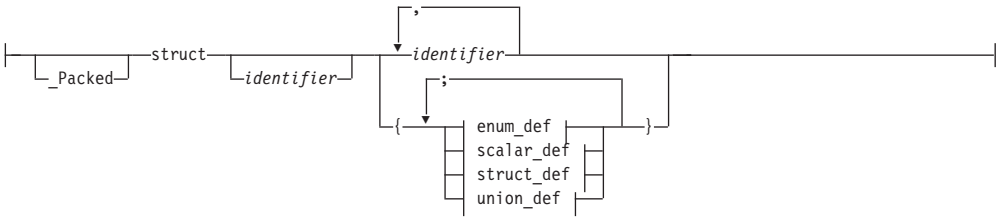


declarator:

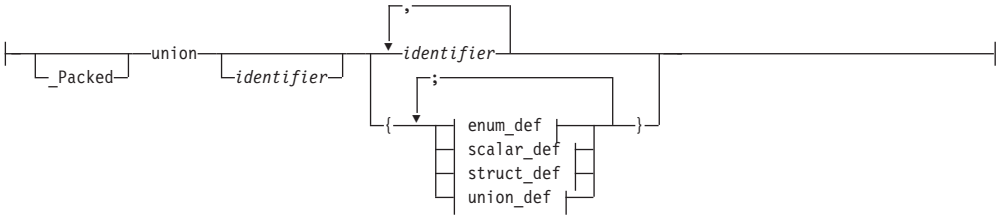
enum_def:



struct_def:

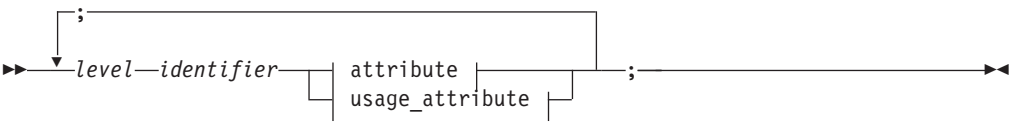


union_def:



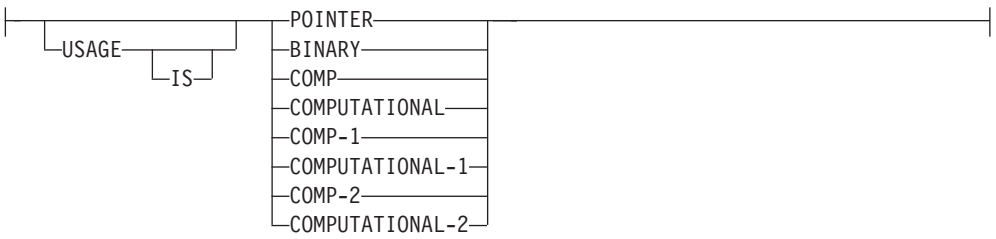
Declarations (COBOL)

Use declarations to create session variables effective during a Debug Tool session.



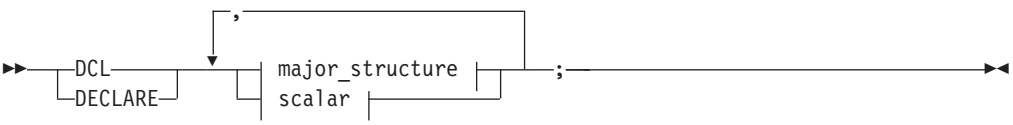
attribute:

usage_attribute:

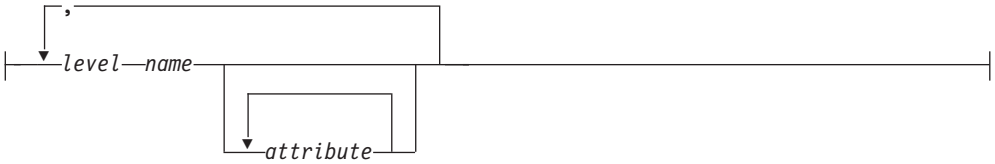


DECLARE command (PL/I)

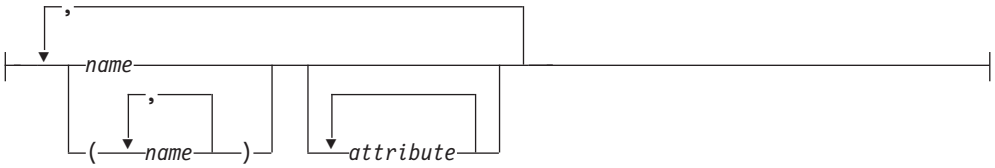
The DECLARE command creates session variables effective during a Debug Tool session.



major_structure:

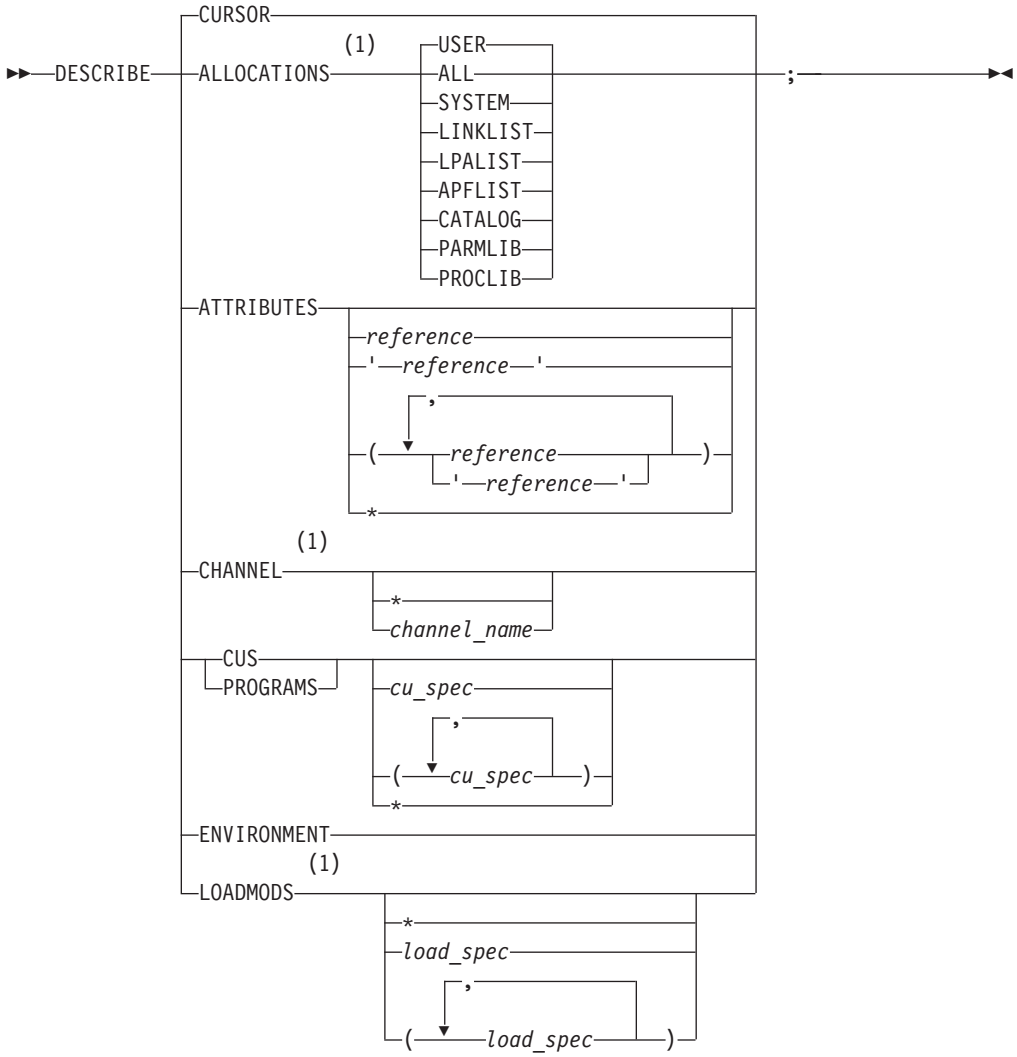


scalar:



DESCRIBE command

The DESCRIBE command displays the file allocations or attributes of references, compile units, known load modules, and the run-time environment.

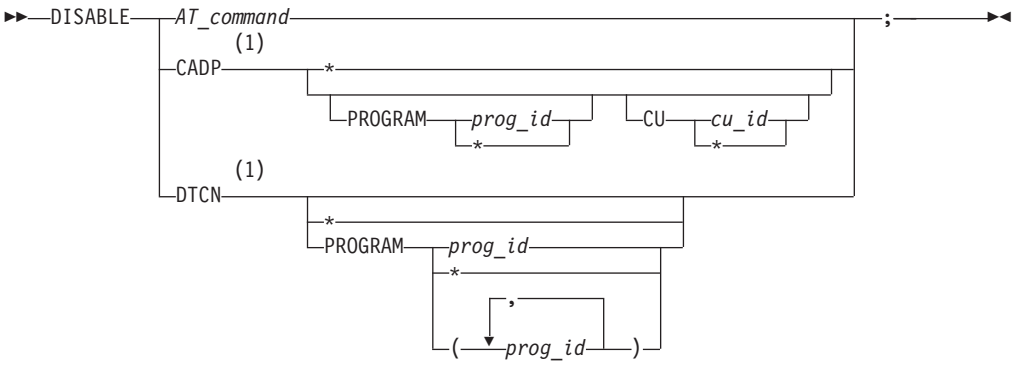


Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16)

DISABLE command

The DISABLE command makes an AT breakpoint inoperative or prevents Debug Tool from being started by CADP or DTCN. However, the breakpoint is not cleared. Later, you can make the breakpoint operative or allow Debug Tool to be started by CADP or DTCN by using the ENABLE command.



Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16)

DISABLE prefix (full-screen mode)

Disables a statement breakpoint or offset breakpoint when you issue this command through the Source window prefix area.

DO command (assembler, disassembly, and non-Language Environment COBOL)

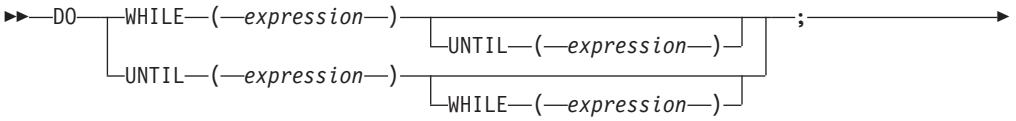
The DO command performs one or more commands that are collected into a group.

DO command (PL/I)

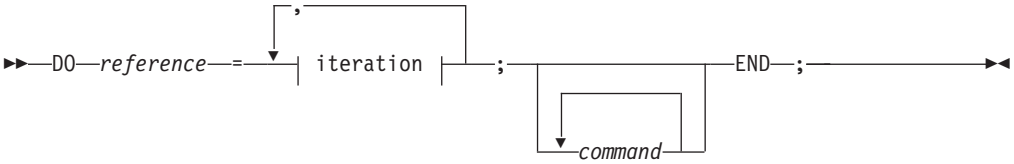
The DO command allows one or more commands to be collected into a group that can (optionally) be repeatedly executed.

