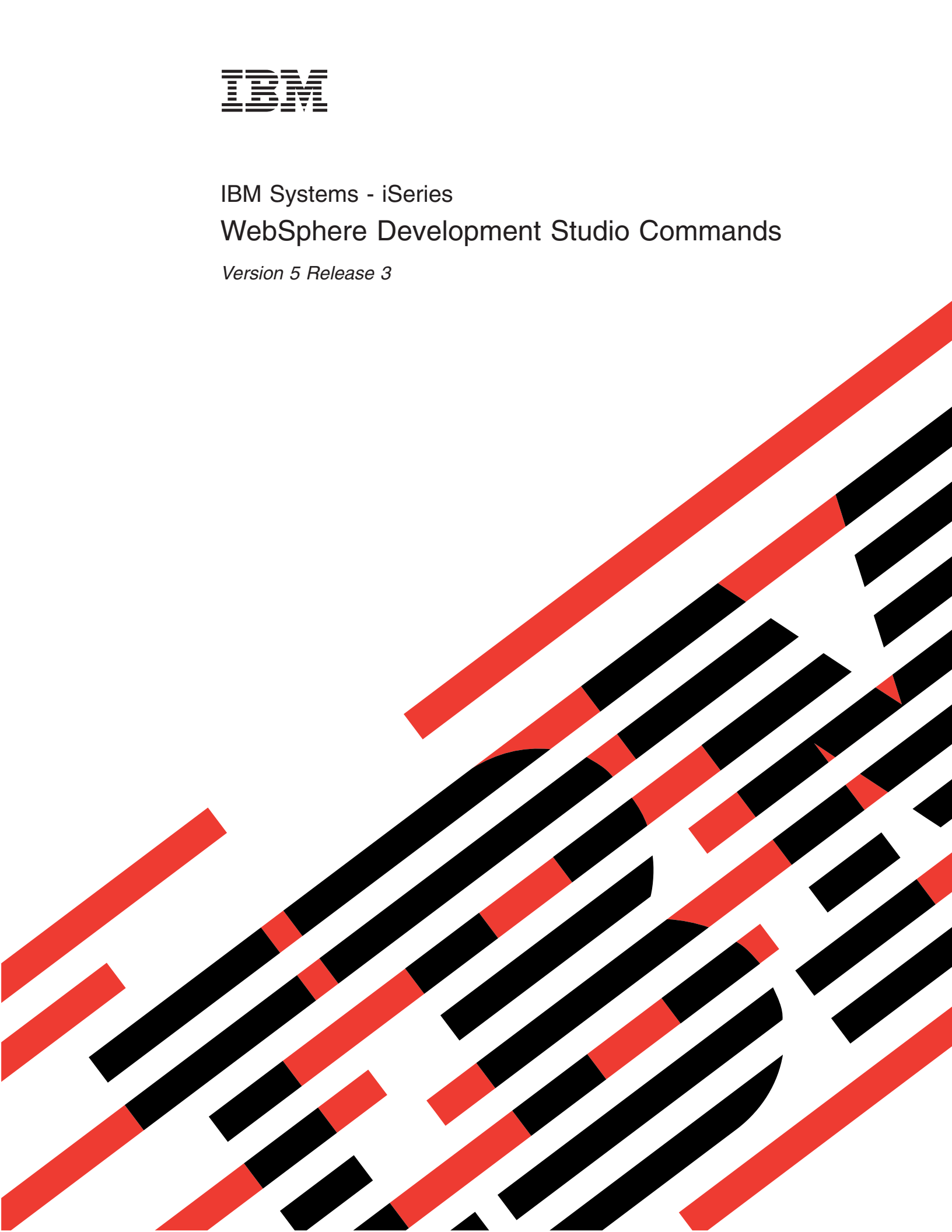

IBM Systems - iSeries

WebSphere Development Studio Commands

Version 5 Release 3

IBM Systems - iSeries

WebSphere Development Studio Commands

Version 5 Release 3

Note

Before using this information and the product it supports, be sure to read the information in "Notices," on page 349.

First Edition (May 2004)

This edition applies to version 5, release 3, modification 0 of WebSphere Development Studio (product number 5722-WDS) and to all subsequent releases and modifications until otherwise indicated in new editions. This version does not run on all reduced instruction set computer (RISC) models nor does it run on CICS models.

Contents

Change PDM Defaults (CHGPDMDFT)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Change PDM Defaults (CHGPDMDFT) command enables you to assign Programming Development Manager (PDM) defaults for a specific user. The function is similar to the Change Defaults panel shown when F18 is pressed when using PDM, except that this command can be used to change the PDM defaults for another user and can be run in batch.

Restrictions:

- You must have object management (*OBJMGT) and use (*USE) authorities to the user profile of the user whose PDM defaults are being changed.

Top

Parameters

Keyword	Description	Choices	Notes
USER	User	*Simple name*	Required, Positional 1
OBJLIB	Object library	*Name*, ***SAME**, *SRCLIB, *CURLIB	Optional
RPLOBJ	Replace object	***SAME**, *NO, *YES	Optional
CRTBCH	Create/compile in batch	***SAME**, *YES, *NO	Optional
RUNBCH	Run in batch	***SAME**, *NO, *YES	Optional
SAVRSTOPT	Save/restore option	***SAME**, *SINGLE, *ALL	Optional
JOBD	Job description	Single values: ***SAME**, *USRPRF Other values: *Qualified object name*	Optional
	Qualifier 1: Job description	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
CHGTYPTXT	Change type and text	***SAME**, *YES, *NO	Optional
FILE	Option file	Single values: ***SAME** Other values: *Qualified object name*	Optional
	Qualifier 1: Option file	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
MBR	Option file member	*Name*, ***SAME**	Optional
FULLSCN	Full screen mode	***SAME**, *NO, *YES	Optional
LOGCOM	Log option commands	***SAME**, *NO, *YES	Optional
EXITENT	Exit lists on ENTER	***SAME**, *NO, *YES	Optional
DSPINFMSG	Display informational messages	***SAME**, *YES, *NO	Optional

Top

User (USER)

Specifies the user whose PDM defaults are to be changed.

This is a required parameter.

simple-name
Specify the name of the user whose default values are to be changed. A user profile with the specified name must exist on the system.

Top

Object library (OBJLIB)

Specifies the library where objects created by compiling source file members (from the Work with Members Using PDM display) are to be stored.

***SAME**
If this parameter was previously set, the value does not change; otherwise *SRCLIB is used.

***SRCLIB**
Use the library in which the source member exists as the object library.

***CURLIB**
Use the current library as the object library. If no current library is defined, QGPL is used.

name Specify the name of the library where objects resulting from compilation of source file members are to be stored.

Top

Replace object (RPLOBJ)

Specifies if the existing object is deleted and replaced with the new object created when compiling a member or creating a module.

***SAME**
If this parameter was previously set, the value does not change; otherwise *NO is used.

***NO** The existing object is not deleted before starting to compile a member or create a module. If the object exists, the Confirm Member Compile display appears.

***YES** The existing object is deleted before starting to compile a member or create a module. If the compilation fails, the object is not restored.

Top

Create/compile in batch (CRTBCH)

Specifies whether to submit a job to batch when compiling members or creating modules.

***SAME**
If this parameter was previously set, the value does not change; otherwise *YES is used.

***YES** Compile members or create modules in batch.

***NO** Compile members or create modules interactively.

Top

Run in batch (RUNBCH)

Specifies whether to submit a job to batch when running objects.

***SAME**
If this parameter was previously set, the value does not change; otherwise *NO is used.

***NO** Objects do not run in batch.

***YES** Objects run in batch.

Top

Save/restore option (SAVRSTOPT)

Specifies whether to save or restore objects and members individually or with one command.

***SAME**
If this parameter was previously set, the value does not change; otherwise *SINGLE is used.

***SINGLE**
Save or restore the selected objects or members individually, each with its own separate command.

***ALL** Save or restore the selected objects or members all at the same time with one command.

Top

Job description (JOBD)

Specifies the name of the job description for submitting a job in batch mode.

Single values

***SAME**
If this parameter was previously set, the value does not change; otherwise *LIBL/QBATCH is used.

***USRPRF**
The job description defined in the user profile of the user specified for the **User (USER)** parameter is used.

Qualifier 1: Job description

name Specify the name of the job description to be used.

Qualifier 2: Library

***LIBL** All libraries in the job library list will be searched for the specified job description.

***CURLIB**
The current library for the job will be searched for the specified job description. If no current library is defined, QGPL is used as the current library.

name Specify the name of the library to be searched for the specified job description.

Top

Change type and text (CHGTYPTXT)

Specifies if the **Type** and **Text** prompts can be changed by typing over them on the Work with Members Using PDM display.

***SAME**
If this parameter was previously set, the value does not change; otherwise *YES is used.

***YES** The **Type** and **Text** prompts can be changed by typing over them on the Work with Members Using PDM display.

***NO** The **Type** and **Text** prompts cannot be changed by typing over them on the Work with Members Using PDM display.

Top

Option file (FILE)

Specifies the file that contains the member with the user-defined options. The user-defined options in this file are the active user-defined options. The user-defined option file has a particular format. For more information on this format and how to copy a user-defined options file, refer to the AS/400 Programming Development Manager User's Guide and Reference.

Single values

***SAME**
If this parameter was previously set, the value does not change; otherwise FILE(*LIBL/QAUOOPT) is used.

Qualifier 1: Option file

name Specify the name of active user-defined options file.

Qualifier 2: Library

***LIBL** All libraries in the job library list will be searched for the active user-defined options file.

***CURLIB**
The current library for the job will be searched for the active user-defined options file. If no current library is defined, QGPL is used as the current library.

name Specify the name of the library to be searched for the active user-defined options file.

Top

Option file member (MBR)

Specifies the member that contains the user-defined options. The user-defined options in this member are the active user-defined options.

***SAME**
If this parameter was previously set, the value does not change; otherwise QAUOOPT is used.

name Specify the name of the member that contains the user-defined options.

Top

Full screen mode (FULLSCN)

Specifies whether the options and function keys are shown on the Work With displays. This option does not apply to the Work with User-Defined Options display.

***SAME**
If this parameter was previously set, the value does not change; otherwise *NO is used.

***NO** Options and function keys are shown on the Work With displays.

***YES** Options and function keys are not shown on the Work With displays.

Top

Log option commands (LOGCOM)

Specifies whether commands resulting from PDM options or user-defined options are logged to the job log.

***SAME**
If this parameter was previously set, the value does not change; otherwise *NO is used.

***NO** Do not log commands resulting from PDM options or user-defined options.

***YES** Log commands resulting from PDM options or user-defined options.

Top

Exit lists on ENTER (EXITENT)

Specifies whether the user can exit list panels with the ENTER key. List panels include Library, Object, Member, User-Defined Options.

***SAME**
If this parameter was previously set, the value does not change; otherwise *NO is used.

***NO** The user cannot exit list panels with the ENTER key.

***YES** The user can exit list panels with the ENTER key.

Top

Display informational messages (DSPINFMSG)

Specifies if the informational message about the new tools (RSE and CODE) is displayed or not.

***SAME**
If this parameter was previously set, the value does not change; otherwise *YES is used.

***YES** The informational message is displayed.

***NO** The informational message is not displayed.

Top

Examples

Example 1: Set Defaults for New PDM User

```
CHGPDMDFT   USER(USER2)
```

This command sets the PDM defaults if USER2 is a new PDM user. If the defaults were already set, they are not changed.

Example 2: Change Defaults for Existing PDM User

```
CHGPDMDFT   USER(USER2)  CRTBCH(*NO)
            RUNBCH(*YES)
```

This command changes the PDM default for the existing PDM user USER2, so that program compiles and modules creation are done interactively, and objects run are done in batch.

Top

Error messages

***ESCAPE Messages**

PDM0004
Library &1 was not found.

PDM0010
Library name &1 is invalid.

CPF0001
Error found on &1 command.

CPF2204
User profile &1 not found.

CPF2209
Library &1 not found.

CPF2228
Not authorized to change user profile.

Top

Compare Physical File Member (CMPPFM)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

This command allows you to compare source physical file members.

Error messages for CMPPFM

None

Top

Parameters

Keyword	Description	Choices	Notes
NEWFILE	New file	*Qualified object name*	Required, Positional 1
	Qualifier 1: New file	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
NEWMBR	New member	Single values: ***FIRST**, *ALL Other values (up to 25 repetitions): *Character value*	Optional, Positional 2
OLDFILE	Old file	*Qualified object name*	Optional, Positional 3
	Qualifier 1: Old file	*Name*, ***NEWFILE**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
OLDMBR	Old member	Single values: ***NEWMBR**, *FIRST Other values (up to 25 repetitions): *Name*	Optional, Positional 4
CMPTYPE	Compare type	***LINE**, *FILE, *WORD	Optional, Positional 5
RPTTYPE	Report type	***DIFF**, *SUMMARY, *CHANGE, *DETAIL	Optional, Positional 6
OUTPUT	Output	*****, *PRINT, *OUTFILE	Optional, Positional 7
OUTFILE	File to receive output	*Qualified object name*	Optional, Positional 8
	Qualifier 1: File to receive output	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
OUTMBR	Output member options	*Element list*	Optional, Positional 9
	Element 1: Member to receive output	*Name*, ***FIRST**	
	Element 2: Replace or add records	***REPLACE**, *ADD	
SRCTYPE	Select source type	*Name*, ***ALL**	Optional, Positional 10

Keyword	Description	Choices	Notes
OPTION	Process option	Values (up to 10 repetitions): *IGNORECASE, *COUNT, *CBLSRCCOL, *RPGSRCCOL, *RPGLESRCCOL, *OMTDUP, *OMTREFMT, *OMTBASCMT, *OMTBLANK, *OMTCCMT, *OMTCBLCMT, *OMTCLCMT, *OMTCMDCMT, *OMTDDSCMT, *OMTPASCMT, *OMTPLICMT, *OMTRPGCMT, *OMTRPGLECMT, *FLGMOVLIN, *CHGFLGS, *CHANGES, *LONGLINES, *NARROW, *CMPSEQDAT, *COUNTREFMT, *WIDE	Optional, Positional 11
STMTFILE	Statement file	*Qualified object name*	Optional, Positional 12
	Qualifier 1: Statement file	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
STMTMBR	Statement member	*Name*, ***FIRST**	Optional, Positional 13

Top

New file (NEWFILE)

Identifies the new physical file containing the members to be compared.

***LIBL** Use the library list of the job.

***CURLIB**
Use the current library of the job.

library-name
Use the specified library.

new-file-name
Use the specified new file.

Top

New member (NEWMBR)

Identifies the members to be compared in the new file.

***FIRST**
Use the first member of the file.

new-file-member-name
Use the specified member or list of members.

To enter multiple values for this parameter, type a plus sign (+) in the + for more prompt, and press the Enter key.

generic*
Use the members that match the specified pattern.

***ALL** Use all members in the file.

Top

Old file (OLDFILE)

Identifies the old physical file containing the members to be compared.

***LIBL** Use the library list of the job.

***CURLIB**
Use the current library of the job.

library-name
Use the specified library.

***NEWFILE**
Use the old file with the same name as is specified for the NEWFILE keyword.

old-file-name
Use the specified old file.

Top

Old member (OLDMBR)

Identifies the members to be compared in the old file.

***NEWMBR**
Use the same member or member list as is specified for the NEWMBR keyword.

old-file-member-name
Use the specified member or list of members.

To enter multiple values for this parameter, type a plus sign (+) in the + for more prompt, and press the Enter key.

***FIRST**
Use the first member of the file.

Top

Compare type (CMPTYPE)

Specifies the type of comparison to be performed.

***LINE** Compare for differences on a line level, identifying inserted and deleted lines.

***FILE** Compare for differences on a file level, without reporting where the differences are. The results of this type of comparison indicate whether the members compared are different or the same, and provide the names of any nonpaired members. This method produces only summary information, but is the fastest type of comparison.

***WORD**
Compare for differences on a word level. This comparison is similar to the *LINE comparison, except words on adjacent lines can be matched. Words are delimited by blanks or the end of a line. The members are processed as long sequences of words without line boundaries or record lengths. The output of this comparison does not necessarily maintain the original spacing between words. Blanks may be added in the output listing so that you can clearly see the differences.

Top

Report type (RPTTYPE)

Specifies the listing type for the result report.

***DIFF** List only the differences between the members being compared, followed by a summary. The differences are flagged in the listing.

***SUMMARY**
List a summary of the results of the comparison, without showing the detailed differences. A group comparison generates an individual summary line for each member in the group, as well as the list of processing options.

***CHANGE**
Provide the same information as the *DIFF report type, with 10 lines before and after the differences. The extra lines allow you to see the differences within the context of the surrounding data.

***DETAIL**
List the entire new file member (and deletions from the old file member), indicate the differences, and provide a summary of the results.

Top

Output (OUTPUT)

Specifies whether the result of the comparison is displayed, printed, or stored to a physical file.

***** Display the result.

***PRINT**
Print the result to the spooled file.

***OUTFILE**
Store the result to a physical file.

Top

File to receive output (OUTFILE)

Specifies the file to which the output results are directed.

***LIBL** Use the library list of the job.

***CURLIB**
Use the current library of the job.

library-name
Use the specified library.

output-physical-file-name
Use the specified physical file.

Top

Output member options (OUTMBR)

Specifies the member to which the output is directed. Do not specify a value for this parameter if the OUTFILE keyword specifies a printer file.

***FIRST**
Use the first member of the file.

output-file-member-name
Use the specified member.

***REPLACE**
If the member exists, replace it.

***ADD** Add this member to the file.

Top

Select source type (SRCTYPE)

Specifies the source member type to be compared. One type or all types can be selected.

***ALL** Compare all source members.

source-member-type
Compare only the members with the specified source type.

Top

Process option (OPTION)

Specifies a list of process options to customize the comparison. You can specify up to 10 process options in the list.

To enter multiple values for this parameter, type a plus sign (+) in the + for more prompt, and press the Enter key.

Note: When you use the OMTxxx options to omit comments, the utility may recognize some non-comment strings as comments.

For example, if the string /*...*/ is imbedded within a program's executable code, the string may or may not be ignored during processing if you have used the *OMTCCMT process option.

***CBLSRCCOL**
Compare only COBOL source columns (7-72 inclusive). This option limits the scope of the comparison to the source code column area. This option is valid for line and word comparisons.

***CHANGES**
List only changed entries in the summary. Normally, all paired members are listed in a group comparison. When you specify *CHANGES in a group comparison, only the member pairs with changes are listed in the summary section. This option is valid for line, file, and word comparisons.

***CHGFLGS**
Generate listings denoting changes in the new file by placing a change flag (>) in column one of the appropriate line in the new file. Deleted lines are indicated by flagging the line following the deletion. This option is valid for line and word comparisons.

***CMPSEQDAT**
Set the origin of the source sequence to 1. Compares the sequence and date fields of the source physical file member. This option is valid for line and word comparisons.

***COUNT**
Count the lines from nonpaired members in a group (line) comparison, and include the results in the summary. If this option is not used, you only obtain statistics on lines from paired members.

***COUNTREFMT**
Reformatted lines are not flagged, but they are counted for the overall summary statistics. This option is valid for line and word comparisons.

***FLGMOVLIN**
Flag moved lines. Identify inserted lines in the new file that match deleted lines in the old file. This option is valid only for line comparisons.

***IGNORECASE**
Ignore differences due to case (upper, lower, mixed). This option is valid for line and word comparisons.

***LONGLINES**
Create a listing with 198 columns, reflecting up to 176 columns from the files. This option is valid for line comparisons.

***NARROW**
Create a 132 listing file with 55 columns for each side. Inserted and deleted lines are flagged and appear side-by-side in the listing output. This option is valid for line comparisons.

***OMTBASCMT**
Omit BASIC comments. BASIC comments are blanked out and excluded.

***OMTBLANK**
Omit lines in which the columns being compared are blank. This option is valid for line and word comparisons.

***OMTCBLCMT**
Omit COBOL comments. COBOL comments and blank lines are excluded from the compare set, to yield a listing with all comments removed or blanked out. This option is valid for line and word comparisons.

***OMTCCMT**
Omit C comments. C comments and blank lines are excluded from the compare set, to yield a listing with all comments removed or blanked out. This option is valid for line and word comparisons.

***OMTCLCMT**
Omit CL comments. CL program comments and blank lines are excluded from the compare set, to yield a listing with all comments removed or blanked out. This option is valid for line and word comparisons.

***OMTCMDCMT**
Omit CMD comments. CL command comments and blank lines are excluded from the compare set, to yield a listing with all comments removed or blanked out. This option is valid for line and word comparisons.

***OMTDDSCMT**
Omit DDS comments. DDS comments and blank lines are excluded from the compare set, to yield a listing with all comments removed or blanked out. This option is valid for line and word comparisons.

***OMTDUP**
Omit duplicate lines. Old file source lines that match new file source lines are omitted from the side-by-side listing. This option is valid for line comparisons.

***OMTPASCMT**
Omit Pascal comments. Pascal comments and blank lines are excluded from the compare set, to yield a listing with all comments removed or blanked out. This option is valid for line and word comparisons.

***OMTPLICMT**
Omit PLI comments. PLI comments and blank lines are excluded from the compare set, to yield a listing with all comments removed or blanked out. This option is valid for line and word comparisons.

***OMTREFMT**
Omit reformatted lines. Reformatted lines in the old file member are omitted from the listing. Reformatted lines in the new file member are included in the listing. Normally both are listed. This option is valid for line comparisons.

***OMTRPGCMT**
Omit RPG comments. RPG comments and blank lines are excluded from the compare set, to yield a listing with all comments removed or blanked out. This option is valid for line and word comparisons.

***OMTRPGLECMT**
Omit ILE RPG comments. ILE RPG comments and blank lines are excluded from the compare set, to yield a listing with all comments removed or blanked out. This option is valid for line and word comparisons.

***RPGLESRCCOL**
Compare only ILE RPG source columns (6-100 inclusive). This limits the comparison scope to the source code column area. This option is valid for line and word comparisons.

***RPGSRCCOL**
Compare only RPG source columns (6-74 inclusive). This limits the comparison scope to the source code column area. This option is valid for line and word comparisons.

***WIDE**
Create a wide 198 side-by-side (80 columns per side) listing. This option is valid for line and word comparisons.

Top

Statement file (STMTFILE)

Specifies a user-defined source physical file that holds process statements. Records in this file can be of any length, but only the first 80 bytes are read.

***LIBL** Use the library list of the job.

***CURLIB**
Use the current library of the job.

library-name
Use the specified library.

statement-file-name
Use the specified file as the statement file.

Top

Statement member (STMTMBR)

Specifies the file member containing the process statements to use in the comparison.

***FIRST**
Use the first member of the file.

statement-file-member-name
Use the specified member.

Top

Examples

None

Top

Error messages

None

Top

Copy DBCS Master Sort Table (CPYIGCSRT)

Parameters
Examples
Error messages

Where allowed to run:

- Batch job (*BATCH)
- Interactive job (*INTERACT)
- Interactive program (*IPGM)
- Batch REXX procedure (*BREXX)
- Interactive REXX procedure (*IREXX)
- Using QCMDEXEC, QCAEXEC, or QCAPCMD API (*EXEC)

Threadsafe: No

The CPYIGCSRT (Copy DBCS Master Sort Table) command copies the DBCS master sort table for Japanese DBCS characters either from a data file to the DBCS master sort table object or from the DBCS master sort table object to a data file. Use the CPYIGCSRT command to copy characters to or from the master file that is moving to or has moved from the System/36 system.

Error messages for CPYIGCSRT

None

Top

Parameters

Keyword	Description	Choices	Notes
OPTION	Copy direction	*OUT, *IN	Required, Positional 1
FILE	File	*Qualified object name*	Optional, Positional 2
	Qualifier 1: File	*Name*, **#KAMAST**	
	Qualifier 2: Library	*Name*, ***CURLIB**, *LIBL	
MBR	Member	*Name*, ***FIRST**, *FILE	Optional, Positional 3

Top

Copy direction (OPTION)

Specifies how the DBCS master sort table is copied.

The possible values are:

***IN** The DBCS master sort table is copied from a file.

***OUT** The DBCS master sort table is copied to a file.

Top

File (FILE)

Specifies the name of the file to use for copying.

The possible values are:

#KAMAST
This file is used if you do not specify another file name. #KAMAST is the name of the master file on the System/36 system.

file-name
The name of a physical file with a record length of 20 characters. If you are copying the sort table from this file, the file must already exist.

Top

Member (MEMBER)

Specifies the physical file member to use for copying.

The possible values are:

***FIRST**
The first member in the file is used.

***FILE** The file name identifies the file member to be used.

member-name
The name of a physical file member to use for copying.

Top

Examples

None

Top

Error messages

None

Top

Create Bound C Program (CRTBNDC)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Create Bound C Program (CRTBNDC) command starts the ILE C compiler. You can use this command in either batch or interactive mode, or from a CL program. The compiler attempts to create a program object based on the ILE C statements in the source code. The full compilation sequence is always run.

Note: When the CRTBNDC command is invoked, a temporary *MODULE object is created in the QTEMP library. The name of the temporary module object is the same as the name you specified on the PGM parameter of the CRTBNDC command. If a *MODULE object with that name already exists in QTEMP, an error message is generated and compilation stops. The temporary module object, which is used with the CRTBNDC command, is deleted after compilation stops whether or not compilation is successful.

Error messages for CRTBNDC

***ESCAPE Messages**

CZM1613
The compilation failed.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Required, Positional 1
	Qualifier 1: Program	*Name*	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QCSRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 3
SRCSTMF	Source stream file	*Path name*	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OUTPUT	Output options	Single values: ***NONE**, '*none' Other values: *Element list*	Optional
	Element 1: Output file name	*Path name*, *PRINT, '*print'	
	Element 2: Title	*Character value*, ***BLANK**	
	Element 3: Subtitle	*Character value*, ***BLANK**	

Keyword	Description	Choices	Notes
OPTION	Compiler options	Values (up to 35 repetitions): *NOAGR, *AGR, *DIGRAPH, *NODIGRAPH, *NOEVENTF, *EVENTF, *NOEXPMAC, *EXPMAC, *NOFULL, *FULL, *GEN, *NOGEN, *NOINCDIRFIRST, *INCDIRFIRST, *LOGMSG, *NOLOGMSG, *NOSECLVL, *SECLVL, *NOSHOWINC, *SHOWINC, *NOSHOWSKP, *SHOWSKP, *SHOWSRC, *NOSHOWSRC, *NOSHOWSYS, *SHOWSYS, *NOSHOWUSR, *SHOWUSR, *STDINC, *NOSTDINC, *NOSTDLOGMSG, *STDLOGMSG, *NOSTRUCREF, *STRUCREF, *NOSYSINCPATH, *SYSINCPATH, *NOXREF, *XREF, *NOXREFREF, *XREFREF	Optional
CHECKOUT	Checkout options	Values (up to 39 repetitions): ***NONE**, *USAGE, *ALL, *NOCOND, *COND, *NOCONST, *CONST, *NOEFFECT, *EFFECT, *NOENUM, *ENUM, *NOEXTERN, *EXTERN, *NOGENERAL, *GENERAL, *NOGOTO, *GOTO, *NOINIT, *INIT, *NOPARM, *PARM, *NOPORT, *PORT, *NOPPCHECK, *PPCHECK, *NOPPTRACE, *PPTRACE, *NOREACH, *REACH, *NOTRUNC, *TRUNC, *NOUNUSED, *UNUSED	Optional
OPTIMIZE	Optimization	**10**, 20, 30, 40	Optional
INLINE	Inline options	*Element list*	Optional
	Element 1: Inliner	***OFF**, *ON	
	Element 2: Mode	***NOAUTO**, *AUTO	
	Element 3: Threshold	1-65535, **250**, *NOLIMIT	
	Element 4: Limit	1-65535, **2000**, *NOLIMIT	
	Element 5: Report	***NO**, *YES	
DBGVIEW	Debugging view	***NONE**, *ALL, *STMT, *SOURCE, *LIST	Optional
DEFINE	Define names	Single values: ***NONE** Other values (up to 32 repetitions): *Character value*	Optional
LANGLVL	Language level	***EXTENDED**, *ANSI	Optional
ALIAS	Alias	Values (up to 3 repetitions): ***ANSI**, *NOANSI, *ADDRTAKEN, *NOADDRTAKEN, *ALLPTRS, *NOALLPTRS, *TYPEPTR, *NOTYPEPTR	Optional
SYSIFCOPT	System interface options	Values (up to 2 repetitions): *NOIFSIO, *IFSIO, *IFS64IO, *NOASYNCSIGNAL, *ASYNCSIGNAL	Optional
LOCALETYPE	Locale object type	***LOCALE**, *LOCALEUCS2, *LOCALEUTF, *CLD	Optional
FLAG	Message flagging level	**0**, 10, 20, 30	Optional
MSGLMT	Compiler messages	*Element list*	Optional
	Element 1: Message limit	0-32767, ***NOMAX**	
	Element 2: Message limit severity	0, 10, 20, **30**	
REPLACE	Replace program object	***YES**, *NO	Optional
USRPRF	User profile	***USER**, *OWNER	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
ENBPFRCOL	Enable performance collection	*Element list*	Optional
	Element 1: Collection level	***PEP**, *ENTRYEXIT, *FULL	
	Element 2: Procedures	*NONLEAF, *ALLPRC	
PFROPT	Performance options	Values (up to 2 repetitions): *SETFPCA, *NOSETFPCA, *NOSTRDONLY, *STRDONLY	Optional
PRFDTA	Profiling data	***NOCOL**, *COL	Optional

Keyword	Description	Choices	Notes
TERASPACE	Teraspace options	Single values: ***NO** Other values: *Element list*	Optional
	Element 1: Teraspace enabled	***YES**	
	Element 2: Use teraspace interfaces	***NOTSIFC**, *TSIFC	
STGMDL	Storage model	***SNGLVL**, *TERASPACE	Optional
DTAMDL	Data model	***P128**, *LLP64	Optional
PACKSTRUCT	Pack structure	***NATURAL**, 1, 2, 4, 8, 16	Optional
ENUM	Enum size	***SMALL**, 1, 2, 4, *INT	Optional
MAKEDEP	Dependency information	*Path name*, ***NONE**	Optional
INCDIR	Include directory	Single values: ***NONE** Other values (up to 32 repetitions): *Path name*	Optional
CSOPT	Compiler services option	*Character value*, ***NONE**	Optional
LICOPT	Licensed internal code options	*Character value*, ***NONE**	Optional
DFTCHAR	Default char type	***UNSIGNED**, *SIGNED	Optional
TGTCCSID	Target CCSID	1-65535, ***SOURCE**, *JOB, *HEX	Optional

Top

Program (PGM)

Specifies the program name and library for the ILE C program object being created.

program-name
Enter a name for the program object.

The possible library values are:

***CURLIB**
The program object is stored in the current library. If a job does not have a current library, the program object is created in the QGPL library.

library-name
Enter the name of the library where the program object being created will be stored.

Top

Source file (SRCFILE)

Specifies the source file name and library of the file containing the ILE C source code that you want to compile.

QCSRC
The source file named QCSRC contains the member with the ILE C source code that you want to compile.

source-file-name
Enter the name of the file that contains the member with the ILE C source code.

The possible library values are:

***LIBL** The library list is searched to find the library where the source file is located.

***CURLIB**
The current library is searched for the source file. If a job does not have a current library, QGPL is searched for the source file.

library-name
Enter the name of the library that contains the source file.

Top

Source member (SRCMBR)

Specifies the name of the member containing the source code to be compiled.

***PGM** The program name supplied on the PGM parameter is used as the source member name.

member-name
Enter the name of the member that contains the source code.

Top

Source stream file (SRCSTMF)

Specifies the path name of the stream file containing the source code that you want to compile.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'. If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by pre-pending the job's current working directory to the path name.

The SRCMBR and SRCFILE parameters cannot be specified with the SRCSTMF parameter.

Top

Text 'description' (TEXT)

Specifies the text that briefly describes the program object.

***SRCMBRTXT**
The text description associated with the source file member is used for the program object. If the source file is an inline file, a stream file, or a device file, the text will be blank.

***BLANK**
Specifies that no text appears.

'description'
Specify no more than 50 characters of text, enclosed in apostrophes.

Top

Output options (OUTPUT)

Specifies whether a compiler listing is produced.

Single Value

***NONE**
Does not generate the compiler listing. When a listing is not required, this parameter value should be used to improve compile-time performance. When *NONE is specified, any listing-related parameter values specified for the OPTION parameter are ignored.

Element 1: Output File Name

***PRINT**
Generates a spooled file containing the listing.

'path-name'
Specify the path name of a stream file to hold the listing.

Element 2: Title

***BLANK**
Specifies that no text appears.

'title' Specify a title string for the listing file (maximum 80 characters).

Element 3: Subtitle

***BLANK**
Specifies that no text appears.

'subtitle'
Specify a subtitle string for the listing file (maximum 80 characters).

Top

Compiler options (OPTION)

Specifies the options to use when the ILE C source code is compiled. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

The possible options are:

***NOAGR**
Does not generate aggregate maps in the listing.

***AGR** Generate maps of all aggregates in the listing. The maps include structures and unions. The structure maps show padding of members. This option overrides the *STRUCREF option.

***DIGRAPH**
Allow the use of digraphs in the source code.

***NODIGRAPH**
Do not allow the use of digraphs in the source code.

***NOEVENTF**
Do not create an event file for use by CoOperative Development Environment/400 (CODE/400).

***EVENTF**
Create an event file for use by CoOperative Development Environment/400 (CODE/400). The event file is created as a member in file EVFEVENT in the library where the module or program object being created will be stored. If the file EVFEVENT does not exist it is automatically created. The event file member name is the same as the name of the object being created. CODE/400 uses this file to provide error feedback integrated with the CODE/400 editor. An event file is normally created when you create a module or program object from within CODE/400.

***NOEXPMAC**
Macros are not expanded in the listing unless a syntax error is encountered within the macro.

***EXPMAC**
Expand all macros in the listing. This parameter conflicts with DBGVIEW(*ALL) and DBGVIEW(*LIST). Compilation will stop with an error message if OPTION(*EXPMAC) is used together with DBGVIEW(*ALL) or DBGVIEW(*LIST).

***NOFULL**
Do not turn on all listing options.

***FULL** Turn on all listing options.

***NOINCDIRFIRST**
Include directories specified as INCDIR parameters are not included before the standard header file include path.

***INCDIRFIRST**
Include directories specified as INCDIR parameters are included before the standard header file include path.

***LOGMSG**
Puts the compilation messages in the job log.

When you specify this option and the FLAG parameter, messages with the severity specified on the FLAG parameter (and higher) are placed in the job log.

When you specify this option and a maximum number of messages on the MSGLMT parameter, compilation stops when the number of messages, at the specified severity, have been placed in the job log.

***NOLOGMSG**
Does not put the compilation messages in the job log.

***NOSECLVL**
Does not generate second-level message text in the listing.

***SECLVL**
Generates second-level message text in the listing. An OUTPUT option must be specified for this option to take effect.

***NOSHOWINC**
Does not expand the user include files or the system include files in the source section of the listing or in the debug views.

***SHOWINC**
Expands both the user include files and the system include files in the source section of the listing or in the debug views. An OUTPUT option or DBGVIEW parameter value of *ALL, *SOURCE or *LIST must be specified.

***NOSHOWSKP**
Does not include the statements that the preprocessor has ignored in the source section of the listing or in the debug listing view. The preprocessor ignores statements as a result of a preprocessor directive evaluating to false (zero).

***SHOWSKP**
Includes all the statements in the source section of the listing or in the debug listing view, regardless of whether or not the preprocessor has skipped them. An OUTPUT option or a DBGVIEW parameter value of *ALL or *LIST must be specified.

***SHOWSRC**
Show the source code in the listing. This option can be modified by the *SHOWINC, *SHOWSYS or *SHOWUSR options.

***NOSHOWSRC**
Does not show the source code in the listing. This option may be modified by the *EXPMAC, *SHOWINC, *SHOWSYS or *SHOWUSR options.

***NOSHOWSYS**
Do not expand the system include files on the #include directive in the source section of the listing or in the debug views. System include files are enclosed in angle brackets (< >) following the #include directive.

***SHOWSYS**
Expands the system include files on the #include directive in the source section of the listing or in the debug views. An OUTPUT option or DBGVIEW parameter value of *ALL, *SOURCE or *LIST must be specified. System include files are enclosed in angle brackets (< >) following the #include directive.

***NOSHOWUSR**
Do not expand user include files in the listing or debug views. User include files are enclosed in double quotation marks (″ ″) following the #include directive.

***SHOWUSR**
Expands the user include files on the #include directive in the source section of the listing or in the debug views. An OUTPUT option or DBGVIEW parameter value of *ALL, *SOURCE or *LIST must be specified. User include files are enclosed in double quotation marks (″ ″) following the #include directive.

***STDINC**
The system supplied header files are included in the search path for the compile.

***NOSTDINC**
The system supplied header files are not included in the search path for the compile.

***NOSTDLOGMSG**
Compilation messages are not sent to the stdout stream.

***STDLOGMSG**
Compilation messages are sent to the stdout stream.

***NOSTRUCREF**
Do not generate maps of all referenced struct or union variables in the listing file.

***STRUCREF**
Generate maps of all referenced struct or union variables in the listing file.

***NOSYSINCPATH**
The search path for user includes is not affected.

***SYSINCPATH**
Changes the search path of user includes to the system include search path. In function this option is equivalent to changing the double-quotes in the user #include directive (#include ″file_name″) to angle brackets (#include <file_name>).

***NOXREF**
Does not generate the cross-reference table in the listing.

***XREF** Generates the cross-reference table containing a list of the identifiers in the source code together with the numbers of the lines in which they appear. An OUTPUT option must be specified.

***NOXREFREF**
Do not produce a cross-reference table of referenced identifiers in the listing.

***XREFREF**
Produce a cross-reference table of referenced variables, structures, and function names in the listing file. The table shows the line numbers where the identifiers are declared. An OUTPUT option must be specified.

Checkout options (CHECKOUT)

Specifies options you may select to generate informational messages that indicate possible programming errors. When you specify an option more than once, or when two options conflict, the last one specified is used.

Note: CHECKOUT may produce many messages. To prevent these messages from going to the job log specify OPTION(*NOLOGMSG *NOSTDLOGMSG) along with an OUTPUT parameter to place the checkout messages in the listing file.

***NONE**
Disables all of the options for CHECKOUT.

***USAGE**
This is equivalent to specifying *ENUM, *EXTERN, *INIT, *PARM, *PORT, *GENERAL and *TRUNC.

***ALL** Enables all of the options for CHECKOUT.

***NOCOND**
Do not warn about possible redundancies or problems in conditional expressions.

***COND**
Warn about possible redundancies or problems in conditional expressions.

***NOCONST**
Do not warn about operations involving constants.

***CONST**
Warn about operations involving constants.

***NOEFFECT**
Do not warn about statements with no effect.

***EFFECT**
Warn about statements with no effect.

***NOENUM**
Does not list the usage of enumerations.

***ENUM**
Lists the usage of enumerations.

***NOEXTERN**
Does not list the unused variables that have external declarations.

***EXTERN**
Lists the unused variables that have external declarations.

***NOGENERAL**
Does not list the general checkout messages.

***GENERAL**
Lists the general checkout messages.

***NOGOTO**
Does not list the occurrence and usage of goto statements.

***GOTO**
Lists the occurrence and usage of goto statements.

***NOINIT**
Does not list the automatic variables that are not explicitly initialized.

***INIT** Lists the automatic variables that are not explicitly initialized.

***NOPARM**
Does not list the function parameters that are not used.

***PARM**
Lists the function parameters that are not used.

***NOPORT**
Does not list nonportable usage of the C language.

***PORT**
Lists nonportable usage of the C language.

***NOPPCHECK**
Does not list the preprocessor directives.

***PPCHECK**
Lists all preprocessor directives.

***NOPPTRACE**
Does not list the tracing of include files by the preprocessor.

***PPTRACE**
Lists the tracing of include files by the preprocessor.

***NOREACH**
Do not warn about unreachable statements.

***REACH**
Warn about unreachable statements.

***NOTRUNC**
Do not warn about the possible truncation or loss of data.

***TRUNC**
Warn about the possible truncation or loss of data.

***NOUNUSED**
Do not check for unused auto or static variables.

***UNUSED**
Check for unused auto or static variables.

Top

Optimization (OPTIMIZE)

Specifies the levels of optimization of the generated object.

10 Generated code is not optimized. This level has the shortest compile time. This level allows variables to be displayed and modified while debugging.

20 Some optimization is performed on the code. This level allows user variables to be displayed but not modified while debugging.

30 Full optimization is performed on the generated code. During a debug session, user variables may not be modified but may be displayed. The presented values may not be the current value of the variable.

40 All optimizations done at level 30 are performed on the generated code. In addition, code is eliminated from procedure prologue and epilogue routines that enable instruction trace and call

trace system functions. Eliminating this code enables the creation of leaf procedures. A leaf procedure is a procedure that contains no calls to other procedures. Procedure call performance to a leaf procedure is significantly faster than to a normal procedure.

Top

Inline options (INLINE)

Specifies whether the compiler should consider replacing a function call with the called function's instructions. Inlining a function eliminates the overhead of a call and can result in better optimization. Small functions that are called many times are good candidates for inlining.

Element 1: Inliner

Specifies whether or not inlining will be used.

***OFF** Specifies that inlining will not be performed on the compilation unit.

***ON** Specifies that inlining will be performed on the compilation unit. If a debug view is specified, the inliner is turned off.

Element 2: Mode

Specifies whether or not the inliner should attempt to automatically inline functions depending on their Threshold and Limit values.

***NOAUTO**
Specifies that only the functions that have been specified with the #pragma inline directive should be considered candidates for inlining.

***AUTO**
Specifies that the inliner should determine if a function can be inlined based on the Threshold and Limit values specified. The #pragma noinline directive overrides *AUTO.

Element 3: Threshold

Specifies the maximum size of a function that can be a candidate for automatic inlining. The size is measured in Abstract Code Units (ACUs). ACUs are proportional in size to the executable code in the function. Source code is translated into ACUs by the compiler.

250 Specifies a threshold of 250.

number-of-ACUs
Specifies a threshold from 1 to 65535 ACUs.

***NOLIMIT**
Defines the threshold as the maximum size of the program object.

Element 4: Limit

Specifies the maximum relative size a function can grow before auto-inlining stops.

2000 Specifies a limit of 2000 ACUs.

***NOLIMIT**
Limit is defined as the maximum size of the program object. System limits may be encountered.

number-of-ACUs
A limit from 1 to 65535 ACUs may be specified.

Element 5: Report

Specifies whether or not to produce an inliner report with the compiler listing.

***NO** The inliner report is not produced.

***YES** The inliner report is produced as part of the compiler listing. An OUTPUT option must be specified to produce the inliner report.

Top

Debugging view (DBGVIEW)

Specifies which level of debugging is available for the module in the created program object. It also specifies which source views are available for source level debugging. Requesting a debug view will turn inlining off.

***NONE**
Debug capability is not inserted into the program object.

***ALL** Enables all of the debug options (*STMT, *SOURCE and *LIST)

***STMT**
Allows the program object to be debugged using program statement numbers and symbolic identifiers.

Note: To debug a module object using the *STMT option you need a listing.

***SOURCE**
Generates the source view for debugging the program object. OPTION parameter values *NOSHOWINC, *SHOWINC, *SHOWSYS, and *SHOWUSR determine the content of the source view created.

Note: In order to use this view for debugging, the root source file should not be modified, renamed or moved after the program object is created.

***LIST** Generates the listing view for debugging the program object. OPTION parameter values *SHOWINC, *SHOWUSR, *SHOWSYS, and *NOSHOWINC determine the content of the listing view created.

Top

Define names (DEFINE)

Specifies preprocessor macros that take effect before the file is processed by the compiler. Using the format DEFINE(macro) is equivalent to DEFINE('macro=1').

***NONE**
No macro is defined.

'name'* or *'name=value'
A maximum of 32 macros may be defined. Each macro name is enclosed in apostrophes. The maximum length of a macro name is 80 characters. The apostrophes are not part of the 80 character string. The apostrophes are required for case-sensitive macro names.

Note: Macros defined in the command override any macro definition of the same name in the source but a warning message is generated by the compiler. Function-like macros such as #define max(a,b) ((a)>(b):(a)?(b)) cannot be defined on the command line.

Top

Language level (LANGLVL)

Specifies the capabilities of the compiler and which prototypes are declared when the source is created.

***EXTENDED**
Defines the preprocessor variable __EXTENDED__ and undefines other language-level variables. This parameter should be used when all the capabilities of ILE C are required.

***ANSI**
Defines the preprocessor variables __ANSI__ and __STDC__ and undefines other language-level variables. Only ANSI-standard C is made available.

Note: The ILE C compiler always predefines the __ILEC400__ macro.

Top

Alias (ALIAS)

Specifies the aliasing assertion to be applied in the program object being created.

***ANSI**
The program object will only allow pointers to point to an object of the same type.

***NOANSI**
The program object will not use the *ANSI aliasing rules.

***ADDRTAKEN**
The program object will have its class of variables disjoint from pointers unless their address is taken.

***NOADDRTAKEN**
The program object will not use the *ADDRTAKEN aliasing rules.

***ALLPTRS**
The program object will not allow any two pointers to be aliased.

***NOALLPTRS**
The program object will not use the *ALLPTRS aliasing rules.

***TYPEPTR**
The program object will not allow any two pointers of different types to be aliased.

***NOTYPEPTR**
The program object will not use the *TYPEPTR aliasing rules.

Top

System interface options (SYSIFCOPT)

Specifies which system interface options will be used for the module object being created. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

***NOIFSIO**
The module object will use the iSeries Data Management file system for C stream I/O operations.

***IFSIO**
The module object will use the Integrated File System for C stream I/O operations.

***IFS64IO**
The module object will use the Integrated File System for 64-bit C stream I/O operations.

***NOASYNCSIGNAL**
Does not enable runtime mapping of synchronous signalling functions to asynchronous signalling functions.

***ASYNCSIGNAL**
Enable runtime mapping of synchronous signalling functions to asynchronous signalling functions. Specifying this option causes C runtime to map the synchronous signal() and raise() functions to the asynchronous sigaction() and kill() functions respectively.

Top

Locale object type (LOCALETYPE)

Specifies the type of locale support to be used by the program object being created.

***LOCALE**
Program objects created with this option use the locale support provided by *LOCALE objects.

***LOCALEUCS2**
Program objects created with this option use the locale support provided by *LOCALE objects. Wide-character types will contain two-byte universal character set values.

***CLD** Program objects created with this option use the locale support provided by *CLD objects.

***LOCALEUTF**
Program objects created with this option use the locale support provided by *LOCALE objects. Wide-character types will contain four-byte utf-32 values. Narrow character types will contain utf-8 values.

Top

Message flagging level (FLAG)

Specifies the level of messages that are to be displayed in the listing.

0 All messages starting at the informational level are displayed.

10 All messages starting at the warning level are displayed.

20 All messages starting at the error level are displayed.

30 All messages starting at the severe error level are displayed.

Top

Compiler messages (MSGLMT)

Specifies the maximum number of messages at the given message severity that can occur before the compilation is stopped.

Element 1: Message Limit

Specifies the maximum number of messages that can occur at, or above, the message severity level specified.

***NOMAX**
Compilation continues regardless of the number of messages that have occurred at the message severity level specified.

message-limit
Specify the number of messages that can occur. The valid range is 0 to 32767.

Element 2: Message Severity

Specifies the message severity that can stop the compilation if the *message-limit* number of messages at the specified severity or above occur.

30 Specifies that a *message-limit* of messages at severity 30 can occur before compilation stops.

0 Specifies that a *message-limit* of messages at severity 0 or above can occur before compilation stops.

10 Specifies that a *message-limit* of messages at severity 10 or above can occur before compilation stops.

20 Specifies that a *message-limit* of messages at severity 20 or above can occur before compilation stops.

Top

Replace program object (REPLACE)

Specifies whether the existing version of the object will be replaced by the current version.

***YES** The existing object is replaced by the new version. The old version is moved to the QRPLOBJ library and renamed based on the system date and time. The text description of the replaced object is changed to the name of the original object. The old object is deleted at the next IPL if it has not been explicitly deleted.

***NO** The existing object is not replaced. When an object with the same name is found in the specified library, a message is displayed and compilation stops.

Top

User profile (USRPRF)

Specifies the user profile that is used when the created program object is run, including the authority that the program object has for each object. The profile of either the program owner or the program user is used to control which objects can be used by the program object.

***USER**
The profile of the user running the program object is used.

***OWNER**
The user profiles of both the program's owner and the program's user are used when the program object is processed. The collective sets of object authority in both user profiles are used to find and access objects during program processing. Authority from the owning user profile's group profile is not included in the authority for the running program object.

Top

Authority (AUT)

Specifies the authority granted to users who do not have specific authority to the object, who are not on the authorization list, or whose group has no specific authority to the object.

***LIBCRTAUT**
Public authority for the object is taken from the CRTAUT keyword of the target library (the

library that contains the created object). This value is determined when the object is created. When the CRTAUT value for the library changes after the object is created, the new value does not affect any existing objects in the library.

***ALL** Provides authority for all operations on the object except those limited to the owner or controlled by authorization list management authority. Any user can control the object's existence, specify the security for it, change it, and perform basic functions on it, including transfer its ownership.

***CHANGE**
Provides all data authority and the authority to perform all operations on the object except those limited to the owner or controlled by object authority and object management authority. The object can be changed and basic functions can be performed on it.

***USE** Provides object operational authority; read authority; and authority for basic read-only operations on the object, such as binding of a module object. Users without specific authority are prevented from changing the object.

***EXCLUDE**
Users without special authority cannot access the object.

authorization-list-name
Enter the name of an authorization list of users and authorities to which the object is added. The object is secured by this authorization list, and the public authority for the object is set to *AUTL. The authorization list must exist on the system when the command is issued.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which the user intends to use the object being created.

In the examples given for the *CURRENT and *PRV values, and when specifying the release-level value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V4R5M0 is version 4, release 5, modification level 0.

***CURRENT**
The object will be used on the release of the operating system currently running on the user's system. For example, if V4R5M5 is running on the system, *CURRENT means the user intends to use the object on a system with V4R5M5 installed. The user can also use the object on a system with any subsequent release of the operating system installed.

Note: If V4R5M5 is running on the system, and the object will be used on a system with V4R5M0 installed, specify TGTRLS(V4R5M0) not TGTRLS(*CURRENT).

***PRV** The object will be used on the previous release with modification level 0 of the operating system. For example, if V4R5M5 is running on the user's system, *PRV means the user intends to use the object on a system with V4R4M0 installed. The user can also use the object on a system with any subsequent release of the operating system installed.

release-level
Specify the release in the format VxRxMx. The object can be used on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level. They change with each new release. If you specify a release level which is earlier than the earliest release level supported by this command, an error message is sent.

Top

Enable performance collection (ENBPFRCOL)

Specifies whether performance measurement code should be generated in the object. The data collected can be used by the system performance tool to profile an application's performance. Generating performance measurement code in a created object will result in slightly larger objects and may affect performance.

***PEP** Performance statistics are gathered on the entry and exit of the program entry procedure only. Choose this value when you want to gather overall performance information for an application.

***ENTRYEXIT *NONLEAF**
Performance statistics are gathered on the entry and exit of all procedures of the program object that are not leaf procedures. This includes the program PEP routine.

This choice would be useful if you only wanted to capture information on those routines that invoke other routines in your application.

***ENTRYEXIT *ALLPRC**
Performance statistics are gathered on the entry and exit of all the procedures of the program object (including those that are leaf procedures). This includes the program PEP routine.

This choice would be useful if you wanted to capture information on all routines. Use this option when you know that all the program objects called by your application were created with either the *PEP, *ENTRYEXIT or *FULL option. Otherwise, if your application calls other program objects that are not enabled for performance measurement, the performance tool will charge their use of resources against your application. This would make it difficult for you to determine where resources are actually being used.

***FULL *NONLEAF**
Performance statistics are gathered on entry and exit of all procedures that are not leaf procedures. Also, statistics are gathered before and after each call to an external procedure.

***FULL *ALLPRC**
Performance statistics are gathered on the entry and exit of all procedures including leaf procedures. Also statistics are gathered before and after each call to an external procedure.

Use this option when you think that your application will call other program objects that were not created with either *PEP, *ENTRYEXIT or *FULL. This option allows the performance tools to distinguish between resources that are used by your application and those used by program objects it calls (even if those program objects are not enabled for performance measurement). This option is the most expensive but allows for selectively analyzing various program objects in an application.

Top

Performance options (PFROPT)

Specifies various options available to boost performance. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

The possible options are:

***SETFPCA**
Causes the compiler to set the floating point computational attributes to achieve ANSI semantics for floating point computations.

***NOSETFPCA**
No computational attributes will be set. This option should only be used when the object being created does not have any floating point computations in it.

***NOSTRDONLY**
Specifies that the compiler must place strings into writeable memory.

***STRDONLY**
Specifies that the compiler may place strings into read-only memory.

Top

Profiling data (PRFDTA)

Specifies the program profiling data attribute for the program object. Program profiling is an advanced optimization technique used to reorder procedures and code within the procedures based on statistical data (profiling data).

***NOCOL**
The program object is not enabled to collect profiling data.

***COL** The program object is enabled to collect profiling data. *COL can be specified only when the optimization level is 30 or greater.

Top

Teraspace options (TERASPACE)

Specifies whether the program object is enabled to work with teraspace storage. This includes teraspace storage allocated by the program object and parameters passed from other teraspace-enabled program and service program objects.

Element 1: Teraspace Enabled

***NO** The program object is not enabled to handle addressing of storage allocated from teraspace.

***YES** The program object is enabled to handle addressing of storage allocated from teraspace, including parameters passed from other teraspace-enabled program and service program objects.

Element 2: Use Teraspace Interfaces

***NOTSIFC**
The program object will default to use the non-teraspace versions of the storage functions.

***TSIFC**
The program object will default to use the teraspace versions of the storage functions. The compiler will define macro variable __TERASPACE__.

Top

Storage model (STGMDL)

Specifies the type of storage to be used by the object created.

***SNGLVL**
The object created will use single-level storage.

***TERASPACE**
The object created will use teraspace storage.

Top

Data model (DTAMDL)

Specifies the sizes (in bytes) of variables declared as int, long, pointer.

***P128** Causes the sizes of int, long, pointer to be 4, 4, 16 respectively.

***LLP64**
Causes the sizes of int, long, pointer to be 4, 4, 8 respectively. The compiler will define the macro __LLP64_IFC__.

Top

Pack structure (PACKSTRUCT)

Specifies the alignment boundary to use for members of a structure.

***NATURAL**
Structure members are aligned on their natural boundaries. For example, a short integer will be two-byte aligned. 16-byte pointers will always align on 16-byte boundaries.

1 Pack structure members on a 1-byte alignment.

2 Pack structure members on a 2-byte alignment.

4 Pack structure members on a 4-byte alignment.

8 Pack structure members on a 8-byte alignment.

16 Pack structure members on a 16-byte alignment.

Top

Enum size (ENUM)

Specifies the number of bytes the compiler uses to represent an enumeration.

***SMALL**
Make all enum variables the smallest size that can represent the range of values.

1 Make all enum variables 1 byte.

2 Make all enum variables 2 bytes.

4 Make all enum variables 4 bytes.

***INT** Use the ANSI-standard enum size, which is 4 bytes.

Top

Dependency information (MAKEDEP)

Specifies whether or not to generate dependency information into a file. This information can be used by a make tool.

***NONE**
Do not generate dependency information.

'path-name'
Specify a path name for the stream file in which to store the dependency information.

Top

Include directory (INCDIR)

Specifies one or more directories to add to the search path used by the compiler to find include files. Use of INCDIR overrides the INCLUDE environment variable.

The search path can be further modified by using the following parameters on the OPTION keyword:

- *INCDIRFIRST or *NOINCDIRFIRST
- *SYSINCPATH or *NOSYSINCPATH
- *STDINC or *NOSTDINC

***NONE**
Unless modified, the default system include directory and the source directory will be searched for user include files.

'directory'
Specify up to 32 directories in which to search for include files. In addition to the specified directories, the source directory is searched for user include files.

Top

Compiler services option (CSOPT)

Specifies one or more compiler service options. This parameter allows IBM to provide switchable compiler capability between releases.

***NONE**
No compiler service option is selected.

'compiler-service-options-string'
The selected compiler service options are used when creating the module object. Valid strings will be described in PTF cover letters or in release notes.

Top

Licensed internal code options (LICOPT)

Specifies one or more Licensed Internal Code compile-time options. This parameter allows individual compile-time options to be selected, and is intended for the advanced programmer who understands the potential benefits and drawbacks of each selected type of compiler option.

The possible values are:

***NONE**
No compile-time options are selected.

'Licensed-Internal-Code-options-string'
The selected Licensed Internal Code compile-time options are used when creating the module object. Certain options may reduce your ability to debug the created module object.

Top

Default char type (DFTCHAR)

Specifies the default sign for the char data type.

***UNSIGNED**
Make default char type unsigned.

***SIGNED**
Make default char type signed.

Top

Target CCSID (TGTCCSID)

Specifies the target coded character set identifier used to describe data stored into the resulting program object.

The possible values are:

***SOURCE**
The root source file's CCSID is used.

***JOB** The current job's CCSID is used.

***HEX** The CCSID 65535 is used, which indicates that character data is treated as binary data and is not converted.

coded-character-set-identifier
Specify the CCSID to be used.

Top

Examples

None

Top

Error messages

***ESCAPE Messages**

CZM1613
The compilation failed.

Top

Create Bound COBOL Program (CRTBNDCBL)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The CRTBNDCBL command compiles an ILE COBOL source program into an runnable program in one step. You can use this command interactively, in batch mode, or in a CL program.

All object names specified for the CRTBNDCBL command must follow server naming conventions.