Simple

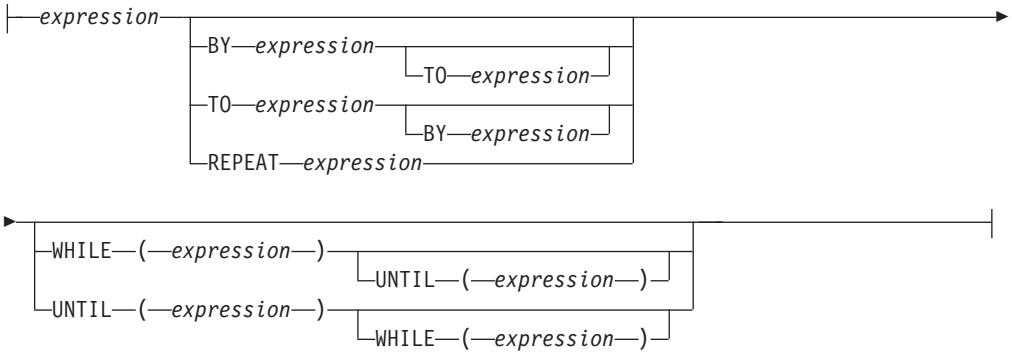
Repeating



Iterative

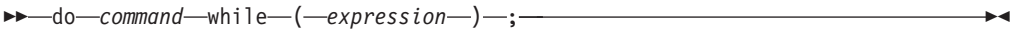


iteration:



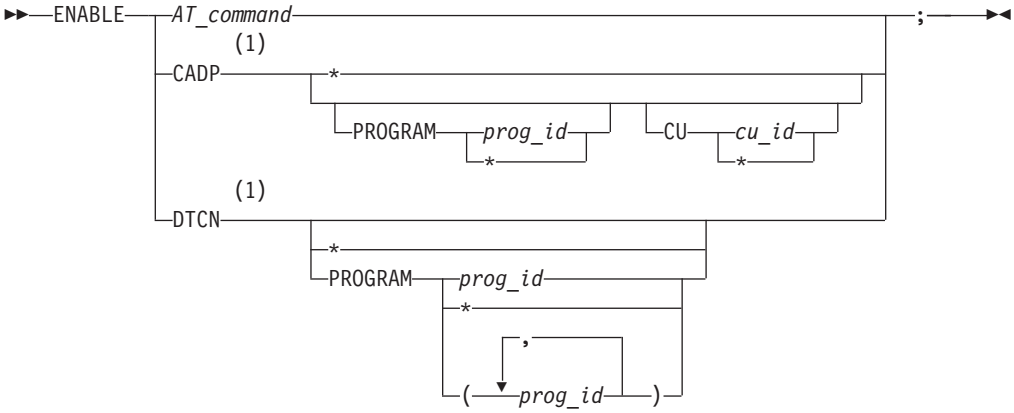
do/while command (C and C++)

The `do/while` command performs a command before evaluating the test expression.



ENABLE command

The ENABLE command activates an AT or pattern match breakpoint after it was disabled.



Notes:

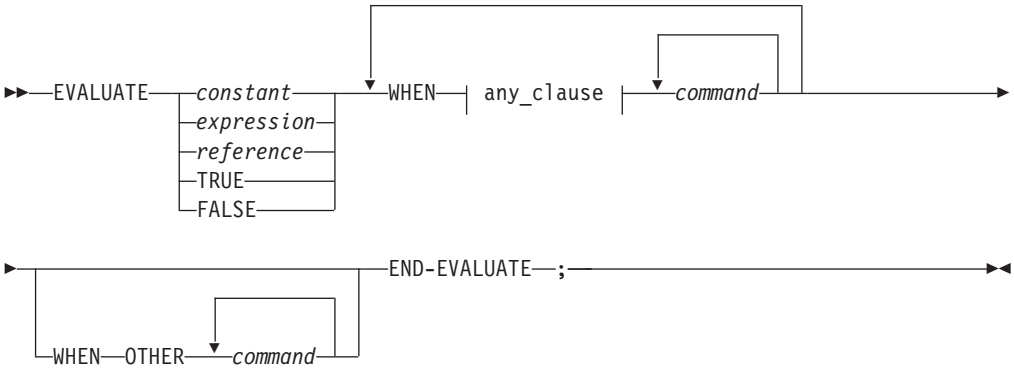
1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16)

ENABLE prefix (full-screen mode)

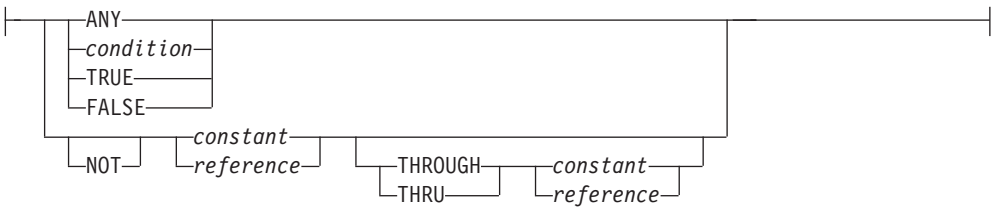
Enables a disabled statement breakpoint or a disabled offset breakpoint when you issue this command through the Source window prefix area.

EVALUATE command (COBOL)

The EVALUATE command provides a shorthand notation for a series of nested IF statements.



any_clause:

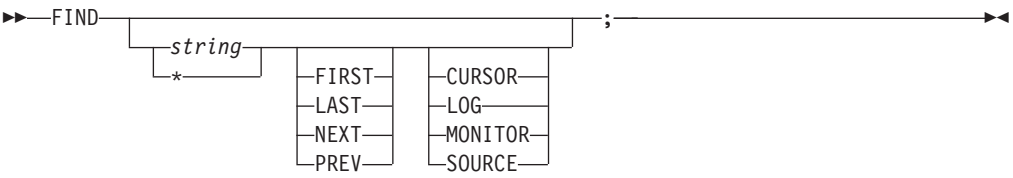


Expression command (C and C++)

The Expression command evaluates the given expression.

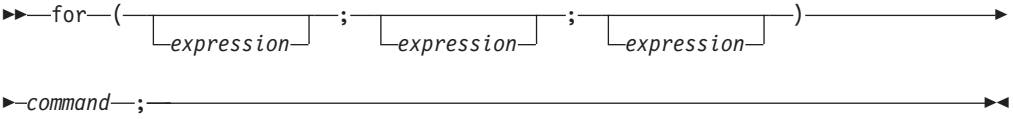
FIND command

The FIND command provides full-screen and batch mode search capability in source object, and full-screen searching of log and monitor objects.



for command (C and C++)

The for command provides iterative looping similar to the C and C++ for statement.



FREE command

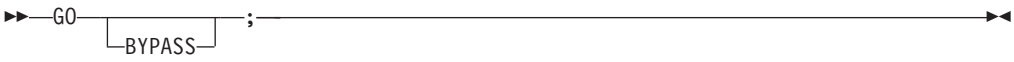
The FREE command releases a file that is currently allocated.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

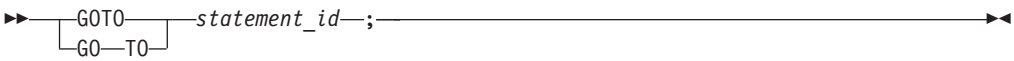
GO command

The GO command causes Debug Tool to start or resume running your program.



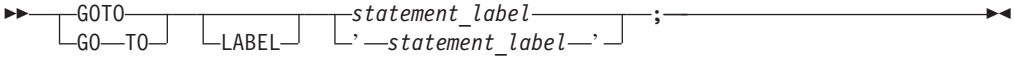
GOTO command

The GOTO command causes Debug Tool to resume program execution at the specified statement id.



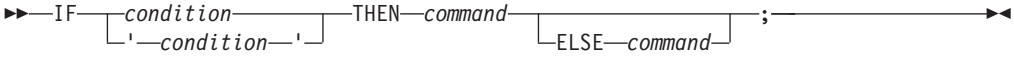
GOTO LABEL command

The GOTO LABEL command causes Debug Tool to resume program execution at the specified statement label. The specified label must be in the same block. If you want Debug Tool to return control to you at the target location, make sure there is a breakpoint at that location.



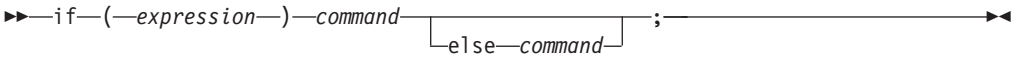
IF command (assembler, disassembly, and non-Language Environment COBOL)

The IF command lets you conditionally perform a command.



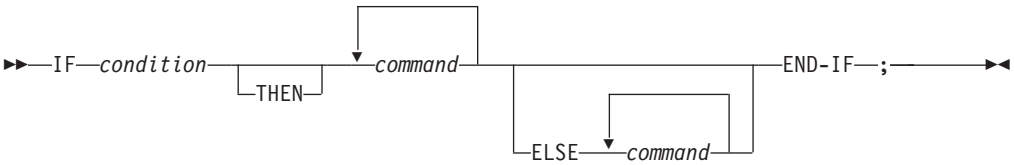
if command (C and C++)

The if command lets you conditionally perform a command.



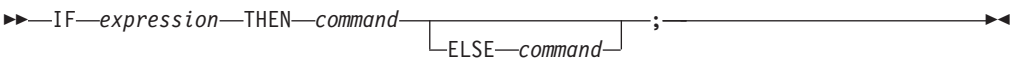
IF command (COBOL)

The IF command lets you conditionally perform a command.



IF command (PL/I)

The IF command lets you conditionally perform a command.

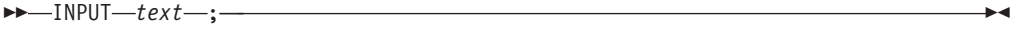


IMMEDIATE command (full-screen mode)

The IMMEDIATE command causes a command within a command list to be performed immediately.

INPUT command (C and C++ and COBOL)

The INPUT command provides data for an intercepted read and is valid only when there is a read pending for an intercepted file.



JUMPTO command

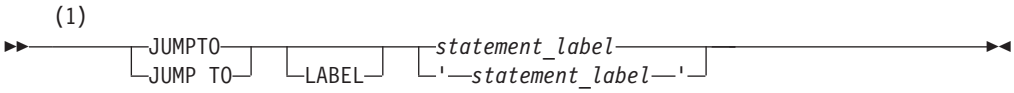
The JUMPTO command moves the resume point to the specified statement but does not resume the program.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

JUMPTO LABEL command

The JUMPTO command moves the resume point to the specified label but does not resume the program.



Notes:

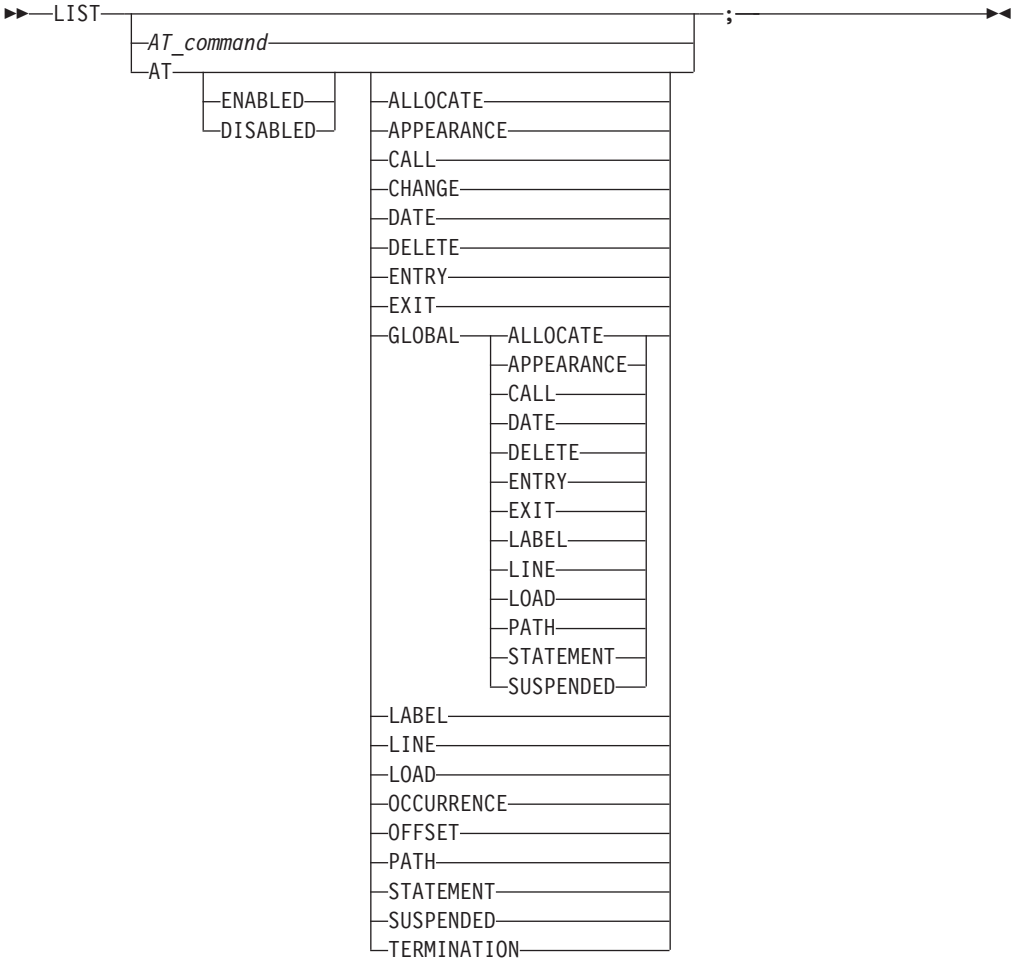
1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

LIST (blank)

Displays the Source Identification panel, where associations are made between source listings or source files shown in the source window and their program units.

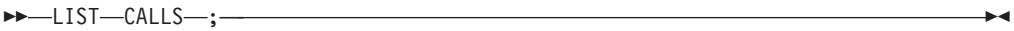
LIST AT

Lists the currently defined breakpoints, including the action taken when the specified breakpoint is activated.



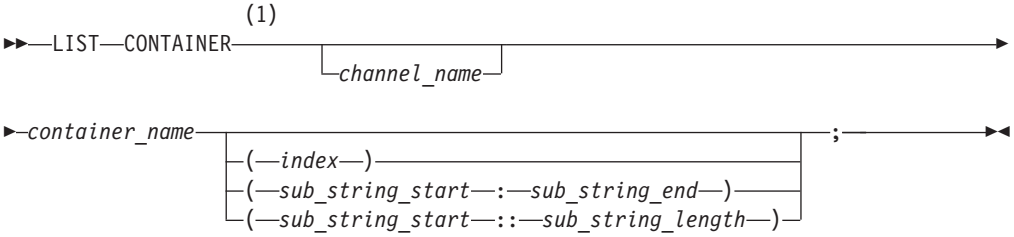
LIST CALLS

Displays the dynamic chain of active blocks.



LIST CONTAINER

Displays the contents of a CICS container.



Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

LIST CURSOR (full-screen mode)

Provides a cursor controlled method for displaying variables, structures, and arrays.

LIST DTCN or CADP

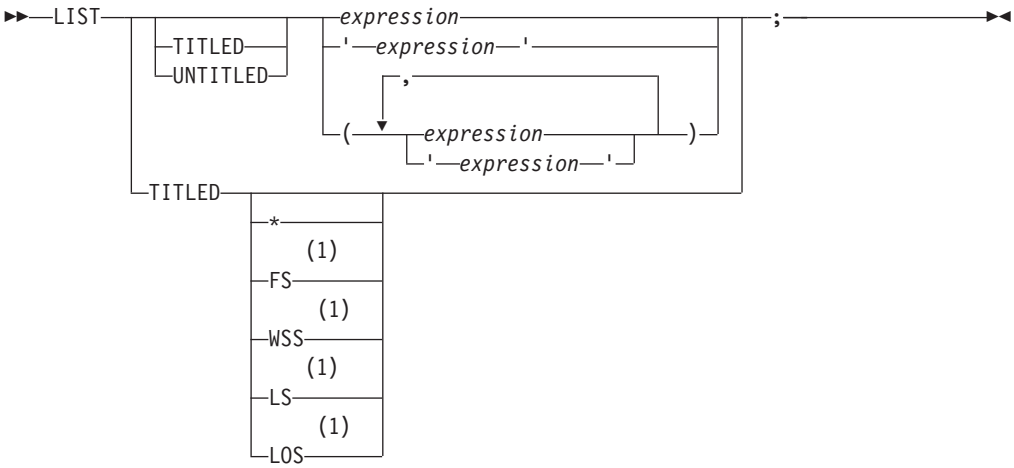
List the programs and compile units that were disabled by the `DISABLE` command.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16)

LIST expression

Displays values of expressions.

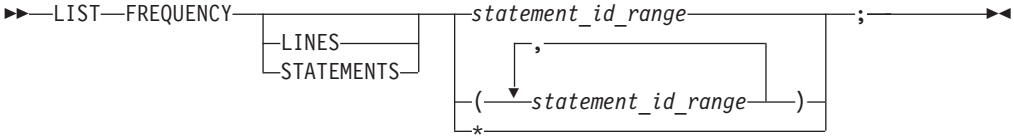


Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

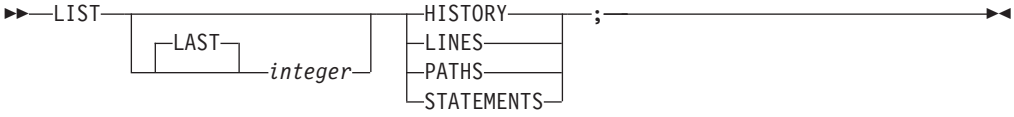
LIST FREQUENCY

Lists statement execution counts.



LIST LAST

Displays a list of recent entries in the history table.



LIST LINE NUMBERS

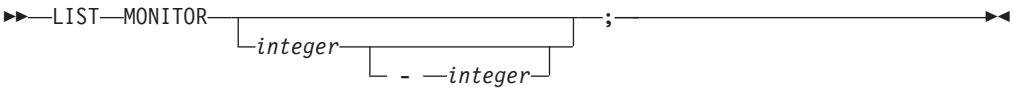
Equivalent to `LIST STATEMENT NUMBERS`.

LIST LINES

Equivalent to `LIST STATEMENTS`.

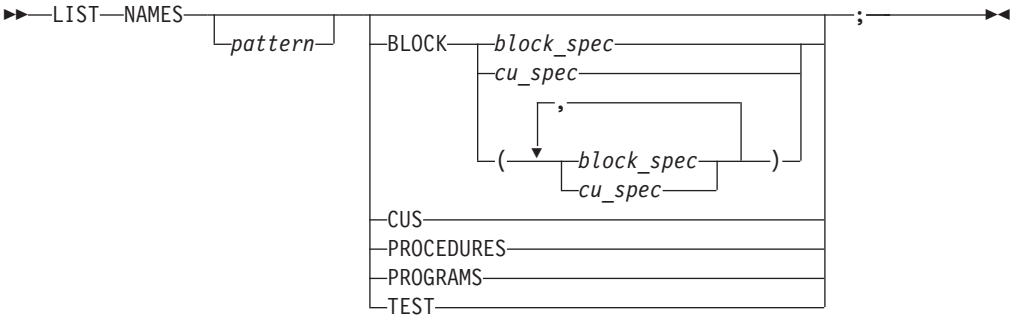
LIST MONITOR

Lists all or selected members of the current set of MONITOR commands.



LIST NAMES

Lists the names of variables, programs, or Debug Tool procedures.

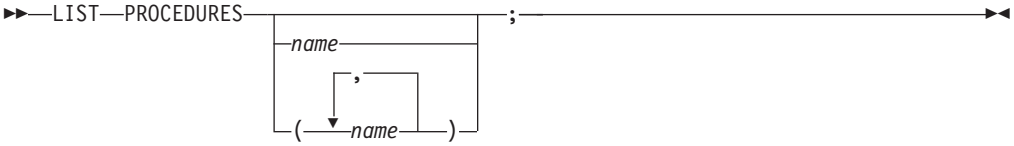


LIST ON (PL/I)

Lists the action (if any) currently defined for the specified PL/I conditions.

LIST PROCEDURES

Lists the commands contained in the specified Debug Tool PROCEDURE definitions.

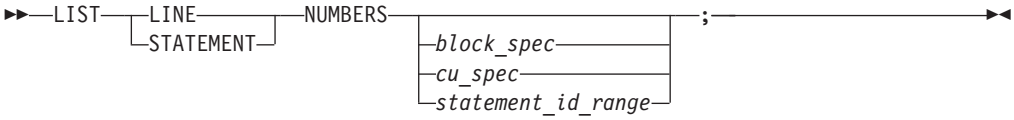


LIST REGISTERS

Displays the current register contents.

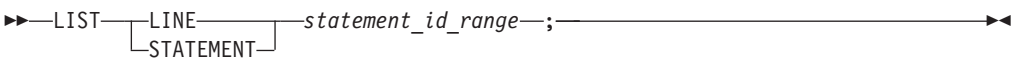
LIST STATEMENT NUMBERS

Lists all statement or line numbers that are valid locations for an AT LINE or AT STATEMENT breakpoint.



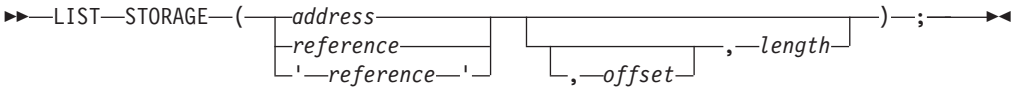
LIST STATEMENTS

Lists one or more statements or lines from a file.



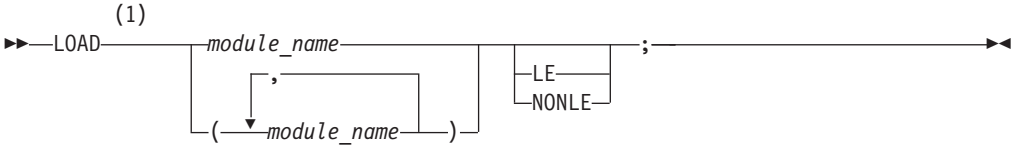
LIST STORAGE

Displays the contents of storage at a particular address in hex format.



LOAD command

Specifies to load the named module using MVS™ LOAD services, EXEC CICS LOAD, or the Language Environment enclave-level load service for debugging purposes.

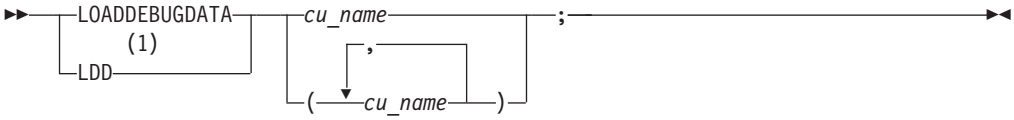


Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

LOADDEBUGDATA (LDD)

The LOADDEBUGDATA command specifies that a compile unit is an assembler compile unit and loads the debug data that corresponds to that assembler compile unit.



Notes:

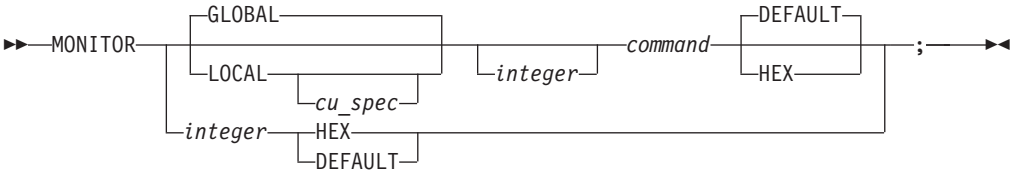
1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

MEMORY

Identifies an address in memory and then display the contents of memory at that location in the Memory window. The Memory window displays memory in dump format.