A description of the parameters for the CRTBNDCBL command follows.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Optional, Positional 1
	Qualifier 1: Program	*Name*, ***PGMID**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QCBLLESRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 3
SRCSTMF	Source stream file	*Path name*	Optional
GENLVL	Generation severity level	0-30, **30**	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OUTPUT	Output	***PRINT**, *NONE	Optional, Positional 4
OPTION	Compiler options	Values (up to 50 repetitions): *SOURCE, *NOSOURCE, *SRC, *NOSRC, *NOXREF, *XREF, *GEN, *NOGEN, *NOSEQUENCE, *SEQUENCE, *NOVBSUM, *VBSUM, *NONUMBER, *NUMBER, *LINENUMBER, *NOMAP, *MAP, *NOOPTIONS, *OPTIONS, *QUOTE, *APOST, *NOSECLVL, *SECLVL, *PRTCORR, *NOPRTCORR, *MONOPRC, *NOMONOPRC, *RANGE, *NORANGE, *NOUNREF, *UNREF, *NOSYNC, *SYNC, *NOCRTF, *CRTF, *NODUPKEYCHK, *DUPKEYCHK, *NOINZDLT, *INZDLT, *NOBLK, *BLK, *STDINZ, *NOSTDINZ, *STDINZHEX00, *NODDSFILLER, *DDSFILLER, *NOIMBEDERR, *IMBEDERR, *STDTRUNC, *NOSTDTRUNC, *NOCHGPOSSGN, *CHGPOSSGN, *NOEVENTF, *EVENTF, *MONOPIC, *NOMONOPIC, *NOCRTARKIDX, *CRTARKIDX	Optional, Positional 5
CVTOPT	Conversion options	Values (up to 8 repetitions): *NOVARCHAR, *VARCHAR, *NODATETIME, *DATETIME, *NOPICXGRAPHIC, *PICXGRAPHIC, *NOPICGGRAPHIC, *PICGGRAPHIC, *NOFLOAT, *FLOAT, *NODATE, *DATE, *NOTIME, *TIME, *NOTIMESTAMP, *TIMESTAMP, *NOCVTTODATE, *CVTTODATE, *NOPICNGRAPHIC, *PICNGRAPHIC	Optional

Keyword	Description	Choices	Notes
MSGLMT	Message limit	*Element list*	Optional
	Element 1: Number of messages	0-9999, ***NOMAX**	
	Element 2: Message limit severity	0-30, **30**	
DBGVIEW	Debug view option	*Element list*	Optional
	Element 1: Debug view	***STMT**, *SOURCE, *LIST, *ALL, *NONE	
	Element 2: Compress listing view	***NOCOMPRESSDBG**, *COMPRESSDBG	
OPTIMIZE	Optimize level	***NONE**, *BASIC, *FULL	Optional
FLAGSTD	FIPS flagging	Values (up to 2 repetitions): *NOFIPS, *MINIMUM, *INTERMEDIATE, *HIGH, *NOOBSOLETE, *OBSOLETE	Optional
EXTDSPOPT	Extended display options	Values (up to 3 repetitions): *DFRWRT, *NODFRWRT, *UNDSPCHR, *NOUNDSPCHR, *ACCUPDALL, *ACCUPDNE	Optional
FLAG	Flagging severity	0-99, **0**	Optional
REPLACE	Replace program	***YES**, *NO	Optional
USRPRF	User profile	***USER**, *OWNER	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
LINKLIT	Link literal	***PGM**, *PRC	Optional
SIMPLEPGM	Simple program	***YES**, *NO	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
SRTSEQ	Sort sequence	Single values: ***HEX**, *JOB, *JOBRUN, *LANGIDUNQ, *LANGIDSHR Other values: *Qualified object name*	Optional
	Qualifier 1: Sort sequence	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
LANGID	Language id	*Character value*, ***JOBRUN**, *JOB	Optional
ENBPFRCOL	Enable performance collection	Single values: ***PEP** Other values: *Element list*	Optional
	Element 1: Collection level	***FULL**, *ENTRYEXIT	
BNDDIR	Binding directory	Single values: ***NONE** Other values (up to 50 repetitions): *Qualified object name*	Optional
	Qualifier 1: Binding directory	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB, *USRLIBL	
ACTGRP	Activation group	*Name*, **QILE**, *NEW, *CALLER	Optional
PRFDTA	Profiling data	***NOCOL**, *COL	Optional
CCSID	Coded character set ID	*Integer*, ***JOBRUN**, *HEX, *JOB	Optional
ARITHMETIC	Arithmetic mode	***NOEXTEND**, *EXTEND31, *EXTEND63	Optional
NTLPADCHAR	Padding character	*Element list*	Optional
	Element 1: Single byte to national	*Character value*, ***DEFAULT**	
	Element 2: Double byte to national	*Character value*, ***DEFAULT**	
	Element 3: National to national	*Character value*, ***DEFAULT**	
LICOPT	Licensed Internal Code options	*Character value*	Optional
INCDIR	Include directory	Values (up to 32 repetitions): *Path name*, ***NONE**	Optional

Keyword	Description	Choices	Notes
PGMINFO	Generate program interface	***NO**, *PCML	Optional
INFOSTMF	Program interface stream file	*Path name*	Optional

Top

Program (PGM)

Specifies the program name and library name for the program object you are creating. The program name and library name must conform to server naming conventions. The possible values are:

***PGMID**
The name for the compiled program is taken from the PROGRAM-ID paragraph in the ILE COBOL source program. When SIMPLEPGM(*NO) is specified, the name of the compiled program object is taken from the PROGRAM-ID paragraph of the first ILE COBOL source program in the sequence of source programs (multiple compilation units in a single source file member).

program-name
Enter a name to identify the compiled ILE COBOL program. If you specify a program name for this parameter, and compile a sequence of source programs and if SIMPLEPGM(*YES) is specified, the first program in the sequence uses this name; any other programs use the name specified in the PROGRAM-ID paragraph in the corresponding ILE COBOL source program.

The possible library values are:

***CURLIB**
The created program object is stored in the current library. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the created program object is to be stored.

Top

Source file (SRCFILE)

Specifies the name of the source file and library that contains the ILE COBOL source code to be compiled. This source file should have a record length of 92. The possible values are:

QCBLLESRC
Specifies that the source file, QCBLLESRC, contains the ILE COBOL source to be compiled.

source-file-name
Enter the name of the source file that contains the ILE COBOL source to be compiled.

The possible library values are:

***LIBL** The library list is searched to find the library where the source file is located.

***CURLIB**
The current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the source file is located.

Top

Source member (SRCMBR)

Specifies the name of the member that contains the ILE COBOL source code to be compiled. You can specify this parameter only if the source file referred to in the SRCFILE parameter is a database file. The possible values are:

***PGM** The source file member with the same name as the program name supplied on the PGM parameter is used.

If you do not specify a program name for the PGM parameter, the compiler uses the first source member of the database source file.

source-file-member-name
Enter the name of the member that contains the ILE COBOL source code.

Top

Source stream file (SRCSTMF)

Specifies the path name of the stream file containing the ILE COBOL source code to be compiled.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'.

If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by appending the job's current working directory to the path name.

The SRCMBR and SRCFILE parameters cannot be specified with the SRCSTMF parameter.

Top

Generation severity level (GENLVL)

Specifies the severity level that determines if a module object is created. The severity level corresponds to the severity level of the messages produced during compilation. This parameter applies individually to each compilation unit in a source file member. Other compilation units in the source file member will still be compiled even if a previous compilation unit fails.

The possible values are:

30 No program object is created if errors occur with a severity level equal to or greater than 30.

severity-level
Specify a one or two-digit number, 0 through 30, which is the severity level you want to use to determine if a program object is created. No program object is created if errors occur with a severity level equal to or greater than this severity level.

Top

Text 'description' (TEXT)

Allows you to enter text that briefly describes the program and its function.

***SRCMBRTXT**
The same text that describes the database file member containing the ILE COBOL source code, is used to describe the program object. If the source comes from a device or inline file, specifying *SRCMBRTXT has the same effect as specifying *BLANK.

***BLANK**
No text is specified.

text-description
Enter text briefly describing the program and its function. The text can be a maximum of 50 SBCS characters in length and must be enclosed in single quotation marks. The single quotation marks are not part of the 50-character string.

Top

Output (OUTPUT)

Specifies if the compiler listing is generated or not. The possible values are:

***PRINT**
A compiler listing is generated. If a member is being compiled, the output file has the same name as the member. If a stream file is being compiled and *PGMID is specified in the PGM parameter, the output file has the name COBOLPGM00. Otherwise, it has the same name as the program.

***NONE**
No compiler listing is generated.

Top

Compiler options (OPTION)

Specifies the options to use when the ILE COBOL source code is compiled.

Options specified in the PROCESS statement of an ILE COBOL source program override the corresponding options of the OPTION parameter.

The possible values of the OPTION parameter are:

***SOURCE or *SRC**
The compiler produces a source listing, consisting of the ILE COBOL source program and all compilation-time error messages.

***NOSOURCE or *NOSRC**
The compiler does not produce the source part of the listing. If you do not require a source listing, you should use this option because compilation may take less time.

***NOXREF**
The compiler does not produce a cross-reference listing for the ILE COBOL source program.

***XREF** The compiler produces a cross-reference listing for the ILE COBOL source program.

***GEN** The compiler creates a program object after the ILE COBOL source is compiled.

***NOGEN**
The compiler does not create a program object after the ILE COBOL source program is compiled. You might specify this option if you want only error messages or listings.

***NOSEQUENCE**
The reference numbers are not checked for sequence errors.

***SEQUENCE**
The reference numbers are checked for sequence errors. Sequence errors do not occur if the *LINENUMBER option is specified.

***NOVBSUM**
Verb usage counts are not printed.

***VBSUM**
Verb usage counts are printed.

***NONUMBER**
The source-file sequence numbers are used for reference numbers.

***NUMBER**
The user-supplied sequence numbers (columns 1 through 6) are used for reference numbers.

***LINENUMBER**
The sequence numbers created by the compiler are used for reference numbers. This option combines ILE COBOL program source code and source code introduced by COPY statements into one consecutively numbered sequence. Use this option if you specify FIPS (Federal Information Processing Standards) flagging or SAA flagging.

***NOMAP**
The compiler does not list the Data Division map.

***MAP** The compiler lists the Data Division map.

***NOOPTIONS**
Options in effect are not listed for this compilation.

***OPTIONS**
Options in effect are listed for this compilation.

***QUOTE**
Specifies that the delimiter quotation mark (") is used for nonnumeric literals, hexadecimal literals, and Boolean literals. This also specifies that the value of the figurative constant QUOTE has the EBCDIC value of a quotation mark.

***APOST**
Specifies that the delimiter apostrophe (') is used for nonnumeric literals, hexadecimal literals, and Boolean literals. This option also specifies that the value of the figurative constant QUOTE has the EBCDIC value of an apostrophe.

***NOSECLVL**
Second level message text is not listed for this compilation.

***SECLVL**
Second level message text is listed for this compilation, along with the first-level error text, in the message section of the compiler listing.

***PRTCORR**
Comment lines are inserted in the compiler listing indicating which elementary items were included as a result of the use of the CORRESPONDING phrase.

***NOPRTCORR**
Comment lines are not inserted in the compiler listing when the CORRESPONDING phrase is used.

***MONOPRC**
The program-name (literal or word) found in the PROGRAM-ID paragraph, the CALL, CANCEL, or SET ENTRY statements, and the END PROGRAM header is converted to all upper-case characters (monocasing) and the rules for program-name formation are enforced.

***NOMONOPRC**
The program-name (literal or word) found in the PROGRAM-ID paragraph, the CALL, CANCEL, or SET ENTRY statements, and the END PROGRAM header is not converted to all upper-case characters (no monocasing) and the rules for program-name formation are not enforced. This option allows special characters not allowed for standard COBOL to be used in the CALL target.

***RANGE**

At run time, subscripts are verified to ensure they are within the correct ranges, but index ranges are not verified. Reference modification and compiler-generated substring operations are also checked.

The contents of date-time items are checked to make sure their format is correct, and that they represent a valid date, time, or timestamp.

***NORANGE**

Ranges are not verified at run time.

Note: The *RANGE option generates code for checking subscript ranges. For example, it ensures that you are not attempting to access element 21 of a 20-element array.

The *NORANGE option does not generate code to check subscript ranges. As a result, the *NORANGE option produces faster running code.

***NOUNREF**

Unreferenced data items are not included in the compiled program. This reduces the amount storage used, allowing a larger program to be compiled. You cannot look at or assign to an unreferenced data item during debugging when the *NOUNREF option is chosen. The unreferenced data items still appear in the cross-reference listings produced by specifying OPTION (*XREF).

***UNREF**

Unreferenced data items are included in the compiled program.

***NOSYNC**

The SYNCHRONIZED clause is syntax checked only.

***SYNC**

The SYNCHRONIZED clause is compiled by the compiler. The SYNCHRONIZED clause causes the position of a data item to be aligned such that the right-hand (least-significant) end is on the natural storage boundary. The natural storage boundary is the next nearest 4-byte, 8-byte, or 16-byte boundary in storage depending on the length and type of data being stored. Extra storage is reserved adjacent to the synchronized item to achieve this alignment. Each elementary data item that is described as SYNCHRONIZED is aligned to the natural storage boundary that corresponds to its data storage assignment.

***NOCRTF**

Disk files that are unavailable at the time of an OPEN operation are not created dynamically.

***CRTF**

Disk files that are unavailable at the time of an OPEN operation are created dynamically.

Note: The maximum record length for a file that will be created dynamically is 32&rbl.766. Indexed files will not be dynamically created even though the *CRTF option has been specified.

***NODUPKEYCHK**

Does not check for duplicate primary and alternate record keys for INDEXED files.

***DUPKEYCHK**

Checks for duplicate primary and alternate record keys for INDEXED files.

***NOINZDLT**

Relative files with sequential access are not initialized with deleted records during the CLOSE operation if the files have been opened for OUTPUT. The record boundary is determined by the number of records written at OPEN OUTPUT time. Subsequent OPEN operations allow access only up to the record boundary.

***INZDLT**

Relative files with sequential access are initialized with deleted records during the CLOSE

operation if the files were opened for OUTPUT. Active records in the files are not affected. The record boundary is defined as the file size for subsequent OPEN operations.

***NOBLK**
The compiler allows blocking only of SEQUENTIAL access files with no START statement. The BLOCK CONTAINS clause, if specified, is ignored, except for tape files.

***BLK** When *BLK is used and a BLOCK CONTAINS clause is specified, the compiler allows blocking for DYNAMIC access files and SEQUENTIAL access files with a START statement. Blocking is not allowed for RELATIVE files opened for output operations. The BLOCK CONTAINS clause controls the number of records to be blocked.

When *BLK is used and no BLOCK CONTAINS clause is specified, the compiler allows blocking only of SEQUENTIAL access files with no START statement. The operating system determines the number of records to be blocked.

***STDINZ**
For those items with no VALUE clause, the compiler initializes data items to system defaults.

***NOSTDINZ**
For those items with no VALUE clause, the compiler does not initialize data items to system defaults.

***STDINZHEX00**
For those items with no VALUE clause, the compiler initializes data items to hexadecimal zero.

***NODDSFILLER**
If no matching fields are found by a COPY DDS statement, no field descriptions are generated.

***DDSFILLER**
If no matching fields are found by a COPY DDS statement, a single character FILLER field description, ″07 FILLER PIC X″, is always created.

***NOIMBEDERR**
Error messages are not included in the source listing section of the compiler listing. Error messages only appear in the error message section of the compiler listing.

***IMBEDERR**
First level error messages are included in the source listing section of the compiler listing, immediately following the line where the error occurred. Error messages also appear in the error message section of the compiler listing.

***STDTRUNC**
This option applies only to USAGE BINARY data. When *STDTRUNC is selected, USAGE BINARY data is truncated to the number of digits in the PICTURE clause of the BINARY receiving field.

***NOSTDTRUNC**
This option applies only to USAGE BINARY data. When *NOSTDTRUNC is selected, BINARY receiving fields are truncated only at half-word, full-word, or double-word boundaries. BINARY sending fields are also handled as half-words, full-words, or double-words. Thus, the full binary content of the field is significant. Also, the DISPLAY statement will convert the entire content of a BINARY field, with no truncation.

***NOCHGPOSSGN**
Hexadecimal F is used as the default positive sign for zoned and packed numeric data. Hexadecimal F is the system default for the operating system.

***CHGPOSSGN**
Hexadecimal C is used as the default positive sign for zoned and packed numeric data. This applies to all results of the MOVE, ADD, SUBTRACT, MULTIPLY, DIVIDE, COMPUTE, and INITIALIZE statements, as well as the results of the VALUE clause.

***NOEVENTF**
Do not create an Event File for use by CoOperative Development Environment/400 (CODE/400). CODE/400 uses this file to provide error feedback integrated with the CODE/400 editor. An Event File is normally created when you create a module or program from within CODE/400.

***EVENTF**
Create an Event File for use by CoOperative Development Environment/400 (CODE/400). The Event File is created as a member in file EVFEVENT in the library where the created module or program object is to be stored. If the file EVFEVENT does not exist it is automatically created. The Event File member name is the same as the name of the object being created.

CODE/400 uses this file to provide error feedback integrated with the CODE/400 editor. An Event File is normally created when you create a module or program from within CODE/400.

***MONOPIC**
The PICTURE character-string is converted to all uppercase characters (monocasing).

***NOMONOPIC**
The currency symbol used in the PICTURE character-string is case sensitive. That is, the lowercase letters corresponding to the uppercase letters for the PICTURE symbols A, B, E, G, N, P, S, V, X, Z, CR, and DB are equivalent to their uppercase representations in a PICTURE character-string. All other lowercase letters are not equivalent to their corresponding uppercase representations.

***NOCRTARKIDX**
Temporary alternate record key (ARK) indexes are not created if permanent ones cannot be found.

***CRTARKIDX**
Temporary alternate record key (ARK) indexes are created if permanent ones cannot be found.

Top

Conversion options (CVTOPT)

Specifies how the compiler handles date, time, and timestamp field types, DBCS field types, variable-length character field types, and floating-point field types passed from externally-described files to your program through COPY DDS. The possible values are:

***NOVARCHAR**
Variable-length fields are declared as FILLER fields.

***VARCHAR**
Variable-length fields are declared as group items, and are accessible to the ILE COBOL source program.

***NODATETIME**
Date, time, and timestamp data types are declared as FILLER fields.

***DATETIME**
Date, time, and timestamp DDS data types are given COBOL data item names based on their DDS names. The category of the COBOL data item is alphanumeric, unless one of the CVTOPT parameter values *DATE, *TIME, or *TIMESTAMP is specified. In this case, the category of the COBOL data item is date, time, or timestamp, respectively.

***NOPICXGRAPHIC**
DBCS-graphic data types are declared as FILLER fields.

***PICXGRAPHIC**
Fixed-length DBCS-graphic data types are declared as fixed-length alphanumeric fields, and are accessible to the ILE COBOL source program.

When the *VARCHAR option is also in use, variable-length DBCS-graphic data types are declared as fixed-length group items, and are accessible to the ILE COBOL source program.

***PICGGRAPHIC**

Fixed-length DBCS-graphic data types are declared as fixed-length G-type fields, and are accessible to the ILE COBOL source program.

When the *VARCHAR option is also in use, variable-length DBCS-graphic data types are declared as fixed-length group items (made of a numeric field followed by G type field), and are accessible to the ILE COBOL source program.

***NOPICGGRAPHIC**

DBCS-graphic data types are declared as FILLER fields.

***NOFLOAT**

Floating-point data types are declared as FILLER fields with a USAGE of binary.

***FLOAT**

Floating-point data types are brought into the program with their DDS names and a USAGE of COMP-1 (single-precision) or COMP-2 (double-precision). The fields are made accessible to the ILE COBOL source program.

***NODATE**

DDS date data types are declared as category alphanumeric COBOL date items, for example:

```
06 FILLER PIC X(10).
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***DATE**

DDS date data types are declared as category date COBOL data items, for example:

```
06 FILLER FORMAT DATE '@Y-%m-%d'.
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***NOTIME**

DDS time data types are declared as category alphanumeric COBOL data items, for example:

```
06 FILLER PIC X(10).
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***TIME**

DDS time data types are declared as category time COBOL data items, for example:

```
06 FILLER FORMAT TIME '%H:%M:%S'.
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***NOTIMESTAMP**

DDS timestamp data types are declared as category alphanumeric COBOL data items, for example:

```
06 FILLER PIC X(10).
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***TIMESTAMP**

DDS timestamp data types are declared as category timestamp COBOL data items, for example:

```
06 FILLER FORMAT TIMESTAMP '%H:%M:%S'.
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***NOCVTTODATE**
DDS data types with the DATFMT keyword (excluding the DDS date data type) are declared in ILE COBOL based on their original DDS type.

***CVTTODATE**
DDS data types with the DATFMT keyword (excluding the DDS date data type) are declared in ILE COBOL as date data types.

***NOPICNGRAPHIC**
DBCS-graphic data types are declared as FILLER fields.

***PICNGRAPHIC**
Fixed-length DBCS-graphic data types are declared as fixed-length national data fields, and are accessible to the ILE COBOL source program.

Top

Message limit (MSGLMT)

Specifies the maximum number of messages of a given error severity level that can occur for each compilation unit before compilation stops. As soon as one compilation unit reaches the maximum, compilation stops for the entire source member.

For example, if you specify 3 for the maximum number of messages and 20 for the error severity level then compilation will stop if three or more errors with a severity level of 20 or higher occur. If no messages equal or exceed the given error severity level, compilation continues regardless of the number of errors encountered.

number-of-messages
Specifies the maximum number of messages. The possible values are:

***NOMAX**
Compilation continues until normal completion regardless of the number of errors encountered.

maximum-number
Specifies the maximum number of messages that can occur at or above the specified error severity level before compilation stops. The valid range is 0-9999.

message-limit-severity
Specifies the error severity level used to determine whether or not to stop compilation. The possible values are:

30 Compilation stops if the number of errors with severity level 30 or higher exceeds the maximum number of messages specified.

error-severity-level
Enter a one or two-digit number, 0 through 30, which is the error severity level you want to use to determine whether or not to stop compilation. Compilation stops if the number of errors with this severity level or higher exceeds the maximum number of messages you specified.

Top

Debug view option (DBGVIEW)

Specifies options that control which views of the source program or generated listing is available for debugging the compiled program, and if the debug listing view is compressed or not.

debug-view

Specify the views to be available for debugging. The possible values are:

***STMT**

The compiled program can be debugged using symbolic names and statement numbers.

***SOURCE**

The primary source member as well as copied source members which were included through COPY statements will have source views available for debugging the compiled program. These views are available only if the primary source member and copied source members come from local database source files. Do not change or delete members during the time between compile and debug.

***LIST** A listing view, which shows the source code after the processing of any COPY and REPLACE statements, will be made available for debugging the compiled module. This option increases the size of the compiled module, without affecting the runtime performance of the compiled module.

The listing view will include the cross-reference listing, Data Division map, and verb usage counts when the corresponding compiler options are requested. For example, a cross-reference listing will be included if OPTION(*XREF) is specified.

Listing views can be generated regardless where the primary source members or copied source members come from. Listing views are not affected by changes to or deletion of the source members following the compilation.

***ALL** Equivalent to specifying *STMT, *SOURCE, and *LIST combined.

***NONE**

The compiled program cannot be debugged. This reduces the size of the compiled program, but does not affect its runtime performance. When this option is specified, a formatted dump can not be taken.

compress-listing-view

Specifies if the listing view is compressed or not when *LIST or *ALL is specified in debug-view. The possible values are:

***NOCOMPRESSDBG**

The listing view is not compressed.

***COMPRESSDBG**

The listing view is compressed when *LIST or *ALL is specified in debug-view.

Top

Optimize level (OPTIMIZE)

Specifies the level of optimization of the program. The possible values are:

***NONE**
No optimization is performed on the compiled program. Compilation time is minimized when this option is used. This option allows variables to be displayed and changed during debugging.

***BASIC**
Some optimization (only at the local block level) is performed on the compiled program. This option allows user variables to be displayed but not changed during debugging.

***FULL** Full optimization (at the global level) is performed on the compiled program. This optimization increases compilation time but also generates the most efficient code. This option allows user variables to be displayed but not changed during debugging. The displayed values of the variables may not be their current values. Some variables may not be displayable.

Note: Regardless of the optimization level chosen, all information to allow full optimization is generated. The user can change optimization levels of the program object from *NONE to *FULL using the CHGMOD command without having to recompile the source program.

Top

FIPS flagging (FLAGSTD)

Specifies the options for FIPS flagging. (Select the *LINENUMBER option to ensure that the reference numbers used in the FIPS messages are unique.) The possible values are:

***NOFIPS**
The ILE COBOL source program is not FIPS flagged.

***MINIMUM**
FIPS flag for minimum subset and higher.

***INTERMEDIATE**
FIPS flag for intermediate subset and higher.

***HIGH**
FIPS flag for high subset.

***NOOBSOLETE**
Obsolete language elements are not flagged.

***OBSOLETE**
Obsolete language elements are flagged.

Top

Extended display options (EXTDSPOPT)

Specifies the options to use for extended ACCEPT and extended DISPLAY statements for workstation I/O. The possible values are:

***DFRWRT**
Extended DISPLAY statements are held in a buffer until an extended ACCEPT statement is encountered, or until the buffer is filled.

The contents of the buffer are written to the display when the extended ACCEPT statement is encountered or the buffer is full.

***NODFRWRT**
Each extended DISPLAY statement is performed as it is encountered.

***UNDSPCHR**
Displayable and undisplayable characters are handled by extended ACCEPT and extended DISPLAY statements.

***NOUNDSPCHR**
Only displayable characters are handled by extended ACCEPT and extended DISPLAY statements.

Although you must use this option for display stations attached to remote 3174 and 3274 controllers, you can also use it for local workstations. If you do use this option, your data must contain displayable characters only. If the data contains values less than hexadecimal 20, the results are not predictable, ranging from unexpected display formats to severe errors.

***ACCUPDALL**
All types of data are predisplayed in the extended ACCEPT statements regardless of the existence of the UPDATE phrase.

***ACCUPDNE**
Only numeric edited data are predisplayed in the extended ACCEPT statements that do not contain the UPDATE phrase.

Top

Flagging severity (FLAG)

Specifies the minimum severity level of messages that will appear in the compiler listing. The possible values are:

0 All messages will appear in the compiler listing.

severity-level
Enter a one or two-digit number that specifies the minimum severity level of messages that you want to appear in the compiler listing. Messages that have severity levels of this specified value or higher will appear in the compiler listing.

Top

Replace program (REPLACE)

Specifies if a new program is created when a program of the same name in the specified or implied library already exists. The intermediate module objects created during the processing of the CRTBNDCBL command are not subject to the REPLACE specifications, and have an implied REPLACE(*NO) against the QTEMP library. The intermediate module objects are deleted once the CRTBNDCBL command has completed processing. The possible values of the REPLACE parameter are:

***YES** A new program is created and it replaces any existing program of the same name in the specified or implied library. The existing program of the same name in the specified or implied library is moved to library QRPLOBJ.

***NO** A new program is not created if a program of the same name already exists in the specified library. The existing program is not replaced, a message is displayed, and compilation stops.

Top

User profile (USRPRF)

Specifies the user profile that will run the created program object. The profile of the program owner or the program user is used to run the program and control which objects can be used by the program (including the authority the program has for each object). This parameter is not updated if the program already exists. To change the value of USRPRF, delete the program and recompile using the correct value (or, if the constituent *MODULE object(s) exist, you may choose to invoke the CRTPGM command).

The possible values are:

***USER**
The user profile of the program user is to be used when the program is run.

***OWNER**
The user profiles of both the owner and user of the program are to be used when the program is run. The collective sets of object authorities in both owner and user profiles are to be used to find and access objects during the running of the program object. Any objects that are created when the program is run are owned by the user of the program.

Top

Authority (AUT)

Specifies the authority given to users who do not have specific authority to the program object, who are not on the authorization list, or whose group has no specific authority to the program object. You can change the authority for all users, or for specific users after the program object is created by using the GRTOBJAUT (Grant Object Authority) or RVKOBJAUT (Revoke Object Authority) commands.

The possible values are:

***LIBCRTAUT**
The public authority for the object is taken from the CRTAUT keyword of the target library (the library that is to contain the created program object). This value is determined when the program object is created. If the CRTAUT value for the library changes after the program object is created, the new value does NOT affect any existing objects.

***ALL** Provides authority for all operations on the program object except those limited to the owner or controlled by authorization list management authority. The user can control the program object's existence, specify security for it, change it, and perform basic functions on it, but cannot transfer its ownership.

***CHANGE**
Provides all data authority and the authority for performing all operations on the program object except those limited to the owner or controlled by object authority and object management authority. The user can change the object and perform basic functions on it.

***USE** Provides object operational authority and read authority; authority for basic operations on the program object. The user can perform basic operations on the object but is prevented from changing the object.

Note: A user must have *USE authority to a program to obtain a formatted dump of the variables of the program. To dump variables, the program must also have observable information.

If you do not want some users to be able to dump the variables, then give them only *OBJOPR plus *EXECUTE authority to the program. This will allow them to call the program but not dump its variables. Use EDTOBJAUT, GRTOBJAUT or RVKOBJAUT to change the authority of the created program.

If you do not want any users to be able to dump the variables, then use Change Program (CHGPGM) to remove the program's observable information.

***EXCLUDE**
The user cannot access the program object.

authorization-list-name
The name of an authorization list of users and authorities to which the program is added. The program object is secured by this authorization list, and the public authority for the program object is set to *AUTL. The authorization list must exist on the system when the CRTBNDCBL command is issued. Use the Create Authorization List (CRTAUTL) command to create your own authorization list.

Top

Link literal (LINKLIT)

Specifies the linkage type for external CALL/CANCEL 'literal' target and the SET ENTRY target. You may override this option for specific external CALL/CANCEL 'literal' target and the SET ENTRY target lists by specifying the following sentence in the SPECIAL-NAMES paragraph:

```
LINKAGE TYPE IS implementer-name FOR target-list.
```

The possible values for LINKLIT are:

***PGM** Target for CALL/CANCEL or SET ENTRY is a program object.

***PRC** Target for CALL/CANCEL or SET ENTRY is an ILE procedure.

Top

Simple program (SIMPLEPGM)

Specifies if a PGM object is created for each of the compilation units in the sequence of source programs. This option is meaningful only if the input source member to this command contains a sequence of source programs which generate multiple modules. If this option is specified but the input source member does not have a sequence of source programs, then the option is ignored. The possible values are:

***YES** A program object is created for each of the compilation units in the sequence of source programs. If REPLACE(*NO) is specified and a program object of the same name already exists for a compilation unit in the sequence of source programs, that program object is not replaced and compilation continues at the next compilation unit.

***NO** A single program object is created from all the compilation units in the sequence, with the first compilation unit representing the Program Entry. With SIMPLEPGM(*NO) specified, if one source program in a sequence of source programs fails to generate a module then all subsequent source programs in the sequence will also fail to generate modules.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which you intend to use the program object being created. In the examples given for the *CURRENT and *PRV values, and when specifying the target-release value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V2R3M0 is version 2, release 3, modification level 0.

Valid values for this parameter change every release. The possible values are:

***CURRENT**
The object is to be used on the release of the operating system currently running on your system. For example, if V2R3M5 is running on the system, *CURRENT means that you intend to use the object on a system with V2R3M5 installed. The object can also be used on a system with any subsequent release of the operating system installed.

Note: If V2R3M5 is running on the system, and the object is to be used on a system with V2R3M0 installed, specify TGTRLS(V2R3M0), not TGTRLS(*CURRENT).

***PRV** The object is to be used on the previous release with modification level 0 of the operating system. For example, if V2R3M5 is running on your system, *PRV means you intend to use the object on a system with V2R2M0 installed. The object can also be used on a system with any subsequent release of the operating system installed.

target-release
Specify the release in the format VxRxMx. The object can be used on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. If you specify a target-release that is earlier than the earliest release level supported by this command, an error message is sent indicating the earliest supported release.

Note: The current version of the command may support options that are not available in previous releases of the command. If the command is used to create objects that are to be used on a previous release, it will be processed by the compiler appropriate to that release, and any unsupported options will not be recognized. The compiler will not necessarily issue any warnings regarding options that it is unable to process.

Top

Sort sequence (SRTSEQ)

Specifies the sort sequence used when NLSSORT is associated with an alphabet-name in the ALPHABET clause. The SRTSEQ parameter is used in conjunction with the LANGID parameter to determine which system-defined or user-defined sort sequence table the program object will use. The possible values are:

***HEX** No sort sequence table will be used, and the hexadecimal values of the characters will be used to determine the sort sequence.

***JOB** The sort sequence will be resolved and associated with the program object at compile time using the sort sequence of the compile job. The sort sequence table must exist in the system at compile time. If at run time, the CCSID of the runtime job differs from the CCSID of the compile time job, the sort sequence table loaded at compile time is converted to match the CCSID of the runtime job.

***JOBRUN**
The sort sequence of the program will be resolved and associated with the program at run time. At compile time, the compiler will associate the sort sequence of the compile job with the program. At run time, this sort sequence will be replaced by the sort sequence associated with the job at run time. This value allows a program to be compiled once and used with different sort sequences at run time.

***LANGIDUNQ**
Specifies that the sort sequence table being used must contain a unique weight for each character in the code page. The sort sequence table used will be the unique weighted table associated with the language specified in the LANGID parameter.

***LANGIDSHR**
Specifies that the sort sequence table being used can contain the same weight for multiple

characters in the code page. The sort sequence table used will be the shared weighted table associated with the language specified in the LANGID parameter.

table-name
Enter the name of the sort sequence table to be used. The table contains weights for all characters in a given code page. A weight is associated with the character that is defined at the code point. When using a sort sequence table name, the library in which the object resides can be specified. The valid values for the library are:

***LIBL** The library list is searched to find the library where the sort sequence table is located.

***CURLIB**
The current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the sort sequence table is found.

Top

Language id (LANGID)

Specifies the language identifier which is used in conjunction with the sort sequence. The LANGID parameter is used only when the SRTSEQ value in effect is *LANGIDUNQ or *LANGIDSHR. The possible values are:

***JOBRUN**
The language identifier of the program will be resolved at run time. When the compiled program is run, the language identifier of the job is used. This value allows a program to be compiled once and used with different language identifiers at run time.

***JOB** The language identifier of the program will be resolved at compile time.

language-identifier-name
Enter a valid 3-character language identifier.

Top

Enable performance collection (ENBPFRCOL)

Specifies whether performance measurement code should be generated in the module or program. The data collected can be used by the system performance tool to profile an application's performance. Generating the addition of the performance measurement code in a compiled module or program will result in slightly larger objects and may affect performance.

***PEP** Performance statistics are gathered on the entry and exit of the program entry procedure only. Choose this value when you want to gather overall performance information for an application. This support is equivalent to the support formally provided with the TPST tool. This is the default.

***ENTRYEXIT**
Performance statistics are gathered on the entry and exit of all the procedures of the program. This includes the program PEP routine.

This choice would be useful if you want to capture information on all routines. Use this option when you know that all the programs called by your application were compiled with either the *PEP, *ENTRYEXIT or *FULL option. Otherwise, if your application calls other programs that are

not enabled for performance measurement, the performance tool will charge their use of resources against your application. This would make it difficult for you to determine where resources are actually being used.

***FULL** Performance statistics are gathered on the entry and exit of all procedures. Also statistics are gathered before and after each call to an external procedure.

Use this option when you think that your application will call other programs that were not compiled with either *PEP, *ENTRYEXIT or *FULL. This option allows the performance tools to distinguish between resources that are used by your application and those used by programs it calls (even if those programs are not enabled for performance measurement). This option is the most expensive but allows for selectively analyzing various programs in an application.

Top

Binding directory (BNDDIR)

Specifies the list of binding directories that are used in symbol resolution.

***NONE**
No binding directory is specified.

binding-directory-name
Specify the name of the binding directory used in symbol resolution. The directory name can be qualified with one of the following library values:

***LIBL** The system searches the library list to find the library where the binding directory is stored. This is the default.

***CURLIB**
The current library for the job is searched. If no library is specified as the current library for the job, library QGPL is used.

***USRLIBL**
Only the libraries in the user portion of the job's library list are searched.

library-name
Specify the name of the library to be searched.

Top

Activation group (ACTGRP)

Specifies the activation group this program is associated with when it is called.

QILE When this program is called, it is activated into the named activation group QILE. This is the default.

***NEW** When this program is called, it is activated into a new activation group.

***CALLER**
When this program is called, it is activated into the caller's activation group.

activation-group-name
Specify the name of the activation group to be used when this program is called.

Top

Profiling data (PRFDTA)

Specifies the program profiling data attribute for the program. Program profiling is an advanced optimization technique used to reorder procedures and code within the procedures based on statistical data (profiling data).

***NOCOL**
This program is not enabled to collect profiling data. This is the default.

***COL** The program is enabled to collect profiling data. *COL can be specified only when the optimization level of the module is *FULL.

Note: If you use the BNDDIR parameter to bind additional modules and service programs, these additional objects are not affected when *COL or *NOCOL is specified for the program. The program profiling data attribute for a module is set when the module is created.

Top

Coded character set ID (CCSID)

Specifies the coded character set identifier (CCSID) that records in files, and data associated with LOCALEs, are converted to at run time.

***JOBRUN**
The CCSID of the program is resolved at run time. When the compiled program is run, the current job's default CCSID is used.

***JOB** The current job's default CCSID at compile time is used.

***HEX** The CCSID 65535 is used, which indicates that data in the fields is treated as bit data, and is not converted.

coded-character-set-identifier
Specifies the CCSID to be used.

Top

Arithmetic mode (ARITHMETIC)

Specifies the arithmetic mode for numeric data. The possible values are:

***NOEXTEND**
This option specifies the default arithmetic mode for numeric data. The intermediate result of a fixed-point arithmetic expression can be up to 30 digits and numeric literals may only have a maximum length of 18 digits.

***EXTEND31**
Use this option to increase the precision of intermediate results for fixed-point arithmetic. The intermediate result of a fixed-point arithmetic expression can be up to 31 digits and numeric literals may have a maximum length of 31 digits.

***EXTEND63**
Use this option to increase the precision of intermediate results for fixed-point arithmetic. The intermediate result of a fixed-point arithmetic expression can be up to 63 digits and numeric literals may have a maximum length of 63 digits.

Top

Padding character (NTLPADCHAR)

Specifies the national padding character (NTLPADCHAR) used when padding occurs in the following conversion situations:

1. Single byte character to national character.

2. Double byte character to national character.

3. National character to national character.

***DEFAULT**
This option specifies the default padding characters as follows:

1. Single byte character to national character (NX″0020″)

2. Double byte character to national character (NX″3000″)

3. National character to national character (NX″3000″)

national hexadecimal literal
Specifies any valid national hexadecimal literal of length 1 in format NX″ ″ or NX′ ′.

Top

Licensed Internal Code options (LICOPT)

Specifies one or more Licensed Internal Code compile-time options. This parameter allows individual compile-time options to be selected, and is intended for the advanced programmer who understands the potential benefits and drawbacks of each selected type of compiler option.

Top

Include directory (INCDIR)

Specifies one or more directories to add to the search path used by the compiler to find copy files. The compiler will search the directories specified here if the copy files in the source program can not be resolved.

***NONE**
No user directories are searched for copy files. By default, the current directory will still be searched.

′directory′
Specify up to 32 directories in which to search for copy files. In addition to the specified directories, the current directory is also searched for copy files.

Top

Generate program interface (PGMINFO)

Specifies whether program interface information should be generated into a stream file. The possible values are:

***NO** This option specifies the default which does not generate program interface information.

***PCML**
Specifies that PCML (Program Call Markup Language) should be generated into a stream file.

The generated PCML makes it easier for JAVA methods to call this COBOL program, with less Java code. The name of a stream file that will contain the generated PCML must be specified on the INFOSTMF option.

Top

Program interface stream file (INFOSTMF)

Specifies the path name of the stream file to contain the generated program interface information specifed on the PGMINFO option.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'.

If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by appending the job's current working directory to the path name.

This parameter can only be specified when the PGMINFO parameter has a value other than *NO.

Top

Examples

Example 1: Compiling a Source Program into a Program Object

```
CRTBNDCBL   PGM(MYLIB/XMPLE1)  SRCFILE(MYLIB/QCBLLESRC)
            SRCMBR(XMPLE1)  OUTPUT(*PRINT)
            TEXT('My ILE COBOL program')
```

This command calls the ILE COBOL compiler to create a program named XMPLE1. The source program is in member XMPLE1 of source file QCBLLESRC in library MYLIB. A compiler listing is created.

Top

Error messages

***ESCAPE Messages**

LNC9001
Compile failed. &1 not created.

LNC9006
TGTRLS(&1) specified, but compiler is not installed.

LNC9007
The product library is damaged, or the user is not allowed to use it.

LNC9015
TGTRLS(&1) is not valid.

Top

Create Bound C++ Program (CRTBNDCPP)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Create Bound C++ Program (CRTBNDCPP) command starts the ILE C++ compiler. You can use this command in either batch or interactive mode, or from a CL program. The compiler attempts to create a program object based on the ILE C++ statements in the source code.

Note: When the CRTBNDCPP command is invoked, a temporary module object is created in the QTEMP library. The name of the temporary module object is the same as the name you specified on the PGM parameter of the CRTBNDCPP command. If a module object with that name already exists in QTEMP, an error message is generated and compilation stops. The temporary module object, which is used with the CRTBNDCPP command, is deleted after compilation stops whether or not compilation is successful.

Error messages for CRTBNDCPP

***ESCAPE Messages**

CZS1613
The compilation failed.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Required, Positional 1
	Qualifier 1: Program	*Name*	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QCPPSRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 3
SRCSTMF	Source stream file	*Path name*	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OUTPUT	Output options	Single values: ***NONE**, '*none' Other values: *Element list*	Optional
	Element 1: Output file name	*Path name*, *PRINT, '*print'	
	Element 2: Title	*Character value*, ***BLANK**	
	Element 3: Subtitle	*Character value*, ***BLANK**	

Keyword	Description	Choices	Notes
OPTION	Compiler options	Values (up to 35 repetitions): *NOBITSIGN, *BITSIGN, *NOEVENTF, *EVENTF, *NOEXPMAC, *EXPMAC, *NOLOGMSG, *LOGMSG, *NOLONGLONG, *LONGLONG, *NORTTI, *RTTIALL, *RTTITYPE, *RTTICAST, *NOSHOWINC, *SHOWINC, *NOSHOWSRC, *SHOWSRC, *NOSHOWSYS, *SHOWSYS, *NOSHOWUSR, *SHOWUSR, *NOSYSINCPATH, *SYSINCPATH, *NOSTDINC, *STDINC, *NOINCDIRFIRST, *INCDIRFIRST, *NOXREF, *XREF, *NOXREFREF, *XREFREF, *NOFULL, *FULL, *NOSTDLOGMSG, *STDLOGMSG	Optional
CHECKOUT	Checkout options	Values (up to 45 repetitions): ***NONE**, *USAGE, *ALL, *NOCLASS, *CLASS, *NOCOND, *COND, *NOEFFECT, *EFFECT, *NOGENERAL, *GENERAL, *NOLANG, *LANG, *NOPARM, *PARM, *NOPORT, *PORT, *NOREACH, *REACH, *NOTEMP, *TEMP, *NOTRUNC, *TRUNC, *NOUNUSED, *UNUSED	Optional
OPTIMIZE	Optimization	**10**, 20, 30, 40	Optional
INLINE	Inline options	*Element list*	Optional
	Element 1: Inliner	***OFF**, *ON	
	Element 2: Mode	***NOAUTO**, *AUTO	
	Element 3: Threshold	1-65535, **250**, *NOLIMIT	
	Element 4: Limit	1-65535, **2000**, *NOLIMIT	
	Element 5: Report	***NO**, *YES	
DBGVIEW	Debugging view	***NONE**, *ALL, *STMT, *SOURCE, *LIST	Optional
DEFINE	Define names	Single values: ***NONE** Other values (up to 32 repetitions): *Character value*	Optional
LANGLVL	Language level	***EXTENDED**, *ANSI, *LEGACY	Optional
ALIAS	Alias	Values (up to 3 repetitions): ***ANSI**, *NOANSI, *ADDRTAKEN, *NOADDRTAKEN, *ALLPTRS, *NOALLPTRS, *TYPEPTR, *NOTYPEPTR	Optional
SYSIFCOPT	System interface options	***IFS64IO**, *IFSIO, *NOIFSIO	Optional
LOCALETYPE	Locale object type	***LOCALE**, *LOCALEUCS2, *LOCALEUTF	Optional
FLAG	Message flagging level	**0**, 10, 20, 30	Optional
MSGLMT	Compiler messages	*Element list*	Optional
	Element 1: Message limit	0-32767, ***NOMAX**	
	Element 2: Message limit severity	0, 10, 20, **30**	
REPLACE	Replace program object	***YES**, *NO	Optional
USRPRF	User profile	***USER**, *OWNER	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
ENBPFRCOL	Enable performance collection	*Element list*	Optional
	Element 1: Collection level	***PEP**, *ENTRYEXIT, *FULL	
	Element 2: Procedures	*NONLEAF, *ALLPRC	
PFROPT	Performance options	Values (up to 2 repetitions): *SETFPCA, *NOSETFPCA, *NOSTRDONLY, *STRDONLY	Optional
PRFDTA	Profiling data	***NOCOL**, *COL	Optional

Keyword	Description	Choices	Notes
TERASPACE	Teraspace options	Single values: ***NO** Other values: *Element list*	Optional
	Element 1: Teraspace enabled	***YES**	
	Element 2: Use teraspace interfaces	***NOTSIFC**, *TSIFC	
STGMDL	Storage model	***SNGLVL**, *TERASPACE	Optional
DTAMDL	Data model	***P128**, *LLP64	Optional
RTBND	Run time Binding	***DEFAULT**, *LLP64	Optional
PACKSTRUCT	Pack structure	***NATURAL**, 1, 2, 4, 8, 16	Optional
ENUM	Enum size	***SMALL**, 1, 2, 4, *INT	Optional
MAKEDEP	Dependency information	*Path name*, ***NONE**	Optional
INCDIR	Include directory	Single values: ***NONE** Other values (up to 32 repetitions): *Path name*	Optional
CSOPT	Compiler services option	*Character value*, ***NONE**	Optional
LICOPT	Licensed internal code options	*Character value*, ***NONE**	Optional
DFTCHAR	Default char type	***UNSIGNED**, *SIGNED	Optional
TGTCCSID	Target CCSID	1-65535, ***SOURCE**, *JOB, *HEX	Optional
TEMPLATE	Template options	*Element list*	Optional
	Element 1: Temporary include directory	*Path name*, ***NONE**, *TEMPINC	
	Element 2: Maximum generated headers	1-99999, **1**	
	Element 3: Template validity checking	***NO**, *WARN, *ERROR	

Top

Program (PGM)

Specifies the name and library for the created program object.

program-name

Enter a name for the program object.

The possible library values are:

***CURLIB**

The program object is stored in the current library. If a job does not have a current library, the program object is created in the QGPL library.

library-name

Specify the name of the library where the program object being created will be stored.

Top

Source file (SRCFILE)

Specifies the source file name and library of the file containing the ILE C++ source code that you want to compile.

QCPPSRC
The source file named QCPPSRC contains the member with the ILE C++ source code that you want to compile.

source-file-name
Enter the name of the source file that contains the member with the ILE C++ source code.

The possible library values are:

***LIBL** The library list is searched to find the library where the source file is located.

***CURLIB**
The current library is searched for the source file. If a job does not have a current library, QGPL is searched for the source file.

library-name
Enter the name of the library that contains the source file.

Top

Source member (SRCMBR)

Specifies the name of the member containing the source code to be compiled.

***PGM** The program name supplied on the PGM parameter is used as the source member name.

member-name
Enter the name of the member that contains the source code.

Top

Source stream file (SRCSTMF)

Specifies the path name of the stream file containing the source code that you want to compile.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'. If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by pre-pending the job's current working directory to the path name.

The SRCMBR and SRCFILE parameters cannot be specified with the SRCSTMF parameter.

Top

Text 'description' (TEXT)

Specifies the text that briefly describes the program object.

***SRCMBRTXT**
The text description associated with the source file member is used for the program object. If the source file is an inline file, a stream file, or a device file, the text will be blank.

***BLANK**
Specifies that no text appears.

'description'
Specify no more than 50 characters of text, enclosed in apostrophes.

Top

Output options (OUTPUT)

Specifies whether a compiler listing is produced.

Single Value

***NONE**
Does not generate the compiler listing. When a listing is not required, this parameter value should be used to improve compile-time performance. When *NONE is specified, any listing-related parameter values specified for the OPTION parameter are ignored.

Element 1: Output File Name

***PRINT**
Generates a spooled file containing the listing.

'path-name'
Specify the path name of a stream file to hold the listing.

Element 2: Title

***BLANK**
Specifies that no text appears.

'title' Specify a title string for the listing file (maximum 80 characters).

Element 3: Subtitle

***BLANK**
Specifies that no text appears.

'subtitle'
Specify a subtitle string for the listing file (maximum 80 characters).

Top

Compiler options (OPTION)

Specifies the options to use when the ILE C++ source code is compiled. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

The possible options are:

***NOBITSIGN**
Specify bitfields as unsigned.

***BITSIGN**
Specify bitfields as signed.

***NOEVENTF**
Do not create an event file for use by CoOperative Development Environment/400 (CODE/400).

***EVENTF**
Create an event file for use by CoOperative Development Environment/400 (CODE/400). The event file is created as a member in file EVFEVENT in the library where the module or program object being created will be stored. If the file EVFEVENT does not exist, it is automatically created. The event file member name is the same as the name of the object being created.

CODE/400 uses this file to provide error feedback integrated with the CODE/400 editor. An Event File is normally created when you create a module or program object from within CODE/400.

***NOEXPMAC**
Macros are not expanded in the listing unless a syntax error is encountered within the macro.

***EXPMAC**
Expand all macros in the listing.

***NOFULL**
Do not turn on all listing options.

***FULL** Turn on all listing options.

***NOINCDIRFIRST**
Include directories specified as INCDIR parameters are not included before the standard header file include path.

***INCDIRFIRST**
Include directories specified as INCDIR parameters are included before the standard header file include path.

***LOGMSG**
Puts the compilation messages in the job log.

When you specify this option and the FLAG parameter, messages with the severity specified on the FLAG parameter (and higher) are placed in the job log.

When you specify this option and a maximum number of messages on the MSGLMT parameter, compilation stops when the number of messages, at the specified severity, have been placed in the job log.

***NOLOGMSG**
Does not put the compilation messages in the job log.

***LONGLONG**
Allows the use of the long long data type.

***NOLONGLONG**
Do not allow the use of the long long data type.

***NORTTI**
Do not generate run-time type identification (RTTI) information.

***RTTIALL**
Generate the information needed for the RTTI typeid and the dynamic_cast operator.

***RTTITYPE**
Generate the information needed for the RTTI typeid operator only.

***RTTICAST**
Generate the information needed for the dynamic_cast operator only.

***NOSHOWINC**
Does not expand the user include files or the system include files in the source section of the listing or in the debug views.

***SHOWINC**
Expands both the user include files and the system include files in the source section of the listing or in the debug views. An OUTPUT option, or DBGVIEW parameter value of *ALL, *SOURCE or *LIST must be specified.

***SHOWSRC**
Show the source code in the listing. This option can be modified by the *SHOWINC, *SHOWSYS or *SHOWUSR options.

***NOSHOWSRC**
Does not show the source code in the listing. This option may be modified by the *EXPMAC, *SHOWINC, *SHOWSYS or *SHOWUSR options.

***NOSHOWSYS**
Do not expand the system include files on the #include directive in the source section of the listing or in the debug views. System include files are enclosed in angle brackets (< >) following the #include directive.

***SHOWSYS**
Expands the system include files on the #include directive in the source section of the listing or in the debug views. An output option, or DBGVIEW parameter of *ALL, *SOURCE or *LIST must be specified. System include files are enclosed in angle brackets (< >) following the #include directive.

***NOSHOWUSR**
Do not expand user include files in the listing or debug views. User include files are enclosed in double quotation marks (″ ″) following the #include directive.

***SHOWUSR**
Expands the user include files on the #include directive in the source section of the listing or in the debug views. An OUTPUT option, or DBGVIEW parameter value of *ALL, *SOURCE or *LIST must be specified. User include files are enclosed in double quotation marks (″ ″) following the #include directive.

***STDINC**
The system-supplied header files are included in the search path for the compile.

***NOSTDINC**
The system-supplied header files are not included in the search path for the compile.

***NOSTDLOGMSG**
Compilation messages are not sent to the stdout stream.

***STDLOGMSG**
Compilation messages are sent to the stdout stream.

***NOSYSINCPATH**
The search path for user includes is not affected.

***SYSINCPATH**
Changes the search path of user includes to the system include search path. In function this option is equivalent to changing the double-quotes in the user #include directive (#include ″file_name″) to angle brackets (#include <file_name>).

***NOXREF**
Does not generate the cross-reference table in the listing.

***XREF** Generates the cross-reference table containing a list of the identifiers in the source code together with the numbers of the lines in which they appear. An OUTPUT option must be specified.

***NOXREFREF**
Do not produce a cross-reference table of referenced identifiers in the listing.

***XREFREF**
Produce a cross-reference table of referenced variables, structures, and function names in the listing file. The table shows the line numbers where the identifiers are declared. An OUTPUT option must be specified.

Checkout options (CHECKOUT)

Specifies options you may select to generate informational messages that indicate possible programming errors. When you specify an option more than once, or when two options conflict, the last one specified is used.

***NONE**
Disables all of the options for CHECKOUT.

***USAGE**
This is equivalent to specifying *COND.

***ALL** Enables all of the options for CHECKOUT.

***NOCLASS**
Do not display information about class use.

***CLASS**
Display information about class use.

***NOCOND**
Do not warn about possible redundancies or problems in conditional expressions.

***COND**
Warn about possible redundancies or problems in conditional expressions.

***NOEFFECT**
Do not warn about statements with no effect.

***EFFECT**
Warn about statements with no effect.

***NOGENERAL**
Do not generate general checkout messages.

***GENERAL**
Generate general checkout messages.

***NOLANG**
Do not display information about the effects of the language level.

***LANG**
Display information about the effects of the language level.

***NOPARM**
Do not warn about unused parameters.

***PARM**
Warn about unused parameters.

***NOPORT**
Do not warn about non-portable language constructs.

***PORT**
Warn about non-portable language constructs.

***NOREACH**
Do not warn about unreachable statements.

***REACH**
Warn about unreachable statements.

***NOTEMP**
Do not generate messages if the compiler creates temporary variables.

***TEMP**
Generate messages if the compiler creates temporary variables.

***NOTRUNC**
Do not warn about the possible truncation or loss of data.

***TRUNC**
Warn about the possible truncation or loss of data.

***NOUNUSED**
Do not check for unused auto or static variables.

***UNUSED**
Check for unused auto or static variables.

Top

Optimization (OPTIMIZE)

Specifies the levels of optimization of the generated object.

10 Generated code is not optimized. This level has the shortest compile time. This level allows variables to be displayed and modified while debugging.

20 Some optimization is performed on the code. This level allows user variables to be displayed but not modified while debugging.

30 Full optimization is performed on the generated code. During a debug session, user variables may not be modified but may be displayed. The presented values may not be the current value of the variable.

40 All optimizations done at level 30 are performed on the generated code. In addition, code is eliminated from procedure prologue and epilogue routines that enable instruction trace and call trace system functions. Eliminating this code enables the creation of leaf procedures. A leaf procedure is a procedure that contains no calls to other procedures. Procedure call performance to a leaf procedure is significantly faster than to a normal procedure.

Top

Inline options (INLINE)

Specifies whether the compiler should consider replacing a function call with the called function's instructions. Inlining a function eliminates the overhead of a call and can result in better optimization. Small functions that are called many times are good candidates for inlining.

Element 1: Inliner

Specifies whether or not inlining will be used.

***OFF** Specifies that inlining will not be performed on the compilation unit.

***ON** Specifies that inlining will be performed on the compilation unit. If a debug view is specified, the inliner is turned off.

Element 2: Mode

Specifies whether or not the inliner should attempt to automatically inline functions depending on their Threshold and Limit values.

***NOAUTO**
Specifies that only the functions that have been specified with the #pragma inline directive should be considered candidates for inlining.

***AUTO**
Specifies that the inliner should determine if a function can be inlined based on the Threshold and Limit values specified. The #pragma noinline directive overrides *AUTO.

Element 3: Threshold

Specifies the maximum size of a function that can be a candidate for automatic inlining. The size is measured in Abstract Code Units (ACUs). ACUs are proportional in size to the executable code in the function. Source code is translated into ACUs by the compiler.

250 Specifies a threshold of 250.

number-of-ACUs
Specifies a threshold from 1 to 65535 ACUs.

***NOLIMIT**
Defines the threshold as the maximum size of the program object.

Element 4: Limit

Specifies the maximum relative size a function can grow before auto-inlining stops.

2000 Specifies a limit of 2000 ACUs.

***NOLIMIT**
Limit is defined as the maximum size of the program object. System limits may be encountered.

number-of-ACUs
A limit from 1 to 65535 ACUs may be specified.

Element 5: Report

Specifies whether or not to produce an inliner report with the compiler listing.

***NO** The inliner report is not produced.

***YES** The inliner report is produced as part of the compiler listing. An OUTPUT option must be specified to produce the inliner report.

Top

Debugging view (DBGVIEW)

Specifies which level of debugging is available for the module in the created program object. It also specifies which source views are available for source level debugging. Requesting a debug view will turn inlining off.

***NONE**
Debug capability is not inserted into the program object.

***ALL** Enables all of the debug options (*STMT, *SOURCE and *LIST)

***STMT**
Allows the program object to be debugged using program statement numbers and symbolic identifiers.

Note: To debug a module object using the *STMT option you need a listing.

***SOURCE**
Generates the source view for debugging the program object. OPTION parameter values *NOSHOWINC, *SHOWINC, *SHOWSYS, and *SHOWUSR determine the content of the source view created.

Note: In order to use this view for debugging, the root source file should not be modified, renamed or moved after the program object is created.

***LIST** Generates the listing view for debugging the program object. OPTION parameter values *SHOWINC, *SHOWUSR, *SHOWSYS, and *NOSHOWINC determine the content of the listing view created.

Top

Define names (DEFINE)

Specifies preprocessor macros that take effect before the file is processed by the compiler. Using the format DEFINE(macro) is equivalent to DEFINE('macro=1').

***NONE**
No macro is defined.

'name'* or *'name=value'
A maximum of 32 macros may be defined. Each macro name is enclosed in apostrophes. The maximum length of a macro name is 80 characters. The apostrophes are not part of the 80 character string. The apostrophes are required for case-sensitive macro names.

Note: Macros defined in the command override any macro definition of the same name in the source but a warning message is generated by the compiler. Function-like macros such as #define max(a,b) ((a)>(b):(a)?(b)) cannot be defined on the command line.

Top

Language level (LANGLVL)

Specifies the capabilities of the compiler and which prototypes are declared when the source is created.

***EXTENDED**
Defines the preprocessor variable __EXTENDED__ and undefines other language-level variables. This parameter should be used when all the capabilities of ILE C++ are required.

***ANSI**
Defines the preprocessor variables __ANSI__, __STDC__ and __cplusplus98_interface__, and undefines other language-level variables. Only ANSI-standard C++ is made available.

***LEGACY**
This option allows some of the source constructs acceptable to earlier compilers.

Top

Alias (ALIAS)

Specifies the aliasing assertion to be applied in the program object being created.

***ANSI**
The program object will only allow pointers to point to an object of the same type.

***NOANSI**
The program object will not use the *ANSI aliasing rules.

***ADDRTAKEN**
The program object will have its class of variables disjoint from pointers unless their address is taken.

***NOADDRTAKEN**
The program object will not use the *ADDRTAKEN aliasing rules.

***ALLPTRS**
The program object will not allow any two pointers to be aliased.

***NOALLPTRS**
The program object will not use the *ALLPTRS aliasing rules.

***TYPEPTR**
The program object will not allow any two pointers of different types to be aliased.

***NOTYPEPTR**
The program object will not use the *TYPEPTR aliasing rules.

Top

System interface options (SYSIFCOPT)

Specifies which system interface options will be used for the program object being created. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

***IFS64IO**
The program object will use the Integrated File System for 64-bit C stream I/O operations.

***IFSIO**
The program object will use the Integrated File System for C stream I/O operations.

***NOIFSIO**
The program object will use the iSeries Data Management file system for C stream I/O operations.

Top

Locale object type (LOCALETYPE)

Specifies the type of locale support to be used by the program object being created.

***LOCALE**
Program objects created with this option use the locale support provided by *LOCALE objects.

***LOCALEUCS2**
Program objects created with this option use the locale support provided by *LOCALE objects. Wide-character types will contain two-byte universal character set values.

***LOCALEUTF**
Program objects created with this option use the locale support provided by *LOCALE objects. Wide-character types will contain four-byte utf-32 values. Narrow character types will contain utf-8 values.

Top

Message flagging level (FLAG)

Specifies the level of messages that are to be displayed in the listing.

0 All messages starting at the informational level are displayed.

10 All messages starting at the warning level are displayed.

20 All messages starting at the error level are displayed.

30 All messages starting at the severe error level are displayed.

Top

Message limit (MSGLMT)

Specifies the maximum number of messages at the given message severity that can occur before the compilation is stopped.

Element 1: Message Limit

Specifies the maximum number of messages that can occur at, or above, the message severity level specified.

***NOMAX**
Compilation continues regardless of the number of messages that have occurred at the message severity level specified.

message-limit
Specify the number of messages that can occur. The valid range is 0 to 32767.

Element 2: Message Severity

Specifies the message severity that can stop the compilation if the *message-limit* number of messages at the specified severity or above occur.

30 Specifies that a *message-limit* of messages at severity 30 can occur before compilation stops.

0 Specifies that a *message-limit* of messages at severity 0 or above can occur before compilation stops.

10 Specifies that a *message-limit* of messages at severity 10 or above can occur before compilation stops.

20 Specifies that a *message-limit* of messages at severity 20 or above can occur before compilation stops.

Top

Replace program object (REPLACE)

Specifies whether the existing version of the object will be replaced by the current version.

***YES** The existing object is replaced by the new version. The old version is moved to the QRPLOBJ library and renamed based on the system date and time. The text description of the replaced object is changed to the name of the original object. The old object is deleted at the next IPL if it has not been explicitly deleted.

***NO** The existing object is not replaced. When an object with the same name is found in the specified library, a message is displayed and compilation stops.

Top

User profile (USRPRF)

Specifies the user profile that is used when the created program object is run, including the authority that the program object has for each object. The profile of either the program owner or the program user is used to control which objects can be used by the program object.

***USER**
The profile of the user running the program object is used.

***OWNER**
The user profiles of both the program's owner and the program's user are used when the program object is processed. The collective sets of object authority in both user profiles are used to find and access objects during program processing. Authority from the owning user profile's group profile is not included in the authority for the running program object.