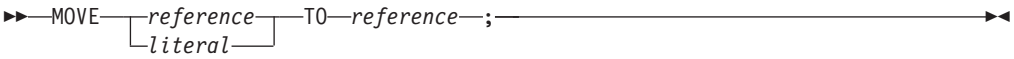
MONITOR command

The MONITOR command defines or redefines a command whose output is displayed in the monitor window (full-screen mode), terminal output (line mode), or log file (batch mode).



MOVE command (COBOL)

The MOVE command transfers data from one area of storage to another.

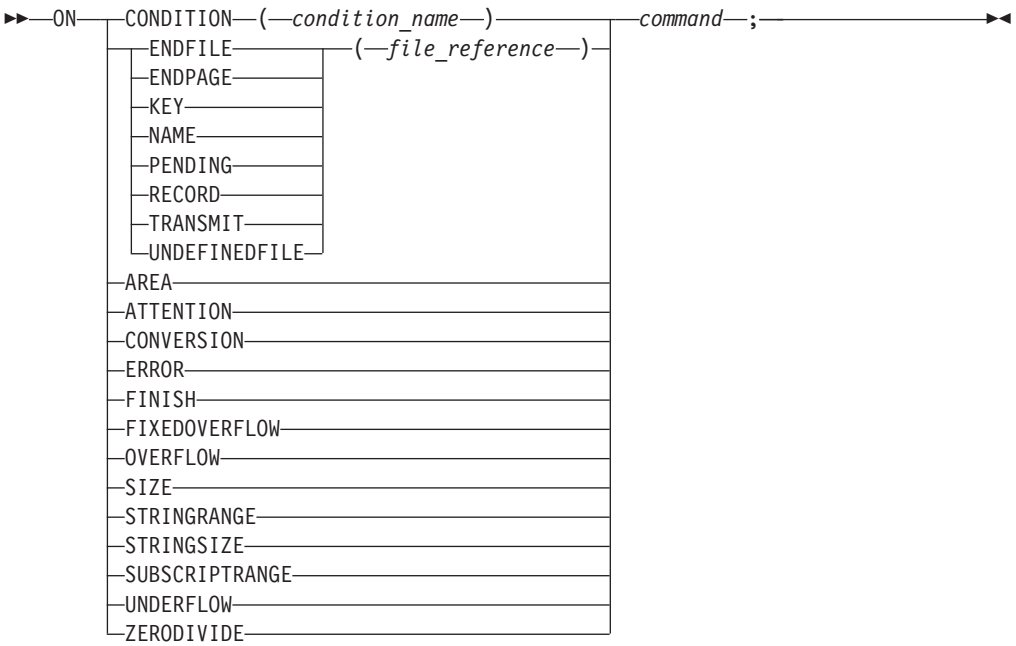


Null command

The Null command is a semicolon written where a command is expected.

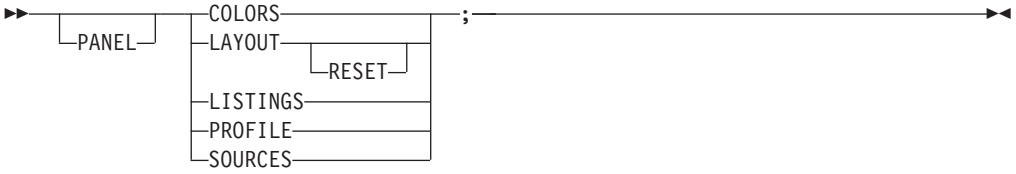
ON command (PL/I)

The ON command establishes the actions to be executed when the specified PL/I condition is raised.



PANEL command (full-screen mode)

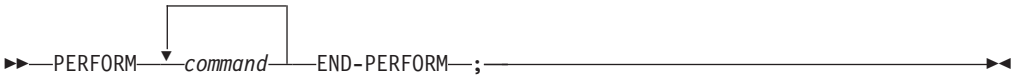
The PANEL command displays special panels.



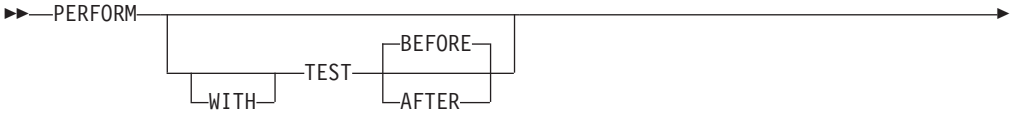
PERFORM command (COBOL)

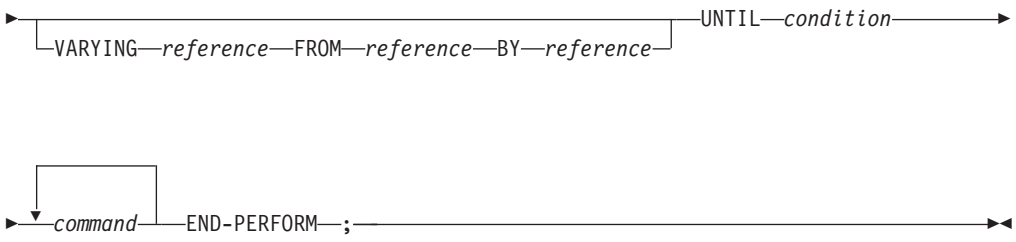
The PERFORM command transfers control explicitly to one or more statements and implicitly returns control to the next executable statement after execution of the specified statements is completed.

Simple:



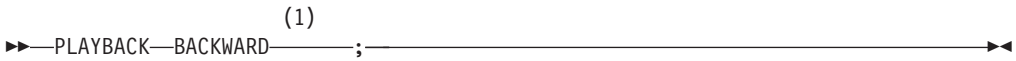
Repeating:





PLAYBACK BACKWARD command

The PLAYBACK BACKWARD command indicates to Debug Tool to perform STEP and RUNTO commands backward, starting from the current point and going to previous points.

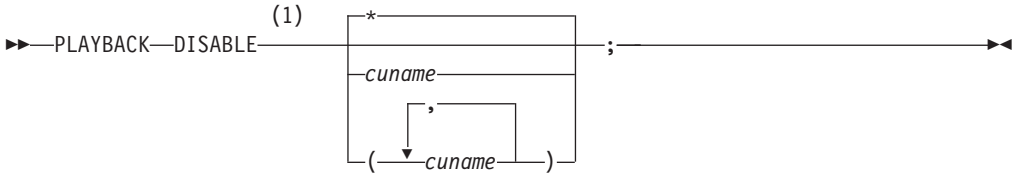


Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

PLAYBACK DISABLE command

The PLAYBACK DISABLE command informs Debug Tool to stop recording run-time environment information and discard any information recorded thus far.

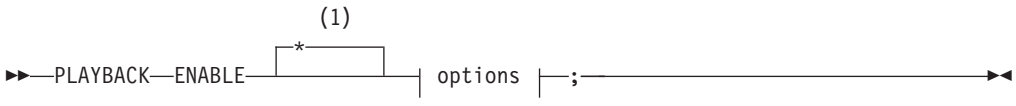


Notes:

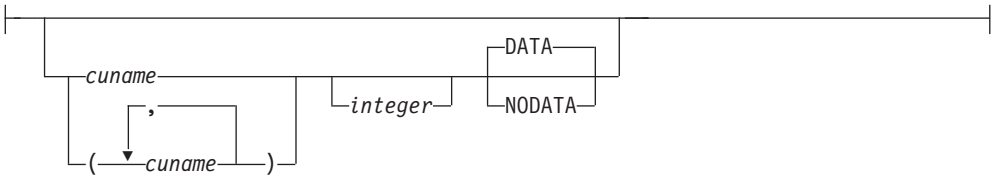
1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

PLAYBACK ENABLE command

The PLAYBACK ENABLE command informs Debug Tool to begin recording the application run-time environment information (steps history and data history).



options:



Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

PLAYBACK FORWARD command

The PLAYBACK FORWARD command indicates to Debug Tool to perform STEP and RUNTO commands forward, starting from the current point and going to the next point.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

PLAYBACK START command

The PLAYBACK START command suspends normal debugging and informs Debug Tool to replay the steps it recorded.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

PLAYBACK STOP command

The PLAYBACK STOP command stops replaying recorded statements and resumes normal debugging at the point where the PLAYBACK START command was entered.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

Prefix commands (full-screen mode)

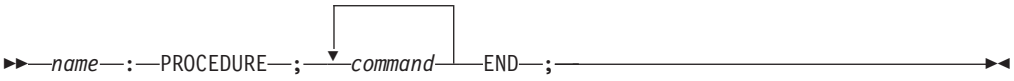
The Prefix commands apply only to source listing lines and are typed into the prefix area in the source window. For details, see the section corresponding to the command name.

The following table summarizes the forms of the Prefix commands.

"AT Prefix (full-screen mode)" on page 5	Defines a statement breakpoint through the Source window prefix area.
"CLEAR prefix (full-screen mode)" on page 9	Clears a breakpoint through the Source window prefix area.
"DISABLE prefix (full-screen mode)" on page 15	Disables a breakpoint through the Source window prefix area.
"ENABLE prefix (full-screen mode)" on page 17	Enables a disabled breakpoint through the Source window prefix area.
"QUERY prefix (full-screen mode)" on page 32	Queries what statements have breakpoints through the Source window prefix area.
"RUNTO prefix command (full-screen mode)" on page 33	Runs the program to the location that the cursor or statement identifier indicate in the Source window prefix area.
"SHOW prefix command (full-screen mode)" on page 43	Specifies what relative statement or verb within the line is to have its frequency count shown in the suffix area.

PROCEDURE command

The PROCEDURE command allows the definition of a group of commands that can be accessed by using the CALL procedure command.

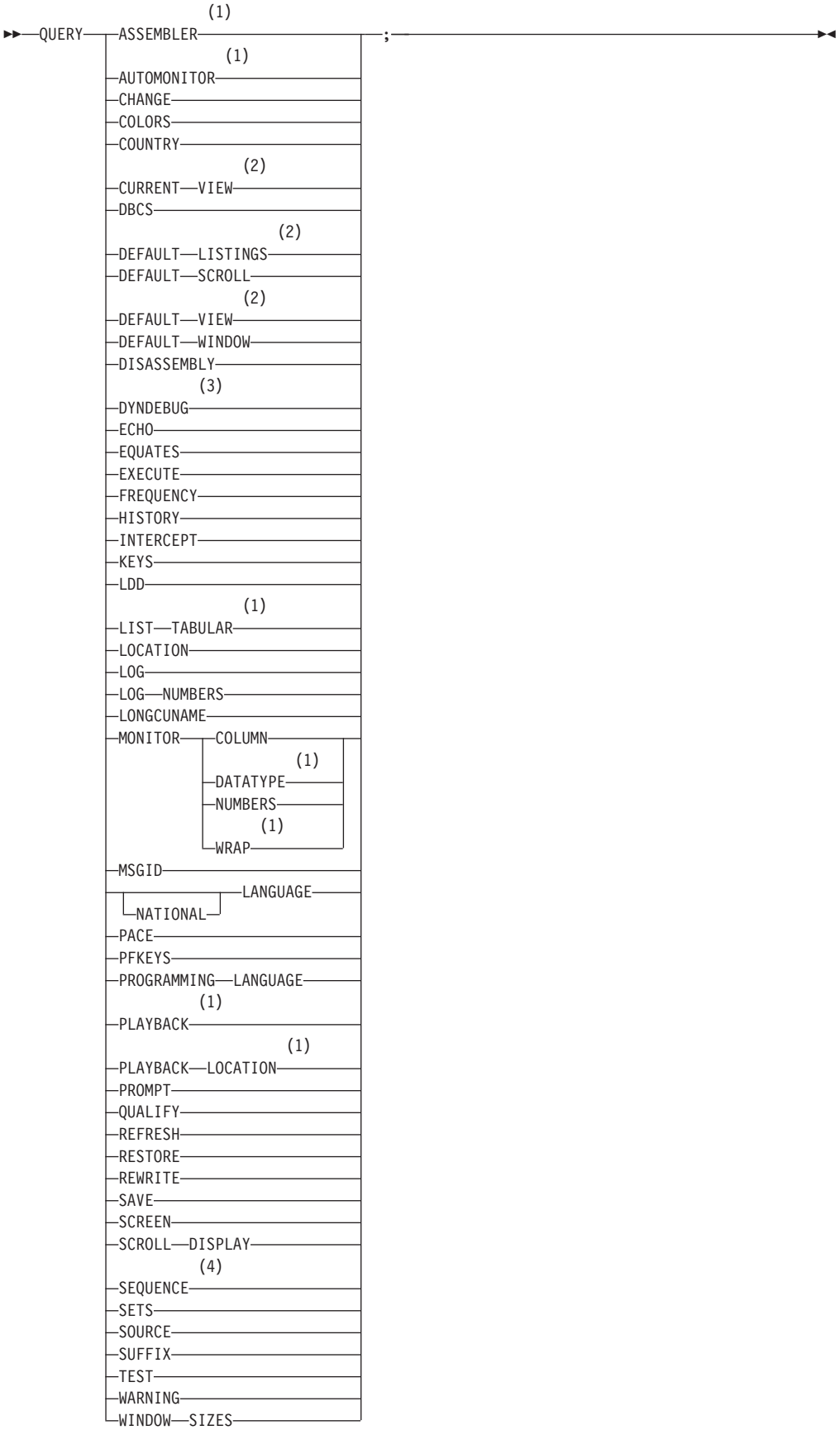


QUALIFY RESET

This command is equivalent to SET QUALIFY RESET.

QUERY command

The QUERY command displays the current value of the specified Debug Tool setting, the current setting of all the Debug Tool settings, or the current location in the suspended program.


QUERY
ASSEMBLER (1)
AUTOMONITOR (1)
CHANGE
COLORS
COUNTRY
CURRENT VIEW (2)
DBCS
DEFAULT LISTINGS (2)
DEFAULT SCROLL
DEFAULT VIEW (2)
DEFAULT WINDOW
DISASSEMBLY
DYNDEBUG (3)
ECHO
EQUATES
EXECUTE
FREQUENCY
HISTORY
INTERCEPT
KEYS
LDD
LIST TABULAR (1)
LOCATION
LOG
LOG NUMBERS
LONGCUNAME
MONITOR
COLUMN
DATATYPE (1)
NUMBERS
WRAP (1)
MSGID
LANGUAGE
NATIONAL
PACE
PFKEYS
PROGRAMMING LANGUAGE
PLAYBACK (1)
PLAYBACK LOCATION (1)
PROMPT
QUALIFY
REFRESH
RESTORE
REWRITE
SAVE
SCREEN
SCROLL DISPLAY
SEQUENCE (4)
SETS
SOURCE
SUFFIX
TEST
WARNING
WINDOW SIZES
;

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

2 You can use this command in remote debug mode.

3 Available only if the Dynamic Debug facility is installed.

4 Only for PL/I.

QUERY prefix (full-screen mode)

Queries what statements on a particular line have statement breakpoints when you issue this command through the Source window prefix area.

QUIT command

The QUIT command ends a Debug Tool session and if an expression is specified, sets the return code.

QQUIT command

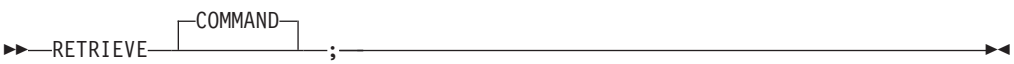
The QQUIT command ends a Debug Tool session without further prompting.

RESTORE command

The RESTORE command enables you to explicitly restore the settings, breakpoints, and monitor specifications that were previously saved by the SET SAVE AUTO command when Debug Tool terminated.

RETRIEVE command (full-screen mode)

The RETRIEVE command displays the last command entered on the command line.



RUN command

The RUN command is synonymous to the GO command.

RUNTO command

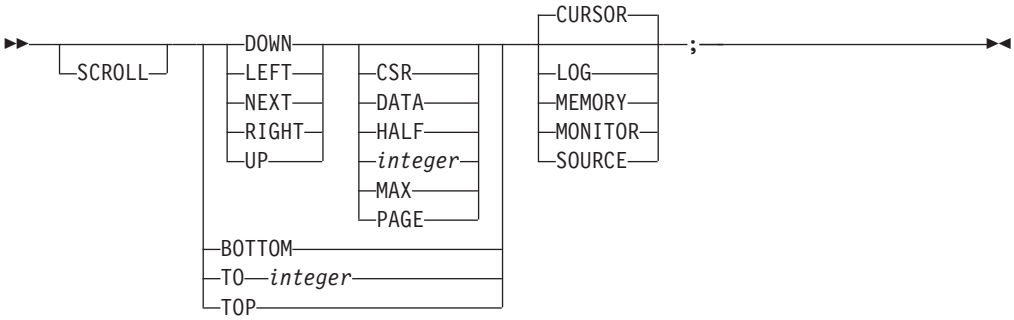
The RUNTO command runs your program to a valid executable statement without setting a breakpoint.

RUNTO prefix command (full-screen mode)

Runs to the statement when you issue this command through the Source window prefix area.

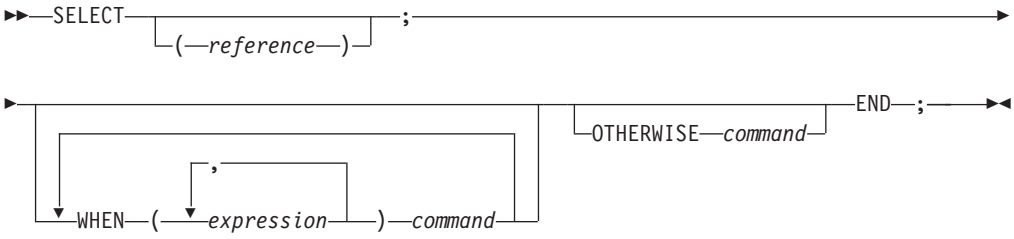
SCROLL command (full-screen mode)

The SCROLL command provides horizontal and vertical scrolling in full-screen mode.



SELECT command (PL/I)

The SELECT command chooses one of a set of alternate commands.



SET ASSEMBLER

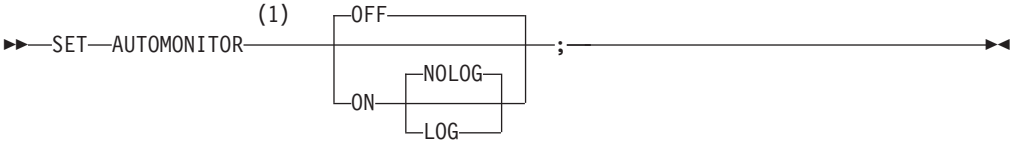
The SET ASSEMBLER command displays additional information that is useful when you debug an assembler program.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

SET AUTOMONITOR

Controls the monitoring of data items at the currently executing statement.

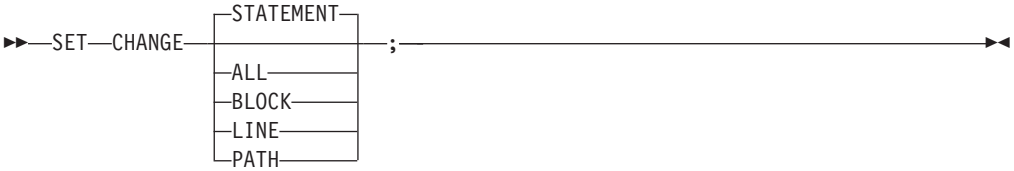


Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

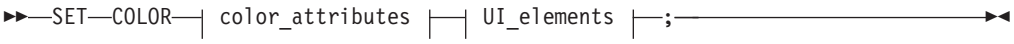
SET CHANGE

Controls the frequency of checking the AT CHANGE breakpoints.

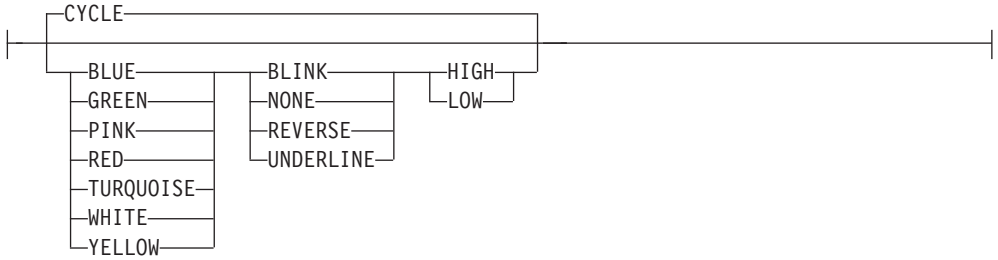


SET COLOR (full-screen and line mode)

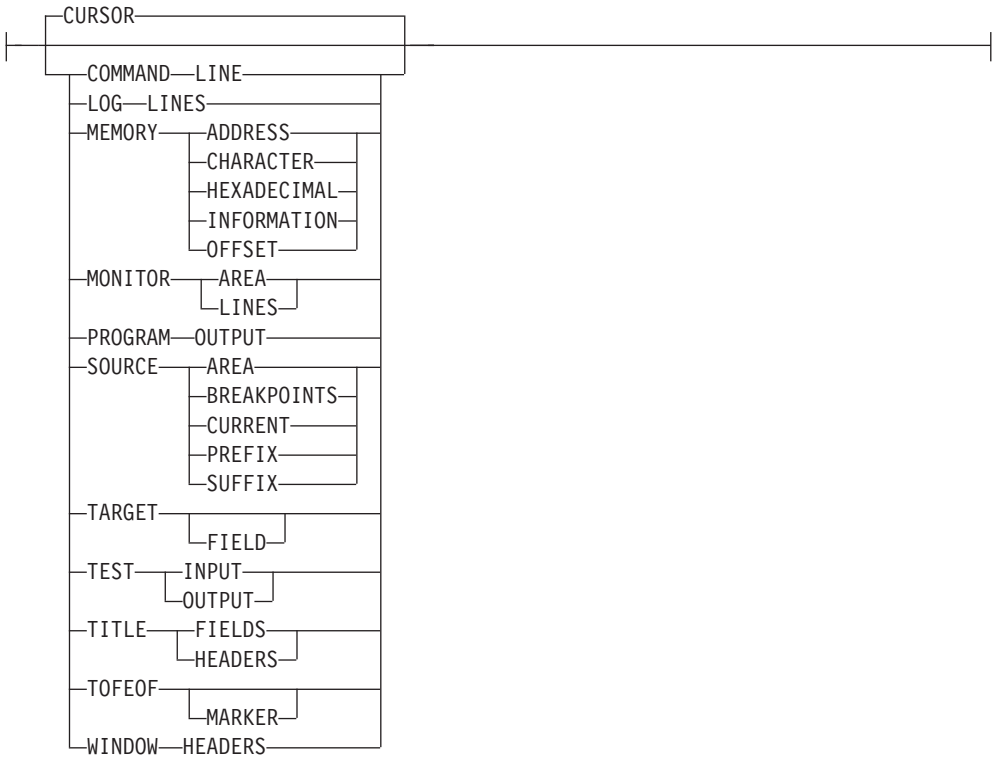
Provides control of the color, highlighting, and intensity attributes when the SCREEN setting is ON.



color_attributes:



UI_elements:



SET COUNTRY

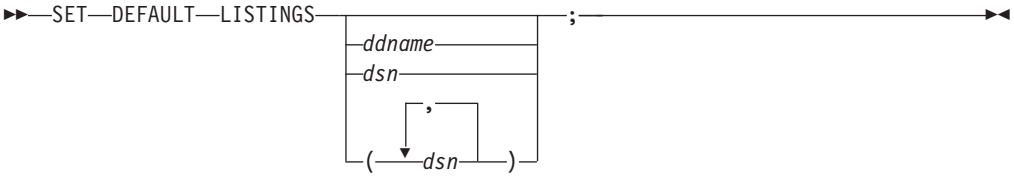
Changes the current national country setting for the application program.