Top

Authority (AUT)

Specifies the authority granted to users who do not have specific authority to the object, who are not on the authorization list, or whose group has no specific authority to the object.

***LIBCRTAUT**
Public authority for the object is taken from the CRTAUT keyword of the target library (the library that contains the created object). This value is determined when the object is created. When the CRTAUT value for the library changes after the object is created, the new value does not affect any existing objects in the library.

***ALL** Provides authority for all operations on the object except those limited to the owner or controlled by authorization list management authority. Any user can control the object's existence, specify the security for it, change it, and perform basic functions on it, including transfer its ownership.

***CHANGE**
Provides all data authority and the authority to perform all operations on the object except those limited to the owner or controlled by object authority and object management authority. The object can be changed and basic functions can be performed on it.

***USE** Provides object operational authority; read authority; and authority for basic read-only operations on the object, such as binding of a module object. Users without specific authority are prevented from changing the object.

***EXCLUDE**
Users without special authority cannot access the object.

authorization-list-name
Enter the name of an authorization list of users and authorities to which the object is added. The object is secured by this authorization list, and the public authority for the object is set to *AUTL. The authorization list must exist on the system when the command is issued.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which the user intends to use the object being created.

In the examples given for the *CURRENT and *PRV values, and when specifying the release-level value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V4R5M0 is version 4, release 5, modification level 0.

***CURRENT**
The object will be used on the release of the operating system currently running on the user's system. For example, if V4R5M5 is running on the system, *CURRENT means the user intends to use the object on a system with V4R5M5 installed. The user can also use the object on a system with any subsequent release of the operating system installed.

Note: If V4R5M5 is running on the system, and the object will be used on a system with V4R5M0 installed, specify TGTRLS(V4R5M0) not TGTRLS(*CURRENT).

***PRV** The object will be used on the previous release with modification level 0 of the operating system. For example, if V4R5M5 is running on the user's system, *PRV means the user intends to use the object on a system with V4R4M0 installed. The user can also use the object on a system with any subsequent release of the operating system installed.

release-level
Specify the release in the format VxRxMx. The object can be used on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level. They change with each new release. If you specify a release level which is earlier than the earliest release level supported by this command, an error message is sent.

Top

Enable performance collection (ENBPFRCOL)

Specifies whether performance measurement code should be generated in the object. The data collected can be used by the system performance tool to profile an application's performance. Generating performance measurement code in a created object will result in slightly larger objects and may affect performance.

***PEP** Performance statistics are gathered on the entry and exit of the program entry procedure only. Choose this value when you want to gather overall performance information for an application.

***ENTRYEXIT *NONLEAF**
Performance statistics are gathered on the entry and exit of all procedures of the program object that are not leaf procedures. This includes the program PEP routine.

This choice would be useful if you only wanted to capture information on those routines that invoke other routines in your application.

***ENTRYEXIT *ALLPRC**
Performance statistics are gathered on the entry and exit of all the procedures of the program object (including those that are leaf procedures). This includes the program PEP routine.

This choice would be useful if you wanted to capture information on all routines. Use this option when you know that all the program objects called by your application were created with either the *PEP, *ENTRYEXIT or *FULL option. Otherwise, if your application calls other program objects that are not enabled for performance measurement, the performance tool will charge their use of resources against your application. This would make it difficult for you to determine where resources are actually being used.

***FULL *NONLEAF**
Performance statistics are gathered on entry and exit of all procedures that are not leaf procedures. Also, statistics are gathered before and after each call to an external procedure.

***FULL *ALLPRC**
Performance statistics are gathered on the entry and exit of all procedures including leaf procedures. Also statistics are gathered before and after each call to an external procedure.

Use this option when you think that your application will call other program objects that were not created with either *PEP, *ENTRYEXIT or *FULL. This option allows the performance tools to distinguish between resources that are used by your application and those used by program objects it calls (even if those program objects are not enabled for performance measurement). This option is the most expensive but allows for selectively analyzing various program objects in an application.

Top

Performance options (PFROPT)

Specifies various options available to boost performance. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

The possible options are:

***SETFPCA**
Causes the compiler to set the floating point computational attributes to achieve ANSI semantics for floating point computations.

***NOSETFPCA**
No computational attributes will be set. This option should only be used when the object being created does not have any floating point computations in it.

***NOSTRDONLY**
Specifies that the compiler must place strings into writeable memory.

***STRDONLY**
Specifies that the compiler may place strings into read-only memory.

Top

Profiling data (PRFDTA)

Specifies the program profiling data attribute for the program object. Program profiling is an advanced optimization technique used to reorder procedures and code within the procedures based on statistical data (profiling data).

***NOCOL**
The program object is not enabled to collect profiling data.

***COL** The program object is enabled to collect profiling data. *COL can be specified only when the optimization level is 30 or greater.

Top

Teraspace options (TERASPACE)

Specifies whether the program object is enabled to work with teraspace storage. This includes teraspace storage allocated by the program object and parameters passed from other teraspace-enabled program and service program objects.

Element 1: Teraspace Enabled

***NO** The program object is not enabled to handle addressing of storage allocated from teraspace.

***YES** The program object is enabled to handle addressing of storage allocated from teraspace, including parameters passed from other teraspace-enabled program and service program objects.

Element 2: Use Teraspace Interfaces

***NOTSIFC**
The program object will default to use the non-teraspace versions of the storage functions.

***TSIFC**
The program object will default to use the teraspace versions of the storage functions. The compiler will define macro variable __TERASPACE__.

Top

Storage model (STGMDL)

Specifies the type of storage to be used by the object created.

***SNGLVL**
The object created will use single-level storage.

***TERASPACE**
The object created will use teraspace storage.

Top

Data model (DTAMDL)

Specifies the sizes (in bytes) of variables declared as int, long, pointer.

***P128** Causes the sizes of int, long, pointer to be 4, 4, 16 respectively.

***LLP64**
Causes the sizes of int, long, pointer to be 4, 4, 8 respectively. The compiler will define the macro __LLP64_IFC__.

Top

Run time Binding (RTBND)

Specifies the run time binding directory for the object created.

***DEFAULT**
The object created will use the default binding directory.

***LLP64**
The object created will use the 64-bit run time binding directory. This value is only available in conjunction with teraspace storage model, 64-bit data model and teraspace storage functions interface options. The compiler will define the macro __LLP64_RTBND__.

Top

Pack structure (PACKSTRUCT)

Specifies the alignment boundary to use for members of a structure.

***NATURAL**
Structure members are aligned on their natural boundaries. For example, a short integer will be two-byte aligned. 16-byte pointers will always align on 16-byte boundaries.

1 Pack structure members on a 1-byte alignment.

2 Pack structure members on a 2-byte alignment.

4 Pack structure members on a 4-byte alignment.

8 Pack structure members on a 8-byte alignment.

16 Pack structure members on a 16-byte alignment.

Top

Enum size (ENUM)

Specifies the number of bytes the compiler uses to represent an enumeration.

***SMALL**
Make all enum variables the smallest size that can represent the range of values.

1 Make all enum variables 1 byte.

2 Make all enum variables 2 bytes.

4 Make all enum variables 4 bytes.

***INT** Use the ANSI-standard enum size, which is 4 bytes.

Top

Dependency information (MAKEDEP)

Specifies whether or not to generate dependency information into a file. This information can be used by a make tool.

***NONE**
Do not generate dependency information.

'path-name'
Specify a path name for the stream file in which to store the dependency information.

Top

Include directory (INCDIR)

Specifies one or more directories to add to the search path used by the compiler to find include files.

The search path can be further modified by using the following parameters on the OPTION keyword:

- *INCDIRFIRST or *NOINCDIRFIRST
- *SYSINCPATH or *NOSYSINCPATH
- *STDINC or *NOSTDINC

***NONE**
Unless modified, the default system include directory and the source directory will be searched for user include files.

'directory'
Specify up to 32 directories in which to search for include files. In addition to the specified directories, the source directory is searched for user include files.

Top

Compiler services option (CSOPT)

Specifies one or more compiler service options. This parameter allows IBM to provide switchable compiler capability between releases.

***NONE**
No compiler service option is selected.

'compiler-service-options-string'
The selected compiler service options are used when creating the module object. Valid strings will be described in PTF cover letters or in release notes.

Top

Licensed internal code options (LICOPT)

Specifies one or more Licensed Internal Code compile-time options. This parameter allows individual compile-time options to be selected, and is intended for the advanced programmer who understands the potential benefits and drawbacks of each selected type of compiler option.

The possible values are:

***NONE**
No compile-time options are selected.

'Licensed-Internal-Code-options-string'
The selected Licensed Internal Code compile-time options are used when creating the module object. Certain options may reduce your ability to debug the created module object.

Top

Default char type (DFTCHAR)

Specifies the default sign for the char data type.

***UNSIGNED**
Make default char type unsigned.

***SIGNED**
Make default char type signed.

Top

Target CCSID (TGTCCSID)

Specifies the target coded character set identifier used to describe data stored into the resulting program object.

The possible values are:

***SOURCE**
The root source file's CCSID is used.

***JOB** The current job's CCSID is used.

***HEX** The CCSID 65535 is used, which indicates that character data is treated as binary data and is not converted.

coded-character-set-identifier
Specify the CCSID to be used.

Top

Template options (TEMPLATE)

Specifies template options to the compiler.

Element 1: Template Include Directory

***NONE**
Templates are not generated.

***TEMPINC**
Templates are generated into a directory named tempinc which is created in the directory where the root source file was found. If the source file is not a stream file, a file named TEMPINC will be created in the library where the source file resides. The TEMPLATE(*TEMPINC) and TMPLREG parameters are mutually exclusive.

'directory'
Specify the directory where the compiler will generate templates.

Element 2: Maximum Generated Headers

1 The maximum number of generated headers that contain templates.

number-of-header-files
Specify an integer value from 1 to 99999 for the maximum number of generated header files.

Element 3: Template Validity Checking

Controls whether parsing and semantic checking are applied to template definition implementations or only to template instantiations. The compiler has the option to producte warning or error messages. Available parameters are:

***NO** Do not parse to reduce the number of errors issued in code written for previous versions of the compiler.

***WARN**
Issue warning messages for semantic errors. Issue error messages for errors found while parsing.

***ERROR**
Treat problems in template implementations as errors, even if the template is not instantiated.

Top

Examples

None

Top

Error messages

***ESCAPE Messages**

CZS1613
The compilation failed.

Top

Create Bound RPG Program (CRTBNDRPG)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Create Bound RPG Program (CRTBNDRPG) command compiles and binds the RPG source code to create an ILE program object (*PGM). You can use this command interactively or in batch mode.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Optional, Positional 1
	Qualifier 1: Program	*Name*, ***CTLSPEC**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QRPGLESRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 3
SRCSTMF	Source stream file	*Path name*	Optional
GENLVL	Generation severity level	0-20, **10**	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
DFTACTGRP	Default activation group	***YES**, *NO	Optional
ACTGRP	Activation group	*Name*, **QILE**, *NEW, *CALLER	Optional
BNDDIR	Binding directory	Single values: ***NONE** Other values (up to 50 repetitions): *Qualified object name*	Optional
	Qualifier 1: Binding directory	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB, *USRLIBL	
OPTION	Compiler options	Values (up to 20 repetitions): *XREF, *NOXREF, *GEN, *NOGEN, *SECLVL, *NOSECLVL, *SHOWCPY, *NOSHOWCPY, *EXPDDS, *NOEXPDDS, *EXT, *NOEXT, *NOSHOWSKP, *SHOWSKP, *NOSRCSTMT, *SRCSTMT, *DEBUGIO, *NODEBUGIO, *NOEVENTF, *EVENTF	Optional
DBGVIEW	Debugging views	***STMT**, *SOURCE, *LIST, *COPY, *ALL, *NONE	Optional
OUTPUT	Output	***PRINT**, *NONE	Optional
OPTIMIZE	Optimization level	***NONE**, *BASIC, *FULL	Optional
INDENT	Source listing indentation	*Character value*, ***NONE**	Optional
CVTOPT	Type conversion options	Single values: ***NONE** Other values (up to 4 repetitions): *DATETIME, *GRAPHIC, *VARCHAR, *VARGRAPHIC	Optional
SRTSEQ	Sort sequence	Single values: ***HEX**, *JOB, *JOBRUN, *LANGIDUNQ, *LANGIDSHR Other values: *Qualified object name*	Optional
	Qualifier 1: Sort sequence	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	

Keyword	Description	Choices	Notes
LANGID	Language identifier	*Name*, ***JOBRUN**, *JOB	Optional
REPLACE	Replace program	***YES**, *NO	Optional
USRPRF	User profile	***USER**, *OWNER	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
TRUNCNBR	Truncate numeric	***YES**, *NO	Optional
FIXNBR	Fix numeric	Single values: ***NONE** Other values (up to 2 repetitions): *ZONED, *INPUTPACKED	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
ALWNULL	Allow null values	***NO**, *INPUTONLY, *USRCTL, *YES	Optional
DEFINE	Define condition names	Values (up to 32 repetitions): *Simple name*, ***NONE**	Optional
ENBPFRCOL	Enable performance collection	***PEP**, *ENTRYEXIT, *FULL	Optional
PRFDTA	Profiling data	***NOCOL**, *COL	Optional
LICOPT	Licensed Internal Code options	*Character value*, **X''**	Optional
INCDIR	Include directory	Values (up to 32 repetitions): *Path name*, ***NONE**	Optional
PGMINFO	Generate program interface	***NO**, *PCML	Optional
INFOSTMF	Program interface stream file	*Path name*	Optional
PPGENOPT	Preprocessor options	Single values: ***NONE**, *DFT Other values (up to 3 repetitions): *RMVCOMMENT, *NORMVCOMMENT, *EXPINCLUDE, *NOEXPINCLUDE, *SEQSRC, *NOSEQSRC	Optional
PPSRCFILE	Output source file	*Qualified object name*	Optional
	Qualifier 1: Output source file	*Name*	
	Qualifier 2: Library	*Name*, ***CURLIB**	
PPSRCMBR	Output source member	*Name*, ***PGM**	Optional
PPSRCSTMF	Output stream file	*Path name*, *SRCSTMF	Optional

Top

Program (PGM)

Specifies the program name and library name for the program object (*PGM) you are creating. The program name and library name must conform to server naming conventions. If no library is specified, the created program is stored in the current library.

***CTLSPEC**
The name for the compiled program is taken from the name specified in the DFTNAME keyword of the control specification. If the program name is not specified on the control specification and the source member is from a database file, the member name, specified by the SRCMBR parameter, is used as the program name. If the source is not from a database file then the program name defaults to RPGPGM.

program-name
Enter the name of the program object.

***CURLIB**
The created program object is stored in the current library. If you have not specified a current library, QGPL is used.

library-name
Enter the name of the library where the created program object is to be stored.

Top

Source file (SRCFILE)

Specifies the name of the source file that contains the ILE RPG source member to be compiled and the library where the source file is stored. The recommended source physical file length is 112 characters: 12 for the sequence number and date, 80 for the code and 20 for the comments. This is the maximum amount of source that is shown on the compiler listing.

QRPGLESRC
The default source file QRPGLESRC contains the ILE RPG source member to be compiled.

source-file-name
Enter the name of the source file that contains the ILE RPG source member to be compiled.

***LIBL** The system searches the library list to find the library where the source file is stored. This is the default.

***CURLIB**
The current library is used to find the source file. If you have not specified a current library, QGPL is used.

library-name
Enter the name of the library where the source file is stored.

Top

Source member (SRCMBR)

Specifies the name of the member of the source file that contains the ILE RPG source program to be compiled.

***PGM** Use the name specified for the PGM parameter as the source file member name. The compiled program object will have the same name as the source file member. If no program name is specified by the PGM parameter, the command uses the first member created in or added to the source file as the source member name.

source-file-member-name
Enter the name of the member that contains the ILE RPG source program.

Top

Source stream file (SRCSTMF)

Specifies the path name of the stream file containing the ILE RPG source code to be compiled.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'.

If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by appending the job's current working directory to the path name.

The SRCMBR and SRCFILE parameters cannot be specified with the SRCSTMF parameter.

Top

Generation severity level (GENLVL)

Controls the creation of the program object. The program object is created if all errors encountered during compilation have a severity level less than or equal to the generation severity level specified.

The value must be between 0 and 20 inclusive. For errors greater than severity 20, the program object will not be generated.

10 A program object will be generated when the compile-time errors have a severity level less than or equal to 10. This is the default.

severity-level-value
Enter a number, 0 through 20 inclusive.

Top

Text 'description' (TEXT)

Allows you to enter text that briefly describes the program and its function. The text appears whenever program information is displayed.

***SRCMBRTXT**
The text of the source member is used.

***BLANK**
No text appears.

'description'
Enter the text that briefly describes the function of the source specifications. The text can be a maximum of 50 characters and must be enclosed in apostrophes. The apostrophes are not part of the 50-character string. Apostrophes are not required if you are entering the text on the prompt screen.

Top

Default activation group (DFTACTGRP)

Specifies whether the created program is intended to always run in the default activation group.

***YES** When this program is called it will always run in the default activation group. The default activation group is the activation group where all original program model (OPM) programs are run.

Specifying DFTACTGRP(*YES) allows ILE RPG programs to behave like OPM programs in the areas of file sharing, file scoping, and RCLRSC.

ILE static binding is not available when a program is created with DFTACTGRP(*YES). This means that you cannot use the BNDDIR or ACTGRP parameters when creating this program. In addition, any call operation in your source must call a program and not a procedure.

DFTACTGRP(*YES) is useful when attempting to move an application on a program-by-program basis to ILE RPG.

***NO** The program is associated with the activation group specified by the ACTGRP parameter. Static binding is allowed when *NO is specified.

If ACTGRP(*CALLER) is specified and this program is called by a program running in the default activation group, then this program will behave according to ILE semantics in the areas of file sharing, file scoping and RCLRSC.

DFTACTGRP(*NO) is useful when you intend to take advantage of ILE concepts, for example, running in a named activation group or binding to a service program.

Top

Activation group (ACTGRP)

Specifies the activation group this program is associated with when it is called.

QILE When this program is called, it is activated into the named activation group QILE. This is the default.

***NEW** When this program is called, it is activated into a new activation group.

***CALLER**
When this program is called, it is activated into the caller's activation group.

activation-group-name
Specify the name of the activation group to be used when this program is called.

Top

Binding directory (BNDDIR)

Specifies the list of binding directories that are used in symbol resolution.

***NONE**
No binding directory is specified.

binding-directory-name
Specify the name of the binding directory used in symbol resolution. The directory name can be qualified with one of the following library values:

***LIBL** The system searches the library list to find the library where the binding directory is stored. This is the default.

***CURLIB**
The current library for the job is searched. If no library is specified as the current library for the job, library QGPL is used.

***USRLIBL**
Only the libraries in the user portion of the job's library list are searched.

library-name
Specify the name of the library to be searched.

Top

Compiler options (OPTION)

Specifies the options to use when the source member is compiled. You can specify any or all of the options in any order. Separate the options with one or more blank spaces. If an option is specified more than once, the last one is used.

***XREF** Produces a cross-reference listing (when appropriate) for the source member.

***NOXREF**
A cross-reference listing is not produced.

***GEN** Create a program object if the highest severity level returned by the compiler does not exceed the severity specified in the GENLVL option.

***NOGEN**
Do not create a program object.

***NOSECLVL**
Do not print second-level message text on the line following the first-level message text.

***SECLVL**
Print second-level message text on the line following the first-level message text in the Message Summary section.

***SHOWCPY**
Show source records of members included by the /COPY compiler directive.

***NOSHOWCPY**
Do not show source records of members included by the /COPY compiler directive.

***EXPDDS**
Show the expansion of externally described files in the listing and display key field information.

***NOEXPDDS**
Do not show the expansion of externally described files in the listing or display key field information.

***EXT** Show the list of external procedures and fields referenced during the compile on the listing.

***NOEXT**
Do not show the list of external procedures and fields referenced during compilation on the listing.

***NOSHOWSKP**
Do not show ignored statements in the source part of the listing. The compiler ignores statements as a result of /IF, /ELSEIF or /ELSE directives.

***SHOWSKP**
Show all statements in the source part of the listing, regardless of whether or not the compiler has skipped them.

***NOSRCSTMT**
Line Numbers in the listing are assigned sequentially; these numbers are used when debugging using statement numbers. Line Numbers are shown on the left-most column of the listing. The source IDs and SEU Sequence Numbers are shown on the two right-most columns of the listing.

***SRCSTMT**
Statement numbers for debugging are generated using SEU sequence numbers and source IDs as follows:

```
Statement_Number = source_ID * 1000000 +
                   source_SEU_sequence_number
```

SEU Sequence Numbers are shown on the left-most column of the listing. Statement Numbers are shown on the right-most column of the listing; these numbers are used when debugging using statement numbers.

Note: When OPTION(*SRCSTMT) is specified, all sequence numbers in the source files must contain valid numeric values. If there are duplicate sequence numbers in the same source file, the behavior of the debugger may be unpredictable and statement numbers for diagnostic messages or cross reference entries may not be meaningful.

***DEBUGIO**
Generate breakpoints for all input and output specifications.

***NODEBUGIO**
Do not generate breakpoints for input and output specifications.

***NOEVENTF**
Do not create an Event File for use by CoOperative Development Environment (CODE). CODE uses this file to provide error feedback integrated with the CODE editor. An Event File is normally created when you create a module or program from within CODE.

***EVENTF**
Create an Event File for use by CoOperative Development Environment (CODE). The Event File is created as a member in file EVFEVENT in the library where the created module or program object is to be stored. If the file EVFEVENT does not exist it is automatically created. The Event File member name is the same as the name of the object being created.

CODE uses this file to provide error feedback integrated with the CODE editor. An Event File is normally created when you create a module or program from within CODE.

Top

Debugging views (DBGVIEW)

Specifies which level of debugging is available for the compiled program object, and which source views are available for source-level debugging.

***STMT**
Allows the program object to be debugged using the Line Numbers or Statement Numbers of the compiler listing. Line Numbers are shown on the left-most column of the source section of the compiler listing when OPTION(*NOSRCSTMT) is specified. Statement Numbers are shown on the right-most column of the source section of the compiler listing when OPTION(*SRCSTMT) is specified.

***SOURCE**
Generates the source view for debugging the compiled program object. This view is not available if the root source member is a DDM file. Also, if changes are made to any source members after the compile and before attempting to debug the program, the views for those source members may not be usable.

***LIST** Generates the listing view for debugging the compiled program object. The information contained in the listing view is dependent on whether *SHOWCPY, *EXPDDS, and *SRCSTMT are specified for the OPTION parameter.

Note: The listing view will not show any indentation which you may have requested using the Indent option.

***COPY**
Generates the source and copy views for debugging the compiled program object. The source view for this option is the same source view generated for the *SOURCE option. The copy view is a debug view which has all the /COPY source members included. These views are not available if the root source member is a DDM file. Also, if changes are made to any source members after the compile and before attempting to debug the program, the views for those source members may not be usable.

***ALL** Generates the listing, source and copy views for debugging the compiled program object. The information contained in the listing view is dependent on whether *SHOWCPY, *EXPDDS, and *SRCSTMT are specified for the OPTION parameter.

***NONE**
Disables all of the debug options for debugging the compiled program object.

Top

Output (OUTPUT)

Specifies if a compiler listing is generated.

***PRINT**
Produces a compiler listing, consisting of the ILE RPG program source and all compile-time messages. The information contained in the listing is dependent on whether *XREF, *SECLVL, *SHOWCPY, *EXPDDS, *EXT, *SHOWSKP, and *SRCSTMT are specified for the OPTION parameter.

***NONE**
Do not generate the compiler listing.

Top

Optimization level (OPTIMIZE)

Specifies the level of optimization, if any, of the module.

***NONE**
Generated code is not optimized. This is the fastest in terms of translation time. It allows variables to be displayed and modified while in debug mode.

***BASIC**
Some optimization is performed on the generated code. This allows user variables to be displayed but not modified while in debug mode.

***FULL** Optimization which generates the most efficient code. Translation time is the longest. User variables may not be modified but may be displayed, although the presented values may not be the current values.

Top

Source listing indentation (INDENT)

Specifies whether structured operations should be indented in the source listing for enhanced readability. Also specifies the characters that are used to mark the structured operation clauses.

Note: Any indentation which you request here will not be reflected in the listing debug view which is created when you specify DBGVIEW(*LIST).

***NONE**
Structured operations will not be indented in the source listing.

character-value
The source listing is indented for structured operation clauses. Alignment of statements and clauses are marked using the characters you choose. You can choose any character string up to 2 characters in length. If you want to use a blank in your character string, you must enclose the string in single quotation marks.

Note: The indentation may not appear as expected if there are errors in the program.

Top

Type conversion options (CVTOPT)

Specifies how the ILE RPG compiler handles date, time, timestamp, graphic data types, and variable-length data types which are retrieved from externally described database files.

***NONE**
Ignores variable-length database data types and use the native RPG date, time, timestamp and graphic data types.

***DATETIME**
Specifies that date, time, and timestamp database data types are to be declared as fixed-length character fields.

***GRAPHIC**
Specifies that double-byte character set (DBCS) graphic data types are to be declared as fixed-length character fields.

***VARCHAR**
Specifies that variable-length character data types are to be declared as fixed-length character fields.

***VARGRAPHIC**
Specifies that variable-length double-byte character set (DBCS) graphic data types are to be declared as fixed-length character fields.

Top

Sort sequence (SRTSEQ)

Specifies the sort sequence table that is to be used in the ILE RPG source program.

***HEX** No sort sequence table is used.

***JOB** Use the SRTSEQ value for the job when the *PGM is created.

***JOBRUN**
Use the SRTSEQ value for the job when the *PGM is run.

***LANGIDUNQ**
Use a unique weighted table. This special value is used in conjunction with the LANGID parameter to determine the proper sort sequence table.

***LANGIDSHR**
Use a shared weighted table. This special value is used in conjunction with the LANGID parameter to determine the proper sort sequence table.

sort-table-name
Enter the qualified name of the sort sequence table to be used with the program.

***LIBL** The system searches the library list to find the library where the sort sequence table is stored.

***CURLIB**
The current library is used to find the sort sequence table. If you have not specified a current library, QGPL is used.

library-name
Enter the name of the library where the sort sequence table is stored.

Top

Language identifier (LANGID)

Specifies the language identifier to be used when the sort sequence is *LANGIDUNQ and *LANGIDSHR. The LANGID parameter is used in conjunction with the SRTSEQ parameter to select the sort sequence table.

***JOBRUN**
Use the LANGID value associated with the job when the RPG program is executed.

***JOB** Use the LANGID value associated with the job when the RPG program is created.

language-identifier
Use the language identifier specified. (For example, FRA for French and DEU for German.)

Top

Replace program (REPLACE)

Specifies if a new program is created when a program of the same name already exists in the specified (or implied) library. The intermediate modules created during the processing of the CRTBNDRPG command are not subject to the REPLACE specifications, and have an implied REPLACE(*NO) against the QTEMP library. The intermediate modules are deleted once the CRTBNDRPG command has completed processing.

***YES** A new program is created in the specified library. The existing program of the same name in the specified library is moved to library QRPLOBJ.

***NO** A new program is not created if a program of the same name already exists in the specified library. The existing program is not replaced, a message is displayed, and compilation stops.

Top

User profile (USRPRF)

Specifies the user profile that will run the created program object. The profile of the program owner or the program user is used to run the program and control which objects can be used by the program (including the authority the program has for each object). This parameter is not updated if the program already exists. To change the value of USRPRF, delete the program and recompile using the correct value (or, if the constituent *MODULE objects exist, you may choose to invoke the CRTPGM command).

***USER**
The program runs under the user profile of the program's user.

***OWNER**
The program runs under the user profile of both the program's user and owner. The collective set of object authority in both user profiles are used to find and access objects while the program is running. Any object created during the program are owned by the program's user.

Top

Authority (AUT)

Specifies the authority given to users who do not have specific authority to the object, who are not on the authorization list, and whose user group has no specific authority to the object. The authority can be altered for all or for specified users after the program is created with the CL commands Grant Object Authority (GRTOBJAUT) or Revoke Object Authority (RVKOBJAUT). For further information on these commands, see the CL concepts and reference topic in the iSeries Information Center at http://www.ibm.com/eserver/iseries/infocenter.

***LIBCRTAUT**
The public authority for the object is taken from the CRTAUT keyword of the target library (the library that contains the object). The value is determined when the object is created. If the CRTAUT value for the library changes after the create, the new value will not affect any existing objects.

***ALL** Authority for all operations on the program object except those limited to the owner or controlled by authorization list management authority. The user can control the program object's existence, specify this security for it, change it, and perform basic functions on it, but cannot transfer its ownership.

***CHANGE**
Provides all data authority and the authority to perform all operations on the program object except those limited to the owner or controlled by object authority and object management authority. The user can change the object and perform basic functions on it.

***USE** Provides object operational authority and read authority; authority for basic operations on the program object. The user is prevented from changing the object.

Note: A user must have *USE authority to a program to obtain a formatted dump of the variables of the program. To dump variables, the program must also have observable information.

If you do not want some users to be able to dump the variables, then give them only *OBJOPR plus *EXECUTE authority to the program. This will allow them to call the program but not dump its variables. Use EDTOBJAUT, GRTOBJAUT or RVKOBJAUT to change the authority of the created program.

If you do not want any users to be able to dump the variables, then use Change Program (CHGPGM) to remove the program's observable information.

***EXCLUDE**
The user is prevented from accessing the object.

authorization-list name
Enter the name of an authorization list of users and authorities to which the program is added. The program object will be secured by this authorization list, and the public authority for the program object will be set to *AUTL. The authorization list must exist on the system when the CRTBNDRPG command is issued.

Note: Use the AUT parameter to reflect the security requirements of your system. The security facilities available are described in detail in iSeries Security Reference, SC41-5302.

Top

Truncate numeric (TRUNCNBR)

Specifies if the truncated value is moved to the result field or if an error is generated when numeric overflow occurs while running the program.

Note: The TRUNCNBR option does not apply to calculations performed within expressions. (Expressions are found in the Extended-Factor 2 field.) If overflow occurs for these calculations, an error will always occur.

***YES** Ignore numeric overflow and move the truncated value to the result field.

***NO** When numeric overflow is detected, a run-time error is generated.

Top

Fix numeric (FIXNBR)

Specifies whether decimal data that is not valid is fixed by the compiler.

***NONE**
Indicates that decimal data that is not valid will result in decimal errors during run time if used.

***ZONED**
Zoned decimal data that is not valid will be fixed by the compiler on the conversion to packed data. Blanks in numeric fields will be treated as zeros. Each decimal digit will be checked for validity. If a decimal digit is not valid, it is replaced with zero. If a sign is not valid, the sign will be forced to a positive sign code of hex 'F'. If the sign is valid, it will be changed to either a positive sign hex 'F' or a negative sign hex 'D' as appropriate. If the resulting packed data is not valid, it will not be fixed.

***INPUTPACKED**
Indicates that if packed decimal data that is not valid is encountered while processing input specifications, the internal variable will be set to zero.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which you intend to use the object being created. In the examples given for the *CURRENT and *PRV values, and when specifying the target-release value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V2R3M0 is version 2, release 3, modification level 0.

Valid values for this parameter change every release. The possible values are:

***CURRENT**
The object is to be used on the release of the operating system currently running on your system. For example, if V2R3M5 is running on the system, *CURRENT means that you intend to use the object on a system with V2R3M5 installed. You can also use the object on a system with any subsequent release of the operating system installed.

Note: If V2R3M5 is running on the system, and the object is to be used on a system with V2R3M0 installed, specify TGTRLS(V2R3M0), not TGTRLS(*CURRENT).

***PRV** The object is to be used on the previous release with modification level 0 of the operating system. For example, if V2R3M5 is running on your system, *PRV means you intend to use the object on a system with V2R2M0 installed. You can also use the object on a system with any subsequent release of the operating system installed.

target-release
Specify the release in the format VxRxMx. You can use the object on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. If you specify a target-release that is earlier than the earliest release supported by this command, an error message is sent indicating the earliest supported release.

Note: The current version of the command may support options that are not available in previous releases of the command. If the command is used to create objects that are to be used on a previous release, it will be processed by the compiler appropriate to that release, and any unsupported options will not be recognized. The compiler will not necessarily issue any warnings regarding options that it is unable to process.

Top

Allow null values (ALWNULL)

Specifies how the ILE RPG program will be allowed to use records containing null-capable fields from externally described database files.

***NO** Specifies that the ILE RPG program will not process records with null-value fields from externally

described files. If you attempt to retrieve a record containing null values, no data in the record is accessible to the ILE RPG program and a data-mapping error occurs.

***INPUTONLY**
Specifies that the ILE RPG program can successfully read records with null-capable fields containing null values from externally described input-only database files. When a record containing null values is retrieved, no data mapping errors occur and the database default values are placed into any fields which contain null values. The program cannot do any of the following:

- use null-capable key fields
- create or update records containing null-capable fields
- determine whether a null-capable field is actually null while the program is running
- set a null-capable field to be null.

***USRCTL**
Specifies that the ILE RPG program can read, write and update records with null values from externally described database files. Records with null keys can be retrieved using keyed operations. The program can determine whether a null-capable field is actually null, and it can set a null-capable field to be null for output or update. The programmer is responsible for ensuring that fields containing null values are used correctly within the program.

***YES** Same as *INPUTONLY.

Top

Define condition names (DEFINE)

Specifies condition names that are defined before the compilation begins. Using the parameter DEFINE(condition-name) is equivalent to coding the /DEFINE condition-name directive on the first line of the source file.

***NONE**
No condition names are defined. This is the default.

condition-name
Up to 32 condition names can be specified. Each name can be up to 50 characters long. The condition names will be considered to be defined at the start of compilation.

Top

Enable performance collection (ENBPFRCOL)

Specifies whether performance collection is enabled.

***PEP** Performance statistics are gathered on the entry and exit of the program entry procedure only. This applies to the actual program-entry procedure for a program, not the main procedure of the modules within the program. This is the default.

***ENTRYEXIT**
Performance statistics are gathered on the entry and exit of all procedures of the program.

***FULL** Performance statistics are gathered on entry and exit of all procedures. Also, statistics are gathered before and after each call to an external procedure.

Top

Profiling data (PRFDTA)

Specifies the program profiling data attribute for the program. Program profiling is an advanced optimization technique used to reorder procedures and code within the procedures based on statistical data (profiling data).

***NOCOL**
This program is not enabled to collect profiling data. This is the default.

***COL** The program is enabled to collect profiling data. *COL can be specified only when the optimization level of the module is *FULL, and when compiling with a target release of *CURRENT.

Note: If you use the BNDDIR parameter to bind additional modules and service programs, these additional objects are not affected when *COL or *NOCOL is specified for the program. The program profiling data attribute for a module is set when the module is created.

Top

Licensed Internal Code options (LICOPT)

Specifies one or more Licensed Internal Code compile-time options. This parameter allows individual compile-time options to be selected, and is intended for the advanced programmer who understands the potential benefits and drawbacks of each selected type of compiler option.

Top

Include directory (INCDIR)

Specifies one or more directories to add to the search path used by the compiler to find copy files. The compiler will search the directories specified here if the copy files in the source program can not be resolved.

***NONE**
No user directories are searched for copy files. By default, the source directory will still be searched.

'directory'
Specify up to 32 directories in which to search for copy files. In addition to the specified directories, the source directory is also searched for copy files.

Top

Generate program interface (PGMINFO)

Specifies whether program interface information should be generated into a stream file. The possible values are:

***NO** This option specifies the default which does not generate program interface information.

***PCML**
Specifies that PCML (Program Call Markup Language) should be generated into a stream file. The generated PCML makes it easier for Java methods to call this RPG program, with less Java code. The name of a stream file that will contain the generated PCML must be specified on the INFOSTMF option.

Top

Program interface stream file (INFOSTMF)

Specifies the path name of the stream file to contain the generated program interface information specifed on the PGMINFO option.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'.

If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by appending the job's current working directory to the path name.

This parameter can only be specified when the PGMINFO parameter has a value other than *NO.

Top

Preprocessor options (PPGENOPT)

Specifies the preprocessor generation options to use when the source code is compiled.

The possible options are:

***NONE**
Run the entire compiler against the source file. Do not copy the preprocessor output to a file.

***DFT** Run the preprocessor against the input source. *RMVCOMMENT, *EXPINCLUDE and *NOSEQSRC will be used as the options for generating the preprocessor output. Use PPSRCFILE and PPSRCMBR to specify an output source file and member, or PPSRCSTMF to specify a stream file to contain the preprocessor output.

***RMVCOMMENT**
Remove comments, blank lines, and most directives during preprocessing. Retain only the RPG specifications and any directives necessary for the correct interpretation of the specifications.

***NORMVCOMMENT**
Preserve comments, blank lines and listing-control directives (for example /EJECT, /TITLE) during preprocessing. Transform source-control directives (for example /COPY, /IF) to comments during preprocessing.

***EXPINCLUDE**
Expand /INCLUDE directives in the generated output file.

***NOEXPINCLUDE**
/INCLUDE directives are placed unchanged in the generated output file.

Note: /COPY directives are always expanded.

***SEQSRC**
If PPSRCFILE is specified, the generated output member has sequential sequence numbers, starting at 000001 and incremented by 000001.

***NOSEQSRC**
If PPSRCFILE is specified, the generated output member has the same sequence numbers as the original source read by the preprocessor.

Top

Output source file (PPSRCFILE)

Specifies the source file name and library for the preprocessor output.

source-file-name
Specify the name of the source file for the preprocessor output.

The possible library values are:

***CURLIB**
The preprocessor output is created in the current library. If a job does not have a current library, the preprocessor output file is created in the QGPL library.

library-name
Specify the name of the library for the preprocessor output.

Top

Output source member (PPSRCMBR)

Specifies the name of the source file member for the preprocessor output.

***PGM** The name supplied on the PGM parameter is used as the preprocessor output member name.

member-name
Specify the name of the member for the preprocessor output.

Top

Output stream file (PPSRCSTMF)

Specifies the path name of the stream file for the preprocessor output.

***SRCSTMF**
The path name supplied on the SRCSTMF parameter is used as the preprocessor output path name. The file will have the extension '.i'.

'path-name'
Specify the path name for the preprocessor output stream file.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'.

If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by appending the job's current working directory to the path name.

Top

Examples

Example 1: Compiling a Source Program into a Program Object

```
CRTBNDRPG   PGM(MYLIB/XMPLE1)
            SRCFILE(MYLIB/QRPGSRC)  SRCMBR(XMPLE1)
            OUTPUT(*PRINT)  TEXT('My RPG IV Program')
```

This command calls the compiler for ILE RPG to create a program named XMPLE1. The source program is in member XMPLE1 of source file QRPGLESRC in library MYLIB. A compiler listing is created.

Top

Error messages

***ESCAPE Messages**

RNS9310
Compilation failed. Program &1 not created in library &2.

Top

Create COBOL Module (CRTCBLMOD)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The CRTCBLMOD command compiles an ILE COBOL source program into a module. You can use this command interactively, in batch mode, or in a CL program.

All object names specified for the CRTCBLMOD command must follow server naming conventions.

A description of the parameters for the CRTCBLMOD command follows.

Top

Parameters

Keyword	Description	Choices	Notes
MODULE	Module	*Qualified object name*	Optional, Positional 1
	Qualifier 1: Module	*Name*, ***PGMID**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QCBLLESRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***MODULE**	Optional, Positional 3
SRCSTMF	Source stream file	*Path name*	Optional
GENLVL	Generation severity level	0-30, **30**	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OUTPUT	Output	***PRINT**, *NONE	Optional, Positional 4
OPTION	Compiler options	Values (up to 50 repetitions): *SOURCE, *NOSOURCE, *SRC, *NOSRC, *NOXREF, *XREF, *GEN, *NOGEN, *NOSEQUENCE, *SEQUENCE, *NOVBSUM, *VBSUM, *NONUMBER, *NUMBER, *LINENUMBER, *NOMAP, *MAP, *NOOPTIONS, *OPTIONS, *QUOTE, *APOST, *NOSECLVL, *SECLVL, *PRTCORR, *NOPRTCORR, *MONOPRC, *NOMONOPRC, *RANGE, *NORANGE, *NOUNREF, *UNREF, *NOSYNC, *SYNC, *NOCRTF, *CRTF, *NODUPKEYCHK, *DUPKEYCHK, *NOINZDLT, *INZDLT, *NOBLK, *BLK, *STDINZ, *NOSTDINZ, *STDINZHEX00, *NODDSFILLER, *DDSFILLER, *NOIMBEDERR, *IMBEDERR, *STDTRUNC, *NOSTDTRUNC, *NOCHGPOSSGN, *CHGPOSSGN, *NOEVENTF, *EVENTF, *MONOPIC, *NOMONOPIC, *NOCRTARKIDX, *CRTARKIDX	Optional, Positional 5
CVTOPT	Conversion options	Values (up to 8 repetitions): *NOVARCHAR, *VARCHAR, *NODATETIME, *DATETIME, *NOPICXGRAPHIC, *PICXGRAPHIC, *NOPICGGRAPHIC, *PICGGRAPHIC, *NOFLOAT, *FLOAT, *NODATE, *DATE, *NOTIME, *TIME, *NOTIMESTAMP, *TIMESTAMP, *NOCVTTODATE, *CVTTODATE, *NOPICNGRAPHIC, *PICNGRAPHIC	Optional

Keyword	Description	Choices	Notes
MSGLMT	Message limit	*Element list*	Optional
	Element 1: Number of messages	0-9999, ***NOMAX**	
	Element 2: Message limit severity	0-30, **30**	
DBGVIEW	Debug view option	*Element list*	Optional
	Element 1: Debug view	***STMT**, *SOURCE, *LIST, *ALL, *NONE	
	Element 2: Compress listing view	***NOCOMPRESSDBG**, *COMPRESSDBG	
OPTIMIZE	Optimize level	***NONE**, *BASIC, *FULL	Optional
FLAGSTD	FIPS flagging	Values (up to 2 repetitions): *NOFIPS, *MINIMUM, *INTERMEDIATE, *HIGH, *NOOBSOLETE, *OBSOLETE	Optional
EXTDSPOPT	Extended display options	Values (up to 3 repetitions): *DFRWRT, *NODFRWRT, *UNDSPCHR, *NOUNDSPCHR, *ACCUPDALL, *ACCUPDNE	Optional
FLAG	Flagging severity	0-99, **0**	Optional
REPLACE	Replace module	***YES**, *NO	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
LINKLIT	Link literal	***PGM**, *PRC	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
SRTSEQ	Sort sequence	Single values: ***HEX**, *JOB, *JOBRUN, *LANGIDUNQ, *LANGIDSHR Other values: *Qualified object name*	Optional
	Qualifier 1: Sort sequence	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
LANGID	Language id	*Character value*, ***JOBRUN**, *JOB	Optional
ENBPFRCOL	Enable performance collection	Single values: ***PEP** Other values: *Element list*	Optional
	Element 1: Collection level	***FULL**, *ENTRYEXIT	
PRFDTA	Profiling data	***NOCOL**, *COL	Optional
CCSID	Coded character set ID	*Integer*, ***JOBRUN**, *HEX, *JOB	Optional
ARITHMETIC	Arithmetic mode	***NOEXTEND**, *EXTEND31, *EXTEND63	Optional
NTLPADCHAR	Padding character	*Element list*	Optional
	Element 1: Single byte to national	*Character value*, ***DEFAULT**	
	Element 2: Double byte to national	*Character value*, ***DEFAULT**	
	Element 3: National to national	*Character value*, ***DEFAULT**	
LICOPT	Licensed Internal Code options	*Character value*	Optional
INCDIR	Include directory	Values (up to 32 repetitions): *Path name*, ***NONE**	Optional
PGMINFO	Generate program interface	***NO**, *PCML	Optional
INFOSTMF	Program interface stream file	*Path name*	Optional

Top

Module (MODULE)

Specifies the module name and library name for the module object you are creating. The module name and library name must conform to server naming conventions. The possible values are:

***PGMID**
The name for the module is taken from the PROGRAM-ID paragraph in the outermost ILE COBOL source program of the compilation unit.

module-name
Enter a name to identify the compiled ILE COBOL module. If you specify a module name for this parameter, and compile a sequence of source programs (multiple compilation units in a single source file member) the first module in the sequence uses this name; any other modules use the name specified in the PROGRAM-ID paragraph in the corresponding outermost ILE COBOL source program of the compilation unit.

The possible library values are:

***CURLIB**
The created module object is stored in the current library. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the created module object is to be stored.

Top

Source file (SRCFILE)

Specifies the name of the source file and library that contains the ILE COBOL source code to be compiled. This source file should have a record length of 92. The possible values are:

QCBLLESRC
Specifies that the source file, QCBLLESRC, contains the ILE COBOL source code to be compiled.

source-file-name
Enter the name of the source file that contains the ILE COBOL source code to be compiled.

The possible library values are:

***LIBL** The library list is searched to find the library where the source file is located.

***CURLIB**
The current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the source file is located.

Top

Source member (SRCMBR)

Specifies the name of the member that contains the ILE COBOL source code to be compiled. You can specify this parameter only if the source file referred to in the SRCFILE parameter is a database file. The possible values are:

***MODULE**
The source file member with the same name as the module name specified on the MODULE parameter, is used.

If you do not specify a module name for the MODULE parameter, the first member of the database source file is used.

source-file-member-name
Enter the name of the member that contains the ILE COBOL source code.

Top

Source stream file (SRCSTMF)

Specifies the path name of the stream file containing the ILE COBOL source code to be compiled.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'.

If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by appending the job's current working directory to the path name.

The SRCMBR and SRCFILE parameters cannot be specified with the SRCSTMF parameter.

Top

Generation severity level (GENLVL)

Specifies the severity level that determines if a module object is created. The severity level corresponds to the severity level of the messages produced during compilation. This parameter applies individually to each compilation unit in a source file member. Other compilation units in the source file member will still be compiled even if a previous compilation unit fails.

The possible values are:

30 No module object is created if errors occur with a severity level equal to or greater than 30.

severity-level
Specify a one or two-digit number, 0 through 30, which is the severity level you want to use to determine if a module object is to be created. No module object is created if errors occur with a severity level equal to or greater than this severity level.

Top

Text 'description' (TEXT)

Allows you to enter text that briefly describes the module and its function.

***SRCMBRTXT**
The same text that describes the database file member containing the ILE COBOL source code, is used to describe the module object. If the source comes from a device or inline file, specifying *SRCMBRTXT has the same effect as specifying *BLANK.

***BLANK**
No text is specified.

text-description
Enter text briefly describing the module and its function. The text can be a maximum of 50 SBCS characters in length and must be enclosed in single quotation marks. The single quotation marks are not part of the 50-character string.

Top

Output (OUTPUT)

Specifies if the compiler listing is generated or not. The possible values are:

***PRINT**
A compiler listing is generated. If a member is being compiled, the output file has the same name as the member. If a stream file is being compiled and *PGMID is specified in the PGM parameter, the output file has the name COBOLPGM00. Otherwise, it has the same name as the program.

***NONE**
No compiler listing is generated.

Top

Compiler options (OPTION)

Specifies the options to use when the ILE COBOL source code is compiled.

Options specified in the PROCESS statement of an ILE COBOL source program override the corresponding options of the OPTION parameter.

The possible values of the OPTION parameter are:

***SOURCE or *SRC**
The compiler produces a source listing, consisting of the ILE COBOL source program and all compilation-time error messages.

***NOSOURCE or *NOSRC**
The compiler does not produce the source part of the listing. If you do not require a source listing, you should use this option because compilation may take less time.

***NOXREF**
The compiler does not produce a cross-reference listing for the ILE COBOL source program.

***XREF** The compiler produces a cross-reference listing for the ILE COBOL source program.

***GEN** The compiler creates a module object after the ILE COBOL source is compiled.

***NOGEN**
The compiler does not create a module object after the ILE COBOL source program is compiled. You might specify this option if you want only error messages or listings.

***NOSEQUENCE**
The reference numbers are not checked for sequence errors.

***SEQUENCE**
The reference numbers are checked for sequence errors. Sequence errors do not occur if the *LINENUMBER option is specified.

***NOVBSUM**
Verb usage counts are not printed.

***VBSUM**
Verb usage counts are printed.

***NONUMBER**
The source-file sequence numbers are used for reference numbers.

***NUMBER**
The user-supplied sequence numbers (columns 1 through 6) are used for reference numbers.

***LINENUMBER**
The sequence numbers created by the compiler are used for reference numbers. This option combines ILE COBOL program source code and source code introduced by COPY statements into one consecutively numbered sequence. Use this option if you specify FIPS (Federal Information Processing Standards) flagging or SAA flagging.

***NOMAP**
The compiler does not list the Data Division map.

***MAP** The compiler lists the Data Division map.

***NOOPTIONS**
Options in effect are not listed for this compilation.

***OPTIONS**
Options in effect are listed for this compilation.

***QUOTE**
Specifies that the delimiter quotation mark (") is used for nonnumeric literals, hexadecimal literals, and Boolean literals. This option also specifies that the value of the figurative constant QUOTE has the EBCDIC value of a quotation mark.

***APOST**
Specifies that the delimiter apostrophe (') is used for nonnumeric literals, hexadecimal literals, and Boolean literals. This option also specifies that the value of the figurative constant QUOTE has the EBCDIC value of an apostrophe.

***NOSECLVL**
Second level message text is not listed for this compilation.

***SECLVL**
Second level message text is listed for this compilation, along with the first-level error text, in the message section of the compiler listing.

***PRTCORR**
Comment lines are inserted in the compiler listing indicating which elementary items were included as a result of the use of the CORRESPONDING phrase.

***NOPRTCORR**
Comment lines are not inserted in the compiler listing when the CORRESPONDING phrase is used.

***MONOPRC**
The program-name (literal or word) found in the PROGRAM-ID paragraph, the CALL, CANCEL, or SET ENTRY statements, and the END PROGRAM header is converted to all upper-case characters (monocasing) and the rules for program-name formation are enforced.

***NOMONOPRC**
The program-name (literal or word) found in the PROGRAM-ID paragraph, the CALL, CANCEL, or SET ENTRY statements, and the END PROGRAM header is not converted to all upper-case characters (no monocasing) and the rules for program-name formation are not enforced. This option allows special characters not allowed for standard COBOL to be used in the CALL target.

***RANGE**
At run time, subscripts are verified to ensure they are within the correct ranges, but index ranges are not verified. Reference modification and compiler-generated substring operations are also checked.

The contents of date-time items are checked to make sure their format is correct, and that they represent a valid date, time, or timestamp.

***NORANGE**
Ranges are not verified at run time.

Note: The *RANGE option generates code for checking subscript ranges. For example, it ensures that you are not attempting to access element 21 of a 20-element array.

The *NORANGE option does not generate code to check subscript ranges. As a result, the *NORANGE option produces faster running code.

***NOUNREF**
Unreferenced data items are not included in the compiled module. This reduces the amount storage used, allowing a larger program to be compiled. You cannot look at or assign to an unreferenced data item during debugging when the *NOUNREF option is chosen. The unreferenced data items still appear in the cross-reference listings produced by specifying OPTION (*XREF).

***UNREF**
Unreferenced data items are included in the compiled module.

***NOSYNC**
The SYNCHRONIZED clause is syntax checked only.

***SYNC**
The SYNCHRONIZED clause is compiled by the compiler. The SYNCHRONIZED clause causes the position of a data item to be aligned such that the right-hand (least-significant) end is on the natural storage boundary. The natural storage boundary is the next nearest 4-byte, 8-byte, or 16-byte boundary in storage depending on the length and type of data being stored. Extra storage is reserved adjacent to the synchronized item to achieve this alignment. Each elementary data item that is described as SYNCHRONIZED is aligned to the natural storage boundary that corresponds to its data storage assignment.

***NOCRTF**
Disk files that are unavailable at the time of an OPEN operation are not created dynamically.

***CRTF**
Disk files that are unavailable at the time of an OPEN operation are created dynamically.

Note: The maximum record length for a file that will be created dynamically is 32&rbl.766. Indexed files will not be dynamically created even though the *CRTF option has been specified.

***NODUPKEYCHK**
Does not check for duplicate primary and alternate record keys for INDEXED files.

***DUPKEYCHK**
Checks for duplicate primary and alternate record keys for INDEXED files.

***NOINZDLT**
Relative files with sequential access are not initialized with deleted records during the CLOSE operation if the files have been opened for OUTPUT. The record boundary is determined by the number of records written at OPEN OUTPUT time. Subsequent OPEN operations allow access only up to the record boundary.

***INZDLT**
Relative files with sequential access are initialized with deleted records during the CLOSE operation if the files were opened for OUTPUT. Active records in the files are not affected. The record boundary is defined as the file size for subsequent OPEN operations.

***NOBLK**
The compiler allows blocking only of SEQUENTIAL access files with no START statement. The BLOCK CONTAINS clause, if specified, is ignored, except for tape files.

***BLK** When *BLK is used and a BLOCK CONTAINS clause is specified, the compiler allows blocking

for DYNAMIC access files and SEQUENTIAL access files with a START statement. Blocking is not allowed for RELATIVE files opened for output operations. The BLOCK CONTAINS clause controls the number of records to be blocked.

When *BLK is used and no BLOCK CONTAINS clause is specified, the compiler allows blocking only of SEQUENTIAL access files with no START statement. The operating system determines the number of records to be blocked.

***STDINZ**
For those items with no VALUE clause, the compiler initializes data items to system defaults.

***NOSTDINZ**
For those items with no VALUE clause, the compiler does not initialize data items to system defaults.

***STDINZHEX00**
For those items with no VALUE clause, the compiler initializes data items to hexadecimal zero.

***NODDSFILLER**
If no matching fields are found by a COPY DDS statement, no field descriptions are generated.

***DDSFILLER**
If no matching fields are found by a COPY DDS statement, a single character FILLER field description, ″07 FILLER PIC X″, is always created.

***NOIMBEDERR**
Error messages are not included in the source listing section of the compiler listing. Error messages only appear in the error message section of the compiler listing.

***IMBEDERR**
First level error messages are included in the source listing section of the compiler listing, immediately following the line where the error occurred. Error messages also appear in the error message section of the compiler listing.

***STDTRUNC**
This option applies only to USAGE BINARY data. When *STDTRUNC is selected, USAGE BINARY data is truncated to the number of digits in the PICTURE clause of the BINARY receiving field.

***NOSTDTRUNC**
This option applies only to USAGE BINARY data. When *NOSTDTRUNC is selected, BINARY receiving fields are truncated only at half-word, full-word, or double-word boundaries. BINARY sending fields are also handled as half-words, full-words, or double-words. Thus, the full binary content of the field is significant. Also, the DISPLAY statement will convert the entire content of a BINARY field, with no truncation.

***NOCHGPOSSGN**
Hexadecimal F is used as the default positive sign for zoned and packed numeric data. Hexadecimal F is the system default for the operating system.

***CHGPOSSGN**
Hexadecimal C is used as the default positive sign for zoned and packed numeric data. This applies to all results of the MOVE, ADD, SUBTRACT, MULTIPLY, DIVIDE, COMPUTE, and INITIALIZE statements, as well as the results of the VALUE clause.

***NOEVENTF**
Do not create an Event File for use by CoOperative Development Environment/400 (CODE/400). CODE/400 uses this file to provide error feedback integrated with the CODE/400 editor. An Event File is normally created when you create a module or program from within CODE/400.

***EVENTF**
Create an Event File for use by CoOperative Development Environment/400 (CODE/400). The Event File is created as a member in file EVFEVENT in the library where the created module or

program object is to be stored. If the file EVFEVENT does not exist it is automatically created. The Event File member name is the same as the name of the object being created.

CODE/400 uses this file to provide error feedback integrated with the CODE/400 editor. An Event File is normally created when you create a module or program from within CODE/400.

***MONOPIC**
The PICTURE character-string is converted to all uppercase characters (monocasing).

***NOMONOPIC**
The currency symbol used in the PICTURE character-string is case sensitive. That is, the lowercase letters corresponding to the uppercase letters for the PICTURE symbols A, B, E, G, N, P, S, V, X, Z, CR, and DB are equivalent to their uppercase representations in a PICTURE character-string. All other lowercase letters are not equivalent to their corresponding uppercase representations.

***NOCRTARKIDX**
Temporary alternate record key (ARK) indexes are not created if permanent ones cannot be found.

***CRTARKIDX**
Temporary alternate record key (ARK) indexes are created if permanent ones cannot be found.

Top

Conversion options (CVTOPT)

Specifies how the compiler handles date, time, and timestamp field types, DBCS field types, variable-length character field types, and floating-point field types passed from externally-described files to your program through COPY DDS. The possible values are:

***NOVARCHAR**
Variable-length fields are declared as FILLER fields.

***VARCHAR**
Variable-length fields are declared as group items, and are accessible to the ILE COBOL source program.

***NODATETIME**
Date, time, and timestamp data types are declared as FILLER fields.

***DATETIME**
Date, time, and timestamp DDS data types are given COBOL data item names based on their DDS names. The category of the COBOL data item is alphanumeric, unless one of the CVTOPT parameter values *DATE, *TIME, or *TIMESTAMP is specified. In this case, the category of the COBOL data item is date, time, or timestamp, respectively.

***NOPICXGRAPHIC**
DBCS-graphic data types are declared as FILLER fields.

***PICXGRAPHIC**
Fixed-length DBCS-graphic data types are declared as fixed-length alphanumeric fields, and are accessible to the ILE COBOL source program.

When the *VARCHAR option is also in use, variable-length DBCS-graphic data types are declared as fixed-length group items, and are accessible to the ILE COBOL source program.

***PICGGRAPHIC**
Fixed-length DBCS-graphic data types are declared as fixed-length G-type fields, and are accessible to the ILE COBOL source program.

When the *VARCHAR option is also in use, variable-length DBCS-graphic data types are declared as fixed-length group items (made of a numeric field followed by G type field), and are accessible to the ILE COBOL source program.

***NOPICGGRAPHIC**
DBCS-graphic data types are declared as FILLER fields.

***NOFLOAT**
Floating-point data types are declared as FILLER fields with a USAGE of binary.

***FLOAT**
Floating-point data types are brought into the program with their DDS names and a USAGE of COMP-1 (single-precision) or COMP-2 (double-precision). The fields are made accessible to the ILE COBOL source program.