SET DBCS

Controls whether shift-in and shift-out codes are interpreted on input and supplied on DBCS output.

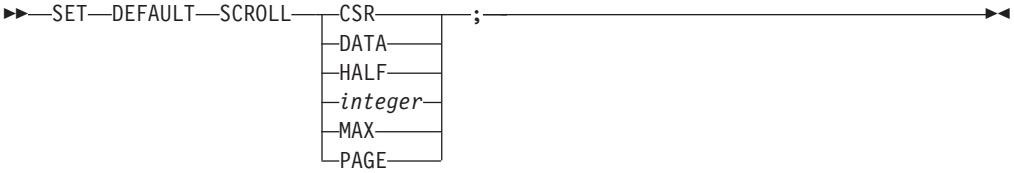
SET DEFAULT LISTINGS

Defines a default partitioned data set DD name or DS name whose members are searched for program source, listings, or separate debug files.



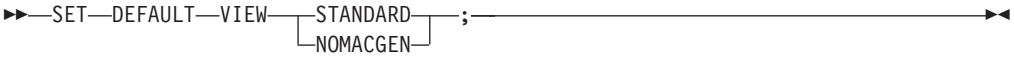
SET DEFAULT SCROLL (full-screen mode)

Sets the default scroll amount that is used when a SCROLL command is issued without the amount specified.



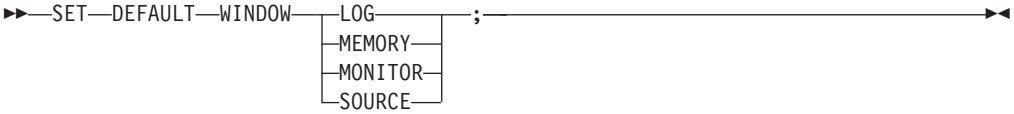
SET DEFAULT VIEW

Controls the default view for assembler compile units.



SET DEFAULT WINDOW (full-screen mode)

Specifies what window is selected when a window referencing command (for example, FIND, SCROLL, or WINDOW) is issued without explicit window identification and the cursor is outside the window areas.



SET DISASSEMBLY

Controls whether the disassembly view is displayed in the Source window.

SET DYNDEBUG

Controls whether to activate the Dynamic Debug facility.

SET ECHO

Controls whether GO and STEP commands are recorded in the log window when they are not subcommands.

SET EQUATE

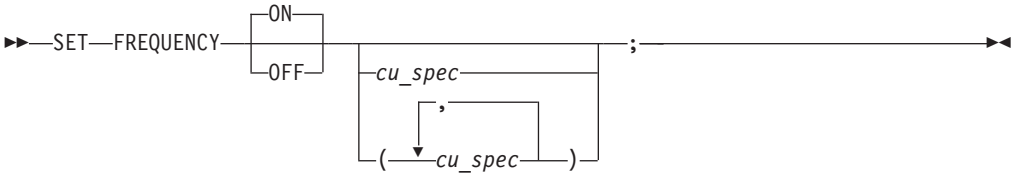
Equates a symbol to a string of characters.

SET EXECUTE

Controls whether commands from all input sources are performed or just syntax checked (primarily for checking USE files).

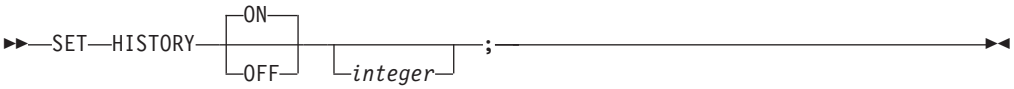
SET FREQUENCY

Controls whether statement executions are counted.



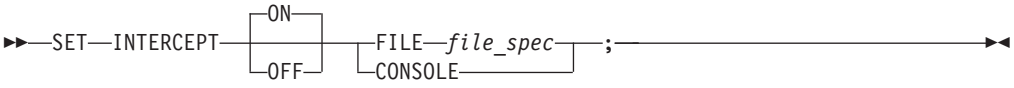
SET HISTORY

Specifies whether entries to Debug Tool are recorded in the history table and optionally adjusts the size of the table.



SET INTERCEPT (C, C++, and COBOL)

Intercepts input to and output from specified files.



SET KEYS (full-screen and line mode)

Controls whether PF key definitions are displayed when the SCREEN setting is ON.

SET LDD

Controls how debug data is loaded for assemblies containing multiple CSECTs.

SET LIST TABULAR

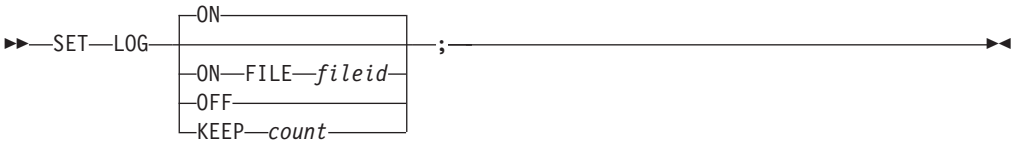
Controls whether to format the output of the LIST command in a tabular format.

Notes:

1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

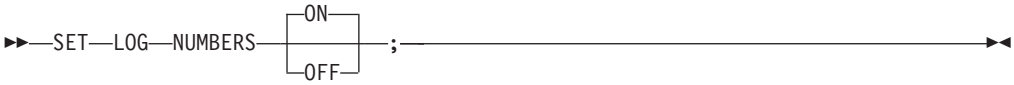
SET LOG

Controls whether each performed command and the resulting output is written to the log file and defines (or redefines) the file that is used.



SET LOG NUMBERS (full-screen and line mode)

Controls whether line numbers are shown in the log window.

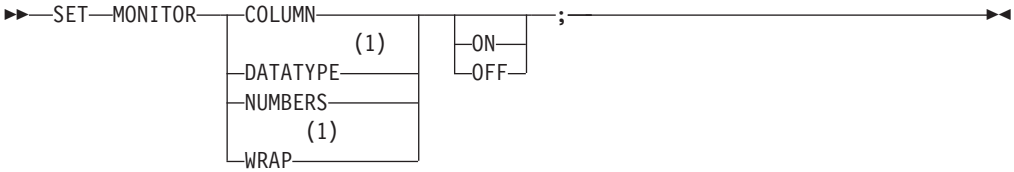


SET LONGCUNAME (C, C++, and PL/I)

Controls whether the CU name is displayed in short or long format.

SET MONITOR (full-screen and line mode)

Controls the format and layout of variable names and values displayed in the Monitor window.



Notes:

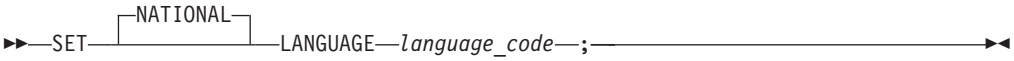
1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

SET MSGID

Controls whether the Debug Tool messages are displayed with the message prefix identifiers.

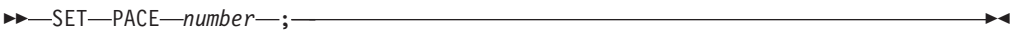
SET NATIONAL LANGUAGE

Switches your application to a different run-time national language that determines what translation is used when a message is displayed.



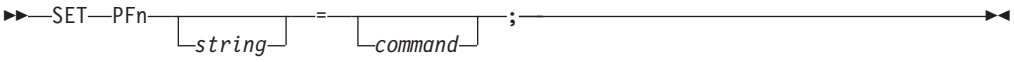
SET PACE

Specifies the maximum speed (in steps per second) of animated execution.



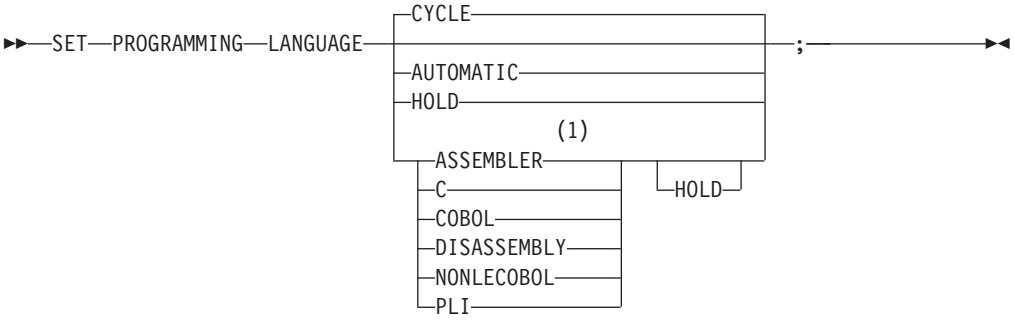
SET PFKEY

Associates a Debug Tool command with a Program Function key (PF key).



SET PROGRAMMING LANGUAGE

Sets the current programming language.



Notes:

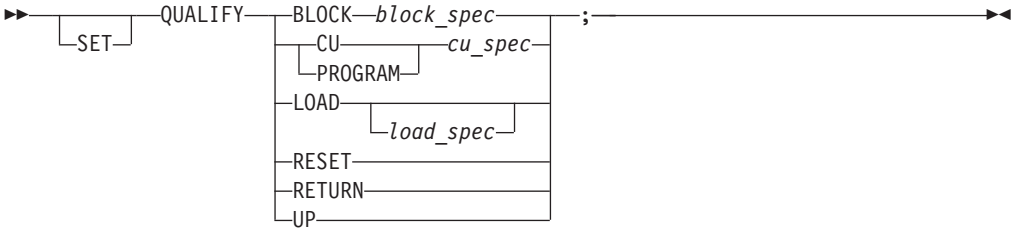
1 Available only with Debug Tool Utilities and Advanced Functions (5655-S16).

SET PROMPT (full-screen and line mode)

Controls whether the current program location is automatically shown as part of the prompt message in line mode.

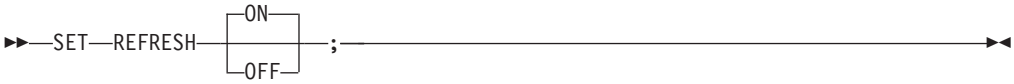
SET QUALIFY

Simplifies the identification of references and statement numbers by resetting the point of view to a new block, compile unit, or load module.



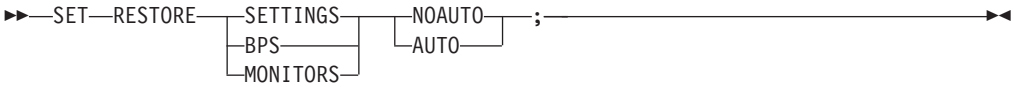
SET REFRESH (full-screen mode)

Controls screen refreshing.



SET RESTORE

Controls the restoring of settings, breakpoints, and monitor specifications.

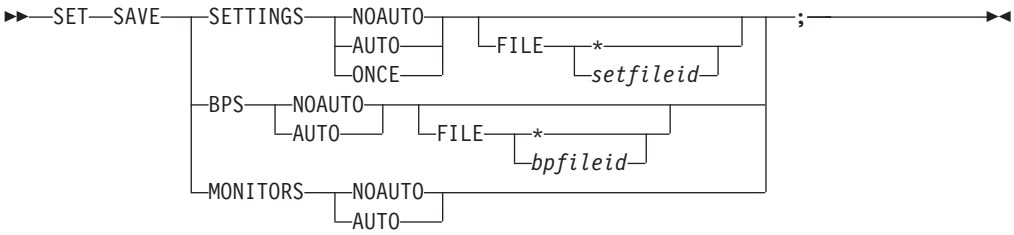


SET REWRITE

Forces a periodic screen rewrite during long sequences of output.

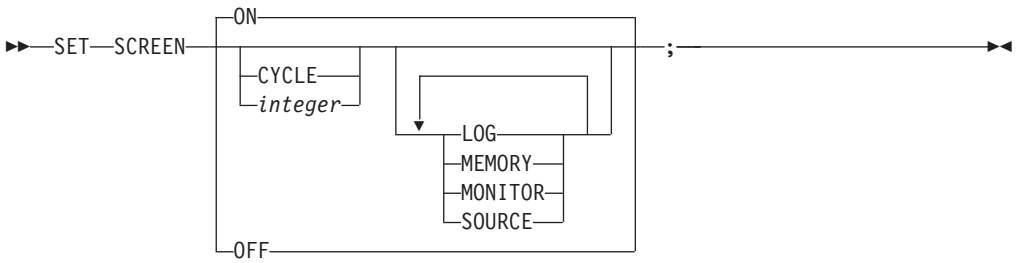
SET SAVE

Controls the saving of settings, breakpoints, and monitor specifications.



SET SCREEN (full-screen and line mode)

Controls how information is displayed on the screen.



SET SCROLL DISPLAY (full-screen mode)

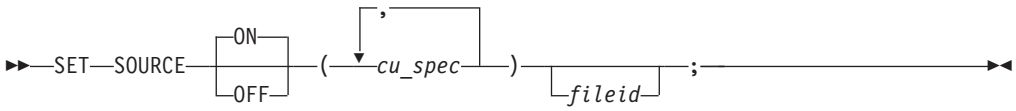
Controls whether the scroll field is displayed when operating in full-screen mode.

SET SEQUENCE (PL/I)

Controls whether Debug Tool interprets data after column 72 in a commands or preference file as a sequence number.

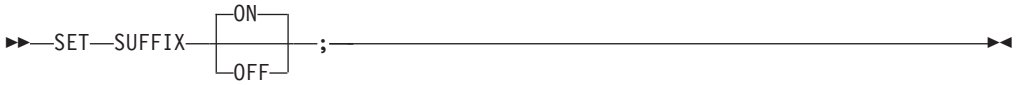
SET SOURCE

Associates a source file, compiler listing or separate debug file with one or more compile units.



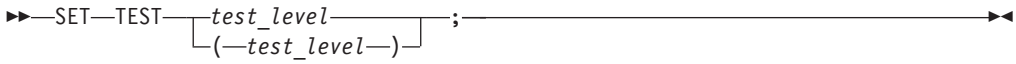
SET SUFFIX (full-screen mode)

Controls the display of frequency counts at the right edge of the Source window when in full-screen mode.



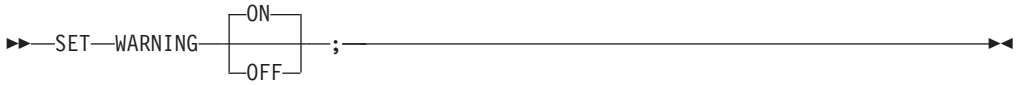
SET TEST

Overrides the initial TEST run-time options specified at invocation.



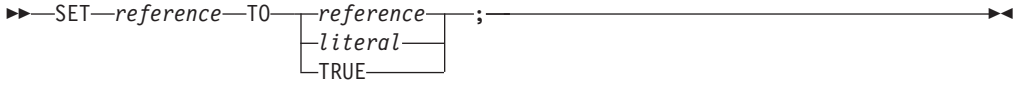
SET WARNING (C, C++, and PL/I)

Controls display of the Debug Tool warning messages and whether exceptions are reflected to the application program.



SET command (COBOL)

The SET command assigns a value to a COBOL reference.



SHOW prefix command (full-screen mode)

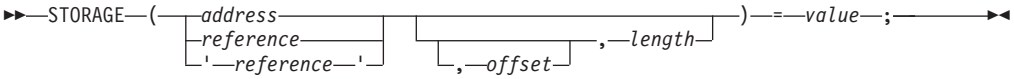
The SHOW prefix command specifies what relative statement (for C) or relative verb (for COBOL) within the line is to have its frequency count temporarily shown in the suffix area.

STEP command

The STEP command causes Debug Tool to dynamically step through a program, executing one or more program statements. In full-screen mode, it provides animated execution.

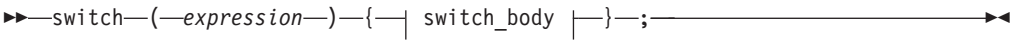
STORAGE command

The STORAGE command enables you to alter up to eight bytes of storage.

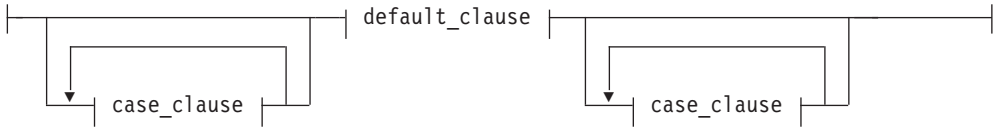


switch command (C and C++)

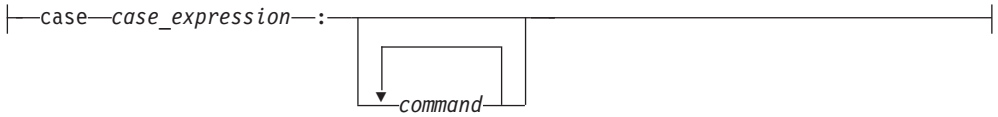
The switch command enables you to transfer control to different commands within the switch body, depending on the value of the switch expression.



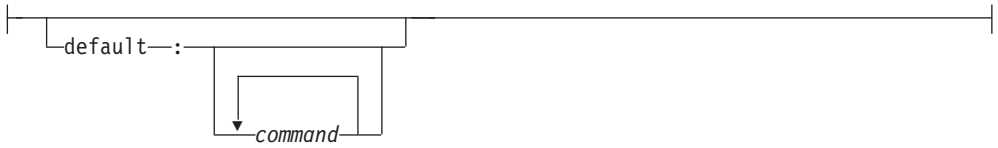
switch_body:



case_clause:



default_clause:

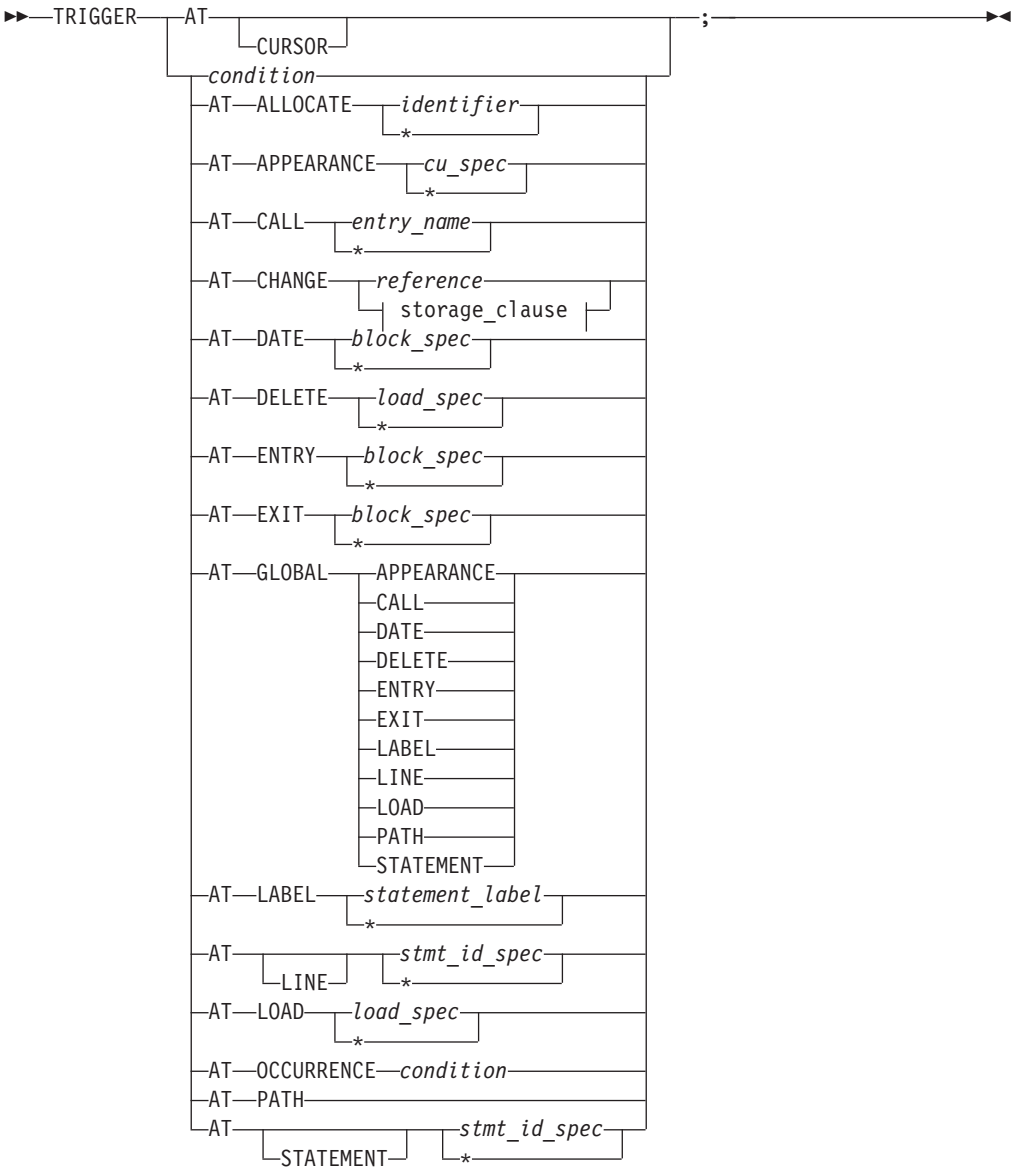


SYSTEM command

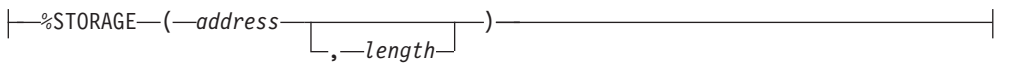
The SYSTEM command lets you issue TSO commands during a Debug Tool session.

TRIGGER command

The TRIGGER command raises the specified AT-condition in Debug Tool, or it raises the specified programming language condition in your program.

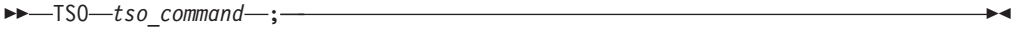


storage_clause:



TSO command (z/OS)

The TSO command lets you issue TSO commands during a Debug Tool session and is valid only in a TSO environment.



USE command

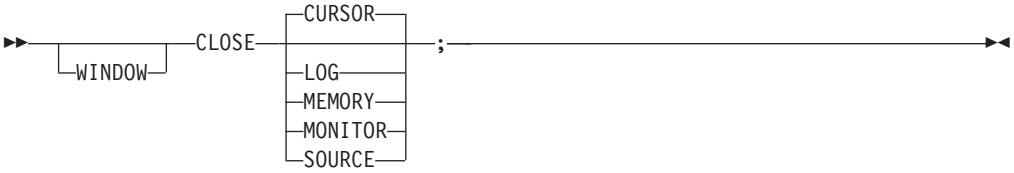
The USE command causes the Debug Tool commands in the specified file or data set to be either performed or syntax checked.

while command (C and C++)

The while command enables you to repeatedly perform the body of a loop until the specified condition is no longer met or evaluates to false.