***NODATE**
DDS date data types are declared as category alphanumeric COBOL data items, for example:

```
06 FILLER PIC X(10).
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***DATE**
DDS date data types are declared as category date COBOL data items, for example:

```
06 FILLER FORMAT DATE '@Y-%m-%d'.
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***NOTIME**
DDS time data types are declared as category alphanumeric COBOL data items, for example:

```
06 FILLER PIC X(8).
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***TIME**
DDS time data types are declared as category time COBOL data items, for example:

```
06 FILLER FORMAT TIME '%H:%M:%S'.
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***NOTIMESTAMP**
DDS timestamp data types are declared as category alphanumeric COBOL data items, for example:

```
06 FILLER PIC X(26).
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***TIMESTAMP**
DDS timestamp data types are declared as category timestamp COBOL data items, for example:

```
06 FILLER FORMAT TIMESTAMP.
```

The COBOL data item name is determined by the *NODATETIME/*DATETIME CVTOPT parameter.

***NOCVTTODATE**
DDS data types with the DATFMT keyword (excluding the DDS date data type) are declared in ILE COBOL based on their original DDS type.

***CVTTODATE**
DDS data types with the DATFMT keyword (excluding the DDS date data type) are declared in ILE COBOL as date data types.

***NOPICNGRAPHIC**
DBCS-graphic data types are declared as FILLER fields.

***PICNGRAPHIC**
Fixed-length DBCS-graphic data types are declared as fixed-length national data fields, and are accessible to the ILE COBOL source program.

Top

Message limit (MSGLMT)

Specifies the maximum number of messages of a given error severity level that can occur for each compilation unit before compilation stops. As soon as one compilation unit reaches the maximum, compilation stops for the entire source member.

For example, if you specify 3 for the maximum number of messages and 20 for the error severity level then compilation will stop if three or more errors with a severity level of 20 or higher occur. If no messages equal or exceed the given error severity level, compilation continues regardless of the number of errors encountered.

number-of-messages
Specifies the maximum number of messages. The possible values are:

***NOMAX**
Compilation continues until normal completion regardless of the number of errors encountered.

maximum-number
Specifies the maximum number of messages that can occur at or above the specified error severity level before compilation stops. The valid range is 0-9999.

message-limit-severity
Specifies the error severity level used to determine whether or not to stop compilation. The possible values are:

30 Compilation stops if the number of errors with severity level 30 or higher exceeds the maximum number of messages specified.

error-severity-level
Enter a one or two-digit number, 0 through 30, which is the error severity level you want to use to determine whether or not to stop compilation. Compilation stops if the number of errors with this severity level or higher exceeds the maximum number of messages you specified.

Top

Debug view option (DBGVIEW)

Specifies options that control which views of the source program or generated listing is available for debugging the compiled module, and if the debug listing view is compressed or not.

debug-view
Specify the views to be available for debugging. The possible values are:

***STMT**
The compiled module can be debugged using symbolic names and statement numbers.

***SOURCE**
The primary source member, as well as copied source members which were included through COPY statements, will have source views available for debugging the compiled module. These views are available only if the primary source member and copied source members come from local database source files. Do not change or delete members during the time between compile and debug.

***LIST** A listing view, which shows the source code after the processing of any COPY and REPLACE statements, will be made available for debugging the compiled module. This option increases the size of the compiled module, without affecting the runtime performance of the compiled module.

The listing view will include the cross-reference listing, Data Division map, and verb usage counts when the corresponding compiler options are requested. For example, a cross-reference listing will be included if OPTION(*XREF) is specified.

Listing views can be generated regardless of where the primary source members or copied source members come from. Listing views are not affected by changes to or deletion of the source members following the compilation.

***ALL** Equivalent to specifying *STMT, *SOURCE, and *LIST combined.

***NONE**
The compiled module cannot be debugged. This reduces the size of the compiled program, but does not affect its runtime performance. When this option is specified, a formatted dump can not be taken.

compress-listing-view
Specifies if the listing view is compressed or not when *LIST or *ALL is specified in debug-view. The possible values are:

***NOCOMPRESSDBG**
The listing view is not compressed.

***COMPRESSDBG**
The listing view is compressed when *LIST or *ALL is specified in debug-view.

Top

Optimize level (OPTIMIZE)

Specifies the level of optimization of the module. The possible values are:

***NONE**
No optimization is performed on the compiled module. Compilation time is minimized when this option is used. This option allows variables to be displayed and changed during debugging.

***BASIC**
Some optimization (only at the local block level) is performed on the compiled module. This option allows user variables to be displayed but not changed during debugging.

***FULL** Full optimization (at the global level) is performed on the compiled module. This optimization increases compilation time but also generates the most efficient code. This option allows user variables to be displayed but not changed during debugging. The displayed values of the variables may not be their current values. Some variables may not be displayable.

Note: Regardless of the optimization level chosen, all information to allow full optimization is generated. The user can change optimization levels of the module object from *NONE to *FULL using the CHGMOD command without having to recompile the source program.

Top

FIPS flagging (FLAGSTD)

Specifies the options for FIPS flagging. (Select the *LINENUMBER option to ensure that the reference numbers used in the FIPS messages are unique.) The possible values are:

***NOFIPS**
The ILE COBOL source program is not FIPS flagged.

***MINIMUM**
FIPS flag for minimum subset and higher.

***INTERMEDIATE**
FIPS flag for intermediate subset and higher.

***HIGH**
FIPS flag for high subset.

***NOOBSOLETE**
Obsolete language elements are not flagged.

***OBSOLETE**
Obsolete language elements are flagged.

Top

Extended display options (EXTDSPOPT)

Specifies the options to use for extended ACCEPT and extended DISPLAY statements for workstation I/O. The possible values are:

***DFRWRT**
Extended DISPLAY statements are held in a buffer until an extended ACCEPT statement is encountered, or until the buffer is filled.

The contents of the buffer are written to the display when the extended ACCEPT statement is encountered or the buffer is full.

***NODFRWRT**
Each extended DISPLAY statement is performed as it is encountered.

***UNDSPCHR**
Displayable and undisplayable characters are handled by extended ACCEPT and extended DISPLAY statements.

***NOUNDSPCHR**
Only displayable characters are handled by extended ACCEPT and extended DISPLAY statements.

Although you must use this option for display stations attached to remote 3174 and 3274 controllers, you can also use it for local workstations. If you do use this option, your data must contain displayable characters only. If the data contains values less than hexadecimal 20, the results are not predictable, ranging from unexpected display formats to severe errors.

***ACCUPDALL**
All types of data are predisplayed in the extended ACCEPT statements regardless of the existence of the UPDATE phrase.

***ACCUPDNE**
Only numeric edited data are predisplayed in the extended ACCEPT statements that do not contain the UPDATE phrase.

Top

Flagging severity (FLAG)

Specifies the minimum severity level of messages that will appear in the compiler listing. The possible values are:

0 All messages will appear in the compiler listing.

severity-level
Enter a one or two-digit number that specifies the minimum severity level of messages that you want to appear in the compiler listing. Messages that have severity levels of this specified value or higher will appear in the compiler listing.

Top

Replace module (REPLACE)

Specifies if a new module is created when a module of the same name in the specified or implied library already exists. The possible values are:

***YES** A new module is created and it replaces any existing module of the same name in the specified or implied library. The existing module of the same name in the specified or implied library is moved to library QRPLOBJ.

***NO** A new module is not created if a module of the same name already exists in the specified or implied library. The existing module is not replaced, a message is displayed, and compilation stops.

Top

Authority (AUT)

Specifies the authority given to users who do not have specific authority to the module object, who are not on the authorization list, or whose group has no specific authority to the module object. You can change the authority for all users, or for specific users after the module object is created by using the GRTOBJAUT (Grant Object Authority) or RVKOBJAUT (Revoke Object Authority) commands.

The possible values are:

***LIBCRTAUT**
The public authority for the object is taken from the CRTAUT keyword of the target library (the library that is to contain the created module object). This value is determined when the module object is created. If the CRTAUT value for the library changes after the module object is created, the new value does NOT affect any existing objects.

***ALL** Provides authority for all operations on the module object except those limited to the owner or controlled by authorization list management authority. The user can control the module object's existence, specify security for it, change it, and perform basic functions on it, but cannot transfer its ownership.

***CHANGE**
Provides all data authority and the authority for performing all operations on the module object except those limited to the owner or controlled by object authority and object management authority. The user can change the object and perform basic functions on it.

***USE** Provides object operational authority and read authority; authority for basic operations on the module object. The user can perform basic operations on the object but is prevented from changing the object.

Note: A user must have *USE authority to a program to obtain a formatted dump of the variables of the program. To dump variables, the program must also have observable information.

If you do not want some users to be able to dump the variables, then give them only *OBJOPR plus *EXECUTE authority to the program or service program containing the module. This will allow them to call the program but not dump its variables. Use EDTOBJAUT, GRTOBJAUT or RVKOBJAUT to change the authority of the created program or service program.

If you do not want any users to be able to dump the variables, then remove the observable information using Change Module (CHGMOD) on the module, or using Change Program (CHGPGM) or Change Service Program (CHGSRVPGM) on the program containing the module.

***EXCLUDE**
The user cannot access the module object.

authorization-list-name
The name of an authorization list of users and authorities to which the module is added. The module object is secured by this authorization list, and the public authority for the module object is set to *AUTL. The authorization list must exist on the system when the CRTCBLMOD command is issued. Use the Create Authorization List (CRTAUTL) command to create your own authorization list.

Top

Link literal (LINKLIT)

Specifies the linkage type for external CALL/CANCEL 'literal' target and the SET ENTRY target. You may override this option for specific external CALL/CANCEL 'literal' target and the SET ENTRY target lists by specifying the following sentence in the SPECIAL-NAMES paragraph:

```
LINKAGE TYPE IS implementer-name FOR target-list.
```

The possible values for LINKLIT are:

***PGM** Target for CALL/CANCEL or SET ENTRY is a program object.

***PRC** Target for CALL/CANCEL or SET ENTRY is an ILE procedure.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which you intend to use the module object being created. In the examples given for the *CURRENT and *PRV values, and when specifying the target-release value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V2R3M0 is version 2, release 3, modification level 0.

Valid values for this parameter change every release. The possible values are:

***CURRENT**
The object is to be used on the release of the operating system currently running on your system. For example, if V2R3M5 is running on the system, *CURRENT means that you intend to use the object on a system with V2R3M5 installed. The object can also be used on a system with any subsequent release of the operating system installed.

Note: If V2R3M5 is running on the system, and the object is to be used on a system with V2R3M0 installed, specify TGTRLS(V2R3M0), not TGTRLS(*CURRENT).

***PRV** The object is to be used on the previous release with modification level 0 of the operating system. For example, if V2R3M5 is running on your system, *PRV means you intend to use the object on a system with V2R2M0 installed. The object can also be used on a system with any subsequent release of the operating system installed.

target-release
Specify the release in the format VxRxMx. The object can be used on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. If you specify a target-release that is earlier than the earliest release level supported by this command, an error message is sent indicating the earliest supported release.

Note: The current version of the command may support options that are not available in previous releases of the command. If the command is used to create objects that are to be used on a previous release, it will be processed by the compiler appropriate to that release, and any unsupported options will not be recognized. The compiler will not necessarily issue any warnings regarding options that it is unable to process.

Top

Sort sequence (SRTSEQ)

Specifies the sort sequence used when NLSSORT is associated with an alphabet-name in the ALPHABET clause. The SRTSEQ parameter is used in conjunction with the LANGID parameter to determine which system-defined or user-defined sort sequence table the module will use. The possible values are:

***HEX** No sort sequence table will be used, and the hexadecimal values of the characters will be used to determine the sort sequence.

***JOB** The sort sequence will be resolved and associated with the module at compile time using the sort sequence of the compile job. The sort sequence table must exist in the system at compile time. If at run time, the CCSID of the runtime job differs from the CCSID of the compile time job, the sort sequence table loaded at compile time is converted to match the CCSID of the runtime job.

***JOBRUN**
The sort sequence of the program will be resolved and associated with the module at run time. At compile time, the compiler will associate the sort sequence of the compile job with the module. At run time, this sort sequence will be replaced by the sort sequence associated with the job at run time. This value allows a module to be compiled once and used with different sort sequences at run time.

***LANGIDUNQ**
Specifies that the sort sequence table being used must contain a unique weight for each character in the code page. The sort sequence table used will be the unique weighted table associated with the language specified in the LANGID parameter.

***LANGIDSHR**
Specifies that the sort sequence table being used can contain the same weight for multiple characters in the code page. The sort sequence table used will be the shared weighted table associated with the language specified in the LANGID parameter.

table-name
Enter the name of the sort sequence table to be used. The table contains weights for all characters in a given code page. A weight is associated with the character that is defined at the code point. When using a sort sequence table name, the library in which the object resides can be specified. The valid values for the library are:

***LIBL** The library list is searched to find the library where the sort sequence table is located.

***CURLIB**
The current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the sort sequence table is found.

Top

Language id (LANGID)

Specifies the language identifier which is used in conjunction with the sort sequence. The LANGID parameter is used only when the SRTSEQ value in effect is *LANGIDUNQ or *LANGIDSHR. The possible values are:

***JOBRUN**
The language identifier of the program will be resolved at run time. When the compiled program is run, the language identifier of the job is used. This value allows a module to be compiled once and used with different language identifiers at run time.

***JOB** The language identifier of the module will be resolved at compile time.

language-identifier-name
Enter a valid 3-character language identifier.

Top

Enable performance collection (ENBPFRCOL)

Specifies whether performance measurement code should be generated in the module or program. The data collected can be used by the system performance tool to profile an application's performance. Generating the addition of the performance measurement code in a compiled module or program will result in slightly larger objects and may affect performance.

***PEP** Performance statistics are gathered on the entry and exit of the program entry procedure only. Choose this value when you want to gather overall performance information for an application. This support is equivalent to the support formally provided with the TPST tool. This is the default.

***ENTRYEXIT**
Performance statistics are gathered on the entry and exit of all the procedures of the program. This includes the program PEP routine.

This choice would be useful if you want to capture information on all routines. Use this option when you know that all the programs called by your application were compiled with either the *PEP, *ENTRYEXIT or *FULL option. Otherwise, if your application calls other programs that are not enabled for performance measurement, the performance tool will charge their use of resources against your application. This would make it difficult for you to determine where resources are actually being used.

***FULL** Performance statistics are gathered on the entry and exit of all procedures. Also statistics are gathered before and after each call to an external procedure.

Use this option when you think that your application will call other programs that were not compiled with either *PEP, *ENTRYEXIT or *FULL. This option allows the performance tools to distinguish between resources that are used by your application and those used by programs it calls (even if those programs are not enabled for performance measurement). This option is the most expensive but allows for selectively analyzing various programs in an application.

Top

Profiling data (PRFDTA)

Specifies the program profiling data attribute for the module. Program profiling is an advanced optimization technique used to reorder procedures and code within the procedures based on statistical data (profiling data).

***NOCOL**
This module is not enabled to collect profiling data. This is the default.

***COL** This module is enabled to collect profiling data.

Note: *COL can be specified only when the optimization level of the module is *FULL.

Top

Coded character set ID (CCSID)

Specifies the coded character set identifier (CCSID) that records in files, and data associated with LOCALEs, are converted to at run time.

***JOBRUN**
The CCSID of the program is resolved at run time. When the compiled program is run, the current job's default CCSID is used.

***JOB** The current job's default CCSID at compile time is used.

***HEX** The CCSID 65535 is used, which indicates that data in the fields is treated as bit data, and is not converted.

coded-character-set-identifier
Specifies the CCSID to be used.

Top

Arithmetic mode (ARITHMETIC)

Specifies the arithmetic mode for numeric data. The possible values are:

***NOEXTEND**
This option specifies the default arithmetic mode for numeric data. The intermediate result of a fixed-point arithmetic expression can be up to 30 digits and numeric literals may only have a maximum length of 18 digits.

***EXTEND31**
Use this option to increase the precision of intermediate results for fixed-point arithmetic. The intermediate result of a fixed-point arithmetic expression can be up to 31 digits and numeric literals may have a maximum length of 31 digits.

***EXTEND63**
Use this option to increase the precision of intermediate results for fixed-point arithmetic. The intermediate result of a fixed-point arithmetic expression can be up to 63 digits and numeric literals may have a maximum length of 63 digits.

Top

Padding character (NTLPADCHAR)

Specifies the national padding character (NTLPADCHAR) used when padding occurs in the following conversion situations:

1. Single byte character to national character.

2. Double byte character to national character.

3. National character to national character.

***DEFAULT**
This option specifies the default padding characters as follows:

1. Single byte character to national character (NX″0020″)

2. Double byte character to national character (NX″3000″)

3. National character to national character (NX″3000″)

national hexadecimal literal
Specifies any valid national hexadecimal literal of length 1 in format NX″ ″ or NX′ ′.

Top

Licensed Internal Code options (LICOPT)

Specifies one or more Licensed Internal Code compile-time options. This parameter allows individual compile-time options to be selected, and is intended for the advanced programmer who understands the potential benefits and drawbacks of each selected type of compiler option.

Top

Include directory (INCDIR)

Specifies one or more directories to add to the search path used by the compiler to find copy files. The compiler will search the directories specified here if the copy files in the source program can not be resolved.

***NONE**
No user directories are searched for copy files. By default, the current directory will still be searched.

'directory'
Specify up to 32 directories in which to search for copy files. In addition to the specified directories, the current directory is also searched for copy files.

Top

Generate program interface (PGMINFO)

Specifies whether program interface information should be generated into a stream file. The possible values are:

***NO** This option specifies the default which does not generate program interface information.

***PCML**
Specifies that PCML (Program Call Markup Language) should be generated into a stream file. The generated PCML makes it easier for JAVA methods to call the procedure in this COBOL module, with less Java code. The name of a stream file that will contain the generated PCML must be specified on the INFOSTMF option.

Top

Program interface stream file (INFOSTMF)

Specifies the path name of the stream file to contain the generated program interface information specifed on the PGMINFO option.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'.

If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by appending the job's current working directory to the path name.

This parameter can only be specified when the PGMINFO parameter has a value other than *NO.

Top

Examples

Example 1: Compiling a Source Program into a Module Object

```
CRTCBLMOD   MODULE(MYLIB/XMPLE1)  SRCFILE(MYLIB/QCBLLESRC)
            SRCMBR(XMPLE1)  OUTPUT(*PRINT)
            TEXT('My ILE COBOL module')
```

This command calls the ILE COBOL compiler to create a module named XMPLE1. The source program is in member XMPLE1 of source file QCBLLESRC in library MYLIB. A compiler listing is created.

Top

Error messages

***ESCAPE Messages**

LNC9001
Compile failed. &1 not created.

LNC9006
TGTRLS(&1) specified, but compiler is not installed.

LNC9007
The product library is damaged, or the user is not allowed to use it.

LNC9015
TGTRLS(&1) is not valid.

Top

Create COBOL Program (CRTCBLPGM)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The CRTCBLPGM command compiles a COBOL source program into a program object for use on the server system. You can use this command interactively, in batch mode, or in a CL program.

All object names specified for the CRTCBLPGM command must be composed of alphanumeric characters, the first of which must be alphabetic. The names cannot exceed 10 characters in length.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Optional, Positional 1
	Qualifier 1: Program	*Name*, ***PGMID**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QLBLSRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 3
GENLVL	Generation severity level	0-29, **29**	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OPTION	Source listing options	Values (up to 50 repetitions): *SOURCE, *NOSOURCE, *SRC, *NOSRC, *NOXREF, *XREF, *GEN, *NOGEN, *NOSEQUENCE, *SEQUENCE, *NOVBSUM, *VBSUM, *NONUMBER, *NUMBER, *LINENUMBER, *NOMAP, *MAP, *NOOPTIONS, *OPTIONS, *QUOTE, *APOST, *SECLVL, *NOSECLVL, *PRTCORR, *NOPRTCORR, *NOSRCDBG, *SRCDBG, *NOLSTDBG, *LSTDBG, *PRINT, *NOPRINT	Optional, Positional 4
GENOPT	Generation options	Values (up to 50 repetitions): *NOLIST, *LIST, *NOXREF, *XREF, *NOPATCH, *PATCH, *NODUMP, *DUMP, *NOATR, *ATR, *RANGE, *NORANGE, *UNREF, *NOUNREF, *NOOPTIMIZE, *OPTIMIZE, *NODDSFILLER, *DDSFILLER, *NOSYNC, *SYNC, *NOCRTF, *CRTF, *NODUPKEYCHK, *DUPKEYCHK, *STDERR, *NOSTDERR, *NOEXTACCDSP, *EXTACCDSP, *NOINZDLT, *INZDLT, *NOBLK, *BLK, *STDINZ, *NOSTDINZ, *FS21DUPKY, *NOFS21DUPKY	Optional, Positional 5
CVTOPT	Conversion options	Values (up to 50 repetitions): *NOVARCHAR, *VARCHAR, *NODATETIME, *DATETIME, *NOGRAPHIC, *GRAPHIC	Optional
SRTSEQ	Sort sequence	Single values: ***HEX**, *JOB, *JOBRUN, *LANGIDUNQ, *LANGIDSHR Other values: *Qualified object name*	Optional
	Qualifier 1: Sort sequence	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
LANGID	Language identifier	*Character value*, ***JOBRUN**, *JOB	Optional

Keyword	Description	Choices	Notes
MSGLMT	Message limit	*Element list*	Optional
	Element 1: Number of messages	1-9999, ***NOMAX**	
	Element 2: Message limit severity	0-29, **29**	
PRTFILE	Print file	*Qualified object name*	Optional
	Qualifier 1: Print file	*Name*, **QSYSPRT**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
FLAGSTD	FIPS flagging	Values (up to 24 repetitions): *NOFIPS, *MINIMUM, *INTERMEDIATE, *HIGH, *NOSEG, *SEG1, *SEG2, *NODEB, *DEB1, *DEB2, *NOOBSOLETE, *OBSOLETE	Optional
SAAFLAG	SAA flagging	***NOFLAG**, *FLAG	Optional
EXTDSPOPT	Extended display options	Values (up to 3 repetitions): *DFRWRT, *NODFRWRT, *UNDSPCHR, *NOUNDSPCHR, *ACCUPDALL, *ACCUPDNE	Optional
FLAG	Flagging severity	0-99, **0**	Optional
REPLACE	Replace program	*NO, ***YES**	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
USRPRF	User profile	***USER**, *OWNER	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
DUMP	Compiler debugging dump	*Element list*	Optional
	Element 1:	1-65535, *	
	Element 2:	1-65535	
ITDUMP	Intermediate text dump	0-31, **0**	Optional

Top

Program (PGM)

Specifies the program name and library name for the COBOL program object you are creating. The possible values are:

***PGMID**
The name for the program object is taken from the PROGRAM-ID paragraph in the COBOL source program.

program-name
Enter a name to identify the compiled COBOL program. If you specify a program name for this parameter, and run the compilation in batch mode, the first program in the batch job uses this name; any other programs use the name specified in the PROGRAM-ID paragraph in the source program.

The possible library values are:

***CURLIB**
If you do not specify a library name, the current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library to contain the created program object.

Top

Source file (SRCFILE)

Specifies the name of the source file that contains the COBOL source to be compiled. The possible values are:

QLBLSRC
The IBM-supplied source file, QLBLSRC, contains the COBOL source to be compiled.

source-file-name
Enter the name of the source file that contains the COBOL source to be compiled. This source file should have a record length of 92.

The possible library values are:

***LIBL** If you do not specify a library name, the system searches the library list to find the library where the source file is located.

***CURLIB**
The current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the source file is located.

Top

Source member (SRCMBR)

Specifies the name of the member that contains the COBOL source to be compiled. You can specify this parameter only if the source file referred to in the SRCFILE parameter is a database file. The possible values are:

***PGM** If you specified a program name for the PGM parameter, the compiler looks for the source program in a member having the same name as the program, and creates an object program with the same name as the program and member.

If you did not specify a program name for the PGM parameter, the compiler looks for the program source in the first member of the database source file, and creates an object program using the name specified in the PROGRAM-ID paragraph.

source-file-member-name
Enter the name of the member that contains the COBOL source.

Top

Generation severity level (GENLVL)

Specifies the severity level that determines if a program object is created. The severity level corresponds to the severity level of the messages produced during compilation of the program. If the severity level of error messages is greater than the value you specify, a program object is not created. For example, if you specify 19 for this parameter, a program object is not created if the severity level of any of the messages is 20 or greater.

The possible values are:

29 If errors occur with a severity level greater than 29, no program object is created.

severity-level
Specify a one or two-digit number, 0 through 29. If errors occur with a severity level greater than this level, no program object is created.

Top

Text 'description' (TEXT)

A brief description of the program and its function. The possible values are:

***SRCMBRTXT**
Use the same text for the program object as that which describes the database file member containing the COBOL source. If the source comes from a device or in-line file, specifying *SRCMBRTXT has the same effect as specifying *BLANK.

***BLANK**
No text is specified.

text-description
Enter the text that briefly describes the program and its function. The text can be a maximum of 50 characters in length and must be enclosed in apostrophes. The apostrophes are not part of the 50-character string.

Top

Source listing options (OPTION)

Specifies the options to use when the COBOL source is compiled. The possible values are:

***SOURCE or *SRC**
The compiler produces a source listing, consisting of the COBOL source code and all compilation-time error messages.

***NOSOURCE or *NOSRC**
The compiler does not produce the source part of the listing. If you do not require a source listing, you should use this option because compilation may take less time.

***NOXREF**
The compiler does not produce a cross-reference listing for the source program.

***XREF** The compiler produces a cross-reference listing for the source program.

***GEN** The compiler creates a program object after the source program is compiled.

***NOGEN**
The compiler does not create a program object after the source program is compiled. You might specify this option if you want only error listings at this time.

***NOSEQUENCE**
The reference numbers are not checked for sequence errors.

***SEQUENCE**
The reference numbers are checked for sequence errors. Sequence errors do not occur if the *LINENUMBER option is specified.

***NOVBSUM**
Verb-usage counts are not printed.

***VBSUM**
Verb-usage counts are printed.

***NONUMBER**
The source-file sequence numbers are used for reference numbers.

***NUMBER**
The user-supplied sequence numbers (columns 1 through 6) are used for reference numbers.

***LINENUMBER**
The sequence numbers created by the compiler are used for reference numbers. This option combines program source code and source code introduced by COPY statements into one consecutively numbered sequence. Use this option if you specify FIPS (Federal Information Processing Standards) flagging or SAA* flagging.

***NOMAP**
The compiler does not list the Data Division map.

***MAP** The compiler lists the Data Division map.

***NOOPTIONS**
Options in effect are not listed for this compilation.

***OPTIONS**
Options in effect are listed for this compilation.

***QUOTE**
Specifies that the delimiter quotation mark (") is used for nonnumeric literals and Boolean literals. This also specifies that the value of the figurative constant QUOTE has the EBCDIC value of a quotation mark.

***APOST**
Specifies that the delimiter apostrophe (') is used for nonnumeric literals and Boolean literals. This also specifies that the value of the figurative constant QUOTE has the EBCDIC value of an apostrophe.

***NOSECLVL**
Second level message text is not listed for this compilation.

***SECLVL**
Second level message text is listed for this compilation.

***PRTCORR**
The compiler inserts comment lines in the compiler listing indicating which elementary items were included as a result of the use of the CORRESPONDING phrase.

***NOPRTCORR**
The compiler does not insert comment lines in the compiler listing when the CORRESPONDING phrase is used.

***NOSRCDBG**
This option determines the kind of information you see on your programmable work station when using the IBM CoOperative Development Environment/400 product to compile your COBOL programs.

The compiler does not produce source-level debugging information. If *NOLSTDBG is also in effect, the compiler does not produce source-level error information either.

***SRCDBG**
This option determines the kind of information you see on your programmable work station when using the IBM CoOperative Development Environment/400 (CODE/400) product to compile your COBOL programs.

The compiler produces source-level error information and source-level debugging information.

You cannot specify *SRCDBG and *LSTDBG together. Specify one or the other.

Note: You can only use the *SRCDBG option if you are using the IBM CODE/400 product to compile your program. If you specify this option but do not have the IBM CODE/400 product installed, the COBOL/400 compiler continues processing, but an error message is issued. For more information on this option, see the CODE/400 Debug Tool and Reference, SC09-1622.

***NOLSTDBG**
This option determines the kind of information you see on your programmable work station when using the IBM CoOperative Development Environment/400 product to compile your COBOL programs.

The compiler does not produce a listing view, or listing-level debugging information. If *NOSRCDBG is also in effect, the compiler does not produce source-level error information either.

***LSTDBG**
This option determines the kind of information you see on your programmable work station when using the IBM CoOperative Development Environment/400 product to compile your COBOL programs.

The compiler produces a listing view, source-level error information, and listing-level debugging information.

You cannot specify *SRCDBG and *LSTDBG together. Specify one or the other.

Note: You can only use the *SRCDBG option if you are using the IBM CODE/400 product to compile your program. If you specify this option but do not have the IBM CODE/400 product installed, the COBOL/400 compiler continues processing, but an error message is issued. For more information on this option, see the CODE/400 Debug Tool User's Guide and Reference, SC09-1622.

***PRINT**
The compiler produces a spool listing.

***NOPRINT**
The compiler does not produce a spool listing.

Top

Generation options (GENOPT)

Specifies the options to use when the program object is created. The listings could be required if a problem occurs in COBOL. The possible values are:

***NOLIST**
No IRP (intermediate representation of program), associated hexadecimal code, or error messages are listed.

***LIST** The IRP, its associated hexadecimal code, and any error messages are listed.

***NOXREF**
Does not produce a cross-reference listing of the objects defined in the IRP.

***XREF** Produces a cross-reference listing of all objects defined in the IRP.

***NOPATCH**
Does not reserve space in the compiled program for a program patch area.

***PATCH**
Reserves space in the compiled program for a program patch area. The program patch area can be used for debugging purposes.

***NODUMP**
Does not list the program template.

***DUMP**
Lists the program template.

***NOATR**
Does not list the attributes for the IRP source.

***ATR** Lists the attributes for the IRP source.

***RANGE**
At run-time, the system verifies that subscripts are within the correct ranges, but does not verify index ranges. It also checks for reference modification and compiler-generated substring operations.

***NORANGE**
Does not verify ranges at run-time.

***UNREF**
Includes unreferenced data items in the compiled program.

***NOUNREF**
Does not include unreferenced data items in the compiled program. This reduces the number of ODT (object definition table) entries used, allowing a larger program to be compiled. The unreferenced data items still appear in the cross-reference listings produced through the *XREF option.

***NOOPTIMIZE**
The compiler performs only standard optimizations for the program.

***OPTIMIZE**
The program object created may run more efficiently, and may require less storage. However, specifying *OPTIMIZE can substantially increase the time required to compile a program.

***NODDSFILLER**
If no matching fields are found by a COPY DDS statement, no field descriptions are created.

***DDSFILLER**
When no matching fields are found by a COPY DDS statement, a single character FILLER field description, ″07 FILLER PIC X″, is always created.

***NOSYNC**
The SYNCHRONIZED clause is syntax checked.

***SYNC**
The SYNCHRONIZED clause causes the alignment of an elementary item on a natural boundary in storage.

***NOCRTF**
Files that are unavailable at the time of an OPEN operation are not created dynamically.

***CRTF**
Files that are unavailable at the time of an OPEN operation are created dynamically.

***NODUPKEYCHK**
Does not check for duplicate keys for INDEXED files.

***DUPKEYCHK**
Checks for duplicate keys for INDEXED files.

***STDERR**
Standard error handling is used. See the chapter on error handling in the COBOL/400 User's Guide for more information about error handling.

***NOSTDERR**
The error handling method of Version 1, Releases 1 and 2, is used. See the chapter on error handling in the COBOL/400 User's Guide for more information about error handling.

***NOEXTACCDSP**
The compiler does not allow extended ACCEPT or extended DISPLAY statements.

***EXTACCDSP**
The compiler allows extended ACCEPT and extended DISPLAY statements.

***NOINZDLT**
Relative files with sequential access are not initialized with deleted records during the CLOSE operation if the files were opened for OUTPUT. That is, the record boundary is determined by the number of records written at OPEN OUTPUT time. Subsequent OPEN operations allow access only up to the record boundary.

***INZDLT**
Relative files with sequential access are initialized with deleted records during the CLOSE operation if the files were opened for OUTPUT. Active records in the files are not affected. That is, the record boundary is defined as the file size for subsequent OPEN operations.

***NOBLK**
The compiler allows blocking only of SEQUENTIAL access files with no START statement.

If a BLOCK CONTAINS clause is specified, the BLOCK CONTAINS clause is ignored, except for tape files.

***BLK** When used with BLOCK CONTAINS, the compiler allows blocking from DYNAMIC access files and SEQUENTIAL access files with a START statement. Blocking is not allowed for RELATIVE files opened for output operations.

The BLOCK CONTAINS clause controls the number of records to be blocked.

When no BLOCK CONTAINS clause is specified, the compiler allows blocking only of SEQUENTIAL access files with no START statement. The operating system determines the number of records to be blocked.

***STDINZ**
The compiler initializes data items to system defaults, provided that the items are not subject to a VALUE clause.

***NOSTDINZ**
For those items with no VALUE clause, the compiler does not initialize data items to system defaults.

***FS21DUPKY**
The compiler reports a file status of 21 when processing an indexed file with duplicate keys in random or dynamic access mode, if the value of the key is changed between the mandatory READ statement and a following REWRITE or DELETE statement.

***NOFS21DUPKY**
The compiler does not report a file status of 21 when processing an indexed file with duplicate keys in random or dynamic access mode. A REWRITE statement can change the key of a record.

Top

Conversion options (CVTOPT)

Specifies how the compiler handles SAA date, time, and timestamp data types, DBCS-graphic data types, and variable-length character fields passed from externally-described files to your program through COPY DDS. The possible values are:

***NOVARCHAR**
Variable-length fields are ignored, and are declared as FILLER fields.

***VARCHAR**
Variable-length fields are declared as fixed-length group items, and are accessible to the program.

***NODATETIME**
Date, time, and timestamp data types are ignored, and are declared as FILLER fields.

***DATETIME**
Date, time, and timestamp data types are declared as fixed-length character fields, and are accessible to the program.

***NOGRAPHIC**
Graphic data types are ignored, and are declared as FILLER fields.

***GRAPHIC**
Fixed-length graphic data types are declared as fixed-length alphanumeric fields, and are accessible to the program.

When the *VARCHAR option is also in use, variable-length DBCS-graphic data types are declared as fixed-length group items, and are accessible to the program.

Top

Sort sequence (SRTSEQ)

Specifies the sort sequence used when NLSSORT is associated with an alphabet-name in the ALPHABET clause. The SRTSEQ parameter is used in conjunction with the LANGID parameter to determine which system-defined or user-defined sort sequence table the program will use. The possible values are:

***HEX** No sort sequence table will be used, and the hexadecimal values of the characters will be used to determine the sort sequence.

***JOB** The sort sequence of the program will be resolved and associated with the program at compile time. The sort sequence table must exist in the system at compile time.

***JOBRUN**
The sort sequence of the program will be resolved and associated with the program at run time. At compile time, the compiler will associate the sort sequence of the compile job with the program. At run time, this sort sequence will be replaced by the sort sequence associated with the job at run time.

***LANGIDUNQ**
Specifies that the sort sequence table being used must contain a unique weight for each character in the code page. The sort sequence table used will be the unique weighted table associated with the language specified in the LANGID parameter.

***LANGIDSHR**
Specifies that the sort sequence table being used can contain the same weight for multiple characters in the code page. The sort sequence table used will be the shared weighted table associated with the language specified in the LANGID parameter.

table-name
Enter the name of the sort sequence table to be used. The table contains weights for all characters in a given code page. A weight is associated with the character that is defined at the code point. When using a sort sequence table name, the library in which the table resides can be specified. The valid values are:

***LIBL** The library list is searched to find the library where the sort sequence table is located.

***CURLIB**
The current library is searched. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the sort sequence table is found.

Note: The valid PROCESS statement options for SRTSEQ are SRTSEQ(HEX), SRTSEQ(JOB), SRTSEQ(JOBRUN), SRTSEQ(LANGIDUNQ), SRTSEQ(LANGIDSHR), SRTSEQ("table-name"), SRTSEQ("library-name/table-name"), SRTSEQ("LIBL/table-name"), and SRTSEQ("CURLIB/table-name").

Top

Language identifier (LANGID)

Specifies the language identifier which is used in conjunction with the sort sequence. The LANGID parameter is used only when the SRTSEQ value in effect is *LANGIDUNQ or *LANGIDSHR. The possible values are:

***JOBRUN**
The language identifier of the program will be resolved at run time. When the compiled program is run, the language identifier of the job is used. This value allows a program to be compiled once and used with different language identifiers at run time.

***JOB** The language identifier of the program will be resolved at compile time.

language-identifier
Enter a valid 3 character language identifier.

Note: The valid PROCESS statement options for LANGID are LANGID(JOBRUN), LANGID(JOB), and LANGID("language-identifier").

Top

Message limit (MSGLMT)

Controls compilation by indicating the maximum number of error messages of a given error severity level that can occur before compilation stops.

For example, you can stop compilation if more than three errors with a severity level of 20 or higher occur. In this example, you would specify 3 for the maximum number of error messages, and 20 for the maximum error severity level. If three errors of severity level 20 or higher occur, compilation continues, but when a fourth is encountered, compilation stops. If no messages equal or exceed the maximum severity level, compilation continues regardless of the number of errors encountered.

message-limit
The possible values for the maximum number of error messages are:

***NOMAX**
Compilation continues until normal completion regardless of the number of errors encountered.

1-9999 Compilation stops if the number of errors of the specified severity level or higher exceeds the number you specify. If no messages equal or exceed the maximum severity level, compilation continues regardless of the number of errors encountered.

message-severity
The possible values for the maximum error severity level are:

29 Compilation stops if the number of errors with severity level 29 or higher exceeds the maximum number of error messages specified.

maximum-severity-level
Specify a one or two-digit number, 0 through 29. Compilation stops if the number of errors with the specified severity level or higher exceeds the maximum number of error messages you specify.

Top

Print file (PRTFILE)

Specifies the name of the file to which the compiler listing is directed and the library where the file is located. The file should have a minimum record length of 132. If a file with a record length less than 132 is specified, information is lost.

The possible values are:

QSYSPRT
If you do not specify a file name, the compiler listing is directed to QSYSPRT, an IBM-supplied file.

file-name
Enter the name of the file to which the compiler listing is directed.

The possible library values are:

***LIBL** The system searches the library list to find the library where the file is located.

***CURLIB**
The current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library in which the file is located.

Top

FIPS flagging (FLAGSTD)

Specifies the options for FIPS flagging. The possible values are:

***NOFIPS**
The source program is not FIPS flagged.

***MINIMUM**
FIPS flag for minimum subset and higher.

***INTERMEDIATE**
FIPS flag for intermediate subset and higher.

***HIGH**
FIPS flag for high subset.

***NOSEG**
Optional module SEGMENTATION is not FIPS flagged.

***SEG1** FIPS Flag for optional module SEGMENTATION level 1 and higher.

***SEG2** FIPS Flag for optional module SEGMENTATION level 2.

***NODEB**
Optional module DEBUG is not FIPS flagged.

***DEB1**
FIPS Flag for optional module DEBUG level 1 and higher.

***DEB2**
FIPS Flag for optional module DEBUG level 2.

***NOOBSOLETE**
Obsolete language elements are not flagged.

***OBSOLETE**
Obsolete language elements are flagged.

Top

SAA flagging (SAAFLAG)

Specifies if you want flagging of COBOL/400* functions that are not supported by SAA COBOL. The possible values are:

***NOFLAG**
SAA COBOL flagging is not performed.

***FLAG**
SAA COBOL flagging is performed.

Top

Extended display options (EXTDSPOPT)

Specifies the options to use for extended ACCEPT and extended DISPLAY statements for workstation I/O. The possible values are:

***DFRWRT**
Extended DISPLAY statements are held in a buffer until an extended ACCEPT statement is encountered, or until the buffer is filled.

If an extended ACCEPT statement is not encountered before the buffer is filled, the contents of the buffer are written to the display. When an extended ACCEPT statement is encountered, the current contents of the buffer are written to the display.

***NODFRWRT**
Each extended DISPLAY statement is performed as it is encountered.

***UNDSPCHR**
Displayable and undisplayable characters are handled by extended ACCEPT and extended DISPLAY statements.

***NOUNDSPCHR**
Only displayable characters are handled by extended ACCEPT and extended DISPLAY statements.

Although you must use this option for display stations attached to remote 3174 and 3274 controllers, you can also use it for local work stations. If you do use this option, your data must contain displayable characters. If the data contains values less than hexadecimal 20, the results are unpredictable, and can range from unexpected display formats to severe errors.

***ACCUPDALL**
All types of data are predisplayed in the extended ACCEPT statements regardless of the existence of the UPDATE phrase.

***ACCUPDNE**
Only numeric edited data are predisplayed in the extended ACCEPT statements that do not contain the UPDATE phrase.

Top

Flagging severity (FLAG)

Specifies the minimum severity level of messages to be printed. The possible values are:

0 All messages are printed.

severity-level
Enter a one or two-digit number that specifies the minimum severity level of messages that are printed. Messages that have severity levels of the specified value or higher are listed.

Top

Replace program (REPLACE)

Specifies if a new program object is created when a program object of the same name in the same library already exists. The possible values are:

***YES** A new program object is created and any existing program object of the same name in the specified library is moved to library QRPLOBJ.

***NO** A new program object is not created if a program object of the same name already exists in the specified library.

Top

Target release (TGTRLS)

Specifies the release level of the operating system on which you intend to use the object being created.

You can specify an exact release level in the format VxRxMx, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V3R1M0 is Version 3, Release 1, Modification level 0.

Note: To use the object on the target system, you must save the object to the target release level specified on the create command and then restore it on the target system.

For example, if your system is running V3R1M0 and you want to create a program object for distribution to a V2R3M0 system, you must create the program with TGTRLS(V2R3M0) or TGTRLS(*PRV), save the program with TGTRLS(V2R3M0) or TGTRLS(*PRV), and restore the program on the V2R3M0 system. The program object can also be restored on a V3R1M0 system.

Note: The program may be able to be restored on a release level earlier than the one you specified on the create command. Use DSPPGM to determine the earliest release the program can run.

***CURRENT**
The object is to be used on the release of the operating system currently running on your system. For example, if V3R1M0 is running on the system, *CURRENT means you intend to use the object on a system with V3R1M0 installed. You can also use the object on a system with any subsequent release of the operating system installed.

***PRV** The object is to be used on the previous release with modification level 0 of the operating system.

For example, if V3R1M0 is running on your system, *PRV means you intend to use the object on a system with V2R3M0 installed. You can also use the object on a system with any subsequent release of the operating system installed.

release-level
Specify the release in the format VxRxMx. The object can be used on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. To see the list of valid values, press F4=Prompt on the TGTRLS parameter of the command.

Top

User profile (USRPRF)

Specifies the user profile that will run the compiled COBOL program. The profile of the program owner or the program user is used to run the program and control which objects can be used by the program (including the authority the program has for each object).

The possible values are:

***USER**
The user profile of the program user is to be used when the program is run.

***OWNER**
The user profiles of both the program's owner and user are to be used when the program is run. The collective sets of object authority in both user profiles are to be used to find and access objects during the running of the program. Any objects that are created during the program are owned by the program's user.

Top

Authority (AUT)

Specifies the authority given to users who do not have specific authority to the program object, who are not on the authorization list, or whose group has no specific authority to the program object. You can alter the authority for all users, or for specific users after the program object is created by using the GRTOBJAUT (Grant Object Authority) or RVKOBJAUT (Revoke Object Authority) commands.

The possible values are:

***LIBCRTAUT**
The public authority for the object is taken from the CRTAUT keyword of the target library (the library that is to contain the created program object). This value is determined when the program object is created. If the CRTAUT value for the library changes after the program object is created, the new value does NOT affect any existing objects.

***ALL** Provides authority for all operations on the program object except those limited to the owner or controlled by authorization list management authority. The user can control the program object's existence, specify security for it, change it, and perform basic functions on it, but cannot transfer its ownership.

***CHANGE**
Provides all data authority and the authority for performing all operations on the program object except those limited to the owner or controlled by object authority and object management authority. The user can change the object and perform basic functions on it, such as running and debugging the program object.

***USE** Provides object operational authority and read authority; authority for basic operations on the program object such as running the program. The user is prevented from changing the object.

Note: A user must have *USE authority to a program to obtain a formatted dump of the variables of the program. To dump variables, the program must also have observable information.

If you do not want some users to be able to dump the variables, then give them only *OBJOPR plus *EXECUTE authority to the program. This will allow them to call the program but not dump its variables.

If you do not want any users to be able to dump the variables, then use Change Program (CHGPGM) to remove the program's observable information.

***EXCLUDE**
The user cannot access the program object.

authorization-list-name
Enter the name of an authorization list of users and authorities to which the program is added. The program object is secured by this authorization list, and the public authority for the program object is set to *AUTL. The authorization list must exist on the system when the CRTCBLPGM command is issued.

Top

Compiler debugging dump (DUMP)

An IBM COBOL debugging aid. (For IBM service personnel.)

Top

Intermediate text dump (ITDUMP)

An IBM debugging aid which causes the compiler to dump the internal text at certain times during the compilation. (For IBM service personnel.)

Top

Examples

Example 1: Compiling a Source Program into a Program Object

```
CRTCBLPGM   PGM(MYLIB/XMPLE1)  SRCFILE(MYLIB/QLBLSRC)
            SRCMBR(XMPLE1)  OPTION(*SOURCE)
            TEXT('My COBOL program')
```

This command calls the COBOL compiler to create a program named XMPLE1. The source program is in member XMPLE1 of source file QLBLSRC in library MYLIB. A compiler listing is created.

Top

Error messages

***ESCAPE Messages**

LBL9001
Compile failed. Program not created.

LBL9004
Program already exists. Compile failed.

LBL9006
TGTRLS(&1) specified, but previous compiler is not installed.

LBL9007
The product library is damaged, or the user is not allowed to use it.

SQL9002
Conflict in TGTRLS parameters for SQL precompile and &7 compile.

SQL9003
&7 Compile at wrong level for SQL source.

Top

Create C Module (CRTCMOD)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Create C Module (CRTCMOD) command starts the ILE C compiler. You can use this command in either batch or interactive mode, or from a CL program. The compiler attempts to create a module object based on the ILE statements in the source code.

Error messages for CRTCMOD

*ESCAPE Messages

CZM0613
The compilation failed.

Top

Parameters

Keyword	Description	Choices	Notes
MODULE	Module	*Qualified object name*	Required, Positional 1
	Qualifier 1: Module	*Name*	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QCSRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***MODULE**	Optional, Positional 3
SRCSTMF	Source stream file	*Path name*	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OUTPUT	Output options	Single values: ***NONE**, '*none' Other values: *Element list*	Optional
	Element 1: Output file name	*Path name*, *PRINT, '*print'	
	Element 2: Title	*Character value*, ***BLANK**	
	Element 3: Subtitle	*Character value*, ***BLANK**	
OPTION	Compiler options	Values (up to 35 repetitions): *NOAGR, *AGR, *DIGRAPH, *NODIGRAPH, *NOEVENTF, *EVENTF, *NOEXPMAC, *EXPMAC, *NOFULL, *FULL, *GEN, *NOGEN, *NOINCDIRFIRST, *INCDIRFIRST, *LOGMSG, *NOLOGMSG, *NOPPONLY, *PPONLY, *NOSECLVL, *SECLVL, *NOSHOWINC, *SHOWINC, *NOSHOWSKP, *SHOWSKP, *SHOWSRC, *NOSHOWSRC, *NOSHOWSYS, *SHOWSYS, *NOSHOWUSR, *SHOWUSR, *STDINC, *NOSTDINC, *NOSTDLOGMSG, *STDLOGMSG, *NOSTRUCREF, *STRUCREF, *NOSYSINCPATH, *SYSINCPATH, *NOXREF, *XREF, *NOXREFREF, *XREFREF	Optional

Keyword	Description	Choices	Notes
CHECKOUT	Checkout options	Values (up to 39 repetitions): ***NONE**, *USAGE, *ALL, *NOCOND, *COND, *NOCONST, *CONST, *NOEFFECT, *EFFECT, *NOENUM, *ENUM, *NOEXTERN, *EXTERN, *NOGENERAL, *GENERAL, *NOGOTO, *GOTO, *NOINIT, *INIT, *NOPARM, *PARM, *NOPORT, *PORT, *NOPPCHECK, *PPCHECK, *NOPPTRACE, *PPTRACE, *NOREACH, *REACH, *NOTRUNC, *TRUNC, *NOUNUSED, *UNUSED	Optional
OPTIMIZE	Optimization	**10**, 20, 30, 40	Optional
INLINE	Inline options	*Element list*	Optional
	Element 1: Inliner	***OFF**, *ON	
	Element 2: Mode	***NOAUTO**, *AUTO	
	Element 3: Threshold	1-65535, **250**, *NOLIMIT	
	Element 4: Limit	1-65535, **2000**, *NOLIMIT	
	Element 5: Report	***NO**, *YES	
MODCRTOPT	Module creation options	***NOKEEPILDTA**, *KEEPILDTA	Optional
DBGVIEW	Debugging view	***NONE**, *ALL, *STMT, *SOURCE, *LIST	Optional
DEFINE	Define names	Single values: ***NONE** Other values (up to 32 repetitions): *Character value*	Optional
LANGLVL	Language level	***EXTENDED**, *ANSI	Optional
ALIAS	Alias	Values (up to 3 repetitions): ***ANSI**, *NOANSI, *ADDRTAKEN, *NOADDRTAKEN, *ALLPTRS, *NOALLPTRS, *TYPEPTR, *NOTYPEPTR	Optional
SYSIFCOPT	System interface options	Values (up to 2 repetitions): *NOIFSIO, *IFSIO, *IFS64IO, *NOASYNCSIGNAL, *ASYNCSIGNAL	Optional
LOCALETYPE	Locale object type	***LOCALE**, *LOCALEUCS2, *LOCALEUTF, *CLD	Optional
FLAG	Message flagging level	**0**, 10, 20, 30	Optional
MSGLMT	Compiler messages	*Element list*	Optional
	Element 1: Message limit	0-32767, ***NOMAX**	
	Element 2: Message limit severity	0, 10, 20, **30**	
REPLACE	Replace module object	***YES**, *NO	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
ENBPFRCOL	Enable performance collection	*Element list*	Optional
	Element 1: Collection level	***PEP**, *ENTRYEXIT, *FULL	
	Element 2: Procedures	*NONLEAF, *ALLPRC	
PFROPT	Performance options	Values (up to 2 repetitions): *SETFPCA, *NOSETFPCA, *NOSTRDONLY, *STRDONLY	Optional
PRFDTA	Profiling data	***NOCOL**, *COL	Optional
TERASPACE	Teraspace options	Single values: ***NO** Other values: *Element list*	Optional
	Element 1: Teraspace enabled	***YES**	
	Element 2: Use teraspace interfaces	***NOTSIFC**, *TSIFC	
STGMDL	Storage model	***SNGLVL**, *TERASPACE, *INHERIT	Optional
DTAMDL	Data model	***P128**, *LLP64	Optional
PACKSTRUCT	Pack structure	***NATURAL**, 1, 2, 4, 8, 16	Optional
ENUM	Enum size	***SMALL**, 1, 2, 4, *INT	Optional

Keyword	Description	Choices	Notes
MAKEDEP	Dependency information	*Path name*, ***NONE**	Optional
PPGENOPT	Preprocessor options	Single values: ***NONE**, *DFT Other values (up to 2 repetitions): *RMVCOMMENT, *NORMVCOMMENT, *GENLINE, *NOGENLINE	Optional
PPSRCFILE	Output source file	*Qualified object name*	Optional
	Qualifier 1: Output source file	*Name*	
	Qualifier 2: Library	*Name*, ***CURLIB**	
PPSRCMBR	Output source member	*Name*, ***MODULE**	Optional
PPSRCSTMF	Output stream file	*Path name*, *SRCSTMF	Optional
INCDIR	Include directory	Single values: ***NONE** Other values (up to 32 repetitions): *Path name*	Optional
CSOPT	Compiler services option	*Character value*, ***NONE**	Optional
LICOPT	Licensed internal code options	*Character value*, ***NONE**	Optional
DFTCHAR	Default char type	***UNSIGNED**, *SIGNED	Optional
TGTCCSID	Target CCSID	1-65535, ***SOURCE**, *JOB, *HEX	Optional

Top

Module (MODULE)

Specifies the module name and library for the module object.

module-name
Enter a name for the module object.

The possible library values are:

***CURLIB**
The module object is stored in the current library. If a job does not have a current library, the module object is created in the QGPL library.

library-name
Enter the name of the library where the module object will be stored.

Top

Source file (SRCFILE)

Specifies the source file name and library of the file containing the ILE C source code that you want to compile.

QCSRC
The source file named QCSRC contains the member with the ILE C source code that you want to compile.

source-file-name
Enter the name of the file that contains the member with the ILE C source code.

The possible library values are:

***LIBL** The library list is searched to find the library where the source file is located.

***CURLIB**
The current library is searched for the source file. If a job does not have a current library, QGPL is searched for the source file.

library-name
Enter the name of the library that contains the source file.

Top

Source member (SRCMBR)

Specifies the name of the member containing the source code to be compiled.

***MODULE**
The module name supplied on the MODULE parameter is used as the source member name.

member-name
Enter the name of the member that contains the source code.

Top

Source stream file (SRCSTMF)

Specifies the path name of the stream file containing the source code that you want to compile.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'. If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by pre-pending the job's current working directory to the path name.

The SRCMBR and SRCFILE parameters cannot be specified with the SRCSTMF parameter.

Top

Text 'description' (TEXT)

Specifies the text that briefly describes the module object.

***SRCMBRTXT**
The text description associated with the source file member is used for the module object. If the source file is an inline file, a stream file, or a device file, the text will be blank.

***BLANK**
Specifies that no text appears.

'description'
Specify no more than 50 characters of text, enclosed in apostrophes.

Top

Output options (OUTPUT)

Specifies whether a compiler listing is produced.

Single Value

***NONE**
Does not generate the compiler listing. When a listing is not required, this parameter value should be used to improve compile-time performance. When *NONE is specified, any listing-related parameter values specified for the OPTION parameter are ignored.

Element 1: Output File Name

***PRINT**
Generates a spooled file containing the listing.

'path-name'
Specify the path name of a stream file to hold the listing.

Element 2: Title

***BLANK**
Specifies that no text appears.

'title' Specify a title string for the listing file (maximum 80 characters).

Element 3: Subtitle

***BLANK**
Specifies that no text appears.

'subtitle'
Specify a subtitle string for the listing file (maximum 80 characters).

Top

Compiler options (OPTION)

Specifies the options to use when the ILE C source code is compiled. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

The possible options are:

***NOAGR**
Does not generate aggregate maps in the listing.

***AGR** Generate maps of all aggregates in the listing. The maps include structures and unions. The structure maps show padding of members. This option overrides the *STRUCREF option.

***DIGRAPH**
Allow the use of digraphs in the source code.

***NODIGRAPH**
Do not allow the use of digraphs in the source code.

***NOEVENTF**
Do not create an event file for use by CoOperative Development Environment/400 (CODE/400).

***EVENTF**
Create an event file for use by CoOperative Development Environment/400 (CODE/400). The event file is created as a member in file EVFEVENT in the library where the module or program object being created will be stored. If the file EVFEVENT does not exist it is automatically created. The event file member name is the same as the name of the object being created. CODE/400 uses this file to provide error feedback integrated with the CODE/400 editor. An event file is normally created when you create a module or program object from within CODE/400.

***NOEXPMAC**
Macros are not expanded in the listing unless a syntax error is encountered within the macro.

***EXPMAC**
Expand all macros in the listing. This parameter conflicts with DBGVIEW(*ALL) and DBGVIEW(*LIST). Compilation will stop with an error message if OPTION(*EXPMAC) is used together with DBGVIEW(*ALL) or DBGVIEW(*LIST).

***NOFULL**
Do not turn on all listing options.

***FULL** Turn on all listing options.

***GEN** All phases of the compilation process are carried out.

***NOGEN**
Compilation stops after syntax checking. No module object is created.

***NOINCDIRFIRST**
Include directories specified as INCDIR parameters are not included before the standard header file include path.

***INCDIRFIRST**
Include directories specified as INCDIR parameters are included before the standard header file include path.

***LOGMSG**
Puts the compilation messages in the job log.

When you specify this option and the FLAG parameter, messages with the severity specified on the FLAG parameter (and higher) are placed in the job log.

When you specify this option and a maximum number of messages on the MSGLMT parameter, compilation stops when the number of messages, at the specified severity, have been placed in the job log.

***NOLOGMSG**
Does not put the compilation messages in the job log.

***NOPPONLY**
The compiler runs the entire compile sequence when *GEN is left as the default for OPTION.

***PPONLY**
Only the preprocessor is run and the output is saved. The rest of the compilation sequence is not run.

If SRCFILE or SRCMBR is specified, then the output is saved in the source file QACZEXPAND in library QTEMP. The member-name is the same as the name specified on the MODULE parameter. When the job is submitted in batch mode, the output is deleted once the job is complete since the output was saved in QTEMP.

If SRCSTMF is specified, then the output is saved in a stream file in the user's current directory. The file name is the same as the file on SRCSTMF with the extension '.i'.

OPTION(*PPONLY) has been superseded by parameters PPGENOPT, PPSRCFILE, PPSRCMBR and PPSRCSTMF.

***NOSECLVL**
Does not generate second-level message text in the listing.

***SECLVL**
Generates second-level message text in the listing. An OUTPUT option must be specified for this option to take effect.

***NOSHOWINC**
Does not expand the user include files or the system include files in the source section of the listing or in the debug views.

***SHOWINC**
Expands both the user include files and the system include files in the source section of the listing or in the debug views. An OUTPUT option or DBGVIEW parameter value of *ALL, *SOURCE or *LIST must be specified.

***NOSHOWSKP**
Does not include the statements that the preprocessor has ignored in the source section of the listing or in the debug listing view. The preprocessor ignores statements as a result of a preprocessor directive evaluating to false (zero).

***SHOWSKP**
Includes all the statements in the source section of the listing or in the debug listing view, regardless of whether or not the preprocessor has skipped them. An OUTPUT option or a DBGVIEW parameter value of *ALL or *LIST must be specified.

***SHOWSRC**
Show the source code in the listing. This option can be modified by the *SHOWINC, *SHOWSYS or *SHOWUSR options.

***NOSHOWSRC**
Does not show the source code in the listing. This option may be modified by the *EXPMAC, *SHOWINC, *SHOWSYS or *SHOWUSR options.

***NOSHOWSYS**
Do not expand the system include files on the #include directive in the source section of the listing or in the debug views. System include files are enclosed in angle brackets (< >) following the #include directive.

***SHOWSYS**
Expands the system include files on the #include directive in the source section of the listing or in the debug views. An OUTPUT option or DBGVIEW parameter value of *ALL, *SOURCE or *LIST must be specified. System include files are enclosed in angle brackets (< >) following the #include directive.

***NOSHOWUSR**
Do not expand user include files in the listing or debug views. User include files are enclosed in double quotation marks (″ ″) following the #include directive.

***SHOWUSR**
Expands the user include files on the #include directive in the source section of the listing or in the debug views. An OUTPUT option or DBGVIEW parameter value of *ALL, *SOURCE or *LIST must be specified. User include files are enclosed in double quotation marks (″ ″) following the #include directive.

***STDINC**
The system supplied header files are included in the search path for the compile.

***NOSTDINC**
The system supplied header files are not included in the search path for the compile.

***NOSTDLOGMSG**
Compilation messages are not sent to the stdout stream.

***STDLOGMSG**
Compilation messages are sent to the stdout stream.

***NOSTRUCREF**
Do not generate maps of all referenced struct or union variables in the listing file.

***STRUCREF**
Generate maps of all referenced struct or union variables in the listing file.

***NOSYSINCPATH**
The search path for user includes is not affected.

***SYSINCPATH**
Changes the search path of user includes to the system include search path. In function this option is equivalent to changing the double-quotes in the user #include directive (#include ″file_name″) to angle brackets (#include <file_name>).

***NOXREF**
Does not generate the cross-reference table in the listing.

***XREF** Generates the cross-reference table containing a list of the identifiers in the source code together with the numbers of the lines in which they appear. An OUTPUT option must be specified.

***NOXREFREF**
Do not produce a cross-reference table of referenced identifiers in the listing.

***XREFREF**
Produce a cross-reference table of referenced variables, structures, and function names in the listing file. The table shows the line numbers where the identifiers are declared. An OUTPUT option must be specified.

Top

Checkout options (CHECKOUT)

Specifies options you may select to generate informational messages that indicate possible programming errors. When you specify an option more than once, or when two options conflict, the last one specified is used.

Note: CHECKOUT may produce many messages. To prevent these messages from going to the job log specify OPTION(*NOLOGMSG *NOSTDLOGMSG) along with an OUTPUT parameter to place the checkout messages in the listing file.

***NONE**
Disables all of the options for CHECKOUT.

***USAGE**
This is equivalent to specifying *ENUM, *EXTERN, *INIT, *PARM, *PORT, *GENERAL and *TRUNC.

***ALL** Enables all of the options for CHECKOUT.

***NOCOND**
Do not warn about possible redundancies or problems in conditional expressions.

***COND**
Warn about possible redundancies or problems in conditional expressions.

***NOCONST**
Do not warn about operations involving constants.

***CONST**
Warn about operations involving constants.

***NOEFFECT**
Do not warn about statements with no effect.

***EFFECT**
Warn about statements with no effect.

***NOENUM**
Does not list the usage of enumerations.

***ENUM**
Lists the usage of enumerations.

***NOEXTERN**
Does not list the unused variables that have external declarations.

***EXTERN**
Lists the unused variables that have external declarations.

***NOGENERAL**
Does not list the general checkout messages.

***GENERAL**
Lists the general checkout messages.

***NOGOTO**
Does not list the occurrence and usage of goto statements.

***GOTO**
Lists the occurrence and usage of goto statements.

***NOINIT**
Does not list the automatic variables that are not explicitly initialized.

***INIT** Lists the automatic variables that are not explicitly initialized.

***NOPARM**
Does not list the function parameters that are not used.

***PARM**
Lists the function parameters that are not used.

***NOPORT**
Does not list nonportable usage of the C language.

***PORT**
Lists nonportable usage of the C language.

***NOPPCHECK**
Does not list the preprocessor directives.

***PPCHECK**
Lists all preprocessor directives.

***NOPPTRACE**
Does not list the tracing of include files by the preprocessor.

***PPTRACE**
Lists the tracing of include files by the preprocessor.

***NOREACH**
Do not warn about unreachable statements.

***REACH**
Warn about unreachable statements.

***NOTRUNC**
Do not warn about the possible truncation or loss of data.

***TRUNC**
Warn about the possible truncation or loss of data.

***NOUNUSED**
Do not check for unused auto or static variables.

***UNUSED**
Check for unused auto or static variables.

Top

Optimization (OPTIMIZE)

Specifies the levels of optimization of the generated object.

10 Generated code is not optimized. This level has the shortest compile time. This level allows variables to be displayed and modified while debugging.

20 Some optimization is performed on the code. This level allows user variables to be displayed but not modified while debugging.

30 Full optimization is performed on the generated code. During a debug session, user variables may not be modified but may be displayed. The presented values may not be the current value of the variable.

40 All optimizations done at level 30 are performed on the generated code. In addition, code is eliminated from procedure prologue and epilogue routines that enable instruction trace and call trace system functions. Eliminating this code enables the creation of leaf procedures. A leaf procedure is a procedure that contains no calls to other procedures. Procedure call performance to a leaf procedure is significantly faster than to a normal procedure.

Top

Inline options (INLINE)

Specifies whether the compiler should consider replacing a function call with the called function's instructions. Inlining a function eliminates the overhead of a call and can result in better optimization. Small functions that are called many times are good candidates for inlining.

Element 1: Inliner

Specifies whether or not inlining will be used.

***OFF** Specifies that inlining will not be performed on the compilation unit.

***ON** Specifies that inlining will be performed on the compilation unit. If a debug view is specified, the inliner is turned off.

Element 2: Mode

Specifies whether or not the inliner should attempt to automatically inline functions depending on their Threshold and Limit values.

***NOAUTO**
Specifies that only the functions that have been specified with the #pragma inline directive should be considered candidates for inlining.

***AUTO**
Specifies that the inliner should determine if a function can be inlined based on the Threshold and Limit values specified. The #pragma noinline directive overrides *AUTO.

Element 3: Threshold

Specifies the maximum size of a function that can be a candidate for automatic inlining. The size is measured in Abstract Code Units (ACUs). ACUs are proportional in size to the executable code in the function. Source code is translated into ACUs by the compiler.

250 Specifies a threshold of 250.

number-of-ACUs
Specifies a threshold from 1 to 65535 ACUs.

***NOLIMIT**
Defines the threshold as the maximum size of the program object.

Element 4: Limit

Specifies the maximum relative size a function can grow before auto-inlining stops.

2000 Specifies a limit of 2000 ACUs.

***NOLIMIT**
Limit is defined as the maximum size of the program object. System limits may be encountered.

number-of-ACUs
A limit from 1 to 65535 ACUs may be specified.

Element 5: Report

Specifies whether or not to produce an inliner report with the compiler listing.

***NO** The inliner report is not produced.

***YES** The inliner report is produced as part of the compiler listing. An OUTPUT option must be specified to produce the inliner report.

Top

Module creation options (MODCRTOPT)

Specifies the options to use when the module object is created. You can specify them in any order, separated by blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

***NOKEEPILDTA**
Intermediate language data is not stored with the module object.

***KEEPILDTA**
Intermediate language data is stored with the module object.

Top

Debugging view (DBGVIEW)

Specifies which level of debugging is available for the module in the created program or service program object. It also specifies which source views are available for source level debugging. Requesting a debug view will turn inlining off.

***NONE**
Debug capability is not inserted into the module object.

***ALL** Enables all of the debug options (*STMT, *SOURCE and *LIST)

***STMT**
Allows the module object to be debugged using program statement numbers and symbolic identifiers.

Note: To debug a module object using the *STMT option you need a listing.

***SOURCE**
Generates the source view for debugging the module object. OPTION parameter values *NOSHOWINC, *SHOWINC, *SHOWSYS, and *SHOWUSR determine the content of the source view created.

Note: In order to use this view for debugging, the root source file should not be modified, renamed or moved after the module object is created.

***LIST** Generates the listing view for debugging the module object. OPTION parameter values *SHOWINC, *SHOWUSR, *SHOWSYS, and *NOSHOWINC determine the content of the listing view created.

Top

Define names (DEFINE)

Specifies preprocessor macros that take effect before the file is processed by the compiler. Using the format DEFINE(macro) is equivalent to DEFINE('macro=1').

***NONE**
No macro is defined.

'name'* or *'name=value'
A maximum of 32 macros may be defined. Each macro name is enclosed in apostrophes. The maximum length of a macro name is 80 characters. The apostrophes are not part of the 80 character string. The apostrophes are required for case-sensitive macro names.

Note: Macros defined in the command override any macro definition of the same name in the source but a warning message is generated by the compiler. Function-like macros such as #define max(a,b) ((a)>(b):(a)?(b)) cannot be defined on the command line.

Top

Language level (LANGLVL)

Specifies the capabilities of the compiler and which prototypes are declared when the source is created.

***EXTENDED**
Defines the preprocessor variable __EXTENDED__ and undefines other language-level variables. This parameter should be used when all the capabilities of ILE C are required.

***ANSI**
Defines the preprocessor variables __ANSI__ and __STDC__ and undefines other language-level variables. Only ANSI-standard C is made available.

Note: The ILE C compiler always predefines the __ILEC400__ macro.

Top

Alias (ALIAS)

Specifies the aliasing assertion to be applied to the module object being created.

***ANSI**
The module object will only allow pointers to point to an object of the same type.

***NOANSI**
The module object will not use the *ANSI aliasing rules.

***ADDRTAKEN**
The module object will have its class of variables disjoint from pointers unless their address is taken.

***NOADDRTAKEN**
The module object will not use the *ADDRTAKEN aliasing rules.

***ALLPTRS**
The module object will not allow any two pointers to be aliased.

***NOALLPTRS**
The module object will not use the *ALLPTRS aliasing rules.

***TYPEPTR**
The module object will not allow any two pointers of different types to be aliased.

***NOTYPEPTR**
The module object will not use the *TYPEPTR aliasing rules.

Top

System interface options (SYSIFCOPT)

Specifies which system interface options will be used for the module object being created. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

***NOIFSIO**
The module object will use the iSeries Data Management file system for C stream I/O operations.

***IFSIO**
The module object will use the Integrated File System for C stream I/O operations.

***IFS64IO**
The module object will use the Integrated File System for 64-bit C stream I/O operations.

***NOASYNCSIGNAL**
Does not enable runtime mapping of synchronous signalling functions to asynchronous signalling functions.

***ASYNCSIGNAL**
Enable runtime mapping of synchronous signalling functions to asynchronous signalling functions. Specifying this option causes C runtime to map the synchronous signal() and raise() functions to the asynchronous sigaction() and kill() functions respectively.

Top

Locale object type (LOCALETYPE)

Specifies the type of locale support to be used by the module object being created.

***LOCALE**
Module objects created with this option use the locale support provided by *LOCALE objects.

***LOCALEUCS2**
Module objects created with this option use the locale support provided by *LOCALE objects. Wide-character types will contain two-byte universal character set values.

***CLD** Module objects created with this option use the locale support provided by *CLD objects.

***LOCALEUTF**
Module objects created with this option use the locale support provided by *LOCALE objects. Wide-character types will contain four-byte utf-32 values. Narrow character types will contain utf-8 values.

Top

Message flagging level (FLAG)

Specifies the level of messages that are to be displayed in the listing.

0 All messages starting at the informational level are displayed.

10 All messages starting at the warning level are displayed.

20 All messages starting at the error level are displayed.

30 All messages starting at the severe error level are displayed.

Top

Compiler messages (MSGLMT)

Specifies the maximum number of messages at the given message severity that can occur before the compilation is stopped.

Element 1: Message Limit

Specifies the maximum number of messages that can occur at, or above, the message severity level specified.

***NOMAX**
Compilation continues regardless of the number of messages that have occurred at the message severity level specified.

message-limit
Specify the number of messages that can occur. The valid range is 0 to 32767.

Element 2: Message Severity

Specifies the message severity that can stop the compilation if the *message-limit* number of messages at the specified severity or above occur.

30 Specifies that a *message-limit* of messages at severity 30 can occur before compilation stops.

0 Specifies that a *message-limit* of messages at severity 0 or above can occur before compilation stops.

10 Specifies that a *message-limit* of messages at severity 10 or above can occur before compilation stops.

20 Specifies that a *message-limit* of messages at severity 20 or above can occur before compilation stops.

Top

Replace module object (REPLACE)

Specifies whether the existing version of the object will be replaced by the current version.

***YES** The existing object is replaced by the new version. The old version is moved to the QRPLOBJ library and renamed based on the system date and time. The text description of the replaced object is changed to the name of the original object. The old object is deleted at the next IPL if it has not been explicitly deleted.

***NO** The existing object is not replaced. When an object with the same name is found in the specified library, a message is displayed and compilation stops.

Top

Authority (AUT)

Specifies the authority granted to users who do not have specific authority to the object, who are not on the authorization list, or whose group has no specific authority to the object.

***LIBCRTAUT**
Public authority for the object is taken from the CRTAUT keyword of the target library (the library that contains the created object). This value is determined when the object is created. When the CRTAUT value for the library changes after the object is created, the new value does not affect any existing objects in the library.

***ALL** Provides authority for all operations on the object except those limited to the owner or controlled by authorization list management authority. Any user can control the object's existence, specify the security for it, change it, and perform basic functions on it, including transfer its ownership.

***CHANGE**
Provides all data authority and the authority to perform all operations on the object except those limited to the owner or controlled by object authority and object management authority. The object can be changed and basic functions can be performed on it.

***USE** Provides object operational authority; read authority; and authority for basic read-only operations on the object, such as binding of a module object. Users without specific authority are prevented from changing the object.

***EXCLUDE**
Users without special authority cannot access the object.

authorization-list-name
Enter the name of an authorization list of users and authorities to which the object is added. The object is secured by this authorization list, and the public authority for the object is set to *AUTL. The authorization list must exist on the system when the command is issued.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which the user intends to use the object being created.

In the examples given for the *CURRENT and *PRV values, and when specifying the release-level value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V4R5M0 is version 4, release 5, modification level 0.

***CURRENT**
The object will be used on the release of the operating system currently running on the user's

system. For example, if V4R5M5 is running on the system, *CURRENT means the user intends to use the object on a system with V4R5M5 installed. The user can also use the object on a system with any subsequent release of the operating system installed.

Note: If V4R5M5 is running on the system, and the object will be used on a system with V4R5M0 installed, specify TGTRLS(V4R5M0) not TGTRLS(*CURRENT).

***PRV** The object will be used on the previous release with modification level 0 of the operating system. For example, if V4R5M5 is running on the user's system, *PRV means the user intends to use the object on a system with V4R4M0 installed. The user can also use the object on a system with any subsequent release of the operating system installed.

release-level
Specify the release in the format VxRxMx. The object can be used on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level. They change with each new release. If you specify a release level which is earlier than the earliest release level supported by this command, an error message is sent.

Top

Enable performance collection (ENBPFRCOL)

Specifies whether performance measurement code should be generated in the object. The data collected can be used by the system performance tool to profile an application's performance. Generating performance measurement code in a created object will result in slightly larger objects and may affect performance.

***PEP** Performance statistics are gathered on the entry and exit of the program entry procedure only. Choose this value when you want to gather overall performance information for an application.

***ENTRYEXIT *NONLEAF**
Performance statistics are gathered on the entry and exit of all procedures of the program object that are not leaf procedures. This includes the program PEP routine.

This choice would be useful if you only wanted to capture information on those routines that invoke other routines in your application.

***ENTRYEXIT *ALLPRC**
Performance statistics are gathered on the entry and exit of all the procedures of the program object (including those that are leaf procedures). This includes the program PEP routine.

This choice would be useful if you wanted to capture information on all routines. Use this option when you know that all the program objects called by your application were created with either the *PEP, *ENTRYEXIT or *FULL option. Otherwise, if your application calls other program objects that are not enabled for performance measurement, the performance tool will charge their use of resources against your application. This would make it difficult for you to determine where resources are actually being used.

***FULL *NONLEAF**
Performance statistics are gathered on entry and exit of all procedures that are not leaf procedures. Also, statistics are gathered before and after each call to an external procedure.

***FULL *ALLPRC**
Performance statistics are gathered on the entry and exit of all procedures including leaf procedures. Also statistics are gathered before and after each call to an external procedure.

Use this option when you think that your application will call other program objects that were not created with either *PEP, *ENTRYEXIT or *FULL. This option allows the performance tools to distinguish between resources that are used by your application and those used by program

objects it calls (even if those program objects are not enabled for performance measurement). This option is the most expensive but allows for selectively analyzing various program objects in an application.

Top

Performance options (PFROPT)

Specifies various options available to boost performance. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

The possible options are:

***SETFPCA**
Causes the compiler to set the floating point computational attributes to achieve ANSI semantics for floating point computations.

***NOSETFPCA**
No computational attributes will be set. This option should only be used when the object being created does not have any floating point computations in it.

***NOSTRDONLY**
Specifies that the compiler must place strings into writeable memory.

***STRDONLY**
Specifies that the compiler may place strings into read-only memory.

Top

Profiling data (PRFDTA)

Specifies the program profiling data attribute for the module object. Program profiling is an advanced optimization technique used to reorder procedures and code within the procedures based on statistical data (profiling data).

***NOCOL**
The module object is not enabled to collect profiling data.

***COL** The module object is enabled to collect profiling data. *COL can be specified only when the optimization level is 30 or greater.

Top

Teraspace options (TERASPACE)

Specifies whether the module object is enabled to work with teraspace storage. This includes teraspace storage allocated by the module object and parameters passed from other teraspace-enabled program and service program objects.

Element 1: Teraspace Enabled

***NO** The module object is not enabled to handle addressing of storage allocated from teraspace.

***YES** The module object is enabled to handle addressing of storage allocated from teraspace, including parameters passed from other teraspace-enabled program and service program objects.

Element 2: Use Teraspace Interfaces

***NOTSIFC**
The module object will default to use the non-teraspace versions of the storage functions.

***TSIFC**
The module object will default to use the teraspace versions of the storage functions. The compiler will define macro variable __TERASPACE__.

Top

Storage model (STGMDL)

Specifies the type of storage to be used by the object created.

***SNGLVL**
The object created will use single-level storage.

***TERASPACE**
The object created will use teraspace storage.

***INHERIT**
The object created can use either single level or teraspace storage. The type of storage used will depend on the type of storage required by the caller.

Top

Data model (DTAMDL)

Specifies the sizes (in bytes) of variables declared as int, long, pointer.

***P128** Causes the sizes of int, long, pointer to be 4, 4, 16 respectively.

***LLP64**
Causes the sizes of int, long, pointer to be 4, 4, 8 respectively. The compiler will define the macro __LLP64_IFC__.

Top

Pack structure (PACKSTRUCT)

Specifies the alignment boundary to use for members of a structure.

***NATURAL**
Structure members are aligned on their natural boundaries. For example, a short integer will be two-byte aligned. 16-byte pointers will always align on 16-byte boundaries.

1 Pack structure members on a 1-byte alignment.

2 Pack structure members on a 2-byte alignment.

4 Pack structure members on a 4-byte alignment.

8 Pack structure members on a 8-byte alignment.

16 Pack structure members on a 16-byte alignment.

Top

Enum size (ENUM)

Specifies the number of bytes the compiler uses to represent an enumeration.

***SMALL**
Make all enum variables the smallest size that can represent the range of values.

1 Make all enum variables 1 byte.

2 Make all enum variables 2 bytes.

4 Make all enum variables 4 bytes.

***INT** Use the ANSI-standard enum size, which is 4 bytes.

Top

Dependency information (MAKEDEP)

Specifies whether or not to generate dependency information into a file. This information can be used by a make tool.

***NONE**
Do not generate dependency information.

'path-name'
Specify a path name for the stream file in which to store the dependency information.

Top

Preprocessor options (PPGENOPT)

Specifies the preprocessor generation options to use when the source code is compiled.

The possible options are:

***NONE**
Run the entire compiler against the source file. Do not copy the preprocessor output to a file.

***DFT** Run the preprocessor against the input source. *RMVCOMMENT and *GENLINE will be used as the options for generating the preprocessor output. Use PPSRCFILE and PPSRCMBR to specify an output source file and member, or PPSRCSTMF to specify a stream file to contain the preprocessor output.

***RMVCOMMENT**
Remove comments during preprocessing.

***NORMVCOMMENT**
Preserve comments during preprocessing.

***GENLINE**
Produce #line directives in the preprocessor output.

***NOGENLINE**
Suppress #line directives from the preprocessor output.

Top

Output source file (PPSRCFILE)

Specifies the physical file name and library for the preprocessor output.

source-file-name
Specify the name of the physical file for the preprocessor output.

The possible library values are:

***CURLIB**
The preprocessor output is created in the current library. If a job does not have a current library, the preprocessor output file is created in the QGPL library.

library-name
Specify the name of the library for the preprocessor output.

Top

Output source member (PPSRCMBR)

Specifies the name of the physical file member for the preprocessor output.

***MODULE**
The name supplied on the MODULE parameter is used as the preprocessor output member name.

member-name
Specify the name of the member for the preprocessor output.

Top

Output stream file (PPSRCSTMF)

Specifies the path name of the stream file for the preprocessor output.

***SRCSTMF**
The path name supplied on the SRCSTMF parameter is used as the preprocessor output path name. The file will have the extension '.i'.

'path-name'
Specify the path name for the preprocessor output stream file.

Top

Include directory (INCDIR)

Specifies one or more directories to add to the search path used by the compiler to find include files. Use of INCDIR overrides the INCLUDE environment variable.

The search path can be further modified by using the following parameters on the OPTION keyword:

- *INCDIRFIRST or *NOINCDIRFIRST
- *SYSINCPATH or *NOSYSINCPATH
- *STDINC or *NOSTDINC

***NONE**
Unless modified, the default system include directory and the source directory will be searched for user include files.

'directory'
Specify up to 32 directories in which to search for include files. In addition to the specified directories, the source directory is searched for user include files.

Top

Compiler services option (CSOPT)

Specifies one or more compiler service options. This parameter allows IBM to provide switchable compiler capability between releases.

***NONE**
No compiler service option is selected.

'compiler-service-options-string'
The selected compiler service options are used when creating the module object. Valid strings will be described in PTF cover letters or in release notes.

Top

Licensed internal code options (LICOPT)

Specifies one or more Licensed Internal Code compile-time options. This parameter allows individual compile-time options to be selected, and is intended for the advanced programmer who understands the potential benefits and drawbacks of each selected type of compiler option.

The possible values are:

***NONE**
No compile-time options are selected.

'Licensed-Internal-Code-options-string'
The selected Licensed Internal Code compile-time options are used when creating the module object. Certain options may reduce your ability to debug the created module object.

Top

Default char type (DFTCHAR)

Specifies the default sign for the char data type.

***UNSIGNED**
Make default char type unsigned.

***SIGNED**
Make default char type signed.

Top

Target CCSID (TGTCCSID)

Specifies the target coded character set identifier used to describe data stored into the resulting module object.

The possible values are:

***SOURCE**
The root source file's CCSID is used.

***JOB** The current job's CCSID is used.

***HEX** The CCSID 65535 is used, which indicates that character data is treated as binary data and is not converted.

coded-character-set-identifier
Specify the CCSID to be used.

Top

Examples

None

Top

Error messages

***ESCAPE Messages**

CZM0613
The compilation failed.

Top

Create C++ Module (CRTCPPMOD)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Create C++ Module (CRTCPPMOD) command starts the ILE C++ compiler. You can use this command in either batch or interactive mode, or from a CL program. The compiler attempts to create a module object based on the ILE C++ statements in the source code.

Error messages for CRTCPPMOD

***ESCAPE Messages**

CZS0613
The compilation failed.

Top

Parameters

Keyword	Description	Choices	Notes
MODULE	Module	*Qualified object name*	Required, Positional 1
	Qualifier 1: Module	*Name*	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QCPPSRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***MODULE**	Optional, Positional 3
SRCSTMF	Source stream file	*Path name*	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OUTPUT	Output options	Single values: ***NONE**, '*none' Other values: *Element list*	Optional
	Element 1: Output file name	*Path name*, *PRINT, '*print'	
	Element 2: Title	*Character value*, ***BLANK**	
	Element 3: Subtitle	*Character value*, ***BLANK**	
OPTION	Compiler options	Values (up to 35 repetitions): *NOBITSIGN, *BITSIGN, *NOEVENTF, *EVENTF, *NOEXPMAC, *EXPMAC, *NOFULL, *FULL, *GEN, *NOGEN, *NOINCDIRFIRST, *INCDIRFIRST, *LOGMSG, *NOLOGMSG, *LONGLONG, *NOLONGLONG, *NORTTI, *RTTIALL, *RTTITYPE, *RTTICAST, *NOSHOWINC, *SHOWINC, *SHOWSRC, *NOSHOWSRC, *NOSHOWSYS, *SHOWSYS, *NOSHOWUSR, *SHOWUSR, *STDINC, *NOSTDINC, *NOSTDLOGMSG, *STDLOGMSG, *NOSYSINCPATH, *SYSINCPATH, *NOXREF, *XREF, *NOXREFREF, *XREFREF	Optional

Keyword	Description	Choices	Notes
CHECKOUT	Checkout options	Values (up to 45 repetitions): ***NONE**, *USAGE, *ALL, *NOCLASS, *CLASS, *NOCOND, *COND, *NOEFFECT, *EFFECT, *NOGENERAL, *GENERAL, *NOLANG, *LANG, *NOPARM, *PARM, *NOPORT, *PORT, *NOREACH, *REACH, *NOTEMP, *TEMP, *NOTRUNC, *TRUNC, *NOUNUSED, *UNUSED	Optional
OPTIMIZE	Optimization	**10**, 20, 30, 40	Optional
INLINE	Inline options	*Element list*	Optional
	Element 1: Inliner	***OFF**, *ON	
	Element 2: Mode	***NOAUTO**, *AUTO	
	Element 3: Threshold	1-65535, **250**, *NOLIMIT	
	Element 4: Limit	1-65535, **2000**, *NOLIMIT	
	Element 5: Report	***NO**, *YES	
MODCRTOPT	Module creation options	***NOKEEPILDTA**, *KEEPILDTA	Optional
DBGVIEW	Debugging view	***NONE**, *ALL, *STMT, *SOURCE, *LIST	Optional
DEFINE	Define names	Single values: ***NONE** Other values (up to 32 repetitions): *Character value*	Optional
LANGLVL	Language level	***EXTENDED**, *ANSI, *LEGACY	Optional
ALIAS	Alias	Values (up to 3 repetitions): ***ANSI**, *NOANSI, *ADDRTAKEN, *NOADDRTAKEN, *ALLPTRS, *NOALLPTRS, *TYPEPTR, *NOTYPEPTR	Optional
SYSIFCOPT	System interface options	***IFS64IO**, *IFSIO, *NOIFSIO	Optional
LOCALETYPE	Locale object type	***LOCALE**, *LOCALEUCS2, *LOCALEUTF	Optional
FLAG	Message flagging level	**0**, 10, 20, 30	Optional
MSGLMT	Compiler messages	*Element list*	Optional
	Element 1: Message limit	0-32767, ***NOMAX**	
	Element 2: Message limit severity	0, 10, 20, **30**	
REPLACE	Replace module object	***YES**, *NO	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
ENBPFRCOL	Enable performance collection	*Element list*	Optional
	Element 1: Collection level	***PEP**, *ENTRYEXIT, *FULL	
	Element 2: Procedures	*NONLEAF, *ALLPRC	
PFROPT	Performance options	Values (up to 2 repetitions): *SETFPCA, *NOSETFPCA, *NOSTRDONLY, *STRDONLY	Optional
PRFDTA	Profiling data	***NOCOL**, *COL	Optional
TERASPACE	Teraspace options	Single values: ***NO** Other values: *Element list*	Optional
	Element 1: Teraspace enabled	***YES**	
	Element 2: Use teraspace interfaces	***NOTSIFC**, *TSIFC	
STGMDL	Storage model	***SNGLVL**, *TERASPACE, *INHERIT	Optional
DTAMDL	Data model	***P128**, *LLP64	Optional
RTBND	Run time Binding	***DEFAULT**, *LLP64	Optional
PACKSTRUCT	Pack structure	***NATURAL**, 1, 2, 4, 8, 16	Optional
ENUM	Enum size	***SMALL**, 1, 2, 4, *INT	Optional
MAKEDEP	Dependency information	*Path name*, ***NONE**	Optional

Keyword	Description	Choices	Notes
PPGENOPT	Preprocessor options	Single values: ***NONE**, *DFT Other values (up to 2 repetitions): *RMVCOMMENT, *NORMVCOMMENT, *GENLINE, *NOGENLINE	Optional
PPSRCFILE	Output source file	*Qualified object name*	Optional
	Qualifier 1: Output source file	*Name*	
	Qualifier 2: Library	*Name*, ***CURLIB**	
PPSRCMBR	Output source member	*Name*, ***MODULE**	Optional
PPSRCSTMF	Output stream file	*Path name*, *SRCSTMF	Optional
INCDIR	Include directory	Single values: ***NONE** Other values (up to 32 repetitions): *Path name*	Optional
CSOPT	Compiler services option	*Character value*, ***NONE**	Optional
LICOPT	Licensed internal code options	*Character value*, ***NONE**	Optional
DFTCHAR	Default char type	***UNSIGNED**, *SIGNED	Optional
TGTCCSID	Target CCSID	1-65535, ***SOURCE**, *JOB, *HEX	Optional
TEMPLATE	Template options	*Element list*	Optional
	Element 1: Temporary include directory	*Path name*, ***NONE**, *TEMPINC	
	Element 2: Maximum generated headers	1-99999, **1**	
	Element 3: Template validity checking	***NO**, *WARN, *ERROR	
TMPLREG	Template registry	*Path name*, *DFT, ***NONE**	Optional
WEAKTMPL	Weak Template Definitions	***YES**, *NO	Optional

Top

Module (MODULE)

Specifies the module name and library for the module object.

module-name
Enter a name for the module object.

The possible library values are:

***CURLIB**
The module object is stored in the current library. If a job does not have a current library, the module object is created in the QGPL library.

library-name
Enter the name of the library where the module object will be stored.

Top

Source file (SRCFILE)

Specifies the source file name and library of the file containing the ILE C++ source code that you want to compile.

QCPPSRC
The source file named QCPPSRC contains the member with the ILE C++ source code that you want to compile.

source-file-name
Enter the name of the source file that contains the member with the ILE C++ source code.

The possible library values are:

***LIBL** The library list is searched to find the library where the source file is located.

***CURLIB**
The current library is searched for the source file. If a job does not have a current library, QGPL is searched for the source file.

library-name
Enter the name of the library that contains the source file.

Top

Source member (SRCMBR)

Specifies the name of the member containing the source code to be compiled.

***MODULE**
The module name supplied on the MODULE parameter is used as the source member name.

member-name
Enter the name of the member that contains the source code.

Top

Source stream file (SRCSTMF)

Specifies the path name of the stream file containing the source code that you want to compile.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'. If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by pre-pending the job's current working directory to the path name.

The SRCMBR and SRCFILE parameters cannot be specified with the SRCSTMF parameter.

Top

Text 'description' (TEXT)

Specifies the text that briefly describes the module object.

***SRCMBRTXT**
The text description associated with the source file member is used for the module object. If the source file is an inline file, a stream file, or a device file, the text will be blank.

***BLANK**
Specifies that no text appears.

'description'
Specify no more than 50 characters of text, enclosed in apostrophes.

Top

Output options (OUTPUT)

Specifies whether a compiler listing is produced.

Single Value

***NONE**
Does not generate the compiler listing. When a listing is not required, this parameter value should be used to improve compile-time performance. When *NONE is specified, any listing-related parameter values specified for the OPTION parameter are ignored.

Element 1: Output File Name

***PRINT**
Generates a spooled file containing the listing.

'path-name'
Specify the path name of a stream file to hold the listing.

Element 2: Title

***BLANK**
Specifies that no text appears.

'title' Specify a title string for the listing file (maximum 80 characters).

Element 3: Subtitle

***BLANK**
Specifies that no text appears.

'subtitle'
Specify a subtitle string for the listing file (maximum 80 characters).

Top

Compiler options (OPTION)

Specifies the options to use when the ILE C++ source code is compiled. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

The possible options are:

***NOBITSIGN**
Specify bitfields as unsigned.

***BITSIGN**
Specify bitfields as signed.

***NOEVENTF**
Do not create an event file for use by CoOperative Development Environment/400 (CODE/400).

***EVENTF**
Create an event file for use by CoOperative Development Environment/400 (CODE/400). The event file is created as a member in file EVFEVENT in the library where the module or program object being created will be stored. If the file EVFEVENT does not exist, it is automatically created. The event file member name is the same as the name of the object being created. An

Event File is normally created when you create a module or program object from within CODE/400. CODE/400 uses this file to provide error feedback integrated with the CODE/400 editor.

***NOEXPMAC**
Macros are not expanded in the listing unless a syntax error is encountered within the macro.

***EXPMAC**
Expand all macros in the listing.

***NOFULL**
Do not turn on all listing options.

***FULL** Turn on all listing options.

***GEN** All phases of the compilation process are carried out.

***NOGEN**
Compilation stops after syntax checking. No module object is created.

***NOINCDIRFIRST**
Include directories specified as INCDIR parameters are not included before the standard header file include path.

***INCDIRFIRST**
Include directories specified as INCDIR parameters are included before the standard header file include path.

***LOGMSG**
Puts the compilation messages in the job log.

When you specify this option and the FLAG parameter, messages with the severity specified on the FLAG parameter (and higher) are placed in the job log.

When you specify this option and a maximum number of messages on the MSGLMT parameter, compilation stops when the number of messages, at the specified severity, have been placed in the job log.

***NOLOGMSG**
Does not put the compilation messages in the job log.

***LONGLONG**
Allows the use of the long long data type.

***NOLONGLONG**
Do not allow the use of the long long data type.

***NORTTI**
Do not generate run-time type identification (RTTI) information.

***RTTIALL**
Generate the information needed for the RTTI typeid and the dynamic_cast operator.

***RTTITYPE**
Generate the information needed for the RTTI typeid operator only.

***RTTICAST**
Generate the information needed for the dynamic_cast operator only.

***NOSHOWINC**
Does not expand the user include files or the system include files in the source section of the listing or in the debug views.

***SHOWINC**
Expands both the user include files and the system include files in the source section of the listing or in the debug views. An OUTPUT option, or DBGVIEW parameter value of *ALL, *SOURCE or *LIST must be specified.

***SHOWSRC**
Show the source code in the listing. This option can be modified by the *SHOWINC, *SHOWSYS or *SHOWUSR options.

***NOSHOWSRC**
Does not show the source code in the listing. This option may be modified by the *EXPMAC, *SHOWINC, *SHOWSYS or *SHOWUSR options.

***NOSHOWSYS**
Do not expand the system include files on the #include directive in the source section of the listing or in the debug views. System include files are enclosed in angle brackets (< >) following the #include directive.

***SHOWSYS**
Expands the system include files on the #include directive in the source section of the listing or in the debug views. An output option, or DBGVIEW parameter of *ALL, *SOURCE or *LIST must be specified. System include files are enclosed in angle brackets (< >) following the #include directive.

***NOSHOWUSR**
Do not expand user include files in the listing or debug views. User include files are enclosed in double quotation marks (″ ″) following the #include directive.

***SHOWUSR**
Expands the user include files on the #include directive in the source section of the listing or in the debug views. An OUTPUT option, or DBGVIEW parameter value of *ALL, *SOURCE or *LIST must be specified. User include files are enclosed in double quotation marks (″ ″) following the #include directive.

***STDINC**
The system-supplied header files are included in the search path for the compile.

***NOSTDINC**
The system-supplied header files are not included in the search path for the compile.

***NOSTDLOGMSG**
Compilation messages are not sent to the stdout stream.

***STDLOGMSG**
Compilation messages are sent to the stdout stream.

***NOSYSINCPATH**
The search path for user includes is not affected.

***SYSINCPATH**
Changes the search path of user includes to the system include search path. In function this option is equivalent to changing the double-quotes in the user #include directive (#include ″file_name″) to angle brackets (#include <file_name>).

***NOXREF**
Does not generate the cross-reference table in the listing.

***XREF** Generates the cross-reference table containing a list of the identifiers in the source code together with the numbers of the lines in which they appear. An OUTPUT option must be specified.

***NOXREFREF**
Do not produce a cross-reference table of referenced identifiers in the listing.

***XREFREF**
Produce a cross-reference table of referenced variables, structures, and function names in the listing file. The table shows the line numbers where the identifiers are declared. An OUTPUT option must be specified.

Top

Checkout options (CHECKOUT)

Specifies options you may select to generate informational messages that indicate possible programming errors. When you specify an option more than once, or when two options conflict, the last one specified is used.

***NONE**
Disables all of the options for CHECKOUT.

***USAGE**
This is equivalent to specifying *COND.

***ALL** Enables all of the options for CHECKOUT.

***NOCLASS**
Do not display information about class use.

***CLASS**
Display information about class use.

***NOCOND**
Do not warn about possible redundancies or problems in conditional expressions.

***COND**
Warn about possible redundancies or problems in conditional expressions.

***NOEFFECT**
Do not warn about statements with no effect.

***EFFECT**
Warn about statements with no effect.

***NOGENERAL**
Do not generate general checkout messages.

***GENERAL**
Generate general checkout messages.

***NOLANG**
Do not display information about the effects of the language level.

***LANG**
Display information about the effects of the language level.

***NOPARM**
Do not warn about unused parameters.

***PARM**
Warn about unused parameters.

***NOPORT**
Do not warn about non-portable language constructs.

***PORT**
Warn about non-portable language constructs.

***NOREACH**
Do not warn about unreachable statements.

***REACH**
Warn about unreachable statements.

***NOTEMP**
Do not generate messages if the compiler creates temporary variables.

***TEMP**
Generate messages if the compiler creates temporary variables.

***NOTRUNC**
Do not warn about the possible truncation or loss of data.

***TRUNC**
Warn about the possible truncation or loss of data.

***NOUNUSED**
Do not check for unused auto or static variables.

***UNUSED**
Check for unused auto or static variables.

Top

Optimization (OPTIMIZE)

Specifies the levels of optimization of the generated object.

10 Generated code is not optimized. This level has the shortest compile time. This level allows variables to be displayed and modified while debugging.

20 Some optimization is performed on the code. This level allows user variables to be displayed but not modified while debugging.

30 Full optimization is performed on the generated code. During a debug session, user variables may not be modified but may be displayed. The presented values may not be the current value of the variable.

40 All optimizations done at level 30 are performed on the generated code. In addition, code is eliminated from procedure prologue and epilogue routines that enable instruction trace and call trace system functions. Eliminating this code enables the creation of leaf procedures. A leaf procedure is a procedure that contains no calls to other procedures. Procedure call performance to a leaf procedure is significantly faster than to a normal procedure.

Top

Inline options (INLINE)

Specifies whether the compiler should consider replacing a function call with the called function's instructions. Inlining a function eliminates the overhead of a call and can result in better optimization. Small functions that are called many times are good candidates for inlining.

Element 1: Inliner

Specifies whether or not inlining will be used.

***OFF** Specifies that inlining will not be performed on the compilation unit.

***ON** Specifies that inlining will be performed on the compilation unit. If a debug view is specified, the inliner is turned off.

Element 2: Mode

Specifies whether or not the inliner should attempt to automatically inline functions depending on their Threshold and Limit values.

***NOAUTO**
Specifies that only the functions that have been specified with the #pragma inline directive should be considered candidates for inlining.

***AUTO**
Specifies that the inliner should determine if a function can be inlined based on the Threshold and Limit values specified. The #pragma noinline directive overrides *AUTO.

Element 3: Threshold

Specifies the maximum size of a function that can be a candidate for automatic inlining. The size is measured in Abstract Code Units (ACUs). ACUs are proportional in size to the executable code in the function. Source code is translated into ACUs by the compiler.

250 Specifies a threshold of 250.

number-of-ACUs
Specifies a threshold from 1 to 65535 ACUs.

***NOLIMIT**
Defines the threshold as the maximum size of the program object.

Element 4: Limit

Specifies the maximum relative size a function can grow before auto-inlining stops.

2000 Specifies a limit of 2000 ACUs.

***NOLIMIT**
Limit is defined as the maximum size of the program object. System limits may be encountered.

number-of-ACUs
A limit from 1 to 65535 ACUs may be specified.

Element 5: Report

Specifies whether or not to produce an inliner report with the compiler listing.

***NO** The inliner report is not produced.

***YES** The inliner report is produced as part of the compiler listing. An OUTPUT option must be specified to produce the inliner report.

Top

Module creation options (MODCRTOPT)

Specifies the options to use when the module object is created. You can specify them in any order, separated by blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

***NOKEEPILDTA**
Intermediate language data is not stored with the module object.

***KEEPILDTA**
Intermediate language data is stored with the module object.

Top

Debugging view (DBGVIEW)

Specifies which level of debugging is available for the module in the created program or service program object. It also specifies which source views are available for source level debugging. Requesting a debug view will turn inlining off.

***NONE**
Debug capability is not inserted into the module object.

***ALL** Enables all of the debug options (*STMT, *SOURCE and *LIST)

***STMT**
Allows the module object to be debugged using program statement numbers and symbolic identifiers.

Note: To debug a module object using the *STMT option you need a listing.

***SOURCE**
Generates the source view for debugging the module object. OPTION parameter values *NOSHOWINC, *SHOWINC, *SHOWSYS, and *SHOWUSR determine the content of the source view created.

Note: In order to use this view for debugging, the root source file should not be modified, renamed or moved after the module object is created.

***LIST** Generates the listing view for debugging the module object. OPTION parameter values *SHOWINC, *SHOWUSR, *SHOWSYS, and *NOSHOWINC determine the content of the listing view created.

Top

Define names (DEFINE)

Specifies preprocessor macros that take effect before the file is processed by the compiler. Using the format DEFINE(macro) is equivalent to DEFINE('macro=1').

***NONE**
No macro is defined.

'name'* or *'name=value'
A maximum of 32 macros may be defined. Each macro name is enclosed in apostrophes. The maximum length of a macro name is 80 characters. The apostrophes are not part of the 80 character string. The apostrophes are required for case-sensitive macro names.

Note: Macros defined in the command override any macro definition of the same name in the source but a warning message is generated by the compiler. Function-like macros such as #define max(a,b) ((a)>(b):(a)?(b)) cannot be defined on the command line.

Top

Language level (LANGLVL)

Specifies the capabilities of the compiler and which prototypes are declared when the source is created.

***EXTENDED**
Defines the preprocessor variable __EXTENDED__ and undefines other language-level variables. This parameter should be used when all the capabilities of ILE C++ are required.

***ANSI**
Defines the preprocessor variables __ANSI__, __STDC__ and __cplusplus98_interface__, and undefines other language-level variables. Only ANSI-standard C++ is made available.

***LEGACY**
This option allows some of the source constructs acceptable to earlier compilers.

Top

Alias (ALIAS)

Specifies the aliasing assertion to be applied to the module object being created.

***ANSI**
The module object will only allow pointers to point to an object of the same type.

***NOANSI**
The module object will not use the *ANSI aliasing rules.

***ADDRTAKEN**
The module object will have its class of variables disjoint from pointers unless their address is taken.

***NOADDRTAKEN**
The module object will not use the *ADDRTAKEN aliasing rules.

***ALLPTRS**
The module object will not allow any two pointers to be aliased.

***NOALLPTRS**
The module object will not use the *ALLPTRS aliasing rules.

***TYPEPTR**
The module object will not allow any two pointers of different types to be aliased.

***NOTYPEPTR**
The module object will not use the *TYPEPTR aliasing rules.

Top

System interface options (SYSIFCOPT)

Specifies which system interface options will be used for the module object being created. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

***IFS64IO**
The module object will use the Integrated File System for 64-bit C stream I/O operations.

***IFSIO**
The module object will use the Integrated File System for C stream I/O operations.

***NOIFSIO**
The module object will use the iSeries Data Management file system for C stream I/O operations.

Top

Locale object type (LOCALETYPE)

Specifies the type of locale support to be used by the module object being created.

***LOCALE**
Module objects created with this option use the locale support provided by *LOCALE objects.

***LOCALEUCS2**
Module objects created with this option use the locale support provided by *LOCALE objects. Wide-character types will contain two-byte universal character set values.

***LOCALEUTF**
Module objects created with this option use the locale support provided by *LOCALE objects. Wide-character types will contain four-byte utf-32 values. Narrow character types will contain utf-8 values.

Top

Message flagging level (FLAG)

Specifies the level of messages that are to be displayed in the listing.

0 All messages starting at the informational level are displayed.

10 All messages starting at the warning level are displayed.

20 All messages starting at the error level are displayed.

30 All messages starting at the severe error level are displayed.

Top

Message limit (MSGLMT)

Specifies the maximum number of messages at the given message severity that can occur before the compilation is stopped.

Element 1: Message Limit

Specifies the maximum number of messages that can occur at, or above, the message severity level specified.

***NOMAX**
Compilation continues regardless of the number of messages that have occurred at the message severity level specified.

message-limit
Specify the number of messages that can occur. The valid range is 0 to 32767.

Element 2: Message Severity

Specifies the message severity that can stop the compilation if the *message-limit* number of messages at the specified severity or above occur.

30 Specifies that a *message-limit* of messages at severity 30 can occur before compilation stops.

0 Specifies that a *message-limit* of messages at severity 0 or above can occur before compilation stops.

10 Specifies that a *message-limit* of messages at severity 10 or above can occur before compilation stops.

20 Specifies that a *message-limit* of messages at severity 20 or above can occur before compilation stops.

Top

Replace module object (REPLACE)

Specifies whether the existing version of the object will be replaced by the current version.

***YES** The existing object is replaced by the new version. The old version is moved to the QRPLOBJ library and renamed based on the system date and time. The text description of the replaced object is changed to the name of the original object. The old object is deleted at the next IPL if it has not been explicitly deleted.

***NO** The existing object is not replaced. When an object with the same name is found in the specified library, a message is displayed and compilation stops.

Top

Authority (AUT)

Specifies the authority granted to users who do not have specific authority to the object, who are not on the authorization list, or whose group has no specific authority to the object.

***LIBCRTAUT**
Public authority for the object is taken from the CRTAUT keyword of the target library (the library that contains the created object). This value is determined when the object is created. When the CRTAUT value for the library changes after the object is created, the new value does not affect any existing objects in the library.

***ALL** Provides authority for all operations on the object except those limited to the owner or controlled by authorization list management authority. Any user can control the object's existence, specify the security for it, change it, and perform basic functions on it, including transfer its ownership.

***CHANGE**
Provides all data authority and the authority to perform all operations on the object except those limited to the owner or controlled by object authority and object management authority. The object can be changed and basic functions can be performed on it.

***USE** Provides object operational authority; read authority; and authority for basic read-only operations on the object, such as binding of a module object. Users without specific authority are prevented from changing the object.

***EXCLUDE**
Users without special authority cannot access the object.

authorization-list-name
Enter the name of an authorization list of users and authorities to which the object is added. The object is secured by this authorization list, and the public authority for the object is set to *AUTL. The authorization list must exist on the system when the command is issued.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which the user intends to use the object being created.

In the examples given for the *CURRENT and *PRV values, and when specifying the release-level value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V4R5M0 is version 4, release 5, modification level 0.

***CURRENT**
The object will be used on the release of the operating system currently running on the user's system. For example, if V4R5M5 is running on the system, *CURRENT means the user intends to use the object on a system with V4R5M5 installed. The user can also use the object on a system with any subsequent release of the operating system installed.

Note: If V4R5M5 is running on the system, and the object will be used on a system with V4R5M0 installed, specify TGTRLS(V4R5M0) not TGTRLS(*CURRENT).

***PRV** The object will be used on the previous release with modification level 0 of the operating system. For example, if V4R5M5 is running on the user's system, *PRV means the user intends to use the object on a system with V4R4M0 installed. The user can also use the object on a system with any subsequent release of the operating system installed.

release-level
Specify the release in the format VxRxMx. The object can be used on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level. They change with each new release. If you specify a release level which is earlier than the earliest release level supported by this command, an error message is sent.

Top

Enable performance collection (ENBPFRCOL)

Specifies whether performance measurement code should be generated in the object. The data collected can be used by the system performance tool to profile an application's performance. Generating performance measurement code in a created object will result in slightly larger objects and may affect performance.

***PEP** Performance statistics are gathered on the entry and exit of the program entry procedure only. Choose this value when you want to gather overall performance information for an application.

***ENTRYEXIT *NONLEAF**
Performance statistics are gathered on the entry and exit of all procedures of the program object that are not leaf procedures. This includes the program PEP routine.

This choice would be useful if you only wanted to capture information on those routines that invoke other routines in your application.

***ENTRYEXIT *ALLPRC**
Performance statistics are gathered on the entry and exit of all the procedures of the program object (including those that are leaf procedures). This includes the program PEP routine.

This choice would be useful if you wanted to capture information on all routines. Use this option when you know that all the program objects called by your application were created with either the *PEP, *ENTRYEXIT or *FULL option. Otherwise, if your application calls other program objects that are not enabled for performance measurement, the performance tool will charge their use of resources against your application. This would make it difficult for you to determine where resources are actually being used.

***FULL *NONLEAF**
Performance statistics are gathered on entry and exit of all procedures that are not leaf procedures. Also, statistics are gathered before and after each call to an external procedure.

***FULL *ALLPRC**
Performance statistics are gathered on the entry and exit of all procedures including leaf procedures. Also statistics are gathered before and after each call to an external procedure.

Use this option when you think that your application will call other program objects that were not created with either *PEP, *ENTRYEXIT or *FULL. This option allows the performance tools to distinguish between resources that are used by your application and those used by program objects it calls (even if those program objects are not enabled for performance measurement). This option is the most expensive but allows for selectively analyzing various program objects in an application.

Top

Performance options (PFROPT)

Specifies various options available to boost performance. You can specify them in any order, separated by one or more blanks. When an option is specified more than once, or when two options conflict, the last one specified is used.

The possible options are:

***SETFPCA**
Causes the compiler to set the floating point computational attributes to achieve ANSI semantics for floating point computations.

***NOSETFPCA**
No computational attributes will be set. This option should only be used when the object being created does not have any floating point computations in it.

***NOSTRDONLY**
Specifies that the compiler must place strings into writeable memory.

***STRDONLY**
Specifies that the compiler may place strings into read-only memory.

Top

Profiling data (PRFDTA)

Specifies the program profiling data attribute for the module object. Program profiling is an advanced optimization technique used to reorder procedures and code within the procedures based on statistical data (profiling data).

***NOCOL**
The module object is not enabled to collect profiling data.

***COL** The module object is enabled to collect profiling data. *COL can be specified only when the optimization level is 30 or greater.

Top

Teraspace options (TERASPACE)

Specifies whether the module object is enabled to work with teraspace storage. This includes teraspace storage allocated by the module object and parameters passed from other teraspace-enabled program and service program objects.

Element 1: Teraspace Enabled

***NO** The module object is not enabled to handle addressing of storage allocated from teraspace.

***YES** The module object is enabled to handle addressing of storage allocated from teraspace, including parameters passed from other teraspace-enabled program and service program objects.

Element 2: Use Teraspace Interfaces

***NOTSIFC**
The module object will default to use the non-teraspace versions of the storage functions.

***TSIFC**
The module object will default to use the teraspace versions of the storage functions. The compiler will define macro variable __TERASPACE__.

Top

Storage model (STGMDL)

Specifies the type of storage to be used by the object created.

***SNGLVL**
The object created will use single-level storage.

***TERASPACE**
The object created will use teraspace storage.

***INHERIT**
The object created can use either single level or teraspace storage. The type of storage used will depend on the type of storage required by the caller.

Top

Data model (DTAMDL)

Specifies the sizes (in bytes) of variables declared as int, long, pointer.

***P128** Causes the sizes of int, long, pointer to be 4, 4, 16 respectively.

***LLP64**
Causes the sizes of int, long, pointer to be 4, 4, 8 respectively. The compiler will define the macro __LLP64_IFC__.

Top

Run time Binding (RTBND)

Specifies the run time binding directory for the object created.

***DEFAULT**
The object created will use the default binding directory.

***LLP64**
The object created will use the 64-bit run time binding directory. This value is only available in conjunction with teraspace storage model, 64-bit data model and teraspace storage functions interface options. The compiler will define the macro __LLP64_RTBND__.

Top

Pack structure (PACKSTRUCT)

Specifies the alignment boundary to use for members of a structure.

***NATURAL**
Structure members are aligned on their natural boundaries. For example, a short integer will be two-byte aligned. 16-byte pointers will always align on 16-byte boundaries.

1 Pack structure members on a 1-byte alignment.

2 Pack structure members on a 2-byte alignment.

4 Pack structure members on a 4-byte alignment.

8 Pack structure members on a 8-byte alignment.

16 Pack structure members on a 16-byte alignment.

Top

Enum size (ENUM)

Specifies the number of bytes the compiler uses to represent an enumeration.

***SMALL**
Make all enum variables the smallest size that can represent the range of values.

1 Make all enum variables 1 byte.

2 Make all enum variables 2 bytes.

4 Make all enum variables 4 bytes.

***INT** Use the ANSI-standard enum size, which is 4 bytes.

Top

Dependency information (MAKEDEP)

Specifies whether or not to generate dependency information into a file. This information can be used by a make tool.

***NONE**
Do not generate dependency information.

'path-name'
Specify a path name for the stream file in which to store the dependency information.

Top

Preprocessor options (PPGENOPT)

Specifies the preprocessor generation options to use when the source code is compiled.

The possible options are:

***NONE**
Run the entire compiler against the source file. Do not copy the preprocessor output to a file.

***DFT** Run the preprocessor against the input source. *RMVCOMMENT and *GENLINE will be used as

the options for generating the preprocessor output. Use PPSRCFILE and PPSRCMBR to specify an output source file and member, or PPSRCSTMF to specify a stream file to contain the preprocessor output.

***RMVCOMMENT**
Remove comments during preprocessing.

***NORMVCOMMENT**
Preserve comments during preprocessing.

***GENLINE**
Produce #line directives in the preprocessor output.

***NOGENLINE**
Suppress #line directives from the preprocessor output.

Top

Output source file (PPSRCFILE)

Specifies the physical file name and library for the preprocessor output.

source-file-name
Specify the name of the physical file for the preprocessor output.

The possible library values are:

***CURLIB**
The preprocessor output is created in the current library. If a job does not have a current library, the preprocessor output file is created in the QGPL library.

library-name
Specify the name of the library for the preprocessor output.

Top

Output source member (PPSRCMBR)

Specifies the name of the physical file member for the preprocessor output.

***MODULE**
The name supplied on the MODULE parameter is used as the preprocessor output member name.

member-name
Specify the name of the member for the preprocessor output.

Top

Output stream file (PPSRCSTMF)

Specifies the path name of the stream file for the preprocessor output.

***SRCSTMF**
The path name supplied on the SRCSTMF parameter is used as the preprocessor output path name. The file will have the extension '.i'.

'path-name'
Specify the path name for the preprocessor output stream file.

Top

Include directory (INCDIR)

Specifies one or more directories to add to the search path used by the compiler to find include files.

The search path can be further modified by using the following parameters on the OPTION keyword:

- *INCDIRFIRST or *NOINCDIRFIRST
- *SYSINCPATH or *NOSYSINCPATH
- *STDINC or *NOSTDINC

***NONE**
Unless modified, the default system include directory and the source directory will be searched for user include files.

'directory'
Specify up to 32 directories in which to search for include files. In addition to the specified directories, the source directory is searched for user include files.

Top

Compiler services option (CSOPT)

Specifies one or more compiler service options. This parameter allows IBM to provide switchable compiler capability between releases.

***NONE**
No compiler service option is selected.

'compiler-service-options-string'
The selected compiler service options are used when creating the module object. Valid strings will be described in PTF cover letters or in release notes.

Top

Licensed internal code options (LICOPT)

Specifies one or more Licensed Internal Code compile-time options. This parameter allows individual compile-time options to be selected, and is intended for the advanced programmer who understands the potential benefits and drawbacks of each selected type of compiler option.

The possible values are:

***NONE**
No compile-time options are selected.

'Licensed-Internal-Code-options-string'
The selected Licensed Internal Code compile-time options are used when creating the module object. Certain options may reduce your ability to debug the created module object.

Top

Default char type (DFTCHAR)

Specifies the default sign for the char data type.

***UNSIGNED**
Make default char type unsigned.

***SIGNED**
Make default char type signed.

Top

Target CCSID (TGTCCSID)

Specifies the target coded character set identifier used to describe data stored into the resulting module object.

The possible values are:

***SOURCE**
The root source file's CCSID is used.

***JOB** The current job's CCSID is used.

***HEX** The CCSID 65535 is used, which indicates that character data is treated as binary data and is not converted.

coded-character-set-identifier
Specify the CCSID to be used.

Top

Template options (TEMPLATE)

Specifies template options to the compiler.

Element 1: Template Include Directory

***NONE**
Templates are not generated.

***TEMPINC**
Templates are generated into a directory named tempinc which is created in the directory where the root source file was found. If the source file is not a stream file, a file named TEMPINC will be created in the library where the source file resides. The TEMPLATE(*TEMPINC) and TMPLREG parameters are mutually exclusive.

'directory'
Specify the directory where the compiler will generate templates.

Element 2: Maximum Generated Headers

1 The maximum number of generated headers that contain templates.

number-of-header-files
Specify an integer value from 1 to 99999 for the maximum number of generated header files.

Element 3: Template Validity Checking

Controls whether parsing and semantic checking are applied to template definition implementations or only to template instantiations. The compiler has the option to producte warning or error messages. Available parameters are:

***NO** Do not parse to reduce the number of errors issued in code written for previous versions of the compiler.

***WARN**
Issue warning messages for semantic errors. Issue error messages for errors found while parsing.

***ERROR**
Treat problems in template implementations as errors, even if the template is not instantiated.

Top

Template registry (TMPLREG)

Maintains a record of all templates as they are encountered in the source and ensures that only one instantiation of each template is made. The TMPLREG and TEMPLATE(*TEMPINC) parameters are mutually exclusive.

The possible values are:

***NONE**
Do not use the template registry file to keep track of template information.

***DFT** If the source file is a stream file, the template registry file is created in the source directory with the default name 'templateregistry'. If the source file is not a stream file, a file QTMPLREG with the member QTMPLREG will be created in the library where the source resides.

'path-name'
Specify a path name for the stream file in which to store the template registry information.

Top

Weak Template Definitions (WEAKTMPL)

Specifies whether or not weak definitions are used for static members of a template class. Weakly defined static members of a template class will prevent collisions of multiple definitions in a program or service program object.

***YES** Weak definitions will be used for static members of a template class.

***NO** Weak definitions will not be used for static members of a template class.

Top

Examples

None

Top

Error messages

***ESCAPE Messages**

CZS0613
The compilation failed.

Top

Create DFU Display File (CRTDFUDSPF)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Control Language (CL) command CRTDFUDSPF creates a DFU display file.

Error messages for CRTDFUDSPF

None

Top

Parameters

Keyword	Description	Choices	Notes
FILE	Display file	*Qualified object name*	Required, Positional 1
	Qualifier 1: Display file	*Name*	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Required, Positional 2
	Qualifier 1: Source file	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***FILE**	Optional, Positional 3
AUTH	Authority	*Name*, ***CHANGE**, *ALL, *USE, *EXCLUDE	Optional
REPLACE	Replace	***YES**, *NO	Optional
PRINT	Print source listing	***NO**, *YES	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional

Top

Display file (FILE)

Specifies the name of the DFU display file to be created. The name must be the same as that of the program DFU created when you saved the DDS source.

***CURLIB**
Type *CURLIB to use your current library to store the display file when it is created. If no current library exists in the library list QGPL is used.

library-name
Type the name of the library into which you want to create the DFU display file. The library must be the same one into which you created the original DFU program.

display-device-file-name
Type the name of the display device file that is to be created. The display device file name must match the name of the program with which the original device file was created.

Top

Source file (SRCFILE)

Specifies the name of the source file that contains the DDS source.

The possible values are:

source-file-name
Type the name of the source file that contains the DDS for this display device file.

***LIBL** DFU will use your library list to search for a specified source file.

***CURLIB**
Type *CURLIB to use your current library to locate the source file. If no current library entry exists in the library list, QGPL is used.

library-name
Type the qualified name of the library where the source file is located.

Top

Source member (SRCMBR)

Specifies the name of the member in the source file that contains the DDS for this DFU display file.

***FILE** DFU will use the name specified on the FILE parameter as the source file member name.

source-file-member-name
Type the name of the member that contains the DDS source.

Top

Authority (AUTH)

Specifies the authority that you are giving to the display file.

The possible values are:

***LIBCRTAUT**
Sets the public authority of the display file to the value specified in the CRTAUT parameter of the library when the display file is created. If the CRTAUT value for the library changes after the display file is created, the new value does not affect any existing objects in the library.

***CHANGE**
Allows others to perform all operations on the display file except those limited to the owner or controlled by object existence authority or object management authority. Others can perform basic functions on the file. Change authority provides object operational authority and all data authority.

***ALL** Allows others to perform all operations on the display file except those limited to the owner or controlled by authorization list management authority. Others can control the file's existence, specify security for it, change the file, and so on. Others cannot transfer ownership of the file.

***USE** Allows others to perform basic operations such as reading the file. Others may not change the file.

***EXCLUDE**
Prevents others from accessing the display file. Others have no authority.

authorization-list-name
You can specify the name of an authorization list whose authority is used for the display file.

Top

Replace (REPLACE)

Specifies that a new display file is created when a display file of the same name already exists in the same library.

***YES** Leave *YES to replace the existing file and re-create the new file.

You must delete the original (before tailoring) DFU display file before you request to create the new one, if REPLACE is specified as *NO.

***NO** Type *NO if you do not want to replace the existing file.

Top

Print source listing (PRINT)

Specifies to print the listing that is produced when the file is created.

***NO** Leave *NO if you do not want this list printed.

***YES** Type *YES to print a list of the source statements (DDS) used to create the display file. Any errors that occur will also be printed.

Top

Text 'description' (TEXT)

Specifies a character string that briefly describes the display file and its function.

***SRCMBRTXT**
DFU takes the text from the source file member that was used to create the display file.

***BLANK**
No text is specified.

'description'
You may enter up to 50 characters, enclosed in apostrophes, to briefly describe the display file.

Top

Examples

None

Top

Error messages

None

Top

Create RPG Module (CRTRPGMOD)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

Create RPG Module

The Create RPG Module (CRTRPGMOD) command compiles the RPG source code to create a module object (*MODULE). You can use this command in either batch or interactive mode.

Top

Parameters

Keyword	Description	Choices	Notes
MODULE	Module	*Qualified object name*	Optional, Positional 1
	Qualifier 1: Module	*Name*, ***CTLSPEC**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QRPGLESRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***MODULE**	Optional, Positional 3
SRCSTMF	Source stream file	*Path name*	Optional
GENLVL	Generation severity level	0-20, **10**	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OPTION	Compiler options	Values (up to 20 repetitions): *XREF, *NOXREF, *GEN, *NOGEN, *SECLVL, *NOSECLVL, *SHOWCPY, *NOSHOWCPY, *EXPDDS, *NOEXPDDS, *EXT, *NOEXT, *NOSHOWSKP, *SHOWSKP, *NOSRCSTMT, *SRCSTMT, *DEBUGIO, *NODEBUGIO, *NOEVENTF, *EVENTF	Optional
DBGVIEW	Debugging views	***STMT**, *SOURCE, *LIST, *COPY, *ALL, *NONE	Optional
OUTPUT	Output	***PRINT**, *NONE	Optional
OPTIMIZE	Optimization level	***NONE**, *BASIC, *FULL	Optional
INDENT	Source listing indentation	*Character value*, ***NONE**	Optional
CVTOPT	Type conversion options	Single values: ***NONE** Other values (up to 4 repetitions): *DATETIME, *GRAPHIC, *VARCHAR, *VARGRAPHIC	Optional
SRTSEQ	Sort sequence	Single values: ***HEX**, *JOB, *JOBRUN, *LANGIDUNQ, *LANGIDSHR Other values: *Qualified object name*	Optional
	Qualifier 1: Sort sequence	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
LANGID	Language identifier	*Name*, ***JOBRUN**, *JOB	Optional
REPLACE	Replace module	***YES**, *NO	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
TRUNCNBR	Truncate numeric	***YES**, *NO	Optional

Keyword	Description	Choices	Notes
FIXNBR	Fix numeric	Single values: ***NONE** Other values (up to 2 repetitions): *ZONED, *INPUTPACKED	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
ALWNULL	Allow null values	***NO**, *INPUTONLY, *USRCTL, *YES	Optional
DEFINE	Define condition names	Values (up to 32 repetitions): *Simple name*, ***NONE**	Optional
ENBPFRCOL	Enable performance collection	***PEP**, *ENTRYEXIT, *FULL	Optional
PRFDTA	Profiling data	***NOCOL**, *COL	Optional
BNDDIR	Binding directory	Single values: ***NONE** Other values (up to 50 repetitions): *Qualified object name*	Optional
	Qualifier 1: Binding directory	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
LICOPT	Licensed Internal Code options	*Character value*, **X''**	Optional
INCDIR	Include directory	Values (up to 32 repetitions): *Path name*, ***NONE**	Optional
PGMINFO	Generate program interface	***NO**, *PCML	Optional
INFOSTMF	Program interface stream file	*Path name*	Optional
PPGENOPT	Preprocessor options	Single values: ***NONE**, *DFT Other values (up to 3 repetitions): *RMVCOMMENT, *NORMVCOMMENT, *EXPINCLUDE, *NOEXPINCLUDE, *SEQSRC, *NOSEQSRC	Optional
PPSRCFILE	Output source file	*Qualified object name*	Optional
	Qualifier 1: Output source file	*Name*	
	Qualifier 2: Library	*Name*, ***CURLIB**	
PPSRCMBR	Output source member	*Name*, ***MODULE**	Optional
PPSRCSTMF	Output stream file	*Path name*, *SRCSTMF	Optional

Top

Module (MODULE)

Specifies the library name and module name for the module object you are creating. The module name and library name must conform to server naming conventions. If no library is specified, the created module is stored in the current library.

***CTLSPEC**
The name for the compiled module is taken from the name specified in the DFTNAME keyword of the control specification. If the module name is not specified on the control specification, and the source member is from a database file, the member name, specified by the SRCMBR parameter, is used as the module name. If the source is not from a database file then the module name defaults to RPGMOD.

module-name
Enter the name of the module object.

***CURLIB**
The compiled module object is stored in the current library. If you have not specified a current library, QGPL is used.

library-name
Enter the name of the library where the compiled module object is to be stored.

Top

Source file (SRCFILE)

Specifies the name of the source file that contains the ILE RPG source member to be compiled and the library where the source file is stored. The recommended source physical file length is 112 characters: 12 for the sequence number and date, 80 for the code and 20 for the comments. This is the maximum amount of source that is shown on the compiler listing.

QRPGLESRC
The default source file QRPGLESRC contains the ILE RPG source member to be compiled.

source-file-name
Enter the name of the source file that contains the ILE RPG source member to be compiled. This is the default.

***LIBL** The system searches the library list to find the library where the source file is stored. This is the default.

***CURLIB**
The current library is used to find the source file. If you have not specified a current library, QGPL is used.

library-name
Enter the name of the library where the source file is stored.

Top

Source member (SRCMBR)

Specifies the name of the member of the source file that contains the ILE RPG source specifications to be compiled.

***MODULE**
Use the name specified for the MODULE parameter as the source file member name. The compiled module object will have the same name as the source file member. If no module name is specified for the MODULE parameter, the command uses the first member created in or added to the source file as the source member name.

source-file-member-name
Enter the name of the member that contains the ILE RPG source specifications.

Top

Source stream file (SRCSTMF)

Specifies the path name of the stream file containing the ILE RPG source code to be compiled.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'.

If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by appending the job's current working directory to the path name.

The SRCMBR and SRCFILE parameters cannot be specified with the SRCSTMF parameter.

Top

Generation severity level (GENLVL)

Controls the creation of the module object. The module object is created if all errors encountered during compilation have a severity level less than or equal to the generation severity level specified.

The value must be between 0 and 20 inclusive. For errors greater than severity 20, the module object will not be generated.

10 A module object will be generated when the compile-time errors have a severity level less than or equal to 10. This is the default.

severity-level-value
Enter a number, 0 through 20 inclusive.

Top

Text 'description' (TEXT)

Allows you to enter text that briefly describes the module and its function. The text appears whenever module information is displayed.

***SRCMBRTXT**
The text of the source member is used. This is the default.

****BLANK***
No text appears.

'description'
Enter the text that briefly describes the function of the source specifications. The text can be a maximum of 50 characters and must be enclosed in apostrophes. The apostrophes are not part of the 50-character string. Apostrophes are not required if you are entering the text on the prompt screen.

Top

Compiler options (OPTION)

Specifies the options to use when the source member is compiled. You can specify any or all of the options in any order. Separate the options with one or more blank spaces. If an option is specified more than once, the last one is used.

***XREF** Produces a cross-reference listing (when appropriate) for the source member.

****NOXREF***
Does not produce a cross-reference listing.

***GEN** Creates a module object that can be bound using the CRTPGM command if the highest severity level returned by the compiler does not exceed the severity specified in the GENLVL option.

***NOGEN**
A module object will not be created.

***NOSECLVL**
Do not print second-level message text on the line following the first-level message text.

***SECLVL**
Print second-level message text on the line following the first-level message text in the Message Summary section.

***SHOWCPY**
Show source records of members included by the /COPY compiler directive.

***NOSHOWCPY**
Do not show source records of members included by the /COPY compiler directive.

***EXPDDS**
Show the expansion of externally described files in the listing and display key field information.

***NOEXPDDS**
Do not show the expansion of externally described files in the listing or display key field information.

***EXT** Show the list of external procedures and fields referenced during the compile on the listing.

***NOEXT**
Do not show the list of external procedures and fields referenced during compilation on the listing.

***NOSHOWSKP**
Do not show ignored statements in the source part of the listing. The compiler ignores statements as a result of /IF, /ELSEIF or /ELSE directives.

***SHOWSKP**
Show all statements in the source part of the listing, regardless of whether or not the compiler has skipped them.

***NOSRCSTMT**
Line Numbers in the listing are assigned sequentially; these numbers are used when debugging using statement numbers. Line Numbers are shown on the left-most column of the listing. The source IDs and SEU Sequence Numbers are shown on the two right-most columns of the listing.

***SRCSTMT**
Statement numbers for debugging are generated using SEU sequence numbers and source IDs as follows:

```
Statement_Number = source_ID * 1000000 +
                   source_SEU_sequence_number
```

SEU Sequence Numbers are shown on the left-most column of the listing. Statement Numbers are shown on the right-most column of the listing; these numbers are used when debugging using statement numbers.

Note: When OPTION(*SRCSTMT) is specified, all sequence numbers in the source files must contain valid numeric values. If there are duplicate sequence numbers in the same source file, the behavior of the debugger may be unpredictable and statement numbers for diagnostic messages or cross reference entries may not be meaningful.

***DEBUGIO**
Generate breakpoints for all input and output specifications.

***NODEBUGIO**
Do not generate breakpoints for input and output specifications.

***NOEVENTF**
Do not create an Event File for use by CoOperative Development Environment (CODE). CODE uses this file to provide error feedback integrated with the CODE editor. An Event File is normally created when you create a module or program from within CODE.

***EVENTF**
Create an Event File for use by CoOperative Development Environment (CODE). The Event File is created as a member in file EVFEVENT in the library where the created module or program object is to be stored. If the file EVFEVENT does not exist it is automatically created. The Event File member name is the same as the name of the object being created.

CODE uses this file to provide error feedback integrated with the CODE editor. An Event File is normally created when you create a module or program from within CODE.

Top

Debugging views (DBGVIEW)

Specifies which level of debugging is available for the compiled module object, and which source views are available for source-level debugging.

***STMT**
Allows the module object to be debugged using the Line Numbers or Statement Numbers of the compiler listing. Line Numbers are shown on the left-most column of the source section of the compiler listing when OPTION(*NOSRCSTMT) is specified. Statement Numbers are shown on the right-most column of the source section of the compiler listing when OPTION(*SRCSTMT) is specified.

***SOURCE**
Generates the source view for debugging the compiled module object. This view is not available if the root source member is a DDM file. Also, if changes are made to any source members after the compile and before attempting to debug the program, the views for those source members may not be usable.

***LIST** Generates the listing view for debugging the compiled module object. The information contained in the listing view is dependent on whether *SHOWCPY, *EXPDDS, and *SRCSTMT are specified for the OPTION parameter.

Note: The listing view will not show any indentation which you may have requested using the Indent option.

***COPY**
Generates the source and copy views for debugging the compiled module object. The source view for this option is the same source view generated for the *SOURCE option. The copy view is a debug view which has all the /COPY source members included. These views are not available if the root source member is a DDM file. Also, if changes are made to any source members after the compile and before attempting to debug the program, the views for those source members may not be usable.

***ALL** Generates the listing, source and copy views for debugging the compiled module object. The information contained in the listing view is dependent on whether *SHOWCPY, *EXPDDS, and *SRCSTMT are specified for the OPTION parameter.

***NONE**
Disables all of the debug options for debugging the compiled module object.

Top

Output (OUTPUT)

Specifies if a compiler listing is generated.

***PRINT**
Produces a compiler listing, consisting of the ILE RPG module source and all compile-time messages. The information contained in the listing is dependent on whether *XREF, *SECLVL, *SHOWCPY, *EXPDDS, *EXT, *SHOWSKP, and *SRCSTMT are specified for the OPTION parameter.

***NONE**
Do not generate the compiler listing.

Top

Optimization level (OPTIMIZE)

Specifies the level of optimization, if any, of the module.

***NONE**
Generated code is not optimized. This is the fastest in terms of translation time. It allows variables to be displayed and modified while in debug mode.

***BASIC**
Some optimization is performed on the generated code. This allows user variables to be displayed but not modified while in debug mode.

***FULL** Optimization which generates the most efficient code. Translation time is the longest. User variables may not be modified but may be displayed, although the presented values may not be the current values.

Top

Source listing indentation (INDENT)

Specifies whether structured operations should be indented in the source listing for enhanced readability. Also specifies the characters that are used to mark the structured operation clauses.

Note: Any indentation which you request here will not be reflected in the listing debug view which is created when you specify DBGVIEW(*LIST).

***NONE**
Structured operations will not be indented in the source listing. This is the default.

character-value
The source listing is indented for structured operation clauses. Alignment of statements and clauses are marked using the characters you choose. You can choose any character string up to 2 characters in length. If you want to use a blank in your character string, you must enclose the string in single quotation marks.

Note: The indentation may not appear as expected if there are errors in the module.

Top

Type conversion options (CVTOPT)

Specifies how the ILE RPG compiler handles date, time, timestamp, graphic data types, and variable-length data types which are retrieved from externally described database files.

***NONE**
Ignores variable-length database data types and uses the native RPG date, time, timestamp and graphic data types.

***DATETIME**
Specifies that date, time, and timestamp database data types are to be declared as fixed-length character fields.

***GRAPHIC**
Specifies that double-byte character set (DBCS) graphic data types are to be declared as fixed-length character fields.

***VARCHAR**
Specifies that variable-length character data types are to be declared as fixed-length character fields.

***VARGRAPHIC**
Specifies that variable-length double-byte character set (DBCS) graphic data types are to be declared as fixed-length character fields.

Top

Sort sequence (SRTSEQ)

Specifies the sort sequence table that is to be used in the ILE RPG source program.

***HEX** No sort sequence table is used.

***JOB** Use the SRTSEQ value from the job when the module is created.

***JOBRUN**
Use the SRTSEQ value from the job when the module is run (after being bound).

***LANGIDUNQ**
Use a unique weighted table. This special value is used in conjunction with the LANGID parameter to select the proper sort sequence table.

***LANGIDSHR**
Use a shared weighted table. This special value is used in conjunction with the LANGID parameter to select the proper sort sequence table.

sort-table-name
Enter the name of the sort sequence table.

***LIBL** The system searches the library list to find the library where the sort sequence table is stored.

***CURLIB**
The current library is used to find the sort sequence table. If you have not specified a current library, QGPL is used.

library-name
Enter the name of the library where the sort sequence table is stored.

Top

Language identifier (LANGID)

Specifies the language identifier to be used when the sort sequence is *LANGIDUNQ or *LANGIDSHR. The LANGID parameter is used in conjunction with the SRTSEQ parameter to select the sort sequence table.

***JOBRUN**
Use the LANGID value associated with the job when the RPG module is run (after being bound).

***JOB** Use the LANGID value associated with the job when the RPG module is created.

language-identifier
Use the language identifier specified. (For example, FRA for French and DEU for German).

Top

Replace module (REPLACE)

Specifies whether a new module object is created if a module of the same name already exists in the specified library.

***YES** A new module object is created in the specified library. The existing module object of the same name in the specified library is moved to library QRPLOBJ.

***NO** A new module object is not created if a module object of the same name already exists in the specified library.

Top

Authority (AUT)

Specifies the authority given to users who do not have specific authority to the object, who are not on the authorization list, and whose user group has no specific authority to the object. The authority can be altered for all or for specified users after the module is created with the CL commands Grant Object Authority (GRTOBJAUT) or Revoke Object Authority (RVKOBJAUT). For further information on these commands, see the CL concepts and reference topic in the iSeries Information Center at http://www.ibm.com/eserver/iseries/infocenter.

***LIBCRTAUT**
The public authority for the object is taken from the CRTAUT keyword of the target library (the library that contains the object). The value is determined when the object is created. If the CRTAUT value for the library changes after the create, the new value will not affect any existing objects.

***ALL** Authority for all operations on the module object except those limited to the owner or controlled by authorization list management authority. The user can control the module object's existence, specify this security for it, change it, and perform basic functions on it, but cannot transfer its ownership.

***CHANGE**
Provides all data authority and the authority to perform all operations on the module object except those limited to the owner or controlled by object authority and object management authority. The user can change the object and perform basic functions on it.

***USE** Provides object operational authority and read authority; authority for basic operations on the module object such as binding it into a program. The user is prevented from changing the object.

Note: A user must have *USE authority to a program to obtain a formatted dump of the variables of the program. To dump variables, the program must also have observable information.

If you do not want some users to be able to dump the variables, then give them only *OBJOPR plus *EXECUTE authority to the program or service program containing the module. This will allow them to call the program but not dump its variables. Use EDTOBJAUT, GRTOBJAUT or RVKOBJAUT to change the authority of the created program or service program.

If you do not want any users to be able to dump the variables, then remove the observable information using Change Module (CHGMOD) on the module, or using Change Program (CHGPGM) or Change Service Program (CHGSRVPGM) on the program containing the module.

***EXCLUDE**
The user is prevented from accessing the object.

authorization-list name
Enter the name of an authorization list of users and authorities to which the module is added. The module object will be secured by this authorization list, and the public authority for the module object will be set to *AUTL. The authorization list must exist on the system when the CRTRPGMOD command is issued.

Note: Use the AUT parameter to reflect the security requirements of your system. The security facilities available are described in detail in in detail in iSeries Security Reference, SC41-5302.

Top

Truncate numeric (TRUNCNBR)

Specifies if the truncated value is moved to the result field or if an error is generated when numeric overflow occurs while running the program.

Note: The TRUNCNBR option does not apply to calculations performed within expressions. (Expressions are found in the Extended-Factor 2 field.) If overflow occurs for these calculations, an error will always occur.

***YES** Ignore numeric overflow and move the truncated value to the result field.

***NO** When numeric overflow is detected, a run-time error is generated.

Top

Fix numeric (FIXNBR)

Specifies whether decimal data that is not valid is fixed by the compiler.

***NONE**
Indicates that decimal data that is not valid will result in decimal errors during run time if used.

***ZONED**
Zoned decimal data that is not valid will be fixed by the compiler on the conversion to packed data. Blanks in numeric fields will be treated as zeros. Each decimal digit will be checked for validity. If a decimal digit is not valid, it is replaced with zero. If a sign is not valid, the sign will be forced to a positive sign code of hex 'F'. If the sign is valid, it will be changed to either a positive sign hex 'F' or a negative sign hex 'D' as appropriate. If the resulting packed data is not valid, it will not be fixed.

***INPUTPACKED**
Indicates that if packed decimal data that is not valid is encountered while processing input specifications, the internal variable will be set to zero.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which you intend to use the object being created. In the examples given for the *CURRENT and *PRV values, and when specifying the target-release value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V2R3M0 is version 2, release 3, modification level 0.

Valid values for this parameter change every release. The possible values are:

***CURRENT**
The object is to be used on the release of the operating system currently running on your system. For example, if V2R3M5 is running on the system, *CURRENT means that you intend to use the object on a system with V2R3M5 installed. You can also use the object on a system with any subsequent release of the operating system installed.

Note: If V2R3M5 is running on the system, and the object is to be used on a system with V2R3M0 installed, specify TGTRLS(V2R3M0), not TGTRLS(*CURRENT).

***PRV** The object is to be used on the previous release with modification level 0 of the operating system. For example, if V2R3M5 is running on your system, *PRV means you intend to use the object on a system with V2R2M0 installed. You can also use the object on a system with any subsequent release of the operating system installed.

target-release
Specify the release in the format VxRxMx. You can use the object on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. If you specify a target-release that is earlier than the earliest release supported by this command, an error message is sent indicating the earliest supported release.

Note: The current version of the command may support options that are not available in previous releases of the command. If the command is used to create objects that are to be used on a previous release, it will be processed by the compiler appropriate to that release, and any unsupported options will not be recognized. The compiler will not necessarily issue any warnings regarding options that it is unable to process.

Top

Allow null values (ALWNULL)

Specifies how the ILE RPG module will be allowed to use records containing null-capable fields from externally described database files.

***NO** Specifies that the ILE RPG module will not process records with null-value fields from externally described files. If you attempt to retrieve a record containing null values, no data in the record is accessible to the ILE RPG module and a data-mapping error occurs.

***INPUTONLY**
Specifies that the ILE RPG module can successfully read records with null-capable fields containing null values from externally described input-only database files. When a record containing null values is retrieved, no data mapping errors occur and the database default values are placed into any fields which contain null values. The module cannot do any of the following:

- use null-capable key fields
- create or update records containing null-capable fields
- determine whether a null-capable field is actually null while the module is running
- set a null-capable field to be null.

***USRCTL**
Specifies that the ILE RPG module can read, write and update records with null values from externally described database files. Records with null keys can be retrieved using keyed operations. The module can determine whether a null-capable field is actually null, and it can set a null-capable field to be null for output or update. The programmer is responsible for ensuring that fields containing null values are used correctly within the module.

***YES** Same as *INPUTONLY.

Top

Define condition names (DEFINE)

Specifies condition names that are defined before the compilation begins. Using the parameter DEFINE(condition-name) is equivalent to coding the /DEFINE condition-name directive on the first line of the source file.

***NONE**
No condition names are defined. This is the default.

Name Up to 32 condition names can be specified. Each name can be up to 50 characters long. The condition names will be considered to be defined at the start of compilation.

Top

Enable performance collection (ENBPFRCOL)

Specifies whether performance collection is enabled.

***PEP** Performance statistics are gathered on the entry and exit of the Program Entry Procedure only. This applies to the actual Program Entry Procedure for a program, not the main procedure of the modules within the program. This is the default.

***ENTRYEXIT**
Performance statistics are gathered on the entry and exit of all procedures of the module.

***FULL** Performance statistics are gathered on entry and exit of all procedures. Also, statistics are gathered before and after each call to an external procedure.

Top

Profiling data (PRFDTA)

Specifies the program profiling data attribute for the module. Program profiling is an advanced optimization technique used to reorder procedures and code within the procedures based on statistical data (profiling data).

***NOCOL**
This module is not enabled to collect profiling data. This is the default.

***COL** This module is enabled to collect profiling data. *COL can be specified only when the optimization level of the module is *FULL, and when compiling with a target release of *CURRENT.

Top

Binding directory (BNDDIR)

Specifies the list of binding directories that are used in symbol resolution for a module. The search for a binding directory is done when the module is bound at CRTPGM or CRTSRVPGM time.

***NONE**
No binding directory is specified.

binding-directory-name
Specify the name of the binding directory used in symbol resolution. The directory name can be qualified with one of the following library values:

***LIBL** The system searches the library list to find the library where the binding directory is stored. This is the default.

***CURLIB**
The current library for the job is resolved at compile time. It is then used to search for the binding directory when the module is bound. If no library is specified as the current library for the job, library QGPL is used.

library-name
Specify the name of the library to be searched.

Top

Licensed Internal Code options (LICOPT)

Specifies one or more Licensed Internal Code compile-time options. This parameter allows individual compile-time options to be selected, and is intended for the advanced programmer who understands the potential benefits and drawbacks of each selected type of compiler option.

Top

Include directory (INCDIR)

Specifies one or more directories to add to the search path used by the compiler to find copy files. The compiler will search the directories specified here if the copy files in the source program can not be resolved.

***NONE**
No user directories are searched for copy files. By default, the source directory will still be searched.

'directory'
Specify up to 32 directories in which to search for copy files. In addition to the specified directories, the source directory is also searched for copy files.

Top

Generate program interface (PGMINFO)

Specifies whether program interface information should be generated into a stream file. The possible values are:

***NO** This option specifies the default which does not generate program interface information.

***PCML**
Specifies that PCML (Program Call Markup Language) should be generated into a stream file. The generated PCML makes it easier for Java methods to call the procedures in this RPG module, with less Java code. The name of a stream file that will contain the generated PCML must be specified on the INFOSTMF option.

Top

Program interface stream file (INFOSTMF)

Specifies the path name of the stream file to contain the generated program interface information specifed on the PGMINFO option.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'.

If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by appending the job's current working directory to the path name.

This parameter can only be specified when the PGMINFO parameter has a value other than *NO.

Top

Preprocessor options (PPGENOPT)

Specifies the preprocessor generation options to use when the source code is compiled.

The possible options are:

***NONE**
Run the entire compiler against the source file. Do not copy the preprocessor output to a file.

***DFT** Run the preprocessor against the input source. *RMVCOMMENT, *EXPINCLUDE and *NOSEQSRC will be used as the options for generating the preprocessor output. Use PPSRCFILE and PPSRCMBR to specify an output source file and member, or PPSRCSTMF to specify a stream file to contain the preprocessor output.

***RMVCOMMENT**
Remove comments, blank lines, and most directives during preprocessing. Retain only the RPG specifications and any directives necessary for the correct interpretation of the specifications.

***NORMVCOMMENT**
Preserve comments, blank lines and listing-control directives (for example /EJECT, /TITLE) during preprocessing. Transform source-control directives (for example /COPY, /IF) to comments during preprocessing.

***EXPINCLUDE**
Expand /INCLUDE directives in the generated output file.

***NOEXPINCLUDE**
/INCLUDE directives are placed unchanged in the generated output file.

Note: /COPY directives are always expanded.

***SEQSRC**
If PPSRCFILE is specified, the generated output member has sequential sequence numbers, starting at 000001 and incremented by 000001.

***NOSEQSRC**
If PPSRCFILE is specified, the generated output member has the same sequence numbers as the original source read by the preprocessor.

Top

Output source file (PPSRCFILE)

Specifies the source file name and library for the preprocessor output.

source-file-name
Specify the name of the source file for the preprocessor output.

The possible library values are:

***CURLIB**
The preprocessor output is created in the current library. If a job does not have a current library, the preprocessor output file is created in the QGPL library.

library-name
Specify the name of the library for the preprocessor output.

Top

Output source member (PPSRCMBR)

Specifies the name of the source file member for the preprocessor output.

***MODULE**
The name supplied on the MODULE parameter is used as the preprocessor output member name.

member-name
Specify the name of the member for the preprocessor output.

Top

Output stream file (PPSRCSTMF)

Specifies the path name of the stream file for the preprocessor output.

***SRCSTMF**
The path name supplied on the SRCSTMF parameter is used as the preprocessor output path name. The file will have the extension '.i'.

'path-name'
Specify the path name for the preprocessor output stream file.

The path name can be either absolutely or relatively qualified. An absolute path name starts with '/'; a relative path name starts with a character other than '/'.

If absolutely-qualified, the path name is complete. If relatively-qualified, the path name is completed by appending the job's current working directory to the path name.

Top

Examples

Example 1: Compiling a Source Module into a Module Object

```
CRTRPGMOD   MODULE(MYLIB/XMPLE1)
            SRCFILE(MYLIB/QRPGLESRC)  SRCMBR(XMPLE1)
            OUTPUT(*PRINT)  TEXT('My RPG IV module')
```

This command calls the compiler for ILE RPG to create a module named XMPLE1. The source module is in member XMPLE1 of source file QRPGLESRC in library MYLIB. A compiler listing is created.

Top

Error messages

***ESCAPE Messages**

RNS9309
Compilation failed. Module &1 not created in library &2.

Top

Create RPG/400 Program (CRTRPGPGM)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The command Create RPG Program (CRTRPGPGM) is used to start the RPG/400 compiler.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Optional, Positional 1
	Qualifier 1: Program	*Name*, ***CTLSPEC**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QRPGSRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 3
GENLVL	Generation severity level	0-99, **9**	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OPTION	Source listing options	Values (up to 14 repetitions): *SOURCE, *NOSOURCE, *SRC, *NOSRC, *XREF, *NOXREF, *GEN, *NOGEN, *DUMP, *NODUMP, *SECLVL, *NOSECLVL, *SRCDBG, *NOSRCDBG, *LSTDBG, *NOLSTDBG	Optional
GENOPT	Generation options	Values (up to 10 repetitions): *LIST, *NOLIST, *XREF, *NOXREF, *ATR, *NOATR, *DUMP, *NODUMP, *PATCH, *NOPATCH, *OPTIMIZE, *NOOPTIMIZE	Optional
INDENT	Source listing indentation	*Character value*, ***NONE**	Optional
CVTOPT	Type conversion options	Single values: ***NONE** Other values (up to 3 repetitions): *DATETIME, *VARCHAR, *GRAPHIC	Optional
SRTSEQ	Sort sequence	Single values: ***HEX**, *JOB, *JOBRUN, *LANGIDUNQ, *LANGIDSHR Other values: *Qualified object name*	Optional
	Qualifier 1: Sort sequence	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
LANGID	Language identifier	*Name*, ***JOBRUN**, *JOB	Optional
SAAFLAG	SAA flagging	***NOFLAG**, *FLAG	Optional
PRTFILE	Print file	*Qualified object name*	Optional
	Qualifier 1: Print file	*Name*, **QSYSPRT**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
REPLACE	Replace program	***YES**, *NO	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
USRPRF	User profile	***USER**, *OWNER	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
PHSTRC	Phase trace	***NO**, *YES	Optional

Keyword	Description	Choices	Notes
ITDUMP	Intermediate text dump	*Character value*, ***NONE**	Optional
SNPDUMP	Snap dump	*Character value*, ***NONE**	Optional
CODELIST	Codelist	*Character value*, ***NONE**, *ALL	Optional
IGNDECERR	Ignore decimal data error	***NO**, *YES	Optional
ALWNULL	Allow null values	***NO**, *YES	Optional

Top

Program (PGM)

Specifies the program name and library for the compiled RPG program.

***CTLSPEC**
The program name indicated in positions 75 - 80 of the Control specification.

program-name
Enter the name by which the program will be known.

***CURLIB**
The current library will be used. If you have not specified a current library, QGPL will be used.

library-name
Enter the name of the library where the compiled program is to be stored.

Top

Source file (SRCFILE)

Specifies the name of the source file that contains the source program.

QRPGSRC
The default source file, QRPGSRC, contains the RPG program to be compiled.

source-file-name
Enter the source file name that contains the RPG source program to be compiled.

***LIBL** The system searches the library list to find the library where the source file is located.

***CURLIB**
The current library will be used. If you have not specified a current library, QGPL will be used.

library-name
Enter the name of the library where the source file is located.

Top

Source member (SRCMBR)

Specifies the name of the member of the source file.

***PGM** The name specified by the PGM parameter as the source file member name.

source-file-member-name
Enter the name of the member that contains the source program.

Top

Generation severity level (GENLVL)

Specifies the diagnostic-message severity level at which creation of a program object stops.

9 The default severity level is 9.

severity-level-value
Enter a two-digit number 01 through 50.

Top

Text 'description' (TEXT)

Specifies what text is used to describe the program function.

***SRCMBRTXT**
Use the text description from the source file member.

***BLANK**
No text appears.

Top

Source listing options (OPTION)

Specifies options to use when the source program is compiled.

***SOURCE or *SRC**
The compiler supplies a source listing.

***NOSOURCE or *NOSRC**
The compiler does not supply a source listing.

***XREF** The compiler supplies a cross-reference listing.

***NOXREF**
The compiler does not supply a cross-reference listing.

***GEN** A program object is created that can be run after the program compiles.

***NOGEN**
No program object is created.

***NODUMP**
The program template does not print when an error occurs.

***DUMP**
The program template prints when an error occurs.

***NOSECLVL**
Message text is not printed.

***SECLVL**
Message text is printed.

***NOSRCDBG**
Do not produce source-level debugging information. Source-level error information will not be created unless *LSTDBG is specified.

***SRCDBG**
The compiler produces source-level error information and source-level debugging information for use with CoOperative Development Environment/400 (CODE/400). Source-level or listing-level

debugging information is also necessary if you want to use the system debugger (STRDBG OPMSRC(*YES)) to debug OPM and ILE programs at the same time.

You cannot specify *SRCDBG and *LSTDBG together. Specify one or the other.

***NOLSTDBG**
Do not produce a listing view or listing-level debugging information. Source-level error information will not be created unless *SRCDBG is specified.

***LSTDBG**
The compiler produces a listing view, source-level error information, and listing-level debugging information for use with CoOperative Development Environment/400 (CODE/400). Source-level or listing-level debugging information is also necessary if you want to use the system debugger (STRDBG OPMSRC(*YES)) to debug OPM and ILE programs at the same time.

You cannot specify *SRCDBG and *LSTDBG together. Specify one or the other.

Top

Generation options (GENOPT)

Specifies options to use to create object code.

***NOLIST**
Does not supply the intermediate representation of the program (IRP) listing.

***LIST** Lists the intermediate representation of the program (IRP).

***NOXREF**
No cross-reference listing is supplied for the intermediate representation of the program (IRP).

***XREF** Supplies a cross-reference listing of all objects defined in the intermediate representation of the program (IRP).

***NOATR**
Does not supply an attribute listing.

***ATR** Lists the attributes for the intermediate representation of a program (IRP) source program.

***NODUMP**
Does not print the program template when an error occurs.

***DUMP**
Prints a program template.

***NOPATCH**
Does not reserve a program patch area in the compiled program.

***PATCH**
Reserves space in the compiled program for a program patch area.

***NOOPTIMIZE**
The compiler will not perform program optimization.

***OPTIMIZE**
The compiler creates a program for more efficient processing.

Top

Source listing indentation (INDENT)

Specifies that DO statements and IF-ELSE clauses will be indented for readability. Also specifies what character is used to connect corresponding DO-ENDDO pairs and IF-ELSE pairs.

***NONE**
Listings are not indented within DO statements or IF-ELSE clauses.

character-string
Use the given character string to connect corresponding nesting level.

Top

Type conversion options (CVTOPT)

Specifies how the RPG/400 compiler handles date, time, and timestamp database data types, and variable-length data types which are retrieved from externally-described files.

***NONE**
Date, time, timestamp and variable-length database data types are ignored and not accessible in the RPG/400 program.

***DATETIME**
Date, time, and timestamp database data types are to be declared as fixed-length character fields and are accessible in the RPG/400 program.

***VARCHAR**
Variable-length database data types are to be declared as fixed-length character fields and are accessible in the RPG/400 program.

***GRAPHIC**
DBCS-graphic data types are to be declared as fixed length character fields and are accessible in the RPG/400 program.

Note: Choose both of the parameters *VARCHAR and *GRAPHIC if you want variable-length DBCS graphic data types to be declared in your program.

Top

Sort sequence (SRTSEQ)

Specifies the sort sequence table to be used.

Note: To use the values coded in the SRTSEQ and LANGID parameters of the CRTRPGPGM or CRTRPTPGM command, you must specify D in the Alternate-Collating-Sequence field on the Control specification. The alternate collating sequence is retrieved from the system either at compile time or run time. If you use the D option, the alternate collating sequence affects: all character comparison operations; LOKUP and SORTA for character table and arrays; and sequence checking for character compile-time data, and pre-run-time arrays and tables. When the alternate sequence is retrieved at run time, sequence-checking for compile-time data is delayed until run time.

***HEX** The hexadecimal values of the characters are used to determine the sort sequence. This is the default.

***JOB** Uses the SRTSEQ value associated with the job when the RPG program is created.

***JOBRUN**
Uses the SRTSEQ value associated with the job when the RPG program is run.

***LANGIDUNQ**
Uses a unique weighted table. This special value is used in conjunction with the LANGID parameter to select the sort sequence table.

***LANGIDSHR**
Uses a shared weighted table. This special value is used in conjunction with the LANGID parameter to select the sort sequence table.

sort-sequence-table-name
Enter the name of the sort sequence table.

***LIBL** The compiler searches the library list to find the library where the sort sequence table is located. This is the default.

***CURLIB**
The current library is searched to find the sort sequence table. If you have not specified a current library, QGPL is used.

library-name
Enter the name of the library where the sort sequence table is located.

Top

Language identifier (LANGID)

Specifies the language identifier to be used when the sort sequence is *LANGIDUNQ or *LANGIDSHR. The LANGID parameter is used in conjunction with the SRTSEQ parameter to select the sort sequence table.

***JOBRUN**
Uses the LANGID value associated with the job when the RPG program is run. This is the default.

***JOB** Uses the LANGID value associated with the job when the RPG program is created.

language-identifier
Enter the language identifier to be used (for example, FRA for French and DEU for German).

Top

SAA flagging (SAAFLAG)

Specifies if there will be flagging of specifications not supported by SAA RPG.

***NOFLAG**
No flagging will be performed.

***FLAG**
Flagging will be performed.

Top

Print file (PRTFILE)

Specifies the name of the file where the compiler listing is to be placed and the library where the file is located.

QSYSPRT
The compiler listing is placed in the QSYSPRT file.

file-name
Enter the name of the file where the compiler listing is to be placed.

***LIBL** The system searches the library list to find the library.

***CURLIB**
The current library will be used. If you have not specified a current library, QGPL will be used.

library-name
Enter the name of the library where the file is located.

Top

Replace program (REPLACE)

Specifies if a new program object will be created when there is an existing program object of the same name in the same library.

***YES** A new program object will be created and any existing program object of the same name in the specified library will be moved to library QRPLOBJ.

***NO** A new program object will not be created if a program object of the same name already exists in the specified library.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which you intend to use the object being created. In the examples given for the *CURRENT and *PRV values, and when specifying the target-release value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V2R3M0 is version 2, release 3, modification level 0.

Valid values for this parameter change every release.

The possible values are:

***CURRENT**
The object is to be used on the release of the operating system currently running on your system. For example, if V2R3M5 is running on the system, *CURRENT means that you intend to use the object on a system with V2R3M5 installed. You can can also use the object on a system with any subsequent release of the operating system installed.

***PRV** The object is to be used on the previous release with modification level 0 of the operating system. For example, if V2R3M5 is running on your system, *PRV means you intend to use the object on a system with V2R2M0 installed. You can also use the object on a system with any subsequent release of the operating system installed.

target-release
Specify the release in the format VxRxMx. You can use the object on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. If you specify a release-level that is earlier than the earliest release level supported by this command, an error message is sent indicating the earliest supported release.

Note: The program may be able to be restored on a release level earlier than the one you specified on the create command. Use DSPPGM to determine the earliest release the program can run.

Top

User profile (USRPRF)

Specifies the user profile under which the compiled RPG program runs.

***USER**
The program runs under the user profile of the program's user.

***OWNER**
The program runs under the user profiles of both the program's owner and user. The USRPRF parameter will not be updated if the program already exists.

Top

Authority (AUT)

Specifies what authority is granted for the program.

***LIBCRTAUT**
The default public authority for created objects is taken from the CRTAUT keyword which is associated with the target library. The value is determined at create time. If the value of the CRTAUT keyword of the library changes after the create, the new value will not affect any existing objects.

***ALL** Complete authority for the program except for transfer of object ownership.

***CHANGE**
All operations are permitted except those dealing with the object's existence and its management.

***USE** Authority to read or run the compiled program, but not debug or change it.

Note: A user must have *USE authority to a program to obtain a formatted dump of the variables of the program. To dump variables, the program must also have observable information.

If you do not want some users to be able to dump the variables, then give them only *OBJOPR plus *EXECUTE authority to the program. This will allow them to call the program but not dump its variables.

If you do not want any users to be able to dump the variables, then use Change Program (CHGPGM) to remove the program's observable information.

***EXCLUDE**
No authority.

authorization-list name
The name of the authorization list which secures the object. The public authority will be *AUTL.

Top

Phase trace (PHSTRC)

Specifies if phase trace information about the compiler is to be included in the listing.

***NO** Does not provide compiler phase information.

***YES** Provides compiler phase information.

Top

Intermediate text dump (ITDUMP)

Specifies creation of the dynamic listing of intermediate text.

***NONE**
Does not supply intermediate text dump.

phase-name
Enter the last two characters of each phase name.

Top

Snap dump (SNPDUMP)

Specifies if a listing of major data area and intermediate text is supplied.

***NONE**
Does not supply a snap dump.

phase-name
Enter the last two characters of each phase name.

Top

Codelist (CODELIST)

Specifies if a dynamic listing of IRP is supplied for a specific phase.

***NONE**
Does not supply an intermediate IRP dump.

***ALL** Supplies an intermediate IRP dump.

phase-name
Enter the last two characters of each phase name.

Top

Ignore decimal data error (IGNDECERR)

Specifies if decimal data errors are ignored.

***NO** Decimal data errors are not ignored.

***YES** Decimal data errors are ignored.

Top

Allow null values (ALWNULL)

Specifies whether an RPG/400 program will accept null values from null-capable fields in an externally-described input file.

***NO** Specifies that the RPG/400 program will not accept null value fields.

***YES** Specifies that an RPG/400 program will accept null value fields for an externally-described input file.

Top

Examples

Example 1: Compiling a Source Program into a Program Object

```
CRTRPGPGM   PGM(MYLIB/XMPLE1)
            SRCFILE(MYLIB/QRPGSRC)  SRCMBR(XMPLE1)
            OPTION(*SOURCE)  TEXT('My RPG III program')
```

This command calls the compiler for RPG/400 to create a program named XMPLE1. The source program is in member XMPLE1 of source file QRPGSRC in library MYLIB. A compiler listing is created.

Top

Error messages

***ESCAPE Messages**

QRG9001
Compile failed. Program not created.

QRG9004
The release &1 specified on the TGTRLS option is not supported.

QRG9005
The *SRCDBG or *LSTDBG compiler option cannot be processed.

Top

Create Auto Report RPG Program (CRTRPTPGM)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The command Create Auto Report Program (CRTRPTPGM) is used to start the RPG/400 compiler for automatic reports.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Optional, Positional 1
	Qualifier 1: Program	*Name*, ***CTLSPEC**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QRPGSRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 3
GENLVL	Generation severity level	0-99, **9**	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OPTION	Source listing options	Values (up to 14 repetitions): *SOURCE, *NOSOURCE, *SRC, *NOSRC, *XREF, *NOXREF, *GEN, *NOGEN, *DUMP, *NODUMP, *SECLVL, *NOSECLVL, *LSTDBG, *NOLSTDBG	Optional
GENOPT	Generation options	Values (up to 10 repetitions): *LIST, *NOLIST, *XREF, *NOXREF, *ATR, *NOATR, *DUMP, *NODUMP, *PATCH, *NOPATCH, *OPTIMIZE, *NOOPTIMIZE	Optional
INDENT	Source listing indentation	*Character value*, ***NONE**	Optional
CVTOPT	Type conversion options	Single values: ***NONE** Other values (up to 3 repetitions): *DATETIME, *VARCHAR, *GRAPHIC	Optional
SRTSEQ	Sort sequence	Single values: ***HEX**, *JOB, *JOBRUN, *LANGIDUNQ, *LANGIDSHR Other values: *Qualified object name*	Optional
	Qualifier 1: Sort sequence	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
LANGID	Language identifier	*Name*, ***JOBRUN**, *JOB	Optional
PRTFILE	Print file	*Qualified object name*	Optional
	Qualifier 1: Print file	*Name*, **QSYSPRT**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
RPTOPT	Auto report options	Values (up to 10 repetitions): *SOURCE, *NOSOURCE, *SRC, *NOSRC, *FLOW, *NOFLOW, *AST, *NOAST, *DATE, *NODATE, *COMPILE, *NOCOMPILE, *SECLVL, *NOSECLVL	Optional

Keyword	Description	Choices	Notes
OUTFILE	Auto report output file	*Qualified object name*	Optional
	Qualifier 1: Auto report output file	*Name*, ***NONE**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
OUTMBR	Auto report output member	*Name*, ***NONE**	Optional
REPLACE	Replace program	***YES**, *NO	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
USRPRF	User profile	***USER**, *OWNER	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
PHSTRC	Phase trace	***NO**, *YES	Optional
ITDUMP	Intermediate text dump	*Character value*, ***NONE**	Optional
SNPDUMP	Snap dump	*Character value*, ***NONE**	Optional
CODELIST	Codelist	*Character value*, ***NONE**, *ALL	Optional
IGNDECERR	Ignore decimal data error	***NO**, *YES	Optional
ALWNULL	Allow null values	***NO**, *YES	Optional

Top

Program (PGM)

Specifies program name and library for the compiled RPG program.

***CTLSPEC**
The program name indicated in positions 75 - 80 of the Control specification.

program-name
Enter the name by which the program will be known.

***CURLIB**
The current library will be used. If you have not specified a current library, QGPL will be used.

library-name
Enter the name of the library where the compiled program is to be stored.

Top

Source file (SRCFILE)

Specifies the name of the source file that contains the source program.

QRPGSRC
The default source file, QRPGSRC, contains the RPG program to be compiled.

source-file-name
Enter the source file name that contains the RPG source program to be compiled.

***LIBL** The system searches the library list to find the library where the source file is located.

***CURLIB**
The current library will be used. If you have not specified a current library, QGPL will be used.

library-name
Enter the name of the library where the source file is located.

Top

Source member (SRCMBR)

Specifies the name of the member of the source file.

***PGM** The name specified by the PGM parameter as the source file member name.

source-file-member-name
Enter the name of the member that contains the source program.

Top

Generation severity level (GENLVL)

Specifies the diagnostic-message severity level at which creation of a program object stops.

9 The default severity level is 9.

severity-level-value
Enter a two-digit number 01 through 50.

Top

Text 'description' (TEXT)

Specifies what text is used to describe the program function.

***SRCMBRTXT**
Use the text description from the source file member.

***BLANK**
No text appears.

Top

Source listing options (OPTION)

Specifies options to use when the source program is compiled.

***SOURCE or *SRC**
The compiler supplies a source listing.

***NOSOURCE or *NOSRC**
The compiler does not supply a source listing.

***XREF** The compiler supplies a cross-reference listing.

***NOXREF**
The compiler does not supply a cross-reference listing.

***GEN** A program object is created that can be run after the program compiles.

***NOGEN**
No program object is created.

***NODUMP**
The program template does not print when an error occurs.

***DUMP**
The program template prints when an error occurs.

***NOSECLVL**
Message text is not printed.

***SECLVL**
Message text is printed.

***NOLSTDBG**
Do not produce a listing view or listing-level debugging information or source-level error information.

***LSTDBG**
The compiler produces a listing view, source-level error information, and listing-level debugging information for use with CoOperative Development Environment/400 (CODE/400). Listing-level debugging information is also necessary if you want to use the system debugger (STRDBG OPMSRC(*YES)) to debug OPM and ILE programs at the same time.

Top

Generation options (GENOPT)

Specifies options to use to create object code.

***NOLIST**
Does not supply the intermediate representation of the program (IRP) listing.

***LIST** Lists the intermediate representation of the program (IRP).

***NOXREF**
No cross-reference listing is supplied for the intermediate representation of the program (IRP).

***XREF** Supplies a cross-reference listing of all objects defined in the intermediate representation of the program (IRP).

***NOATR**
Does not supply an attribute listing.

***ATR** Lists the attributes for the intermediate representation of a program (IRP) source program.

***NODUMP**
Does not print the program template when an error occurs.

***DUMP**
Prints a program template.

***NOPATCH**
Does not reserve a program patch area in the compiled program.

***PATCH**
Reserves space in the compiled program for a program patch area.

***NOOPTIMIZE**
The compiler will not perform program optimization.

***OPTIMIZE**
The compiler creates a program for more efficient processing.

Top

Source listing indentation (INDENT)

Specifies that DO statements and IF-ELSE clauses will be indented for readability. Also specifies what character is used to connect corresponding DO-ENDDO pairs and IF-ELSE pairs.

***NONE**
Listings are not indented within DO statements or IF-ELSE clauses.

character-string
Use the given character string to connect corresponding nesting level.

Top

Type conversion options (CVTOPT)

Specifies how the RPG/400 compiler handles date, time, and timestamp database data types, and variable-length data types which are retrieved from externally-described files.

***NONE**
Date, time, timestamp and variable-length database data types are ignored and not accessible in the RPG/400 program.

***DATETIME**
Date, time, and timestamp database data types are to be declared as fixed-length character fields and are accessible in the RPG/400 program.

***VARCHAR**
Variable-length database data types are to be declared as fixed-length character fields and are accessible in the RPG/400 program.

***GRAPHIC**
DBCS-graphic data types are to be declared as fixed length character fields and are accessible in the RPG/400 program.

Note: Choose both of the parameters *VARCHAR and *GRAPHIC if you want variable-length DBCS graphic data types to be declared in your program.

Top

Sort sequence (SRTSEQ)

Specifies the sort sequence table to be used.

Note: To use the values coded in the SRTSEQ and LANGID parameters of the CRTRPGPGM or CRTRPTPGM command, you must specify D in the Alternate-Collating-Sequence field on the Control specification. The alternate collating sequence is retrieved from the system either at compile time or run time. If you use the D option, the alternate collating sequence affects: all character comparison operations; LOKUP and SORTA for character table and arrays; and sequence checking for character compile-time data, and pre-run-time arrays and tables. When the alternate sequence is retrieved at run time, sequence-checking for compile-time data is delayed until run time.

***HEX** The hexadecimal values of the characters are used to determine the sort sequence. This is the default.

***JOB** Uses the SRTSEQ value associated with the job when the RPG program is created.

***JOBRUN**
Uses the SRTSEQ value associated with the job when the RPG program is run.

***LANGIDUNQ**
Uses a unique weighted table. This special value is used in conjunction with the LANGID parameter to select the sort sequence table.

***LANGIDSHR**
Uses a shared weighted table. This special value is used in conjunction with the LANGID parameter to select the sort sequence table.

sort-sequence-table-name
Enter the name of the sort sequence table.

***LIBL** The compiler searches the library list to find the library where the sort sequence table is located. This is the default.

***CURLIB**
The current library is searched to find the sort sequence table. If you have not specified a current library, QGPL is used.

library-name
Enter the name of the library where the sort sequence table is located.

Top

Language identifier (LANGID)

Specifies the language identifier to be used when the sort sequence is *LANGIDUNQ or *LANGIDSHR. The LANGID parameter is used in conjunction with the SRTSEQ parameter to select the sort sequence table.

***JOBRUN**
Uses the LANGID value associated with the job when the RPG program is run. This is the default.

***JOB** Uses the LANGID value associated with the job when the RPG program is created.

language-identifier
Enter the language identifier to be used (for example, FRA for French and DEU for German).

Top

Print file (PRTFILE)

Specifies the name of the file where the compiler listing is to be placed and the library where the file is located.

QSYSPRT
The compiler listing is placed in the QSYSPRT file.

file-name
Enter the name of the file where the compiler listing is to be placed.

***LIBL** The system searches the library list to find the library.

***CURLIB**
The current library will be used. If you have not specified a current library, QGPL will be used.

library-name
Enter the name of the library where the file is located.

Top

Auto report options (RPTOPT)

Specifies options to use when the automatic report source program is compiled.

***NOSOURCE or *NOSRC**
The compiler does not supply a source listing.

***SOURCE or *SRC**
The compiler supplies a source listing.

***NOFLOW**
A flow of the major routines that were run is not written.

***FLOW**
A flow of the major routines that were run is written.

***NOAST**
No asterisks are generated for total written lines.

***AST** Asterisks are generated for total written lines.

***DATE**
The date and page number are printed on the first *AUTO page heading line.

***NODATE**
The date and page number are not printed on the first *AUTO page heading line.

***COMPILE**
The RPG/400 compiler is called after the automatic report source compiles.

***NOCOMPILE**
The RPG/400 compiler is not called.

***NOSECLVL**
Message text is not printed.

***SECLVL**
Message text is printed.

Top

Auto report output file (OUTFILE)

Specifies the name of the output file for the RPG source program created by the automatic report compiler.

***NONE**
Auto report creates a temporary file to pass the RPG source program to the compiler.

file-name
Enter the name of the file that contains the generated RPG source program.

***LIBL** The system searches the library list to find the library where the source file is located.

***CURLIB**
The current library will be used. If you have not specified a current library, QGPL will be used.

library-name
Enter the name of the library in which the source file is stored.

Top

Auto report output member (OUTMBR)

Specifies the name of the member of the output file.

***NONE**
Uses the source member name as the member name.

file-member-name
Enter the name of the member that is to receive the output from the automatic report program.

Top

Replace program (REPLACE)

Specifies if a new program object will be created when there is an existing program object of the same name in the same library.

***YES** A new program object will be created and any existing program object of the same name in the specified library will be moved to library QRPLOBJ.

***NO** A new program object will not be created if a program object of the same name already exists in the specified library.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which you intend to use the object being created. In the examples given for the *CURRENT and *PRV values, and when specifying the target-release value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V2R3M0 is version 2, release 3, modification level 0.

Valid values for this parameter change every release.

The possible values are:

***CURRENT**
The object is to be used on the release of the operating system currently running on your system. For example, if V2R3M5 is running on the system, *CURRENT means that you intend to use the object on a system with V2R3M5 installed. You can can also use the object on a system with any subsequent release of the operating system installed.

***PRV** The object is to be used on the previous release with modification level 0 of the operating system. For example, if V2R3M5 is running on your system, *PRV means you intend to use the object on a system with V2R2M0 installed. You can also use the object on a system with any subsequent release of the operating system installed.

target-release
Specify the release in the format VxRxMx. You can use the object on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. If you specify a release-level that is earlier than the earliest release level supported by this command, an error message is sent indicating the earliest supported release.

Note: The program may be able to be restored on a release level earlier than the one you specified on the create command. Use DSPPGM to determine the earliest release the program can run.

Top

User profile (USRPRF)

Specifies the user profile under which the compiled RPG program runs.

***USER**
The program runs under the user profile of the program's user.

***OWNER**
The program runs under the user profiles of both the program's owner and user. The USRPRF parameter will not be updated if the program already exists.

Top

Authority (AUT)

Specifies what authority is granted for the program.

***LIBCRTAUT**
The default public authority for created objects is taken from the CRTAUT keyword which is associated with the target library. The value is determined at create time. If the value of the CRTAUT keyword of the library changes after the create, the new value will not affect any existing objects.

***ALL** Complete authority for the program except for transfer of object ownership.

***CHANGE**
All operations are permitted except those dealing with the object's existence and its management.

***USE** Authority to read or run the compiled program, but not debug or change it.

Note: A user must have *USE authority to a program to obtain a formatted dump of the variables of the program. To dump variables, the program must also have observable information.

If you do not want some users to be able to dump the variables, then give them only *OBJOPR plus *EXECUTE authority to the program. This will allow them to call the program but not dump its variables.

If you do not want any users to be able to dump the variables, then use Change Program (CHGPGM) to remove the program's observable information.

***EXCLUDE**
No authority.

authorization-list name
The name of the authorization list which secures the object. The public authority will be *AUTL.

Top

Phase trace (PHSTRC)

Specifies if phase trace information about the compiler is to be included in the listing.

***NO** Does not provide compiler phase information.

***YES** Provides compiler phase information.

Top

Intermediate text dump (ITDUMP)

Specifies creation of the dynamic listing of intermediate text.

***NONE**
Does not supply intermediate text dump.

phase-name
Enter the last two characters of each phase name.

Top

Snap dump (SNPDUMP)

Specifies if a listing of major data area and intermediate text is supplied.

***NONE**
Does not supply a snap dump.

phase-name
Enter the last two characters of each phase name.

Top

Codelist (CODELIST)

Specifies if a dynamic listing of IRP is supplied for a specific phase.

***NONE**
Does not supply an intermediate IRP dump.

***ALL** Supplies an intermediate IRP dump.

phase-name
Enter the last two characters of each phase name.

Top

Ignore decimal data error (IGNDECERR)

Specifies if decimal data errors are ignored.

***NO** Decimal data errors are not ignored.

***YES** Decimal data errors are ignored.

Top

Allow null values (ALWNULL)

Specifies whether an RPG/400 program will accept null values from null-capable fields in an externally-described input file.

***NO** Specifies that the RPG/400 program will not accept null value fields.

***YES** Specifies that an RPG/400 program will accept null value fields for an externally-described input file.

Top

Examples

Example 1: Compiling a Source Program into a Program Object