WINDOW CLOSE

Closes the specified window in the Debug Tool full-screen session panel.

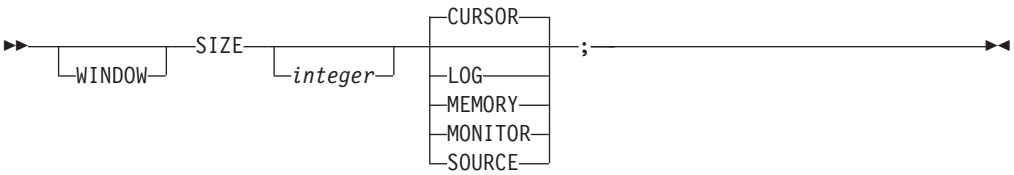


WINDOW OPEN

Opens a previously-closed window in the Debug Tool full-screen session panel.

WINDOW SIZE

Controls the relative size of currently visible windows in the Debug Tool full-screen session panel.

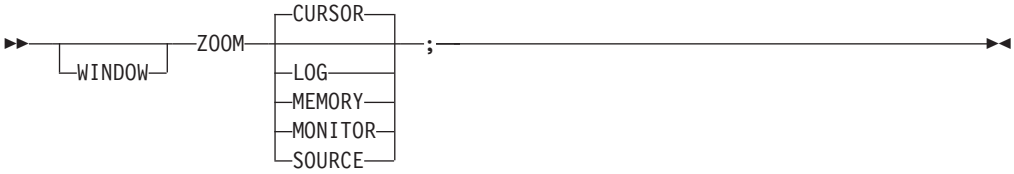


WINDOW SWAP

Replaces the logical window being displayed in a physical window with another logical window. The order of the operands is not important.

WINDOW ZOOM

Expands the indicated window to fill the entire screen or restores the screen to the currently defined window configuration.



Chapter 2. Debug Tool built-in functions

Debug Tool provides you with the following built-in functions:

%DEC (assembler, disassembly, and non-Language Environment COBOL)

Returns the decimal value of an operand.

%GENERATION (PL/I)

Returns a specific generation of a controlled variable in your program.

%HEX

Returns the hexadecimal value of an operand.

%INSTANCES (C, C++, and PL/I)

Returns the maximum value of %RECURSION (the most recent recursion number) for a given block.

%RECURSION (C, C++, and PL/I)

Returns a specific instance of an automatic variable or a parameter in a recursive procedure.

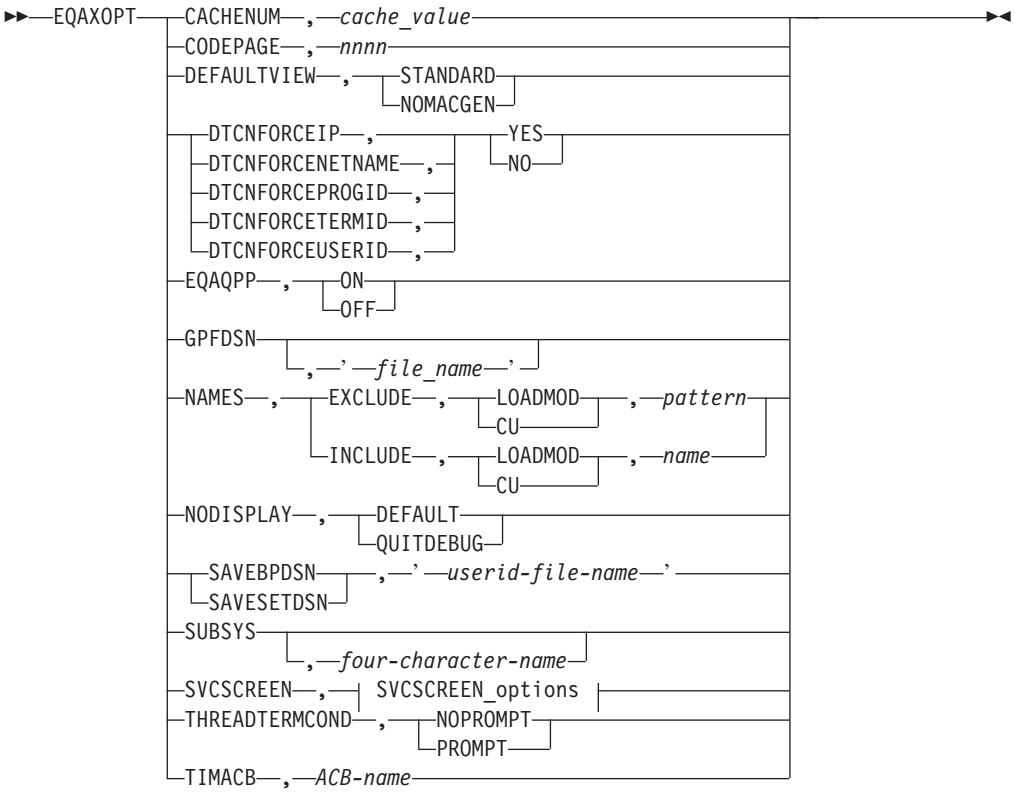
%WHERE (assembler, disassembly, and non-Language Environment COBOL)

Returns a string that is the address of the operand. %WHERE can be used *only* as the outermost expression in the LIST command.

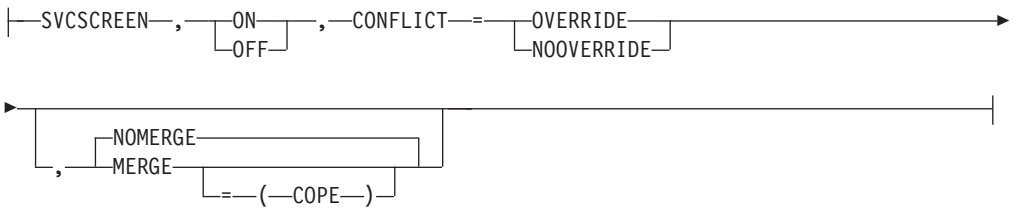
Chapter 3. EQAOPTS options

The EQAOPTS load module allows customization of certain Debug Tool functions, which can apply to the site, a group, or a single user. This topic summarizes the purpose of each EQAOPTS option. See *Debug Tool Customization Guide* for information on how to create the EQAOPTS load module and a complete description of each option.

The follow diagram describes the syntax of this option:



SVCSCREEN_options:



CACHENUM

Specifies the maximum number of program items to be held in an in-memory cache for a CICS debug session. Increase this number to improve performance for applications which have many programs; however, a larger number also increases the storage usage by the debugged task.

Related tasks

Overriding the default number of program elements held in cache in *Debug Tool Customization Guide*

CODEPAGE

Indicates which code page to use so that NLS characters are properly communicated between Debug Tool and a remote debugger and properly displayed in full screen mode.

Related tasks

Specifying a code page (optional) in *Debug Tool Customization Guide*

DEFAULTVIEW

Provides a method of setting the initial value for the `SET DEFAULT VIEW` command.

Related tasks

Requiring users to specify resource types in *Debug Tool Customization Guide*

DTCNFORCE*xxxxxx*

Controls DTCN behavior for the conditions described in Table 2. When you set a DTCNFORCE*xxxxxx* option to YES, DTCN forces users to specify the respective resource type. The default setting is NO.

Table 2.

EQAXOPT option	DTCN field name
DTCNFORCETERMID	Terminal Id
DTCNFORCETRANID	Transaction Id
DTCNFORCEPROGID	Program Id(s)
DTCNFORCEUSERID	User Id
DTCNFORCENETNAME	NetName
DTCNFORCEIP	IP client name or address

Related tasks

Setting the initial value for SET DEFAULT VIEW in *Debug Tool Customization Guide*

EQAQPP

Indicates the presence of Q++ programs.

Related tasks

Configuring for debugging Q++ programs in *Debug Tool Customization Guide*

GPFDSN

Specifies the data set name for the global preferences file.

Related tasks

Specifying global preferences (optional) in *Debug Tool Customization Guide*

NAMES

Provides a method of entering `NAMES` commands that apply before the first load module and any of the compile units contained in that load module are processed.

Related tasks

Supplying NAMES commands for the initial load module (optional) in *Debug Tool Customization Guide*

NODISPLAY

Controls Debug Tool behavior when the terminal using full-screen mode through a VTAM terminal or the remote debugger are not available.

Related tasks

Modifying Debug Tool behavior when requested user interface is not available in *Debug Tool Customization Guide*

SAVEBPDSN and SAVESETDSN

Specifies the data set names to be used to save the breakpoints (SAVEBPDSN) and settings (SAVESETDSN). One qualifier in each of these data set names should be `&&USERID`, which represents the user ID of the current user.

Related tasks

Modifying the name of the default data sets that store settings, breakpoints, and monitor values (optional) in *Debug Tool Customization Guide*

SUBSYS

Provides a 1 to 4 character subsystem name. If an Enterprise PL/I or C/C++ source file is found to have a DSORG of DA or VSAM and this parameter is supplied, then this parameter is passed to SVC 99 (dynamic allocation) through the SUBSYS text unit when Debug Tool allocates the source file.

Related tasks

Specifying SUBSYS to access source code in a library system in *Debug Tool Customization Guide*

SVCSCREEN

Controls Debug Tool's enablement of SVC screening.

Related tasks

Setting the SVC screening option in *Debug Tool Customization Guide*

THREADTERMCOND

Specifies whether Debug Tool should suppress the prompt it displays when the the thread termination condition, FINISH condition, or CEE067 is raised by Language Environment.

Related tasks

Suppressing the prompt Debug Tool displays for FINISH, CEE066, or CEE067 conditions in *Debug Tool Customization Guide*

TIMACB

Specifies an alternate Terminal Information Manager ACB name.

Related tasks

Running the Terminal Interface Manager on more than one LPAR on the same VTAM® network in *Debug Tool Customization Guide*

Notices

This information was developed for products and services offered in the U.S.A. IBM might not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with the local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement might not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Copyright license

Programming interface information

This book is intended to help you debug application programs. This publication documents intended Programming Interfaces that allow you to write programs to obtain the services of Debug Tool.

Trademarks and service marks

The following terms, denoted by an asterisk (*) on the first occurrence in this publication, are trademarks or service marks of International Business Machines Corporation in the United States or other countries:

IBM
The IBM logo
ibm.com
AD/Cycle
AIX
C/370
CICS
COBOL/370
DB2
IBM
IMS
Language Environment
MVS
MVS/ESA
OS/390
Resource Link
VisualAge
VSE/ESA
VTAM
WebSphere
z/OS
z/VM

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

Acrobat®, Adobe®, the Adobe logo, PostScript®, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other contries.

Java™ and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

LINUX is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT®, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

MasterCraft is a trademark of Tata Consultancy Services Ltd.

Readers' Comments — We'd Like to Hear from You

Debug Tool for z/OS
Debug Tool Utilities and Advanced Functions for z/OS
Reference Summary
Version 8.1

Publication No. SC19-1199-01

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Submit your comments using one of these channels:
- Send your comments to the address on the reverse side of this form.
- Send your comments via e-mail to: COMMENTS@US.IBM.COM

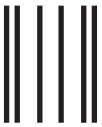
If you would like a response from IBM, please fill in the following information:

Name	Address
Company or Organization	
Phone No.	E-mail address

Readers' Comments — We'd Like to Hear from You
SC19-1199-01

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Reader Comments
DTX/E269
555 Bailey Ave.
San Jose, CA
95141-9989

Fold and Tape | **Please do not staple** | Fold and Tape

SC19-1199-01

Cut or Fold
Along Line

Program Number: 5655-S17

Printed in USA

SC19-1199-01