```
CRTRPTPGM   PGM(MYLIB/XMPLE1)
            SRCFILE(MYLIB/QRPGSRC)  SRCMBR(XMPLE1)
            OPTION(*SOURCE)
            TEXT('My RPG III Auto Report Program')
```

This command calls the RPG/400 Auto Report compiler to create a Auto Report program named XMPLE1. The source program is in member XMPLE1 of source file QRPGSRC in library MYLIB. A compiler listing is created.

Top

Error messages

***ESCAPE Messages**

RPT9001
Auto Report failed.

RPT0082
The Auto Report generation terminated because severe errors occurred.

QRG9004
The release &1 specified on the TGTRLS option is not supported.

Top

Create S/36 COBOL Program (CRTS36CBL)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The CRTS36CBL command compiles a COBOL source program into a program object for use on the System/36 environment. You can use this command interactively, in batch mode, or from a CL program.

All object names you specify in the CRTS36CBL command must be composed of alphanumeric characters, the first of which must be alphabetic. The names cannot exceed 8 characters in length.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Optional, Positional 1
	Qualifier 1: Program	*Name*, ***PGMID**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QS36SRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 3
GENLVL	Generation severity level	0-29, **29**	Optional, Positional 4
NEP	Never-ending program	***NO**, *YES	Optional, Positional 5
MRTMAX	Maximum MRT devices	0-99, **0**	Optional, Positional 6
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OPTION	Source listing options	Values (up to 50 repetitions): *SOURCE, *NOSOURCE, *SRC, *NOSRC, *NOXREF, *XREF, *GEN, *NOGEN, *NOSEQUENCE, *SEQUENCE, *NOVBSUM, *VBSUM, *NONUMBER, *NUMBER, *LINENUMBER, *NOMAP, *MAP, *NOOPTIONS, *OPTIONS, *QUOTE, *APOST, *DEBUG, *NODEBUG, *SECLVL, *NOSECLVL, *PRINT, *NOPRINT	Optional
GENOPT	Generation options	Values (up to 50 repetitions): *NOLIST, *LIST, *NOXREF, *XREF, *NOPATCH, *PATCH, *NODUMP, *DUMP, *NOATR, *ATR, *RANGE, *NORANGE, *UNREF, *NOUNREF, *NOOPTIMIZE, *OPTIMIZE	Optional
PRTFILE	Print file	*Qualified object name*	Optional
	Qualifier 1: Print file	*Name*, **QSYSPRT**	
	Qualifier 2: Print file library	*Name*, ***LIBL**, *CURLIB	
ICFLIB	Library for ICF files	*Name*	Optional
REPLACE	Replace program	*NO, ***YES**	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
USRPRF	User profile	***USER**, *OWNER	Optional

Keyword	Description	Choices	Notes
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
DUMP	Compiler debugging dump	*Element list*	Optional
	Element 1:	1-32767, **1**, *	
	Element 2:	1-32767, **32767**	
ITDUMP	Intermediate text dump	0-31, **0**	Optional
FIXDECDTA	Fix decimal data	*NO, ***YES**	Optional
CPYLIB	Copy file library	*Name*, *LIBL, *CURLIB	Optional

Top

Program (PGM)

Specifies the program name and library name for the COBOL program object you are creating. The possible values are:

***PGMID**
The name for the program object is taken from the PROGRAM-ID paragraph in the COBOL source program.

program-name
Enter a name to identify the compiled COBOL program. If you specify a program name for this parameter, and run the compilation in batch mode, the first program in the batch job uses this name; any other programs use the name specified in the PROGRAM-ID paragraph in the source program.

The possible library values are:

***CURLIB**
The current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library to contain the created program object.

Top

Source file (SRCFILE)

Specifies the name of the source file that contains the COBOL source to be compiled. The possible values are:

QS36SRC
The IBM-supplied source file, QS36SRC, contains the COBOL source to be compiled.

source-file-name
Enter the name of the source file that contains the COBOL source to be compiled. This source file should have a record length of 92.

The possible library values are:

***LIBL** The system searches the library list to find the library where the file is located.

***CURLIB**
The current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the source file is stored.

Top

Source member (SRCMBR)

Specifies the name of the member of the source file that contains the COBOL source to be compiled. You can specify this parameter only if the source file referenced in the SRCFILE parameter is a database file. The possible values are:

***PGM** If you specified a program name for the PGM parameter, the compiler looks for the source program in a member having the same name as the program, and creates a program object with the same name as the program and member.

If you did not specify a program name for the PGM parameter, the compiler looks for the source program in the first member of the database source file, and creates a program object using the name specified in the PROGRAM-ID paragraph.

source-file-member-name
Enter the name of the member that contains the COBOL source.

Top

Generation severity level (GENLVL)

Specifies the severity level that determines if a program object is created. The severity level corresponds to the severity level of the messages produced during compilation of the program. If the severity level of error messages is greater than the value you specify, a program object is not created. For example, if you specify 19 for this parameter, a program object is not created if the severity level of any of the messages is 20 or greater.

29 If errors occur in a program with a severity level greater than 29 a program object is not created.

severity-level
Enter a two-digit number, 00 through 29.

Top

Never-ending program (NEP)

Specifies if this program is a never-ending (or long-running) program. A never-ending program is constantly running and uses system resources (such as disk storage, display stations, or printers) that are not shared with other programs. A never-ending program does not end when all of its requesters are released.

The possible values are:

***NO** The program is not a never-ending program.

***YES** The program is a never-ending program.

Top

Maximum MRT devices (MRTMAX)

Specifies the maximum number of requesting display stations to be allowed during the running of the compiled program. The maximum is 99. The possible values are:

0 There are no requesting display stations.

1-99 A two-digit number, 00 through 99, can be specified.

Top

Text 'description' (TEXT)

A brief description of the program and its function. The possible values are:

***SRCMBRTXT**
Use the same text for the program object as that which describes the database file member containing the COBOL source. If the source comes from a device or in-line file, specifying *SRCMBRTXT has the same effect as specifying *BLANK.

***BLANK**
No text is specified.

'text-description'
Enter the text that briefly describes the program and its function. The text can be a maximum of 50 characters in length and must be enclosed in apostrophes. The apostrophes are not part of the 50-character string.

Top

Source listing options (OPTION)

Specifies the options to use when the COBOL source is compiled. The possible values are:

***SOURCE or *SRC**
The compiler produces a source listing, consisting of the COBOL source input and all compilation-time error messages.

***NOSOURCE or *NOSRC**
The compiler does not produce a source listing.

***NOXREF**
The compiler does not produce a cross-reference listing for the source program.

***XREF** The compiler produces a cross-reference listing for the source program.

***GEN** The compiler creates a program object after the source program is compiled.

***NOGEN**
The compiler performs syntax-checking and lists the appropriate error messages, but does not create a compiled program object.

***SEQUENCE**
The compiler checks the sequence of the source program statements. If statements are not in sequence, a message is printed.

***NOSEQUENCE**
The compiler does not check the sequence of the source program statements.

***NOVBSUM**
Verb-usage counts are not printed.

***VBSUM**
Verb-usage counts are printed.

***NUMBER**
The user-supplied sequence numbers (columns 1 through 6) are used for reference numbers.

***NONUMBER**
The source file sequence numbers (columns 1 through 6) are used for reference numbers.

***LINENUMBER**
The sequence numbers created by the compiler are used for reference numbers. This option combines program source code and source code introduced by COPY statements into one consecutively-numbered sequence. Use this option if you specify FIPS (Federal Information Processing Standards) flagging.

***NOMAP**
The compiler does not list the Data Division map.

***MAP** The compiler lists the Data Division map.

***OPTIONS**
Options in effect are listed for this compilation.

***NOOPTIONS**
Options in effect are not listed for this compilation.

***QUOTE**
Specifies that the quotation mark (") is used to delineate literals.

***APOST**
Specifies that the apostrophe (') is used to delineate literals.

***DEBUG**
If you specify DEBUG for a program that does not contain WITH DEBUGGING MODE clauses, this option has no effect. Treat WITH DEBUGGING ON clause as normal.

***NODEBUG**
Treat WITH DEBUGGING ON clause as comment.

***NOSECLVL**
Second level message text is not listed for this compilation.

***SECLVL**
Second level message text is listed for this compilation.

***PRINT**
The compiler listing created by the COBOLC procedure is printed.

***NOPRINT**
The compiler listing created by the COBOLC procedure is not printed or displayed.

Top

Generation options (GENOPT)

Specifies the options to use when creating the program object. The listings could be required if a problem occurs in COBOL. The possible values are:

***NOLIST**
No IRP (intermediate representation of program), associated hexadecimal code, or error messages are listed.

***LIST** The IRP, its associated hexadecimal code, and any error messages are listed.

***NOXREF**
A cross-reference listing of all objects defined in the IRP is not produced.

***XREF** A cross-reference listing of all objects defined in the IRP is produced.

***NOPATCH**
Space is not reserved in the compiled program for a program patch area.

***PATCH**
Space is reserved in the compiled program for a program patch area. The program patch area can be used for debugging purposes.

***NODUMP**
The program template is not listed.

***DUMP**
The program template is listed.

***NOATR**
The attributes for the IRP source are not listed.

***ATR** The attributes for the IRP source are listed.

***NORANGE**
Does not verify ranges at run-time.

***RANGE**
At run-time, the system verifies that subscripts are within the correct ranges, but does not verify index ranges. It also checks for substring operations in the code created by the compiler.

***UNREF**
Unreferenced data items are included in the compiled program.

***NOUNREF**
Unreferenced data items are not included in the compiled program. This reduces the number of ODT (object definition table) entries used, allowing a larger program to be compiled. The unreferenced data items still appear in the cross-reference listings produced through the *XREF option.

***NOOPTIMIZE**
The compiler performs only standard optimizations for the program.

***OPTIMIZE**
The program object created may run more efficiently, and may require less storage. However, specifying *OPTIMIZE can substantially increase the time required to compile a program.

Top

Print file (PRTFILE)

Specifies the name of the file to which the compiler listing is directed and the library where the file is located. The file should have a minimum record length of 132. If a file with a record length less than 132 is specified, information is lost.

The possible values are:

QSYSPRT
If you do not specify a file name, the compiler listing is directed to QSYSPRT, an IBM-supplied file.

file-name
Enter the name of the file to which the compiler listing is directed.

The possible library values are:

***LIBL** The system searches the library list, *LIBL, to find the library in which the file is located.

***CURLIB**
The library specified as the current library is searched. If you have not assigned a current library, QGPL is used.

library-name
Enter the name of the library in which the file is located.

Top

Library for ICF files (ICFLIB)

Specifies the library containing the ICF record format definitions to be used with the program.

ICF-library-name
Enter the name of the library for ICF files.

Top

Replace program (REPLACE)

Specifies if a new program object is created when a program object of the same name in the same library already exists. The possible values are:

***YES** A new program object is created and any existing program object of the same name in the specified library is moved to library QRPLOBJ.

***NO** A new program object is not created if a program object of the same name already exists in the specified library.

Top

Target release (TGTRLS)

Specifies the release level of the operating system on which you intend to use the object being created.

You can specify an exact release level in the format VxRxMx, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V3R1M0 is Version 3, Release 1, Modification level 0.

Note: To use the object on the target system, you must save the object to the target release level specified on the create command and then restore it on the target system.

For example, if your system is running V3R1M0 and you want to create a program object for distribution to a V2R3M0 system, you must create the program with TGTRLS(V2R3M0) or TGTRLS(*PRV), save the program with TGTRLS(V2R3M0) or TGTRLS(*PRV), and restore the program on the V2R3M0 system. The program object can also be restored on a V3R1M0 system.

Note: The program may be able to be restored on a release level earlier than the one you specified on the create command. Use DSPPGM to determine the earliest release the program can run.

***CURRENT**
The object is to be used on the release of the operating system currently running on your system. For example, if V3R1M0 is running on the system, *CURRENT means you intend to use the object on a system with V3R1M0 installed. You can also use the object on a system with any subsequent release of the operating system installed.

***PRV** The object is to be used on the previous release with modification level 0 of the operating system. For example, if V3R1M0 is running on your system, *PRV means you intend to use the object on a system with V2R3M0 installed. You can also use the object on a system with any subsequent release of the operating system installed.

release-level
Specify the release in the format VxRxMx. The object can be used on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. To see the list of valid values, press F4=Prompt on the TGTRLS parameter of the command.

Top

User profile (USRPRF)

Specifies the user profile that will run the compiled COBOL program. The profile of the program owner or the program user is used to run the program and control which objects can be used by the program (including the authority the program has for each object).

The possible values are:

***USER**
The user profile of the program user is to be used when the program is run.

***OWNER**
The user profiles of both the program's owner and user are to be used when the program is run. The collective sets of object authority in both user profiles are to be used to find and access objects when the program is run. Any objects that are created during the program are owned by the user of the program.

Top

Authority (AUT)

Specifies the authority given to users who do not have specific authority to the program object, who are not on the authorization list, or whose group has no specific authority to the program object. You can alter the authority for all users, or for specific users after the program is created by using the GRTOBJAUT (Grant Object Authority) or RVKOBJAUT (Revoke Object Authority) commands.

The possible values are:

***LIBCRTAUT**
The public authority for the object is taken from the CRTAUT keyword of the target library (the library that is to contain the created program object). This value is determined when the object is created. If the CRTAUT value for the library changes after the object is created, the new value does NOT affect any existing objects.

***ALL** Provides authority for all operations on the program object except those limited to the owner or controlled by authorization list management authority. The user can control the program object's existence, specify security for it, change it, and perform basic functions on it, such as running and debugging the program object.

***CHANGE**
Provides all data authority and the authority to perform all operations on the program object

except those limited to the owner or controlled by object authority and object management authority. The user can change the object and perform basic functions on it, such as running and debugging the program object.

***USE** Provides object operational authority and read authority; authority for basic operations on the program object such as running the program. The user is prevented from changing the object.

Note: A user must have *USE authority to a program to obtain a formatted dump of the variables of the program. To dump variables, the program must also have observable information.

If you do not want some users to be able to dump the variables, then give them only *OBJOPR plus *EXECUTE authority to the program. This will allow them to call the program but not dump its variables.

If you do not want any users to be able to dump the variables, then use Change Program (CHGPGM) to remove the program's observable information.

***EXCLUDE**
The public cannot access the program object.

authorization-list-name
Enter the name of an authorization list of users and authorities to which the program is added. The program object is secured by this authorization list, and the public authority for the program object is set to *AUTL. The authorization list must exist on the system when the CRTS36CBL command is issued.

Top

Compiler debugging dump (DUMP)

An IBM COBOL debugging aid. (For IBM service personnel.)

Top

Intermediate text dump (ITDUMP)

An IBM debugging aid which causes the compiler to dump the internal text at certain times during the compilation. (For IBM service personnel.)

Top

Fix decimal data (FIXDECDTA)

Checks for decimal data errors. A decimal data error occurs when a program attempts to use a packed or zoned number that is not valid. The possible values are:

***YES** The compiler generates code to monitor for and correct decimal data errors.

***NO** The compiler does not generate code to monitor for decimal data errors. If such an error occurs, your program will stop.

Top

Copy file library (CPYLIB)

Specifies the name of the library to be searched when a COPY statement is encountered. If no library is specified, #LIBRARY will be used.

The possible library values are:

***CURLIB**
The library specified as the current library is searched. If you have not assigned a current library, QGPL is used.

***LIBL** The library list is searched to find the library in which the COPY file is located.

library-name
Enter the name of the library in which the COPY file is located.

Top

Examples

Example 1: Compiling a Source Program into a Program Object for use on the System/36 environment

```
CRTS36CBL   PGM(MYLIB/XMPLE1)  SRCFILE(MYLIB/QS36SRC)
            SRCMBR(XMPLE1)  OPTION(*SOURCE)
            TEXT('My COBOL program')
```

This command calls the COBOL compiler to create a program named XMPLE1. The source program is in member XMPLE1 of source file QS36SRC in library MYLIB. A compiler listing is created.

Top

Error messages

***ESCAPE Messages**

CBL1019
Compile failed. Program not created.

SBL9001
Compile failed. Program not created.

SBL9006
TGTRLS(&1) specified, but previous compiler is not installed.

SBL9007
The product library is damaged, or the user is not allowed to use it.

Top

Create RPG II Program (CRTS36RPG)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The command Create System/36 RPG Program (CRTS36RPG) is used to call the System/36-compatible RPG II compiler.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Optional, Positional 1
	Qualifier 1: Program	*Name*, ***CTLSPEC**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QS36SRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 3
GENLVL	Generation severity level	0-99, **21**	Optional
NEP	Never ending program	***NO**, *YES	Optional
MRTMAX	Maximum MRT devices	0-99, **0**	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OPTION	Source listing options	Values (up to 10 repetitions): *SOURCE, *NOSOURCE, *SRC, *NOSRC, *XREF, *NOXREF, *GEN, *NOGEN, *DUMP, *NODUMP, *SECLVL, *NOSECLVL, *CONSOLE, *NOCONSOLE	Optional
GENOPT	Generation options	Values (up to 10 repetitions): *LIST, *NOLIST, *XREF, *NOXREF, *ATR, *NOATR, *DUMP, *NODUMP, *PATCH, *NOPATCH, *OPTIMIZE, *NOOPTIMIZE	Optional
PRTFILE	Print file	*Qualified object name*	Optional
	Qualifier 1: Print file	*Name*, **QSYSPRT**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
ICFLIB	Library for ICF files	*Name*	Optional
REPLACE	Replace program	***YES**, *NO	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional
USRPRF	User profile	***USER**, *OWNER	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
PHSTRC	Phase trace	***NO**, *YES	Optional
ITDUMP	Intermediate text dump	*Character value*, ***NONE**	Optional
SNPDUMP	Snap dump	*Character value*, ***NONE**	Optional
CODELIST	Codelist	*Character value*, ***NONE**, *ALL	Optional
FIXDECDTA	Fix decimal data	***YES**, *NO	Optional

Top

Program (PGM)

Specifies the library and program name by which the compiled RPG program is to be known.

***CTLSPEC**
Specifies the program name used by the system, in positions 75 - 80 of the control specification.

program-name
Enter the name by which the program object will be known.

***CURLIB**
The current library will be used. If you have not specified a current library, #LIBRARY will be used.

library-name
Enter the name of the library in which the compiled program is to be stored.

Top

Source file (SRCFILE)

Specifies the name of the file that contains the source program and the name of the library that contains the file.

QS36SRC
Specifies the default file name.

source-file-name
Enter the name of the source file containing the program to be compiled.

***LIBL** Causes the system to search the library list to find the library containing the source file.

***CURLIB**
The current library will be used. If you have not specified a current library, #LIBRARY will be used.

library-name
Enter the name of the library in which the source file is stored.

Top

Source member (SRCMBR)

Specifies the name of the member of the source file.

***PGM** Specifies the member name for the source program to be compiled.

source-file-member-name
Enter the name of the member containing the source program.

Top

Generation severity level (GENLVL)

Specifies whether or not a program object is to be generated depending on the severity of the errors encountered.

21 Does not generate a program object if error messages have a severity level of 21 or greater.

severity-level-value
Enter a number from 0 through 99. The severity level value of RPG messages does not exceed 50.

Top

Never ending program (NEP)

Specifies whether the current program is to be a never-ending program.

***NO** Specifies that this is not a never-ending program.

***YES** Specifies that this is a never-ending program.

Top

Maximum MRT devices (MRTMAX)

Specifies the maximum number of requesting display stations.

0 This is a SRT program.

maximum-terminals
Enter a number from 0 through 99 to show the maximum number of requesting terminals.

Top

Text 'description' (TEXT)

Allows the user to enter text that describes the program and its function.

***SRCMBRTXT**
Uses the text in the source program.

***BLANK**
Omits the text.

text Enter the description, enclosed in apostrophes, in 50 characters or less.

Top

Source listing options (OPTION)

Specifies the options to use when the source program is compiled.

***SOURCE or *SRC**
Produces a source listing with compile-time errors.

***NOSOURCE or *NOSRC**
Does not produce a source listing.

***XREF** Produces a cross-reference listing and key-field information table.

***NOXREF**
Does not produce a cross-reference listing and key-field information table. This is the default when *NOSOURCE is specified.

***GEN** Creates a program that can be run.

***NOGEN**
Does not create a program that can be run.

***NODUMP**
Does not print the program template when an error occurs.

***DUMP**
Prints a program template when an error occurs.

***NOSECLVL**
Does not print second-level messages.

***SECLVL**
Prints second-level messages following first-level messages.

***CONSOLE**
Creates display formats for a CONSOLE file if the program being compiled contains a CONSOLE file.

***NOCONSOLE**
Does not create display formats for a CONSOLE file.

Top

Generation options (GENOPT)

Specifies the options to use when the program object is created.

***NOLIST**
Does not produce a listing of the intermediate representation of a program (IRP).

***LIST** Produces a listing of the intermediate representation of a program (IRP).

***NOXREF**
Does not print a cross-reference listing for the IRP.

***XREF** Prints a cross-reference listing for the IRP.

***NOATR**
Does not produce an attribute listing for the IRP.

***ATR** Produces an attribute listing for the IRP.

***NODUMP**
Does not print the program template when an error occurs.

***DUMP**
Prints a program template when an error occurs.

***NOPATCH**
Does not reserve a program patch area in the compiled program.

***PATCH**
Reserves a program patch area in the compiled program.

***NOOPTIMIZE**
Does not improve a program for more efficient processing.

***OPTIMIZE**
Improves a program for more efficient processing.

Top

Print file (PRTFILE)

Specifies the name of the file in which the compiler listing is to be placed and the library in which the file is located.

QSYSPRT
If a file name is not specified, the compiler listing is placed in QSYSPRT file.

print-file-name
Enter the name of the file in which the compiler listing is to be placed.

***LIBL** The system searches the library list to find the library.

***CURLIB**
The current library will be used. If you have not specified a current library, #LIBRARY will be used.

library-name
Enter the name of the library in which the file is stored.

Top

Library for ICF files (ICFLIB)

Specifies the library containing the ICF record format definitions to be used with the program.

ICF-library-name
Enter the name of the library for ICF files.

Top

Replace program (REPLACE)

Specifies whether or not a new program object is to be created if a program with the same name already exists in the specified library.

***YES** A new program object will be created and any existing program object of the same name in the specified library will be moved to library QRPLOBJ.

***NO** A new program object will not be created if a program object of the same name exists in the specified library.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which you intend to use the object being created. In the examples given for the *CURRENT and *PRV values, and when specifying the target-release value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V2R3M0 is version 2, release 3, modification level 0.

Valid values for this parameter change every release.

The possible values are:

***CURRENT**
The object is to be used on the release of the operating system currently running on your system. For example, if V2R3M5 is running on the system, *CURRENT means that you intend to use the

object on a system with V2R3M5 installed. You can can also use the object on a system with any subsequent release of the operating system installed.

***PRV** The object is to be used on the previous release with modification level 0 of the operating system. For example, if V2R3M5 is running on your system, *PRV means you intend to use the object on a system with V2R2M0 installed. You can also use the object on a system with any subsequent release of the operating system installed.

target-release
Specify the release in the format VxRxMx. You can use the object on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. If you specify a release-level that is earlier than the earliest release level supported by this command, an error message is sent indicating the earliest supported release.

Note: The program may be able to be restored on a release level earlier than the one you specified on the create command. Use DSPPGM to determine the earliest release the program can run.

Top

User profile (USRPRF)

Specifies the user profile that the compiled RPG program is to run under.

***USER**
Uses the program user's profile when the program runs.

***OWNER**
Uses the user profiles of both the program's owner and user when the program runs. The USRPRF parameter will not be updated if the program already exists.

Top

Authority (AUT)

Specifies what authority is granted for the program.

***LIBCRTAUT**
The default public authority for created objects is taken from the CRTAUT keyword which is associated with the target library. The value is determined at create time. If the value of the CRTAUT keyword of the library changes after the create, the new value will not affect any existing objects.

***ALL** Complete authority for the program except for transferal of object ownership.

***CHANGE**
All operations are permitted except those dealing with the object's existence and its management.

***USE** Permission to read or run the compiled program, but not to debug or to change it.

Note: A user must have *USE authority to a program to obtain a formatted dump of the variables of the program. To dump variables, the program must also have observable information.

If you do not want some users to be able to dump the variables, then give them only *OBJOPR plus *EXECUTE authority to the program. This will allow them to call the program but not dump its variables.

If you do not want any users to be able to dump the variables, then use Change Program (CHGPGM) to remove the program's observable information.

***EXCLUDE**
No authority.

authorization-list
The name of the authorization list which secures the object. The public authority will be *AUTL.

Top

Phase trace (PHSTRC)

Specifies whether trace information about the compiler phases will be included in the listing.

***NO** Does not provide compiler phase information.

***YES** Provides compiler phase information.

Top

Intermediate text dump (ITDUMP)

Produces a dynamic listing of intermediate text for the specified compiler phases.

***NONE**
Does not produce an intermediate text dump.

phase-name
Enter the fourth and fifth characters of from 1 to 25 phase names.

Top

Snap dump (SNPDUMP)

Produces a listing of major data areas and intermediate text following compilation of one or more specified phases.

***NONE**
Does not produce a snap dump.

phase-name
Enter the fourth and fifth characters of from 1 to 25 phase names.

Top

Codelist (CODELIST)

Causes dynamic listing of the intermediate representation of a program (IRP) for the specified compiler phases.

***NONE**
Does not produce IRP listings.

***ALL** Produces an IRP listing for each compiler phase.

phase-name
Enter the fourth and fifth characters of from 1 to 25 phase names.

Top

Fix decimal data (FIXDECDTA)

Specifies whether decimal data which is not valid is corrected or an error is signaled.

***YES** Corrects data which is not valid by setting digit values in the range A through F to zero, and not valid signs to plus.

***NO** Signals an error to the program without correcting the data which is not valid.

Top

Examples

Example 1: Compiling a Source Program into a Program Object

```
CRTS36RPG   PGM(MYLIB/XMPLE1)
            SRCFILE(MYLIB/QS36SRC)  SRCMBR(XMPLE1)
            OPTION(*SOURCE)  TEXT('My RPG II program')
```

This command calls the System/36 Compatible RPG II compiler to create a program named XMPLE1. The source program is in member XMPLE1 of source file QS36SRC in library MYLIB. A compiler listing is created.

Top

Error messages

***ESCAPE Messages**

QRG9001
Compilation failed. Program is not created.

QRG9004
The release &1 specified on the TGTRLS option is not supported.

Top

Create Console Display File (CRTS36RPGR)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Create System/36 RPGR command (CRTS36RPGR) is used to create a CONSOLE display file.

Top

Parameters

Keyword	Description	Choices	Notes
SRCMBR	Source member	*Name*	Required, Positional 1
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QS36SRC**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
OUTLIB	Output library	*Name*, ***CURLIB**	Optional, Positional 3
FMTSRCF	File to receive S/36 formats	Single values: *NONE Other values: *Qualified object name*	Optional, Positional 4
	Qualifier 1: File to receive S/36 formats	*Name*, ***SRCFILE**	
	Qualifier 2: Library	*Name*, ***SRCLIB**	
FMTMBR	Member to receive formats	*Name*, ***CRTDFT**	Optional, Positional 5
DDSSRCF	File to receive DDS	Single values: *NONE Other values: *Qualified object name*	Optional
	Qualifier 1: File to receive DDS	*Name*, ***SRCFILE**	
	Qualifier 2: Library	*Name*, ***SRCLIB**	
DDSMBR	Member to receive DDS	*Name*, ***CRTDFT**	Optional
GEN	Generate console formats	***YES**, *NO	Optional
REPLACE	Replace output members	***YES**, *NO	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional

Top

Source member (SRCMBR)

Specifies the source member that contains the RPG II program specifications.

source-file-member-name
Enter the name of the member containing the source program.

Top

Source file (SRCFILE)

Specifies the file that contains the RPG II source member and the library that contains the file.

QS36SRC
Specifies the default file name.

source-file-name
Enter the name of the source file containing the source member.

***CURLIB**
The current library will be used. If you have not specified a current library, #LIBRARY will be used.

library-name
Enter the name of the library in which the source file is stored.

Top

Output library (OUTLIB)

Specifies the library that contains the display format object member.

***CURLIB**
The current library will be used. If you have not specified a current library, #LIBRARY will be used.

output-library-name
Enter the name of the library in which the object member will be stored.

Top

File to receive S/36 formats (FMTSRCF)

Specifies the file in which the member containing the S and D specifications will be stored and the library that will contain the file.

***SRCFILE**
Uses the file name specified in the source file parameter (SRCFILE).

***NONE**
Specifies that the S and D specifications will not be produced.

S&D-source-file-name
Enter the name of the file that will contain the S and D specifications.

***SRCLIB**
Uses the same library that you specified in the source file library name (SRCFILE Parameter).

S&D-library-name
Enter the name of the library that will contain the S and D source file.

Top

Member to receive formats (FMTMBR)

Specifies the member that will contain the S and D specifications.

***CRTDFT**
Uses the specified source member name (SRCMBR) with 'FM' added to the end of it.

S&D-member-name
Enter the name of the member that will contain the S and D specifications.

Top

File to receive DDS (DDSSRCF)

Specifies the file in which the member containing the DDS specifications will be stored and the library that will contain the file.

***SRCFILE**
Uses the file name specified in the source file parameter (SRCFILE).

***NONE**
Specifies that the DDS specifications will not be produced.

DDS-source-file-name
Enter the name of the source file that will contain the DDS member.

***SRCLIB**
Uses the same library that you specified in the source file library name (SRCFILE Parameter).

DDS-library-name
Enter the name of the library that will contain the DDS source file.

Top

Member to receive DDS (DDSMBR)

Specifies the member that will contain the DDS specifications.

***CRTDFT**
Uses the specified source member name (SRCMBR) with 'A' added to the end of it.

DDS-member-name
Enter the name of the member that will contain the DDS specifications.

Top

Generate console formats (GEN)

Specifies that the display formats for the CONSOLE file are to be created.

***YES** Specifies that display formats will be generated.

***NO** Specifies that display formats will not be generated.

Top

Replace output members (REPLACE)

Specifies whether or not a new CONSOLE display file is to be created if a file with the same name already exists in the specified library.

***YES** Specifies that a new CONSOLE display file will be created and any existing CONSOLE display file of the same name in the specified library will be deleted.

***NO** Specifies that a new CONSOLE display file will not be created if a member of the same name exists in the specified library.

Top

Authority (AUT)

Specifies what authority is granted for the program.

***LIBCRTAUT**
The default public authority for created objects is taken from the CRTAUT keyword which is associated with the target library. The value is determined at create time. If the value of the CRTAUT keyword of the library changes after the create, the new value will not affect any existing objects.

***ALL** Complete authority for the program except for transferal of object ownership.

***CHANGE**
All operations are permitted except those dealing with the object's existence and its management.

***USE** Permission to read or run the compiled program, but not to debug or to change it.

***EXCLUDE**
No authority.

authorization-list
The name of the authorization list which secures the object. The public authority will be *AUTL.

Top

Examples

Example 1: Compiling a Source Program into a Program Object

```
CRTS36RPGR   SRCMBR(XMPLE1) SRCFILE(MYLIB/QS36SRC)
             OUTLIB(MYLIB)
             FMTSRCF(*SRCLIB/QS36DDSSRC) FMTMBR(*CRTDFT)
             DDSSRCF(*SRCLIB/QDDSSRC) DDSMBR(*CRTDFT)
             REPLACE(*YES)
```

This command creates a display file for a CONSOLE file from the RPG II source member XMPLE1. The name of the file will be the name in column 75 of the Control (H) specification in the RPG source, followed by FM. For example, if the Control specification contains XMPLE1 in columns 75-80, the display file will be named XMPLE1FM. Source member XMPLE1FM will be created in file MYLIB/QS36DDSSRC, containing System/36 Formats (S and D specifications). Source member XMPLE1A will be created in file MYLIB/QDDSSRC, containing the DSPF DDS source used to create the file MYLIB/XMPLE1FM.

Top

Error messages

None

Top

Create S/36 RPG II Auto Report (CRTS36RPT)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The command Create System/36 Auto Report Program (CRTS36RPT) is used to call the System/36-compatible RPG II compiler.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Optional, Positional 1
	Qualifier 1: Program	*Name*, ***CTLSPEC**	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, **QS36SRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 3
GENLVL	Generation severity level	0-99, **21**	Optional
NEP	Never ending program	***NO**, *YES	Optional
MRTMAX	Maximum MRT devices	0-99, **0**	Optional
TEXT	Text 'description'	*Character value*, ***SRCMBRTXT**, *BLANK	Optional
OPTION	Source listing options	Values (up to 10 repetitions): *SOURCE, *NOSOURCE, *SRC, *NOSRC, *XREF, *NOXREF, *GEN, *NOGEN, *DUMP, *NODUMP, *SECLVL, *NOSECLVL, *CONSOLE, *NOCONSOLE	Optional
GENOPT	Generation options	Values (up to 10 repetitions): *LIST, *NOLIST, *XREF, *NOXREF, *ATR, *NOATR, *DUMP, *NODUMP, *PATCH, *NOPATCH, *OPTIMIZE, *NOOPTIMIZE	Optional
PRTFILE	Print file	*Qualified object name*	Optional
	Qualifier 1: Print file	*Name*, **QSYSPRT**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
ICFLIB	Library for ICF files	*Name*	Optional
RPTOPT	Auto report options	Values (up to 10 repetitions): *SOURCE, *NOSOURCE, *SRC, *NOSRC, *FLOW, *NOFLOW, *AST, *NOAST, *DATE, *NODATE, *COMPILE, *NOCOMPILE, *SECLVL, *NOSECLVL	Optional
OUTFILE	Save file for auto report	*Qualified object name*	Optional
	Qualifier 1: Save file for auto report	*Name*, ***NONE**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
OUTMBR	Auto report save file member	*Name*, ***NONE**	Optional
REPLACE	Replace program	***YES**, *NO	Optional
TGTRLS	Target release	*Simple name*, ***CURRENT**, *PRV	Optional

Keyword	Description	Choices	Notes
USRPRF	User profile	***USER**, *OWNER	Optional
AUT	Authority	*Name*, ***LIBCRTAUT**, *ALL, *CHANGE, *USE, *EXCLUDE	Optional
PHSTRC	Phase trace	***NO**, *YES	Optional
ITDUMP	Intermediate text dump	*Character value*, ***NONE**	Optional
SNPDUMP	Snap dump	*Character value*, ***NONE**	Optional
CODELIST	Codelist	*Character value*, ***NONE**, *ALL	Optional
FIXDECDTA	Fix decimal data	***YES**, *NO	Optional

Top

Program (PGM)

Specifies the library and program name by which the compiled RPG program is to be known.

***CTLSPEC**
Specifies the program name used by the system, in positions 75 - 80 of the control specification.

program-name
Enter the name by which the program object will be known.

***CURLIB**
The current library will be used. If you have not specified a current library, #LIBRARY will be used.

library-name
Enter the name of the library in which the compiled program is to be stored.

Top

Source file (SRCFILE)

Specifies the name of the file that contains the source program and the name of the library that contains the file.

QS36SRC
Specifies the default file name.

source-file-name
Enter the name of the source file containing the program to be compiled.

***LIBL** Causes the system to search the library list to find the library containing the source file.

***CURLIB**
The current library will be used. If you have not specified a current library, #LIBRARY will be used.

library-name
Enter the name of the library in which the source file is stored.

Top

Source member (SRCMBR)

Specifies the name of the member of the source file.

***PGM** Uses the name specified for the program name (PGM Parameter).

source-file-member-name
Enter the name of the member containing the source program.

Top

Generation severity level (GENLVL)

Specifies whether or not a program object is to be generated depending on the severity of the errors encountered.

21 Does not generate a program object if error messages have a severity level of 21 or greater.

severity-level-value
Enter a number from 0 through 99. The severity level value of RPG messages does not exceed 50.

Top

Never ending program (NEP)

Specifies whether the current program is to be a never-ending program.

***NO** Specifies that this is not a never-ending program.

***YES** Specifies that this is a never-ending program.

Top

Maximum MRT devices (MRTMAX)

Specifies the maximum number of requesting display stations.

0 This is a SRT program.

maximum-terminals
Enter a number from 0 through 99 to show the maximum number of requesting terminals.

Top

Text 'description' (TEXT)

Allows the user to enter text that describes the program and its function.

***SRCMBRTXT**
Uses the text in the source program.

***BLANK**
Omits the text.

text Enter the description, enclosed in apostrophes, in 50 characters or less.

Top

Source listing options (OPTION)

Specifies the options to use when the source program is compiled.

***SOURCE or *SRC**
Produces a source listing with compile-time errors.

***NOSOURCE or *NOSRC**
Does not produce a source listing.

***XREF** Produces a cross-reference listing and key-field information table.

***NOXREF**
Does not produce a source listing and key-field information table. This is the default when *NOSOURCE is specified.

***GEN** Creates a program that can be run.

***NOGEN**
Does not create a program that can be run.

***NODUMP**
Does not print the program template when an error occurs.

***DUMP**
Prints a program template when an error occurs.

***NOSECLVL**
Does not print second-level messages.

***SECLVL**
Prints second-level messages following first-level messages.

***CONSOLE**
Creates display formats for a CONSOLE file if the program being compiled contains a CONSOLE file.

***NOCONSOLE**
Does not create display formats for a CONSOLE file.

Top

Generation options (GENOPT)

Specifies the options to use when the program object is created.

***NOLIST**
Does not produce a listing of the intermediate representation of a program (IRP).

***LIST** Produces a listing of the intermediate representation of a program (IRP).

***NOXREF**
Does not print a cross-reference listing for the IRP.

***XREF** Prints a cross-reference listing for the IRP.

***NOATR**
Does not produce an attribute listing for the IRP.

***ATR** Produces an attribute listing for the IRP.

***NODUMP**
Does not print the program template when an error occurs.

***DUMP**
Prints a program template when an error occurs.

***NOPATCH**
Does not reserve a program patch area in the compiled program.

***PATCH**
Reserves a program patch area in the compiled program.

***NOOPTIMIZE**
Does not improve a program for more efficient processing.

***OPTIMIZE**
Improves a program for more efficient processing.

Top

Print file (PRTFILE)

Specifies the name of the file in which the compiler listing is to be placed and the library in which the file is located.

QSYSPRT
If a file name is not specified, the compiler listing is placed in QSYSPRT file.

print-file-name
Enter the name of the file in which the compiler listing is to be placed.

***LIBL** The system searches the library list to find the library.

***CURLIB**
The current library will be used. If you have not specified a current library, #LIBRARY will be used.

library-name
Enter the name of the library in which the file is stored.

Top

Library for ICF files (ICFLIB)

Specifies the library containing the ICF record format definitions to be used with the program.

ICF-library-name
Enter the name of the library for ICF files.

Top

Auto report options (RPTOPT)

Specifies options to use when the automatic report source program is compiled.

***NOSOURCE or *NOSRC**
Does not produce a source listing.

***SOURCE or SRC**
Produces a source listing.

***NOFLOW**
Does not write a flow of the major routines.

***FLOW**
Writes a flow of the major routines.

***NOAST**
Does not produce asterisks for the total lines of output.

***AST** Produces asterisks for total lines of output.

***DATE**
Prints the date and page number on the first *AUTO page heading line.

***NODATE**
Does not print the date and page number on the first *AUTO page heading line.

***COMPILE**
Calls the RPG compiler after the automatic report source compiles.

***NOCOMPILE**
Does not call the RPG compiler after the automatic report source compiles.

***NOSECLVL**
Does not print second-level messages.

***SECLVL**
Prints second-level messages following first level messages.

Top

Save file for auto report (OUTFILE)

Specifies the name of the output file to contain the compiled automatic report, and the library in which the file will be located.

***NONE**
Creates a file in QTEMP to pass the source program produced to the compiler.

output-file-name
Enter the name of the file that contains the complete RPG II source program.

***LIBL** Causes the system to search the library list to find the library containing the source file.

***CURLIB**
The current library will be used. If you have not specified a current library, #LIBRARY will be used.

library-name
Enter the name of the library in which the source file is stored.

Top

Auto report save file member (OUTMBR)

Specifies the name of the member of the file to contain the output from the automatic report.

***NONE**
Uses the first member in the file as the member name.

file-member-name
Enter the name of the member that is to contain the output from the automatic report.

Top

Replace program (REPLACE)

Specifies whether or not a new program object is to be created if a program with the same name already exists in the specified library.

***YES** A new program object will be created and any existing program object of the same name in the specified library will be moved to library QRPLOBJ.

***NO** A new program object will not be created if a program object of the same name exists in the specified library.

Top

Target release (TGTRLS)

Specifies the release of the operating system on which you intend to use the object being created. In the examples given for the *CURRENT and *PRV values, and when specifying the target-release value, the format VxRxMx is used to specify the release, where Vx is the version, Rx is the release, and Mx is the modification level. For example, V2R3M0 is version 2, release 3, modification level 0.

Valid values for this parameter change every release.

The possible values are:

***CURRENT**
The object is to be used on the release of the operating system currently running on your system. For example, if V2R3M5 is running on the system, *CURRENT means that you intend to use the object on a system with V2R3M5 installed. You can can also use the object on a system with any subsequent release of the operating system installed.

***PRV** The object is to be used on the previous release with modification level 0 of the operating system. For example, if V2R3M5 is running on your system, *PRV means you intend to use the object on a system with V2R2M0 installed. You can also use the object on a system with any subsequent release of the operating system installed.

target-release
Specify the release in the format VxRxMx. You can use the object on a system with the specified release or with any subsequent release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. If you specify a release-level that is earlier than the earliest release level supported by this command, an error message is sent indicating the earliest supported release.

Note: The program may be able to be restored on a release level earlier than the one you specified on the create command. Use DSPPGM to determine the earliest release the program can run.

Top

User profile (USRPRF)

Specifies the user profile that the compiled RPG program is to run under.

***USER**
Uses the program user's profile when the program runs.

***OWNER**
Uses the user profiles of both the program's owner and user when the program runs. The USRPRF parameter will not be updated if the program already exists.

Top

Authority (AUT)

Specifies what authority is granted for the program.

***LIBCRTAUT**
The default public authority for created objects is taken from the CRTAUT keyword which is associated with the target library. The value is determined at create time. If the value of the CRTAUT keyword of the library changes after the create, the new value will not affect any existing objects.

***ALL** Complete authority for the program except for transferal of object ownership.

***CHANGE**
All operations are permitted except those dealing with the object's existence and its management.

***USE** Permission to read or run the compiled program, but not to debug or to change it.

Note: A user must have *USE authority to a program to obtain a formatted dump of the variables of the program. To dump variables, the program must also have observable information.

If you do not want some users to be able to dump the variables, then give them only *OBJOPR plus *EXECUTE authority to the program. This will allow them to call the program but not dump its variables.

If you do not want any users to be able to dump the variables, then use Change Program (CHGPGM) to remove the program's observable information.

***EXCLUDE**
No authority.

authorization-list
The name of the authorization list which secures the object. The public authority will be *AUTL.

Top

Phase trace (PHSTRC)

Specifies whether trace information about the compiler phases will be included in the listing.

***NO** Does not provide compiler phase information.

***YES** Provides compiler phase information.

Top

Intermediate text dump (ITDUMP)

Produces a dynamic listing of intermediate text for the specified compiler phases.

***NONE**
Does not produce an intermediate text dump.

phase-name
Enter the fourth and fifth characters of from 1 to 25 phase names.

Top

Snap dump (SNPDUMP)

Produces a listing of major data areas and intermediate text following compilation of one or more specified phases.

***NONE**
Does not produce a snap dump.

phase-name
Enter the fourth and fifth characters of from 1 to 25 phase names.

Top

Codelist (CODELIST)

Causes dynamic listing of the intermediate representation of a program (IRP) for the specified compiler phases.

***NONE**
Does not produce IRP listings.

***ALL** Produces an IRP listing for each compiler phase.

phase-name
Enter the fourth and fifth characters of from 1 to 25 phase names.

Top

Fix decimal data (FIXDECDTA)

Specifies whether decimal data which is not valid is corrected or an error is signaled.

***YES** Corrects data which is not valid by setting digit values in the range A through F to zero, and not valid signs to plus.

***NO** Signals an error to the program without correcting the data which is not valid.

Top

Examples

Example 1: Compiling a Source Program into a Program Object

```
CRTS36RPT   PGM(MYLIB/XMPLE1)
            SRCFILE(MYLIB/QS36SRC) SRCMBR(XMPLE1)
            OPTION(*SOURCE)
            TEXT('My RPG II Auto Report Program')
```

This command calls the System/36 Compatible RPG II compiler to create a Auto Report program named XMPLE1. The source program is in member XMPLE1 of source file QS36SRC in library MYLIB. A compiler listing is created.

Top

Error messages

***ESCAPE Messages**

RPT9001
Auto Report failed.

QRG9004
The release &1 specified on the TGTRLS option is not supported.

RPT0082
Auto Report generation terminated because severe errors occurred.

Top

Convert RPG Source (CVTRPGSRC)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

Convert RPG Source

The Convert RPG Source (CVTRPGSRC) command converts RPG III or RPG/400 source code to ILE RPG source code.

Top

Parameters

Keyword	Description	Choices	Notes
FROMFILE	From file	*Qualified object name*	Required, Positional 1
	Qualifier 1: From file	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
FROMMBR	From member	*Generic name, name*, *ALL	Required, Positional 2
TOFILE	To file	Single values: *NONE Other values: *Qualified object name*	Optional, Positional 3
	Qualifier 1: To file	*Name*, **QRPGLESRC**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
TOMBR	To member	*Name*, ***FROMMBR**	Optional, Positional 4
EXPCPY	Expand copy member	***NO**, *YES	Optional
CVTRPT	Print conversion report	***YES**, *NO	Optional
SECLVL	Include second level text	***NO**, *YES	Optional
INSRTPL	Insert specification template	***NO**, *YES	Optional
LOGFILE	Log file	Single values: *NONE Other values: *Qualified object name*	Optional
	Qualifier 1: Log file	*Name*, **QRNCVTLG**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
LOGMBR	Log file member	*Name*, ***FIRST**, *LAST	Optional

Top

From file (FROMFILE)

Specifies the name of the source file that contains the RPG III or RPG/400 source code to be converted and the library where the source file is stored. This is a required parameter; there is no default file name.

source-file-name
Enter the name of the source file that contains the source member(s) to be converted.

***LIBL** The system searches the library list to find the library where the source file is stored.

***CURLIB**
The current library is used to find the source file. If you have not specified a current library, then the library QGPL is used.

library-name
Enter the name of the library where the source file is stored.

Top

From member (FROMMBR)

Specifies the name(s) of the source member(s) to be converted. This is a required parameter; there is no default member name.

The valid source member types of source members to be converted are RPG, RPT, RPG38, RPT38, SQLRPG and blank. The Convert RPG Source command does not support source member types RPG36, RPT36 and other non-RPG source member types (for example, CLP and TXT).

source-file-member-name
Enter the name of the source member to be converted.

***ALL** The command converts all the members in the source file specified.

generic-source-file-member-name*
Enter the generic name of members having the same prefix in their names followed by a '*' (asterisk). The command converts all the members having the generic name in the source file specified. For example, specifying FROMMBR(PR*) will result in the conversion of all members whose names begin with 'PR'.

Top

To file (TOFILE)

Specifies the name of the source file that contains the converted source members and the library where the converted source file is stored. The converted source file must exist and should have a record length of 112 characters: 12 for the sequence number and date, 80 for the code and 20 for the comments.

QRPGLESRC
The default source file QRPGLESRC contains the converted source member(s).

***NONE**
No converted source member is generated. The TOMBR parameter value is ignored. CVTRPT(*YES) must also be specified or the conversion will end immediately.

This feature allows you to find some potential problems without having to create the converted source member.

source-file-name
Enter the name of the converted source file that contains the converted source member(s).

The TOFILE source file name must be different from the FROMFILE source file name if the TOFILE library name is the same as the FROMFILE library.

***LIBL** The system searches the library list to find the library where the converted source file is stored.

***CURLIB**
The current library is used to find the converted source file. If you have not specified a current library then the library QGPL is used.

library-name
Enter the name of the library where the converted source file is stored.

Top

To member (TOMBR)

Specifies the name(s) of the converted source member(s) in the converted source file. If the value specified on the FROMMBR parameter is (*ALL) or generic*, then TOMBR must be equal to *FROMMBR.

***FROMMBR**
The member name specified in the FROMMBR parameter is used as the converted source member name. If FROMMBR(*ALL) is specified, then all the source members in the FROMFILE are converted. The converted source members have the same names as those of the original source members. If a generic name is specified in the FROMMBR parameter, then all the source members specified having the same prefix in their names are converted. The converted source members have the same names as those of the original generic source members.

source-file-member-name
Enter the name of the converted source member. If the member does not exist, it will be created.

Top

Expand copy member (EXPCPY)

Specifies whether a /COPY member(s) is expanded into the converted source member. EXPCPY(*YES) should be specified only if you are having conversion problems pertaining to /COPY members.

***NO** Do not expand the /COPY file member(s) into the converted source. This is the default.

***YES** Expand the /COPY file member(s) into the converted source.

Note: If the member is of type RPT or RPT38, EXPCPY(*YES) or EXPCPY(*NO) has no effect because the auto report program will always expand the /COPY members.

Top

Print conversion report (CVTRPT)

Specifies whether or not a conversion report is printed.

***YES** The conversion report is printed. This is the default.

***NO** The conversion report is not printed.

Top

Include second level text (SECLVL)

Specifies whether second-level text is printed in the conversion report.

***NO** Second-level message text is not printed in the conversion report. This is the default.

***YES** Second-level message text is printed in the conversion report.

Top

Insert specification template (INSRTPL)

Specifies if the ILE RPG specification templates (H-, F-, D-, I-, C- and/or O-specification template) are inserted in the converted source member(s).

***NO** A specification template is not inserted in the converted source member. This is the default.

***YES** A specification template is inserted in the converted source member. Each specification template is inserted at the beginning of the appropriate specification section.

Top

Log file (LOGFILE)

Specifies the name of the log file that is used to track the conversion information. Unless *NONE is specified, there must be a log file. The file must already exist, and it must be a physical data file. Create the log file by using the CPYF command with the ″From object″ file QARNCVTLG in library QRPGLE and the ″New object″ file QRNCVTLG in your library.

QRNCVTLG
The default log file QRNCVTLG is used to contain the conversion information.

***NONE**
Conversion information is not written to a log file.

log-file-name
Enter the name of the log file that is to be used to track the conversion information.

***LIBL** The system searches the library list to find the library where the log file is stored.

library-name
Enter the name of the library where the log file is stored.

Top

Log file member (LOGMBR)

Specifies the name of the log file member used to track conversion information. The new information is added to the existing data in the specified log file member.

If the log file contains no members, then a member having the same name as the log file is created.

***FIRST**
The command uses the first member in the specified log file. This is the default.

***LAST**
The command uses the last member in the specified log file.

log-file-member-name
Enter the name of the file member used to track conversion information.

Top

Examples

Example 1: Converting RPG III Source to RPG IV Source

```
CVTRPGSRC   FROMFILE(MYLIB/QRPGSRC) FROMMBR(XMPLE1)
            TOFILE(MYLIB/QRPGLESRC) TOMBR(*FROMMBR)
            EXPCPY(*NO)
            CVTRPT(*YES) LOGFILE(MYLIB/QRNCVTLG)
```

This command converts the RPG III source member XMPLE1 in file MYLIB/QRPGSRC to an RPG IV source member of the same name in file MYLIB/QRPGLESRC. /COPY statements in the RPG III program are not expanded; they remain as /COPY statements in the RPG IV program. A report is printed. The status of each conversion is placed in the file MYLIB/QRNCVTLG.

Top

Error messages

***ESCAPE Messages**

RNS9350
Conversion terminated.

Top

End COBOL Debug (ENDCBLDBG)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

This command deactivates the debugging code that is created when the WITH DEBUGGING MODE clause is used in a COBOL program. This command must be entered for each COBOL program for which debugging is to be stopped.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Required, Positional 1
	Qualifier 1: Program	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	

Top

Program (PGM)

Specifies the name of the compiled COBOL program and the library where it is located. This is a required parameter. The possible values are:

program-name
Enter the name by which the compiled COBOL program is known.

The possible library values are:

***LIBL** The system searches the library list to find the library where the program is located.

***CURLIB**
The current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the created program is located.

Top

Examples

Example 1: Deactivate the debugging code for a COBOL program

```
ENDCBLDBG PGM(MYLIB/XMPLE1)
```

This command deactivates the debugging code that was created for the COBOL program XMPLE1 in library MYLIB.

Top

Error messages

***ESCAPE Messages**

CBE7018
Program &1 not found.

CBE7019
Library &1 not found.

LBE7018
Program &1 not found.

LBE7019
Library &1 not found.

Top

End ISDB (ENDISDB)

Where allowed to run: Interactive environments (*INTERACT *IPGM *IMOD *IREXX *EXEC)
Threadsafe: No

End Interactive Source Debugger

Use this command to end your Interactive Source Debugger session for serviced jobs. There are no parameters for this command.

When you issue this command, all programs that you were debugging are removed from debug mode.

A log of the commands you used in your debugging session is saved in a file named QTEMP/QIXALOG

ISDB overwrites this log file every time you start a new session. If you want to keep a log file to use the commands again, rename the file.

Error messages for ENDISDB

None

Top

Parameters

None

Top

Examples

None

Top

Error messages

None

Top

Find String Using PDM (FNDSTRPDM)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Find String Using PDM (FNDSTRPDM) command allows you to search for character or hexadecimal strings in source physical file members or data physical file members. You can use any Programming Development Manager (PDM) option or one of your own user-defined options on each member that contains a match for the string.

Top

Parameters

Keyword	Description	Choices	Notes
STRING	Find 'string'	*Character value*	Required, Positional 1
FILE	File	*Qualified object name*	Required, Positional 2
	Qualifier 1: File	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
MBR	Member	Values (up to 300 repetitions): *Character value*, *ALL	Required, Positional 3
OPTION	Operation to perform	*Element list*	Required, Positional 4
	Element 1: Option	*Character value*, *EDIT, *COPY, *DLT, *DSP, *PRT, *RNM, *DSPD, *SAVE, *CHGT, *CMPR, *CMPL, *MOD, *MRG, *RUNP, *SDA, *DFU, *RLU, *NONE	
	Element 2: Prompt	***NOPROMPT**, *PROMPT	
COL	Columns to search	*Element list*	Optional
	Element 1: From column	*Character value*, **1**, *RCDLEN	
	Element 2: To column	*Character value*, ***RCDLEN**	
CASE	Kind of match	***IGNORE**, *MATCH	Optional
PRTMBRLIST	Print list	***NO**, *YES	Optional
PRTRCDS	Print records	Single values: ***NONE** Other values: *Element list*	Optional
	Element 1: Number to find	*Character value*, '_____', *ALL	
	Element 2: Print format	***CHAR**, *HEX, *ALTHEX	
	Element 3: Mark record	***MARK**, *NOMARK	
	Element 4: Record overflow	***FOLD**, *TRUNCATE	
PARM	Parameters	*Character value*	Optional

Top

Find 'string' (STRING)

Specifies the string you want to search for

This is a required parameter.

character-value
Specify the string to search for, enclosed in single quotation marks. The string can be character or hexadecimal.

Top

File (FILE)

Specifies the file that contains the members you want to search. The file to be searched can be a source physical file or a data physical file.

This is a required parameter.

Qualifier 1: File

name Specify the name of the physical file that contains the members you want to search.

Qualifier 2: Library

***LIBL** All libraries in the job library list will be searched for the specified file.

***CURLIB**
The current library for the job will be searched for the specified file. If no current library is defined, QGPL is used as the current library.

name Specify the name of the library to be searched for the specified file.

Top

Member (MBR)

Specifies the member or members to be searched. You can use this parameter to search all the members or a subset of members in the specified file.

This is a required parameter.

You can specify 300 values for this parameter.

***ALL** Search all the file members for the specified string.

generic-name
Specify the generic name of the file members to be searched.

The generic name can be in one of the following formats:

ABC* Searches all members with names that begin with the characters ABC. For example, ABC, ABCD, or ABCTEST.

***ABC** Searches all members with names that end with the characters ABC. For example, ABC, DABC, or TESTABC.

B Searches all members that have the character B anywhere in the member name. For example, B, BALL, ABCD.

A*C Searches all members that begin with the character A and end with the character C. For example, AC, ABC, or AZZZC.

″**a***″ Searches all members with quoted names that start with the letter **a**. For example, ″a″, ″aB″, ″aD″.

****ALL** Searches all members with names ending with ALL. For example, ALL, BALL, or TESTALL. The double asterisk is needed in this case, since *ALL is defined as the special value to search all members of the file.

name Specify the name of the member to be searched.

Top

Operation to perform (OPTION)

Specifies the Programming Development Manager (PDM) option that you want performed on each member for which a match for the string is found. The parameter is comprised of two elements, one for choosing an option and one for prompting. The option can be any PDM option that is valid for this type of file, or any user-defined option in your active option file. The valid options differ for source physical files and data physical files.

This is a required parameter.

Element 1: Option

***NONE**
No action is performed on the member that contains the string. Use this value when printing the list of members or records that contain the string.

Source physical file member list options

***EDIT** Edit one or more members using the SEU (source entry utility) editor.

***CHGT**
Change some of the attributes of one or more members.

***CMPL**
Compile one or more members. The system creates an object based on the member being compiled. The member is compiled interactively or in batch mode, depending on what you have specified on the Change Defaults display or the Change PDM Defaults (CHGPDMDFT) command.

The following member types can be compiled: BAS, BAS36, BAS38, C, CBL, CBLLE, CBL36, CBL38, CICSC, CICSCBL, CICSSQLCBL,CLD, CLLE, CLP, CLP38, CMD, CMD38, CPP, DSPF, DSPF36, DSPF38, FTN, ICFF, LF, LF38, MENU, PAS, PF, PF38, PLI, PLI38, PNLGRP, PRTF, PRINT38, QRY38, RMC, RPG, RPGLE, RPG36, RPG38, RPT, RPT36, RPT38, SPADCT, SQLC, SQLCPP, SQLCBL, SQLCBLLE, SQLFTN, SQLPLI, SQLRPG, SQLRPGLE, and TBL.

When the programming development manager compiles a program using the necessary create commands, the object name to create is always specified as the source member name. You may change the object name parameter to another object name by prompting the option or typing the correct parameter on the command line. The programming development manager will check if the object name already exists, and if it does, the Confirm Compile of Member display appears. This display gives you the option of deleting the existing object.

Note: This screen will not appear if the **Replace object** prompt on the Change Defaults display (or CHGPDMDFT command) is set to Y (or *YES).

If you have changed the object name parameter to a special value, PDM will **not** check to see if the object exists. For example, if you compile an RPG program and change the **Program** prompt to *CTLSPEC, the programming development manager will not check if the object exists.

***CMPR**
Compare one or more members.

***COPY**
Copy one or more members to one or more new members. You can also copy members to another file, another library, or both.

***DLT** Delete one or more members from the file.

***DSP** Display one or more members using SEU (source entry utility).

***DSPD**
Display information about one or more members.

***MOD**
Create a module object for an ILE source type.

***MRG** Merge the target member with another member.

***PRT** Print one or more members using SEU (source entry utility).

***RNM** Rename one or more members.

***RUNP**
Run a source member with a member type of REXX, OCL36, BASP, or BASP38. If you try to run a member with a type that cannot be run, you receive an error message. To run an OCL36 procedure, the file name must be QS36PRC. You can have the member run in batch mode or interactively depending on what you specified in the **Run in batch** prompt on the Change Defaults display or CHGPDMDFT command.

***SAVE**
Save a member on diskette or tape.

***SDA** Use SDA (screen design aid) to work with the chosen members.

- If the member type is DSPF or DSPF38, SDA is called to work with a display.
- If the member type is MNU, MNUDDS or MNUCMD, SDA is called to work with a menu.
- If the member type is DSPF36 or MNU36, then the System 36 SDA main menu is displayed.
- If the former type of member MNU is entered, SDA converts this to MNUDDS.
- Note that menu members for PDM have type MNUDDS for the image member and type MNUCMD for the command source member. The two are linked together to constitute a group, so that specifying one of the types means that you also operate on the linked member at the same time.

***RLU** Use RLU (report layout utility) to work with the chosen members.

Data physical file member list options

***CHGT**
Change some of the attributes of one or more members in a physical file.

***CMPR**
Compare one or more members.

***COPY**
Copy one or more members to one or more new members. You can also copy members to another file, another library, or both.

***DFU** Call DFU (data file utility) to change the chosen members.

***DLT** Delete one or more members from the file.

***DSP** Display one or more members.

***DSPD**
Display information about one or more members.

***RNM** Rename one or more members.

***SAVE**
Save a member on diskette or tape.

User-defined member list option

character-value
Specify the name of an option you have defined in your active option file.

Element 2: Prompt

The prompt portion of this parameter specifies whether or not you want to be prompted each time the command for the option is carried out.

***NOPROMPT**
Do not prompt for the command to be carried out for each member that matches the string.

***PROMPT**
Prompt for the command to be carried out for each member that matches the string.

Top

Columns to search (COL)

Specify the starting and ending column numbers of the part of each file record to be searched. This allows you to search the beginning, ending, or middle of each record.

The starting column number cannot exceed the record length.

Element 1: From column

1 Searching will start in column 1 of each record.

***RCDLEN**
Only the last column of each record will be searched.

number
Specify the first column of the part of each record to be searched.

Element 2: To column

***RCDLEN**
The part of each record to be searched extends from the specified starting column to the last column of the record.

number
Specify the last column of the part of each record to be searched.

Top

Kind of match (CASE)

Specifies whether the search is case sensitive.

***IGNORE**
Search the member for the specified string without case sensitivity.

***MATCH**
Search for an exact match to the specified string.

Top

Print list (PRTMBRLIST)

Specified whether the list of those members for which a match is found is to be printed.

***NO** Do not print the list of members that contain a match to the string.

***YES** Print the list of members that contain a match to the specified string.

Top

Print records (PRTRCDS)

Specifies which records that contains the string are to be printed.

Single values

***NONE**
Do not print any of the records that contain the specified string.

Element 1: Number to find

***ALL** Print all the records that contain the specified string.

1-99999
Specify the number of records to be printed that contain the specified string.

Element 2: Print format

***CHAR**
Print the records in character format.

***HEX** Print the records in hexadecimal, over/under style format. This means that the character value is printed with the hexadecimal below it.

***ALTHEX**
Print the records in hexadecimal side-by-side style format.

Element 3: Mark record

You can mark the string on the printed record. The string itself is used as a marker for character searches for quick recognition. For hexadecimal searches, the string is marked with asterisks (*).

***MARK**
Mark the occurrence of the string in the record.

***NOMARK**
Do not mark the occurence of the string in the record.

Element 4: Record overflow

You can specify whether the record will be folded or truncated if it is greater than the length of the print line.

***FOLD**
Print the entire record over multiple print lines.

***TRUNCATE**
Print only the part of the record that fits on a single print line. When *ALTHEX is used, only columns 1 - 32 are printed, and when *CHAR or *HEX are used, columns 1 - 100 are printed.

Top

Parameters (PARM)

Specifies the parameters that you want to be added to the command carried out as a result of the value specified for element 1 of the **Operation to perform (OPTION)** parameter.

character-value

Specify the parameters to be passed to the command associated with the value specified for the OPTION parameter. The default for this parameter is blank which will pass no parameters.

Top

Examples

Example 1: Find String Occurrences in One Member

```
FNDSTRPDM   STRING('Ms') FILE(*LIBL/CUST)
            MBR('NEW')  CASE(*MATCH)
            OPTION(*NONE)  PRTRCDS(*ALL *CHAR)
```

This command finds all records of member NEW in file CUST that contain the string **Ms**. The file will be located using the job list list. The search will be case sensitive. A spooled file is generated that lists each record that contains the search string.

Example 2: Find String in a Set of Members

```
FNDSTRPDM   STRING('*TEST*')  FILE(MYLIB/MYFILE)
            MBR('*APP') CASE(*IGNORE)
            OPTION(FIX)
```

This command finds occurrences of the string ***TEST*** in members of file MYFILE in library MYLIB with member names that end with the letters **APP**. The search is not case sensitive so records with ***test*** or ***Test*** will be considered as matching the search string. For each member that contains the search string, user-defined option FIX will be run from the active option file.

Top

Error messages

***ESCAPE Messages**

PDM0055

Error while processing the &1 command.

Top

Generate C/C++ Source (GENCSRC)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Generate C/C++ Source (GENCSRC) command maps externally-described file information to the equivalent structures that can be included into ILE C or ILE C++ programs. The structures generated from the specified file object are written to either a source file member or a stream file.

Error messages for GENCSRC

***ESCAPE Messages**

CZM2613
GENCSRC failed to generate include file.

Top

Parameters

Keyword	Description	Choices	Notes
OBJ	From object	*Path name*	Optional, Positional 1
SRCFILE	To source file	*Qualified object name*	Optional
	Qualifier 1: To source file	*Name*	
	Qualifier 2: Library	*Name*, ***CURLIB**	
SRCMBR	To source member	*Name*, ***OBJ**	Optional
SRCSTMF	To source stream file	*Path name*	Optional
RCDFMT	Record format	Single values: ***ALL** Other values (up to 20 repetitions): *Name*	Optional
SLTFLD	Select fields	Values (up to 6 repetitions): *INPUT, *OUTPUT, ***BOTH**, *KEY, *INDICATOR, *LVLCHK, *NULLFLDS	Optional
PKDDECFLD	Packed decimal fields	***DECIMAL**, *CHAR	Optional
STRUCTURE	Structures	*NONPACKED, ***PACKED**	Optional
ONEBYTE	1-byte character fields	*CHAR, ***ARRAY**	Optional
UNIONDFN	Union definition name	*Character value*, ***OBJ**, *NONE	Optional
TYPEDEFPFX	Typedef prefix	*Character value*, ***OBJ**, *NONE	Optional

Top

From object (OBJ)

Specifies the path name of the object to map. The path name must identify a *FILE object in the QSYS file system.

Top

To source file (SRCFILE)

Specifies the name of the physical file that will contain the generated structures. The physical file must exist.

Note: Ensure that the record length of the physical file is large enough to avoid truncation of the data.

file-name
Specify the name of an existing file.

The possible library values are:

***CURLIB**
The current library for the job is used to locate the physical file. If no current library entry exists in the library list, QGPL is used.

library-name
Specify the name of the library where the file is located.

Top

To source member (SRCMBR)

Specifies the name of the file member which will contain the mapped structures. If a member by this name does not exist, it will be added automatically. The records in an existing member by this name will be replaced.

***OBJ** The object name derived from the OBJ parameter will be used as the member name.

member-name
Specify the member name for the generated C/C++ structures.

Top

To source stream file (SRCSTMF)

Specifies the name of the stream file that will contain the generated structures. All directories in the stream file's path must exist. If a file by this name does not exist, it will be created automatically. The data in an existing file by this name will be replaced.

file-name
Specify the complete path of the stream file.

Top

Record format (RCDFMT)

Specifies which record formats of the file are to have C/C++ structures generated.

***ALL** All record formats for the file will have structures generated.

format-name
Specify which record formats will have structures generated. A maximum of 20 record formats can be specified.

Top

Select fields (SLTFLD)

Specifies the usage type of fields that will be mapped.

***BOTH**
Fields declared as INPUT, OUTPUT, or BOTH in the DDS are included in the typedef structure. Option and response indicators are included in both structures unless the keyword INDARA is specified in the external file description for device files.

***INPUT**
Fields declared as either INPUT or BOTH are included in the mapped structures. Response indicators are included in the input structure unless the keyword INDARA is specified in the external file description for device files.

***OUTPUT**
Fields declared as either OUTPUT or BOTH are included in the record structures. Option indicators are included in the output structure unless the keyword INDARA is specified in the external file description for device files.

***KEY** Fields declared as keys in the external file description are included. This option is only valid for database files and DDM files.

***INDICATOR**
A separate 99-byte structure for indicators is created when the indicator option is specified. This option is only valid for device files.

***LVLCHK**
A typedef of an array of struct is generated, named _LVLCHK_T. A pointer to an object of type _LVLCHK_T is also generated and is initialized with the level check information (format name and level identifier).

***NULLFLDS**
If there is at least one null-capable field in the record format of the DDS, a null map typedef is generated containing a character field for every field in the format. With this typedef, you can specify which fields are to be null (set value of each null field to '1', otherwise set to '0'). Also, if the *KEY option is used along with the *NULLFLDS option, and there is at least one null-capable key field in the format, an additional typedef is generated containing a character field for every key field in the format.

For physical and logical files, you can specify *INPUT, *BOTH, *KEY, *LVLCHK, and *NULLFLDS. For device files you can specify *INPUT, *OUTPUT, *BOTH, *INDICATORS, and *LVLCHK.

Top

Packed decimal fields (PKDDECFLD)

Specifies the mapping of packed decimal fields.

***DECIMAL**
Packed decimal fields are declared as _Decimal data types.

***CHAR**
Packed decimal fields are declared as character arrays.

Top

Structures (STRUCTURE)

Specifies whether or not packed structures are generated.

***NONPACKED**
Packed structures are not generated.

***PACKED**
Packed structures are generated.

Top

1-byte character fields (ONEBYTE)

Specifies whether an array or a single character is generated for one-byte fields.

***CHAR**
A single-byte character field is generated for one-byte characters.

***ARRAY**
A one-element array of char is generated for one-byte characters.

Top

Union definition name (UNIONDFN)

Specifies the union names generated.

***OBJ** Use the file name derived from the OBJ parameter.

***NONE**
No union is generated.

union-name
Generate a union definition with the name union-name_t. Maximum length is 50 characters.

Top

Typedef prefix (TYPEDEFPFX)

Specifies the prefix for the generated structures.

***OBJ** Use the file name derived from the OBJ parameter.

***NONE**
Do not use a prefix for the generated structures.

prefix-name
Specify a prefix for the structure name. Maximum length is 50 characters.

Top

Examples

None

Top

Error messages

***ESCAPE Messages**

CZM2613
GENCSRC failed to generate include file.

Top

Merge Form Description (MRGFORMD)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

The Merge Form Description (MRGFORMD) command merges a spooled output file with a database file containing a form description, which is designed with the Start Advanced Printer Function (STRAPF) command. The output can be spooled for later printing, or can be directed immediately to a printer.

The Merge Form Description (MRGFORMD) command is part of the IBM Application Development Tools Program, 5722-WDS. For more information on the advanced printer function (APF) tool, refer to the Advanced Printer Function Guide, SC09-1361.

Note: Do not precede an entry with an asterisk unless that entry is a ″special value″ that is shown (on the display itself or in the help information) with an asterisk.

Error messages for MRGFORMD

***ESCAPE Messages**

APF5101
Not able to open printer file &1.

APF5102
Error occurred accessing spooled file.

APF5104
Form description not available.

APF5105
Form description not valid.

APF5106
Not authorized to perform requested function.

APF5107
Error occurred on device &3.

APF5121
File &1 in &2 not correct for APF Utility.

APF9901
Error in APF utility.

APF9910
Not able to complete display file I/O operation.

APF9911
Not able to complete data base file I/O operation.

APF9912
Not able to open advanced printer function file.

Top

Parameters

Keyword	Description	Choices	Notes
FORMD	Form description	*Name*	Required, Positional 1
FILE	File	*Qualified object name*	Required, Positional 2
	Qualifier 1: File	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
SPLF	Spool file	*Name*, ***NONE**	Optional
JOB	Job name	Single values: * Other values: *Qualified job name*	Optional
	Qualifier 1: Job name	*Name*	
	Qualifier 2: User	*Name*	
	Qualifier 3: Number	000000-999999	
SPLNBR	Spooled file number	1-999999, *ONLY, ***LAST**	Optional
COPIES	Copies	1-255, ***FILE**	Optional
DEV	Device	*Name*, ***FILE**, *JOB, *SYSVAL	Optional
SPOOL	Spool the data	*YES, *NO, ***FILE**	Optional
OUTQ	Output queue	Single values: ***FILE**, *JOB, *DEV Other values: *Qualified object name*	Optional
	Qualifier 1: Output queue	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
FORMTYPE	Form type	*Character value*, ***FILE**, *STD	Optional
OUTSPLF	Output spool file	*Name*, ***FRMD**, ' '	Optional
SCHEDULE	Spooled output schedule	***FILE**, *IMMED, *JOBEND, *FILEEND	Optional
JOBD	Job description	Single values: *NONE Other values: *Qualified object name*	Optional
	Qualifier 1: Job description	*Name*, **QBATCH**	
	Qualifier 2: Library	*Name*, *LIBL, *CURLIB	

Top

Form description (FORMD)

Specifies the name of the form description to be used to print a form or to be used in the merge operation.

This is a required parameter.

Top

File (FILE)

Specifies the name and library of the file that contains the form description.

This is a required parameter.

The possible library values are:

***LIBL** The library list is used to find the file.

***CURLIB**
The current library for the job is used to find the file. If no current library entry exists in the library list, QGPL is used.

library-name
Specify the library in which the file will be located.

Top

Spool file (SPLF)

Specifies name of the spooled output file that contains the data to be merged with the form description.

The possible values are:

***NONE**
No spooled output file is to be specified.

spool-file-name
Specify the name of the spooled output file.

Top

Job name (JOB)

Specifies the name of the job that contains the spooled output file to be merged with the form description.

The possible values are:

*****
Specifies that the current job contains the spooled file.

job name
Specify the name of the job that created the spooled file to be merged. If no job name is given, all jobs currently in the system are searched for the simple job name.

user
The user name identifies the user profile under which the job is run.

number
The system assigned job number.

Top

Spooled file number (SPLNBR)

Specifies the number of the spooled output file to be merged with the form description.

The possible values are:

***LAST**
Specifies that the last spooled file with the specified name is to be merged with the forms description.

***ONLY**
Specifies that only one spooled file has the name specified on the Spool file prompt (SPLF parameter).

spooled-file-number
Enter the number of the spooled output file to be merged with the form description.

Top

Copies (COPIES)

Specifies the number of copies of the merged spooled output file that are to be printed.

The possible values are:

***FILE** The number of copies to print is to be taken from the COPIES value specified for the printer device file (QPAPFPRT).

number-of-copies
Enter the number of copies of the merged spooled output file to be printed.

Top

Device (DEV)

Specifies the name of the printer device to be used to print the form or merged output.

The possible values are:

***FILE** The device used as the printer is to be the same as that specified in the DEV parameter in the printer device file (QPAPFPRT).

***SYSVAL**
The device used as the printer is specified through system value QPRTDEV.

***JOB** The device used as the printer is specified through the job's device file.

device-name
Specify the name of the printer device to be used to print the form or merged output.

Top

Spool the data (SPOOL)

Specifies whether or not the data is to be spooled.

The possible values are:

***FILE** The spooled file attribute is to be the same as that used in the printer device file (QPAPFPRT).

***YES** The data is to be spooled.

***NO** The data is not to be spooled.

Top

Output queue (OUTQ)

Specifies the output queue on which the merged spooled output file is to be placed.

The possible values are:

***FILE** The output queue name is to be the same as that specified in the printer device file (QPAPFPRT).

***DEV** Use the default output queue value associated with the printer specified on the Device prompt (DEV parameter).

***JOB** Use the output queue specified in the job description associated with the job for the spooled output.

output-queue-name
Enter the name and library of the output queue that is to contain the spooled database output file. The possible library name values are:

***LIBL** The library list is used to find the output queue.

***CURLIB**
The current library for the job is used to find the output queue. If no current library entry exists in the library list, QGPL is used.

library-name
Specify the library in which the output queue will be located.

Top

Form type (FORMTYPE)

Specifies the type of form on which to print the merged spooled output file.

The possible values are:

***FILE** The merged spooled output file is to be printed on the form type specified in the printer device file (QPAPFPRT).

***STD** The merged spooled output file is to be printed on the standard form type used at your installation.

form-type
Enter the name of the form type on which the spooled output file is to be printed.

Top

Output spool file (OUTSPLF)

Specifies the name of the the merged spooled output file on the output queue.

The possible values are:

***FRMD**
The forms description name is to be used as the name of the the merged spooled output file on the output queue.

output-spool-file-name
Enter the name (10 characters maximum) of the merged spooled output file on the output queue.

Top

Spooled output schedule (SCHEDULE)

Specifies when the merged spooled output file is to be made available to a spool writer.

The possible values are:

***FILE** The merged spooled output file is to be made available to a spool writer as specified on the Spooled output schedule prompt (SCHEDULE parameter). in the printer device file (QPAPFPRT).

***IMMED**
The merged spooled output file is to be made available to a spool writer immediately.

***JOBEND**
The merged spooled output file is to be made available to a spool writer when the current job finishes.

***FILEEND**
The merged spooled output file is to be made available to a spool writer when the end of the current file is reached.

Top

Job description (JOBD)

Specifies the name of the job description to be used to submit the job.

The possible values are:

***NONE**
The printing is to be done under the current job description.

job-description-name
Specify the name and library of the job description to be used to submit the job. The possible library name values are:

***LIBL** The library list is used to locate the job description.

***CURLIB**
The current library for the job is used to find the job description. If no current library entry exists in the library list, QGPL is used.

library-name
Specify the library in which the job description will be located.

Top

Examples

None

Top

Error messages

***ESCAPE Messages**

APF5101
Not able to open printer file &1.

APF5102
Error occurred accessing spooled file.

APF5104
Form description not available.

APF5105
Form description not valid.

APF5106
Not authorized to perform requested function.

APF5107
Error occurred on device &3.

APF5121
File &1 in &2 not correct for APF Utility.

APF9901
Error in APF utility.

APF9910
Not able to complete display file I/O operation.

APF9911
Not able to complete data base file I/O operation.

APF9912
Not able to open advanced printer function file.

Top

Merge Source (MRGSRC)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

This command allows you to merge updates from maintenance file members into target file members.

You provide the MRGSRC command with the names of the following source files:

Root The original version of the source file, on which the updates are based

Maintenance
The source file containing the updates to be merged into the target file

Target The source file into which the updates from the maintenance file are merged

The MRGSRC command compares each target member and maintenance member with its corresponding root member. The results of these comparisons are used to determine the updates that have occurred.

Error messages for MRGSRC

None

Top

Parameters

Keyword	Description	Choices	Notes
TGTFILE	Target file	*Qualified object name*	Required, Positional 1
	Qualifier 1: Target file	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
TGTMBR	Target member	*Name*, *ALL	Required, Positional 2
MAINTFILE	Maintenance file	*Qualified object name*	Optional, Positional 3
	Qualifier 1: Maintenance file	*Name*, ***TARGET**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
MAINTMBR	Maintenance member	*Name*, ***TARGET**	Optional, Positional 4
ROOTFILE	Root file	*Qualified object name*	Optional, Positional 5
	Qualifier 1: Root file	*Name*, ***MAINT**	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	
ROOTMBR	Root member	*Name*, ***MAINT**	Optional, Positional 6
SELECT	Select updates	***YES**, *NO	Optional
RPTONLY	Report only	***NO**, *YES	Optional

Top

Target file (TGTFILE)

Specifies the source physical file into which the maintenance updates are merged.

***LIBL** Use the library list.

***CURLIB**
Use the current library.

library-name
Use the specified library.

file-name
Use the specified file.

Top

Target member (TGTMBR)

Specifies the member into which the maintenance updates are merged.

***ALL** Select all members.

member-name
Select the specified member.

Top

Maintenance file (MAINTFILE)

Specifies the source physical file containing the updates to merge.

***LIBL** Use the library list.

***CURLIB**
Use the current library.

library-name
Use the specified library.

***TARGET**
Use the file specified for the target file.

file-name
Use the specified file.

Top

Maintenance member (MAINTMBR)

Specifies the member containing the updates to merge.

***TARGET**
Select the same member as is specified for the TGTMBR keyword.

member-name
Select the specified member.

Top

Root file (ROOTFILE)

Specifies the source file on which the merge process is based.

***LIBL** Use the library list.

***CURLIB**
Use the current library.

library-name
Use the specified library.

***MAINT**
Use the file specified for the maintenance file.

file-name
Use the specified file.

Top

Root member (ROOTMBR)

Specifies the source file member on which the merge process is based.

***MAINT**
Select the same member as is specified for the MAINTMBR parameter. This parameter is required if TGTMBR(*ALL) is specified.

member-name
Select the specified member.

Top

Select updates (SELECT)

Specifies whether to show the Split Merge display for selecting the maintenance updates.

***YES** Show the Split Merge display so that you can select the maintenance updates to merge into the target member. No report is generated.

***NO** Do not show the Split Merge display, and print a Merge Summary report.

Top

Report only (RPTONLY)

Indicates whether to merge the maintenance updates into the target member or just print a Merge Summary report to show the scope of the updates.

***NO** Perform the merge and print a Merge Summary report.

***YES** Print the Merge Summary report without performing the merge.

Top

Examples

None

Top

Error messages

None

Top

Submit CODE Batch Job (SBMCODEJOB)

Parameters
Examples
Error messages

Where allowed to run:

- Batch job (*BATCH)
- Interactive job (*INTERACT)
- Batch program (*BPGM)
- Interactive program (*IPGM)
- Batch REXX procedure (*BREXX)
- Interactive REXX procedure (*IREXX)
- Using QCMDEXEC, QCAEXEC, or QCAPCMD API (*EXEC)

Threadsafe: No

The CL command SBMCODEJOB submits a command to be run in batch, and optionally informs the work station when the job is complete and returns error feedback information from certain compile commands to the work station.

This command is intended to be used by the ″CoOperative Development Environment/400″ and ″VRPG Client/2″ features of the ″IBM Application Development ToolSet Client Server/400″ (ADTS CS/400) program product.

Top

Parameters

Keyword	Description	Choices	Notes
CMD	Command to run	*Command string*	Required, Positional 1
JOBD	Job description	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Job description	*Name*, ***USRPRF**	
	Qualifier 2: Library	*Name*, *LIBL, *CURLIB	
JOBQ	Job queue	*Qualified object name*	Optional, Positional 3
	Qualifier 1: Job queue	*Name*, ***JOBD**	
	Qualifier 2: Library	*Name*, *LIBL, *CURLIB	
NOTIFY	Notification message	*YES, ***NO**	Optional, Positional 4
SERVER	Host server name	*Character value*, **OS400**	Optional, Positional 5
ADDPARMS	Additional SBMJOB parameters	*Character value*, ***NONE**	Optional, Positional 6

Top

Command to run (CMD)

Specifies the CL command that is to be run in batch.

Top

Job description (JOBD)

Specifies the name that is associated with the job when it is processed by the system.

The possible values are:

***USRPRF**
The job description in the user profile under which the submitted job runs is used as the job description of the submitted job.

job-description-name
Specify the name (library-name/job-description-name) of the job description used for the job.

The possible values are:

***LIBL** All libraries in the job's library list are searched until the first match is found.

***CURLIB**
The current library for the job is used to locate the job description name. If no library is specified as the current library for the job, QGPL is used.

library-name
Specify the name of the library where the job description name is located.

Top

Job queue (JOBQ)

Specifies the name of the job queue in which this job is placed.

The possible values are:

***JOBD**
The submitted job is placed on the job queue named in the specified job description.

job-queue-name
Specify the name (library-name/job-queue-name) of the job queue on which the submitted job is placed.

The possible values are:

***LIBL** All libraries in the job's library list are searched until the first match is found.

***CURLIB**
The current library for the job is used to locate the job queue name. If no library is specified as the current library for the job, QGPL is used.

library-name
Specify the name of the library where the job queue name is located.

Top

Notification message (NOTIFY)

Specify Yes to have the batch job inform the work station of the completion of the submitted command.

To use this option, an ICF file named EVFCICFF must be found in the library list of the batch job. This identifies the location of the work station. See the CODE/400 Installation Guide for more information on setting up the ICF file.

Top

Host server name (SERVER)

Specifies the name of a CODE/400 or VRPG/2 server that has already been started by the Start CODE (STRCODE) command.

This parameter is only used when the command being submitted generates error feedback information for use by the CODE/400 workstation editor. Only commands with the parameters OPTION(*SRCDBG), OPTION(*LSTDBG) and OPTION(*EVENTF) generate such error feedback.

Top

Additional SBMJOB parameters (ADDPARMS)

Specifies parameters to be used in the Submit Job (SBMJOB) command.

The SBMCODEJOB command invokes SBMJOB for you. It uses the system defaults for all SBMJOB parameters except CMD, JOBD and JOBQ. This parameter allows you to specify additional parameters for the SBMJOB command in order to override the system default. For example, to use your job description's library list in the batch job, specify 'INLLIBL(*JOBD)' in this parameter.

You can specify multiple SBMJOB parameters, but each must be separated by one or more blanks.

If 'INLLIBL(*JOBD)' is specified, that job description's library list must include a library with an EVFCICFF ICF file in it.

Top

Examples

None

Top

Error messages

Unknown

Top

Advanced Printer Function (STRAPF)

Where allowed to run: Interactive environments (*INTERACT *IPGM *IREXX *EXEC)
Threadsafe: No

The Start Advanced Printer Function (STRAPF) command calls a menu that offers options for designing customized forms. From this menu, you can design special symbols, the layout of a form (called a form description), print copies of the form description, or merge a spooled file with a form description and print the result.

The advanced printer function (APF) tool uses the special printing capabilities of the IBM 5224 and 5225 Work Station Printers.

There are no parameters for this command.

Error messages for STRAPF

None

Top

Parameters

None

Top

Examples

None

Top

Error messages

None

Top

Start COBOL Debug (STRCBLDBG)

Where allowed to run: All environments (*ALL)
Threadsafe: No

Parameters
Examples
Error messages

This command activates the debugging code that is created when the WITH DEBUGGING MODE clause is used in a COBOL program. This command must be entered for each COBOL program to be debugged in the next COBOL run unit.

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Required, Positional 1
	Qualifier 1: Program	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB	

Top

Program (PGM)

Specifies the name of the compiled COBOL program and the library where it is located. This is a required parameter. The possible values are:

program-name
Specifies the name by which the compiled COBOL program is known.

The possible library values are:

***LIBL** The system searches the library list to find the library where the program is located.

***CURLIB**
The current library is used. If you have not assigned a library as the current library, QGPL is used.

library-name
Enter the name of the library where the created program is stored.

Top

Examples

Example 1: Activate the debugging code for a COBOL program

```
STRCBLDBG PGM(MYLIB/XMPLE1)
```

This command activates the debugging code that was created for the COBOL program XMPLE1 in library MYLIB.

Top

Error messages

***ESCAPE Messages**

LBE7018
Program &1 not found.

LBE7019
Library &1 not found.

Top

Start CGU (STRCGU)

Parameters
Examples
Error messages

Where allowed to run:

- Batch job (*BATCH)
- Interactive job (*INTERACT)
- Interactive program (*IPGM)
- Batch REXX procedure (*BREXX)
- Interactive REXX procedure (*IREXX)
- Using QCMDEXEC, QCAEXEC, or QCAPCMD API (*EXEC)

Threadsafe: No

The STRCGU (Start CGU) command starts the Character Generator Utility (CGU).

Error messages for STRCGU

None

Top

Parameters

Keyword	Description	Choices	Notes
OPTION	CGU option	***SELECT**, 1, 2, 3, 4, 5, 6	Optional, Positional 1

Top

CGU option (OPTION)

Specifies an option from the Character Generator Utility (CGU) menu which you access directly.

The possible values are:

***SELECT**
Displays the Character Generator Utility (CGU) menu.

main-menu-option-number
Type a number between 1 and 6 that corresponds to an option on the CGU menu. If you select this parameter value, the CGU menu does not appear.

Top

Examples

None

Top

Error messages

None

Top

Start CODE (STRCODE)

Parameters
Examples
Error messages

Where allowed to run:

- Batch job (*BATCH)
- Interactive job (*INTERACT)
- Batch program (*BPGM)
- Interactive program (*IPGM)
- Batch REXX procedure (*BREXX)
- Interactive REXX procedure (*IREXX)
- Using QCMDEXEC, QCAEXEC, or QCAPCMD API (*EXEC)

Threadsafe: No

The CL command STRCODE starts a job that runs the host server for the Cooperative Development Environment/400 product. This command should be called in the OS/400 native environment.

Top

Parameters

Keyword	Description	Choices	Notes
SERVER	Host server name	*Character value*, **OS400**	Optional, Positional 1
RMTLOCNAME	Remote location name	*Character value*, *RESOLVE, ***PRV**	Optional, Positional 2
CMNTYPE	Communications type	***PRV**, *APPC, *TCPIP	Optional, Positional 3
PARMS	Additional parameters	*Character value*, ***NONE**	Optional, Positional 7
PORT	TCP/IP port number	1-65534, ***PRV**, *DFT	Optional, Positional 4
USERID	User identifier	*Character value*, ***NONE**	Optional, Positional 5
PASSWORD	Password	*Character value*	Optional, Positional 6

Top

Host server name (SERVER)

Specifies the name that is associated with the job as it is processed by the system.

The possible values are:

OS400 This is the default value.

server-name
Enter the name that is used to identify the job as it is processed by the system.

Note: The following server names are reserved, and should not be used:

- LOCAL

- REMOTE

Top

Remote location name (RMTLOCNAME)

Specifies the name of the remote workstation with which the host server communicates. For TCP/IP connections, this is the IP name or IP address of your PC. For APPC connections, this name is the remote workstation's Logical Unit (LU) or PC local name specifies in the SNA base profile of your communications program.

The possible values are:

***PRV** The value used in the last invocation of this command is the default.

***RESOLVE**
Try to have CODE/400 communications resolve the remote location name. This value is recommended for TCP/IP DHCP users. This value will not work in a batch job.

PC local name
Enter the name that identifies the remote workstation that the host server will communicate with.

Top

Communications type (CMNTYPE)

Specifies the name of the type of protocol with which the host server communicates.

The possible values are:

***PRV** The value used in the last invocation of this command is the default.

***APPC**
The host sever uses APPC protocol to communicate with the workstation.

***TCPIP**
The host server uses TCP/IP protocol to communicate with the workstation.

Top

Top

TCP/IP port number (PORT)

Specifies the name of the port that is associated with the TCP/IP protocol processed by the host server.

The possible values are:

***PRV** The value used in the last invocation of this command is the default.

***DFT** The value used to communicate with the system is the default CODE/400 TCP/IP port. If you wish to change the port, enter the new TCP/IP port value.

Top

User identifier (USERID)

Specifies the user ID to use to establish communications when the workstation has enabled conversation security through the Communication Properties window.

Both this field and the **Password** field are required if conversation security has been set up in The Communication Properties window. The possible values are:

***NONE**
The default value is *NONE.

user ID
Enter the user ID that was specified STRCODE user ID field in the Communication Properties window on the workstation.

Top

Password (PASSWORD)

Specifies the password for the user ID specified in the **User identifier** field. This password should be the same as that specified in the STRCODE password field in the Communication Properties window.

Both this field and the **User identifier** field are required if conversation security Profile has been set up in the Communication Properties window.

Top

Examples

None

Top

Error messages

Unknown

Top

Start CODE Command (STRCODECMD)

Where allowed to run:

- Batch job (*BATCH)
- Interactive job (*INTERACT)
- Batch program (*BPGM)
- Interactive program (*IPGM)
- Batch REXX procedure (*BREXX)
- Interactive REXX procedure (*IREXX)
- Using QCMDEXEC, QCAEXEC, or QCAPCMD API (*EXEC)

Threadsafe: No

The CL command STRCODECMD submits a command to be run on the workstation.

This command requires that you install the Cooperative Development Environmen/400 (CODE/400) product on your workstation.

Top

Parameters

Keyword	Description	Choices	Notes
CMD	Workstation command	*Character value*	Required, Positional 1

Top

Workstation command (CMD)

Specifies the workstation command that is to be run. To find your workstation, the values in the STRCODE *USRSPC object are used. This value will typically be the last one specified on the Start CODE (STRCODE) command. If this object is not found in your library list, CODE communications will attempt to resolve the remote location automatically. CODE communications is unable to automatically resolve remote locations when this command is run in a batch job. In addition, an Intersystem Communications Function (ICF) file is needed for APPC connections. This command searches the library list for the ICF file EVFCICFF and uses it to send the information to your workstation.

Top

Examples

None

Top

Error messages

Unknown

Top

Start ISDB (STRISDB)

Where allowed to run: Interactive environments (*INTERACT *IPGM *IMOD *IREXX *EXEC)
Threadsafe: No

Parameters
Examples
Error messages

Start Interactive Source Debugger

The Interactive Source Debugger (ISDB) tool provides an interactive debugging environment for CL, COBOL, or RPG programs.

Note: The library QTEMP is required for ISDB to run. ISDB uses the library QTEMP to access files such as the log file.

To start it:

1. Specify the necessary STRISDB parameters for your program.
2. Press Enter. ISDB is started.

The source of the program is displayed in the ISDB Source display, and you can debug the program using the features provided there.

Error messages for STRISDB

None

Top

Parameters

Keyword	Description	Choices	Notes
PGM	Program	*Qualified object name*	Required, Positional 1
	Qualifier 1: Program	*Name*	
	Qualifier 2: Library	*Name*, *CURLIB, ***LIBL**	
UPDPROD	Update production files	*YES, ***NO**	Optional, Positional 2
INVPGM	Invoke program	***YES**, *NO, *CMD	Optional, Positional 3
PARM	Parameters for call	Values (up to 40 repetitions): *Character value*	Optional, Positional 4
CMD	Invocation Command	*Command string*	Optional, Positional 5
SRCMBR	Source member	*Name*, ***PGM**	Optional, Positional 6
SRCF	Source file	*Qualified object name*	Optional, Positional 7
	Qualifier 1: Source file	*Name*	
	Qualifier 2: Library	*Name*, *CURLIB, *LIBL	
SRVJOB	Job to service	*, *SELECT	Optional, Positional 8

Top

Program

Use this parameter to specify the name and library of the program to be debugged.

The possible values for the library are:

***LIBL** The library list is used to locate the program. This is the default if no library name is specified.

***CURLIB**
The current library in the library list is used to locate the program. (If you have not specified a current library, QGPL is assumed.)

library-name
Specify the name of the library containing the program to be debugged.

Note: To debug COBOL programs with ISDB, ensure that they are compiled with the *SRCDBG option.

Top

Update production files (UPDPROD)

Use this parameter to specify whether files in production libraries can be changed while they are in debug mode.

The possible values are:

***NO** Files cannot be updated while they are in debug mode.

***YES** Files can be updated while they are in debug mode.

Note that you can change the update production files value from the Source display. To do this:

1. Select the Debug menu-bar choice from the Source display.
2. In the Debug pull-down window, select option 1 (Change debug). The Set Debugging Options window appears.
3. Select either option 1 (Update production files) or option 2 (Do not update production files), and press Enter.

Top

Invoke Program

Use this parameter to specify whether you want ISDB to call your program, do other preparation before you invoke your program, or use a command to invoke your program.

The possible values are:

***YES** ISDB calls your program with the parameters you specify in the PARM field.

Note: You cannot use this parameter for serviced jobs.

***NO** A command is not issued to start the program. A command entry screen (QCMD) is provided so that you can do any preparation required before you issue the command to start the program.

If you start your program from the command entry screen, when your program completes running, control will return to this screen. To End ISDB or Restart, you must first exit this command entry screen (F3 or F12). The Program Termination display will then appear where you can choose to end ISDB or restart.

Note: For servicing other jobs, this is the only option you can use, but the QCMD panel is not invoked.

***CMD** ISDB executes the command specified in the CMD parameter. (This command must invoke your program.)

Top

Parameters

Use this parameter to specify any parameters required to call your program.

This parameter is not intended for the INVPGM(*NO) and INVPGM(*CMD) invocation commands.

Note: Numeric literals (for example, 500) and strings which contain null (X'00') characters should not be used in this field. Instead, use the parameter INVPGM(*CMD) and specify the appropriate CALL command in the CMD parameter. For instance, instead of entering the command:

```
STRISDB PGM(MYPGM) INVPGM(*YES) PARM(123 X'00')
```

Use the following command:

```
STRISDB PGM(MYPGM) INVPGM(*CMD) CMD(CALL PGM(MYPGM)
PARM(123 X'00'))
```

Top

Invocation Command

Use this parameter to specify the command you want to use to invoke your program.

This parameter is not intended for the INVPGM(*NO) and INVPGM(*YES) invocation commands.

Top

Source member

This parameter is optional, since you need to specify the source member only if it is different from the one specified in the object description of the program. The source of the member you specify will be displayed in the Source display when you invoke the program.

You will need to use this parameter if:

- The library, file, or member names of the source have changed since the program was last compiled. (Be sure the member contains the correct program source, or you will get unpredictable results.)
- The program is an RPG Auto Report program. The CRTRPTPGM command has a parameter that lets you place the expanded source into a source physical file.
- The source is located on another server machine and its program was not created using a DDM file.

The possible values are:

***PGM** The source information is retrieved from the object description using the DSPOBJD command.

member_name
Specify the name of the source member you want displayed, and type its file and library names in the parameters provided.

Top

Source file and library

Use these parameters to specify the file and library names of the source member you specified in the Source member prompt. You must specify both names if either one of them is different from the names specified in the object description of the program.

The possible value for the source file name is:

file_name
Specify the name of the file containing the source member you want displayed.

The possible values for the source library name are:

***LIBL** The library list is used to locate the source file.

***CURLIB**
The current library is used to locate the source file. (If you have not specified a current library, QGPL is assumed.)

library_name
Specify the name of the library containing the source file you want displayed.

Top

Job to service (SRVJOB)

Use this parameter to debug a program running in a job different from the one to which you are signed on. It is useful for debugging batch jobs, or other interactive jobs.

Possible values are:

***** Debug in the current job.

***SELECT**
Display the Select Job to Service display that lets you select a job from a list of active jobs. If you select one of these jobs, the STRSRVJOB command is issued and that job is placed in debug mode.

Note: SRVJOB(*SELECT) is not valid with INVPGM(*YES) or INVPGM(*CMD).

Top

Examples

None

Top

Error messages

None

Top

Start PDM (STRPDM)

Where allowed to run: Interactive environments (*INTERACT *IPGM *IREXX *EXEC)
Threadsafe: No

The Start PDM (STRPDM) command calls the Programming Development Manager (PDM) utility. A menu is displayed where you choose options to work with libraries, objects, members, and user-defined options.

This command has no parameters.

Top

Parameters

None

Top

Examples

```
STRPDM
```

This command starts the Programming Development Manager (PDM) utility and displays the initial PDM menu.

Top

Error messages

***ESCAPE Messages**

PDM0055
Error while processing the &1 command.

Top

Start Report Layout Utility (STRRLU)

Where allowed to run: Interactive environments (*INTERACT *IPGM *IREXX *EXEC)
Threadsafe: No

Parameters
Examples
Error messages

The Control Language (CL) command STRRLU is the command used to start the Report Layout Utility (RLU).

Use this display to specify parameter values to start the Report Layout Utility.

Error messages for STRRLU

None

Top

Parameters

Keyword	Description	Choices	Notes
SRCFILE	Source file	Single values: ***PRV** Other values: *Qualified object name*	Optional, Positional 1
	Qualifier 1: Source file	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB, *PRV	
SRCMBR	Source member	*Name*, ***PRV**	Optional, Positional 2
OPTION	Option	**2**, 6	Optional, Positional 3
PAGWIDTH	Page width	1-378, ***SAME**	Optional, Positional 4
TEXT	Text 'description'	*Character value*, ***BLANK**	Optional

Top

Source file (SRCFILE)

Specifies the qualified name of the source physical file that contains the member you changed or created in a previous session, or the source physical file in which you want to store a new member. The possible values are:

***PRV** Specifies the qualified source physical file name you used last.

Name Specify the name of an existing source file.

Warning: Temporary Level 3 Header

Library (LIB)

Specifies the qualified name of the library that contains the member changed or created in a previous session, or the library in which you want to store the source file. The possible library values are:

***LIBL** The library list appears from which you can select the library that contains the source file, or the library in which you want to store the source file.

***CURLIB**
The current library for the job is used to store the source file. If no current library entry exists in the library list, your source file goes into QGPL.

Name Specify the library in which the source file is located.

Top

Source member (SRCMBR)

Specifies the names of the member changed or created. The default value for this parameter changes if you use the SRCFILE parameter. The possible values are:

***PRV** Selects the name of the previous source member used.

Name Specify the name of the member you changed or created in a previous session, or the name you want if you are creating a new member.

Top

Option (OPTION)

Specify an option to work with a report. The possible values are:

2 Changes a report

6 Prints a prototype report.

Top

Page width (PAGWIDTH)

Specifies the page width in positions per line. The default is *SAME. The first time you use RLU, the page width is set to 132 if you do not specify another value. If the member has been previously edited with RLU, the page width is taken from the previous edit session for that member.

***SAME**
Specifies the same report width used when you created or last changed the report.

1 to 378
Type a value between 1 and 378 to determine the width of the page.

Top

Text 'description' (TEXT)

Specify a description to store in the text prompt for the member. The possible values are:

***BLANK**
For a new member, RLU specifies blanks in the Text prompt of the member. For an existing member, this default value does not change the Text prompt of the member.

'description'
Specify text no longer than 50 characters.

Top

Examples

None

Top

Error messages

None

Top

Start RSE Server (STRRSESVR)

Where allowed to run: Interactive environments (*INTERACT *IPGM *IREXX *EXEC)
Threadsafe: No

Parameters
Examples
Error messages

The Start RSE Server (STRRSESVR) command starts the Remote System Explorer (RSE) communications server if it is not already started and associates the current interactive job with the specified RSE connection.

The purpose of STRRSESVR command is to associate an emulator with an RSE connection for running and debugging interactive applications. The job is locked for the exclusive use of the associated RSE connection. The job can be released either by disconnecting the associated RSE connection or using the Release Interactive Job action for the RSE connection.

Top

Parameters

Keyword	Description	Choices	Notes
NAME	Connection name	*Character value*, ***PRV**	Optional, Positional 1
WRKLIB	Working library	*Name*, ***PRV**, *DFT	Optional, Positional 2
RMTLOCNAME	Remote location name	*Character value*, *RESOLVE, ***PRV**	Optional, Positional 3
PORT	TCP/IP port	1-65534, ***PRV**, *DFT	Optional

Top

Connection name (NAME)

Specifies the RSE connection which should be associated with this interactive job. The connection name can be specified by itself or optionally qualified with the RSE profile name.

***PRV** The connnection name specified in the previous invocation of the STRRSESVR command is used. The connection name must be specified if no previous invocation exists.

character-value
Specify the connection name that should be associated with this job.

Top

Working library (WRKLIB)

Specifies the library where the STRRSESVR command will create a data queue to communicate with the RSE server.

***PRV** The library name specified in the previous invocation of the STRRSESVR command is used. Until a library is specified, QGPL is used.

***DFT** Library QGPL is used.

name Specify the name of the library to be used.

Top

Remote location name (RMTLOCNAME)

Specifies the TCP/IP hostname or IP address of the client machine where the RSE communications daemon is running.

***RESOLVE**
Attempt to have the STRRSESVR command resolve the remote location name. This value is recommended for TCP/IP DHCP users.

***PRV** The hostname specified in the previous invocation of the STRRSESVR command is used. Until a hostname is specified, *RESOLVE is used.

character-value
Specify the TCP/IP host name or IP address for the client machine.

Top

TCP/IP port (PORT)

Specifies the port number where the RSE communications daemon is listening.

***PRV** The port number specified in the previous invocation of the STRRSESVR command is used. Until a port number is specified, port 4300 is used.

***DFT** The default port (4300) is used.

1-65534
Specify the TCP/IP port number to be used.

Top

Examples

Example 1: Start RSE Server Specifying Only a Connection Name

```
STRRSESVR  NAME(DEVELOPMENT)
```

This command associates the current interactive job with the RSE connection whose name is DEVELOPMENT.

Example 2: Start RSE Server Specifying Connection Name and Additional Parameters

```
STRRSESVR   NAME(TEST.DEVELOPMENT)  RMTLOCNAME('192.168.1.45')
            WRKLIB(DEVLIB)
```

This command associates the current interactive job with the RSE connection named DEVELOPMENT in the TEST profile on the client machine with the specified IP address. The data queue used to communicate between this job and the RSE is created in the library DEVLIB.

Top

Error messages

***ESCAPE Messages**

RSE2001
Error connecting to Remote Systems Explorer

RSE2002
Connection name not defined.

RSE2003
Connection defined for different hostname.

RSE2004
Error creating data queue.

RSE2005
Working library is not valid.

RSE2006
RSE server failed to start or connect.

RSE2007
Error accessing data queue.

RSE2008
Data queue could not be deleted.

RSE2009
Error resolving remote location IP address.

RSE2010
Connection already in use.

RSE2011
Socket Error

RSE2012
Error retrieving job information.

RSE2013
No previous connection name exists.

Top

Start SDA (STRSDA)

Where allowed to run: Interactive environments (*INTERACT *IPGM *IREXX *EXEC)
Threadsafe: No

Parameters
Examples
Error messages

The CL command STRSDA is the primary command for the IBM Screen Design Aid (SDA) utility. This command can be invoked in all three environments of the system.

Error messages for STRSDA

***ESCAPE Messages**

SDA0001
SDA does not support a &1 workstation.

SDA0002
SDA cannot access its display file.

SDA0003
SDA ended because of a serious error.

SDA0004
Error while processing the STRSDA command.

SDA0005
SDA cannot access its panel group.

SDA0601
SDA is already active.

Top

Parameters

Keyword	Description	Choices	Notes
OPTION	SDA option	***SELECT**, 1, 2, 3	Optional, Positional 1
SRCFILE	Source file	*Qualified object name*	Optional, Positional 2
	Qualifier 1: Source file	*Name*, ***PRV**	
	Qualifier 2: Library	*Name*, ***PRV**, *LIBL, *CURLIB	
SRCMBR	Source member	*Name*, ***PRV**, *SELECT	Optional, Positional 3
OBJLIB	Object library	*Name*, ***PRV**, *CURLIB	Optional
JOBD	Job description	*Qualified object name*	Optional
	Qualifier 1: Job description	*Name*, ***PRV**, *USRPRF	
	Qualifier 2: Library	*Name*, ***PRV**, *LIBL, *CURLIB	
TSTFILE	Test file	*Qualified object name*	Optional
	Qualifier 1: Test file	*Name*, ***PRV**	
	Qualifier 2: Library	*Name*, ***PRV**, *LIBL, *CURLIB	
MODE	Mode	***STD**, *S38, *S36	Optional

Top

SDA option (OPTION)

Specifies which option to use as a value for the SDA main menu. This parameter is ignored if MODE (*S36) is specified.

The possible values are:

***SELECT**
Displays the SDA main menu.

main-menu-option-number
Type a number between 1 and 3 that corresponds to an option on the SDA main menu. If you select this parameter value, the SDA main menu will not appear.

Top

Source file (SRCFILE)

Specifies the name of the source file that contains the source member to be updated or to which a new source member will be added. Only the library qualifier will be used if MODE (*S36) is specified.

The possible values are:

***PRV** Specifies that SDA is to use the name of the source file and library used in your last SDA session (only when MODE is (*STD)).

qualified-source-name
Type the qualified name of an existing source file to be used by SDA. If *CURLIB is specified as the library qualifier, the current library entry of the library list will be searched. If no current library exists in the library list, then library QGPL will be searched. If *LIBL is specified as the library qualifier, *LIBL is used to find the file.

Top

Source member (SRCMBR)

Specifies the name of a new or existing source file member that contains or will contain source for the displays or menus to be updated or created by SDA.

The possible value is:

***PRV** Specifies that SDA is to use the name of the source member used in your last SDA session (only when MODE is (*STD)).

***SELECT**
Specifies that SDA will bring up the Design Screens display. From this display you can press F4 on the Member field to see a list of source members to select from.

source-member-name
Type the name of the source member to be created or updated.

Top

Object library (OBJLIB)

Specifies the name of the library into which the program or display file that SDA creates will be stored.

The possible values are:

***PRV** Specifies that SDA is to use the name of the object library used in your last SDA session (only when MODE is (*STD)).

***CURLIB**
Specifies that SDA is to use the current library entry of the library list. If no current library entry exists in the library list, then library QGPL will be used.

object-library-name
Type the name of the library into which objects created by SDA are to be stored.

Top

Job description (JOBD)

Specifies the qualified name of the job description to be used with batch jobs being submitted by SDA. This parameter is not used if MODE (*S36) is used.

The possible values are:

***PRV** Specifies that SDA is to use the name of the job description and library used in your last SDA session (only when MODE is (*STD)).

job-description-name
Type the qualified name of the job description to be used with submitted jobs. If *CURLIB is specified as the library qualifier, the current library entry of the library list will be searched. If no current library exists in the library list, then library QGPL will be searched. If *LIBL is specified as the library qualifier, *LIBL is used to find the file.

***USRPRF**
Specifies that SDA is to use the name of the job description defined in the user profile.

Top

Test file (TSTFILE)

Specifies the qualified name of the display file to be used for testing. This parameter is ignored if MODE (*S36) is specified.

The possible values are:

***PRV** Specifies that SDA is to use the name of the display file and library used in your last SDA session (only when MODE is (*STD)).

test-file-name
Type the qualified name of the display file to be used for testing. If *CURLIB is specified as the library qualifier, the current library entry of the library list will be searched. If no current library exists in the library list, then library QGPL will be searched. If *LIBL is specified as the library qualifier, *LIBL is used to find the file.

Top

Mode (MODE)

Specifies which version of SDA is to use.

The possible values are:

***STD** Specifies SDA and displays the SDA main menu. The main menu will not appear if the OPTION parameter is specified.

***S36** Specifies the System/36 environment of SDA and displays the System/36 SDA main menu.

***S38** Specifies the System/38 view of SDA and displays the System/38 SDA main menu. The main menu will not appear if the OPTION parameter is specified.

Top

Examples

None

Top

Error messages

***ESCAPE Messages**

SDA0001
SDA does not support a &1 workstation.

SDA0002
SDA cannot access its display file.

SDA0003
SDA ended because of a serious error.

SDA0004
Error while processing the STRSDA command.

SDA0005
SDA cannot access its panel group.

SDA0601
SDA is already active.

Top

Start Source Entry Utility (STRSEU)

Where allowed to run:

- Batch job (*BATCH)
- Interactive job (*INTERACT)
- Interactive program (*IPGM)
- Interactive REXX procedure (*IREXX)
- Using QCMDEXEC, QCAEXEC, or QCAPCMD API (*EXEC)

Threadsafe: No

Parameters
Examples
Error messages

The STRSEU (Start Source Entry Utility) command allows you to create, change, display, or print a source member.

Error messages for STRSEU

*ESCAPE Messages

EDT9007

Error found on &1 command.

Top

Parameters

Keyword	Description	Choices	Notes
SRCFILE	Source file	Single values: ***PRV** Other values: *Qualified object name*	Optional, Positional 1
	Qualifier 1: Source file	*Name*	
	Qualifier 2: Library	*Name*, ***LIBL**, *CURLIB, *PRV	
SRCMBR	Source member	*Name*, ***PRV**, *SELECT	Optional, Positional 2
TYPE	Source type	*Simple name*, ***SAME**, BAS, BASP, BND, C, CBL, CBLLE, CICSC, CICSCBL, CICSCBLLE, CICSMAP, CICSSQLCBL, CL, CLD, CLP, CLLE, CMD, CPP, DFU, DSPF, FTN, ICFF, LF, MENU, MNU, MNUCMD, MNUDDS, PAS, PF, PLI, PNLGRP, PRTF, QRY, REXX, RMC, RPG, RPGLE, RPT, SPADCT, SQLC, SQLCBL, SQLCBLLE, SQLFTN, SQLPLI, SQLRPG, SQLRPGLE, SRT, TBL, TXT, BAS38, BASP38, BSCF38, CBL38, CL38, CLP38, CMD38, CMNF38, DFU38, DSPF38, LF38, MXDF38, PF38, PLI38, PRTF38, QRY38, RPG38, RPT38, SRT38, TXT38, ARS36, ASM36, BAS36, BASP36, BGC36, BGD36, BGF36, CBL36, DFU36, DTA36, DSPF36, FOR36, MNU36, MSGF36, OCL36, PHL36, RPG36, RPT36, SRT36, TXT36, UNS36, WSU36	Optional, Positional 3
OPTION	Option	***BLANK**, ' ', 2, 5, 6	Optional, Positional 4
TEXT	Text 'description'	*Character value*, ***BLANK**	Optional

Top

Source file (SRCFILE)

Specify the names of the source physical file and library that contain the member to be edited or created.

The possible values are:

***PRV** Specifies that SEU is to use the name of the source file and library used in your last SEU session. If you specify *PRV for the Source file parameter, it is not necessary to specify a library.

source-file-name
Type the name of an existing source file to be used. If you specify the source-file-name and a library name, SEU searches the specified library for the source file. If you do not specify the source-file-name with a library name, *LIBL is used.

library-name
Type the name of an existing library to be used. If *CURLIB is specified as the library, SEU searches the **current library** in your library list. If you specify *LIBL as the library, SEU searches the libraries in the library list for the file.

Top

Source member (SRCMBR)

Specifies the name of the source physical file member to be edited or created. The default value for this parameter depends on if you specify the SRCFILE parameter.

The possible values are:

***SELECT**
This is the default value if you specify the SRCFILE parameter. If you choose *SELECT, you will get lists of all members in the specified file and library. Select a member to edit, browse, print, or delete.

***PRV** This is the default value if you do not specify the SRCFILE parameter. *PRV is the name of the previous source physical file member or the Work with Members display.

source-file-member-name
Type the name of the source physical file member you want to create or edit.

Top

Source type (TYPE)

Specifies the type of source member to edit or create. The possible values are:

***SAME**
The default value is the same type as that used when this member was last edited. For a new member the default value is TXT.

TYPE This value allows you to specify the type of source to use. You can specify any combination up to 10 characters, or you can specify a type supported by SEU.

Your members can have any type that is meaningful for you. SEU supports the following member types:

Operating system types
BAS, BASP, BND, C, CBLLE, CBL, CICSC, CICSCBLLE, CICSCBL, CICSMAP, CICSSQLCBL, CL, CLD, CLLE, CLP, CMD, CPP, DFU, DSPF, FTN, ICFF, LF, MENU, MNU, MNUCMD, MNUDDS, PAS, PF, PLI, PNLGRP, PRTF, QRY, REXX, RMC, RPG, RPGLE, RPT, SPADCT, SQLC, SQLCLE, SQLCBL, SQLCBLLE, SQLFTN, SQLPLI, SQLRPG, SQLRPGLE, SRT, TBL, and TXT.

System/38 types
BAS38, BASP38, BSCF38, CBL38, CL38, CLP38, CMD38, CMNF38, DFU38, DSPF38, LF38, MXDF38, PF38, PLI38, PRTF38, QRY38, RPG38, RPT38, SRT38, and TXT38.

System/36 types
ARS36, ASM36, BAS36, BASP36, BGC36, BGD36, BGF36, CBL36, DFU36, DSPF36, DTA36, FOR36, MNU36, MSGF36, OCL36, PHL36, RPG36, RPT36, SRT36, TXT36, UNS36, and WSU36.

In addition to the types listed above, you can also choose your own member type names.

Top

Option (OPTION)

Specifies the function to perform on the selected member. The default value depends on if you specify a member name. If you do not specify a member name, the default is *BLANK, which indicates no action. If you specify a member name, the default value is 2 (Edit). This specifies an Edit session for the member.

The possible value is:

***BLANK or ' '**
This is the default value if you do not specify a member name. *BLANK specifies no action.

2=Edit a member
Type 2 after you have selected the option parameter to go to the Edit display.

5=Browse a member
Type 5 after you have selected the option parameter to go to the Browse display.

6=Print Member
Type 6 to print the member specified.

Top

Text 'description' (TEXT)

Specify a character string that describes the member in the text field for the member.

The possible values are:

***BLANK**
Specifies that SEU should enter blanks in the text field of a new member. This default does not change the text field of an existing member.

Description
Specify a character string of up to 50 characters to describe a member. Enclose the string in apostrophes to use leading or trailing blanks.

Top

Examples

None

Top

Error messages

***ESCAPE Messages**

EDT9007
Error found on &1 command.

Top

Work with Libraries Using PDM (WRKLIBPDM)

Where allowed to run: Interactive environments (*INTERACT *IPGM *IREXX *EXEC)
Threadsafe: No

Parameters
Examples
Error messages

The Work with Libraries Using PDM (WRKLIBPDM) command enables you to work with a single library or multiple libraries. Using this command, you can bypass the Programming Development Manager (PDM) menu and the Specify Libraries to Work With display.

Top

Parameters

Keyword	Description	Choices	Notes
LIB	Library	*Character value*, ***PRV**, *LIBL, *USRLIBL, *ALL, *ALLUSR, *CURLIB	Optional, Positional 1
ASP	ASP number	1-32, ***ALL**	Optional
ASPDEV	ASP device	*Name*, *, *SYSBAS, *CURASPGRP	Optional

Top

Library (LIB)

Specifies the libraries to work with.

***PRV** Work with the same library or libraries that you worked with in your previous WRKLIBPDM session.

***LIBL** Work with all of the libraries in the job's library list.

***USRLIBL**
Work with all libraries in the user portion of the job's library list.

***ALL** Work with all libraries on the system, including QSYS and QTEMP.

***ALLUSR**
Work with all non-system libraries, including all user libraries. The libraries are listed alphabetically by library name.

***CURLIB**
Work with the current library for the job. If you have not specified a current library for the job, library QGPL is assumed.

generic-name
Specify the generic name of the libraries to work with.

The generic name can be in one of the following formats:

ABC* Displays a list of all libraries that begin with the characters ABC. For example, ABC, ABCD, or ABCTEST.

***ABC** Displays a list of all libraries ending with the characters ABC. For example, ABC, DABC, or TESTABC.

B Displays a list of all libraries that have the character B anywhere in the name. For example, B, BALL, ABCD.

A*C Displays a list of all libraries that begin with the character A and end with the character C. For example, AC, ABC, or AZZZC.

″a*″ Displays a list of all libraries with quoted names that start with the letter **a**. For example, ″a″, ″aB″, ″aD″.

****ALL** Displays a list of all libraries ending with ALL. For example, ALL, BALL, or TESTALL. The double asterisk is needed in this case, since *ALL is defined as the special value to display a list of all libraries.

name Specify the name of the single library to work with.

Top

ASP number (LIB)

Specifies the auxiliary storage pool (ASP) for the libraries that should be shown on the Work with Libraries display. This parameter is ignored when a value of *LIBL, *CURLIB or *USRLIBL is entered for the library parameter. If a number is specified for this parameter, the ASP device (ASPDEV) parameter value must be *.

***ALL** All ASPs defined by the value for the ASP device (ASPDEV) parameter will be searched.

1-32 Specify the number of the system or basic user ASP to be searched. ASP 1 is the system ASP, which is always configured. Basic user ASPs are 2-32, and must designate an ASP that is configured on the system. For information on configuring an ASP, see the Backup and Recovery book, SC41-5304.

Top

ASP device (LIB)

Specifies the auxiliary storage pool (ASP) device name where storage for the library being displayed is allocated. If the library is in an ASP that is not part of the thread's library name space, this parameter must be specified to ensure the correct library is displayed. If a number is specified for the ASP number (ASP) parameter, the ASPDEV parameter value must be *.

***** The ASPs that are currently part of the thread's library name space will be searched to find the library. This includes the system ASP (ASP 1), all defined basic user ASPs (ASPs 2-32), and, if the thread has an ASP group, the primary and secondary ASPs in the thread's ASP group.

***SYSBAS**
The system ASP (ASP 1) and all defined basic user ASPs (ASPs 2-32) will be searched to find the library. No primary or secondary ASPs will be searched, even if the thread has an ASP group.

***CURASPGRP**
If the thread has an ASP group, the primary and secondary ASPs in the thread's ASP group will be searched to find the library. The system ASP (ASP 1) and defined basic user ASPs (ASPs 2-32) will not be searched. If no ASP group is associated with the thread an error will be issued.

name Specify the device name of the primary or secondary ASP to be searched. The primary or secondary ASP must have been activated (by varying on the ASP device) and have a status of 'Available'. The system ASP (ASP 1) and configured basic user ASPs (ASPs 2-32) will not be searched.

Top

Examples

Example 1: Work with Libraries in Job Library List

```
WRKLIBPDM   LIB('*LIBL')
```

This command allows you to work with all of the libraries that are in the library list of the current job. The libraries are listed in the same order as the library list.

Example 2: Work with a Generic Set of Libraries

```
WRKLIBPDM   LIB('*PAY*')
```

This command allows you to work with the subset of libraries that contain the letters **PAY** in the library name. The libraries are listed in alphabetical order.

Top

Error messages

***ESCAPE Messages**

PDM0055

Error while processing the &1 command.

Top

Work with Members Using PDM (WRKMBRPDM)

Where allowed to run: Interactive environments (*INTERACT *IPGM *IREXX *EXEC)
Threadsafe: No

Parameters
Examples
Error messages

The Work with Members Using PDM (WRKMBRPDM) command enables you to work with a list of members in one database file. Using this command, you can bypass the Programming Development Manager (PDM) menu and the Specify Members to Work With display.

Top

Parameters

Keyword	Description	Choices	Notes
FILE	File	Single values: ***PRV** Other values: *Qualified object name*	Optional, Positional 1
	Qualifier 1: File	*Name*	
	Qualifier 2: Library	*Name*, *PRV, ***LIBL**, *CURLIB	
MBR	Member	*Character value*, ***ALL**, *PRV	Optional, Positional 2
MBRTYPE	Member type	*Character value*, ***ALL**, *PRV, *BLANK, BAS, BAS36, BAS38, BASP, BASP38, C, CBL, CBLLE, CBL36, CBL38, CICSC, CICSCBL, CICSCBLLE, CICSMAP, CICSSQLCBL, CLD, CLLE, CLP, CLP38, CMD, CMD38, CPP, DSPF, DSPF36, DSPF38, FTN, ICFF, LF, LF38, MENU, MNU, MNUCMD, MNUDDS, MNU36, MSGF36, OCL36, PAS, PF, PF38, PLI, PLI38, PNLGRP, PRTF, PRTF38, QRY38, REXX, RMC, RPG, RPGLE, RPG36, RPG38, RPT, RPT36, RPT38, SPADCT, SQLC, SQLCPP, SQLCBL, SQLCBLLE, SQLFTN, SQLPLI, SQLRPG, SQLRPGLE, TBL, TXT	Optional, Positional 3

Top

File (FILE)

Specifies the database file that contains the members you want to work with. The file can be a source physical file or a data physical file.

Single values

***PRV** Work with the members of the same file that was used in your previous WRKMBRPDM session.

Qualifier 1: File

name Specify the name of the physical file that contains the members you want to work with.

Qualifier 2: Library

***LIBL** Type *LIBL followed by a file name to search the current library list for the file you want to work with. If the command WRKMBRPDM FILE(file name) is used, PDM will search the library list for the file specified.

***PRV** Work with a physical file in the same library that was used in your previous WRKMBRPDM session. You can specify a different file in the previous library by using the following format: FILE(*PRV/QRPGSRC).

***CURLIB**
Work with a physical file in the current library for the job. If no current library is defined, QGPL is the default value.

name Specify the name of the library that contains the file and members that you want to work with.

Top

Member (MBR)

Specifies the member or members you want to work with. You can use this parameter to work with all the members or a subset of members in the specified file.

***ALL** Work with all members in the specified file.

***PRV** Work with the same set of members that was used in your previous WRKMBRPDM session.

generic-name
Specify the generic name of the file members that you want to work with.

The generic name can be in one of the following formats:

ABC* Displays a list of all members that begin with the characters ABC. For example, ABC, ABCD, or ABCTEST.

***ABC** Displays a list of all members ending with the characters ABC. For example, ABC, DABC, or TESTABC.

B Displays a list of all members that have the character B anywhere in the name. For example, B, BALL, ABCD.

A*C Displays a list of all members that begin with the character A and end with the character C. For example, AC, ABC, or AZZZC.

″**a***″ Displays a list of all members with quoted names that start with the letter **a**. For example, ″a″, ″aB″, ″aD″.

****ALL** Displays a list of all members ending with ALL. For example, ALL, BALL, or TESTALL. The double asterisk is needed in this case, since *ALL is defined as the special value to display a list of all members.

name Specify the name of the single file member that you want to work with.

Top

Member type (MBRTYPE)

Specifies the member type for members you want to work with. You can use this parameter to work with all the member types in a specified file or a subset of members which matches a specific or generic member type.

***ALL** Work with file members of all member types, including those with no type.

***PRV** Work with members with the same member type that was used in your previous WRKMBRPDM session.

***BLANK**
Work with only those members with no member type value.

member-type
Specify a member type to display a list of all members of that particular type.

You can use a member type that you have created, or use one of the following standard member types used by PDM commands.

BAS BASIC

BAS36
BASIC System/36

BAS38
BASIC System/38

BASP BASIC Native Procedure

BASP38
BASIC System/38 Native Procedure

C C Language

CBL COBOL

CBLLE
Integrated Language Environment* COBOL/400

CBL36 COBOL System/36

CBL38 COBOL System/38

CICSC
CICS C

CICSCBL
CICS* COBOL

CICSMAP
CICS Map

CICSSQLCBL
CICS DB2/400 Query Manager COBOL

CLD C Locale Description

CLLE Control Language Integrated Language Environment*

CLP Control Language

CLP38 System/38 Control Language

CMD Command

CMD38
Command System/38

CPP C++

DSPF Display File

DSPF36
Display File System/36

DSPF38
Display File System/38

FTN FORTRAN/400

ICFF Inter-System Communications Function File

LF Logical File

LF38 Logical File System/38

MENU
UIM MENU

MNU Menu

MNUCMD
Menu Command

MNUDDS
Menu Data Description Specifications

MNU36
Menu System/36

MSGF36
Message File For System/36

OCL36
System/36 Operator Control Language

PAS Pascal

PF Physical File

PF38 Physical File System/38

PLI PL/I

PLI38 PL/I System/38

PNLGRP
Panel Group

PRTF Printer File

PRTF38
Printer File System/38

QRY38
S/38 QUERY

REXX Restructured Extended Executor Language

RMC RM/COBOL-85**

RPG RPG/400

RPGLE
Integrated Language Environment RPG/400

RPG36
RPG System/36

RPG38
RPG System/38

RPT RPG Auto Report

RPT36 RPG Auto Report System/36

RPT38 RPG Auto Report System/38

SPADCT
Spelling Aid Dictionary

SQLC DB2/400 Query Manager C

SQLCPP
DB2/400 Query Manager C++

SQLCBL
DB2/400 Query Manager COBOL

SQLCBLLE
DB2/400 Query Manager Integrated Language Environment COBOL/400

SQLCLE
DB2/400 Query Manager C/400 Integrated Language Environment

SQLFTN
DB2/400 Query Manager FORTRAN

SQLPLI
DB2/400 Query Manager PL/I

SQLRPG
DB2/400 Query Manager RPG

SQLRPGLE
DB2/400 Query Manager Integrated Language Environment RPG/400

TBL Table

TXT Text

generic-member-type
Specify the generic member type of the file members that you want to work with.

The generic member type can be in one of the following typical formats:

RPG* Displays a list of all members whose member type begins with the characters RPG. For example, RPG, RPG36, and RPG38.

***C** Displays a list of all members whose member type ends with the character C. For example, C and SQLC.

I Displays a list of all members that have the character I anywhere in the member type. For example, ICFF, PLI, PLI38, and SQLPLI.

R*36 Displays a list of all members whose member type begins with the character R and ends with the characters 36. For example, RPG36 and RPT36.

″**a***″ Displays a list of all members with quoted member types that start with the letter **a**. For example, ″a″, ″aB″, ″aD″.

****ALL** Displays a list of all members whose member type ends with ALL. For example, ALL, BALL, or TESTALL. The double asterisk is needed in this case, since *ALL is defined as the special value to display a list of members with all member types.

Top

Examples

Example 1: Work with All Members in a File

```
WRKMBRPDM   FILE(*PRV)  MBR('*ALL')
```

This command allows you to work with all of the members of the same file you worked with in the previous WRKMBRPDM session.

Example 2: Work with Members of One Type

```
WRKMBRPDM   FILE(*LIBL/MYSRCFILE)  MBRTYPE('CLP')
```

This command allows you to work with all members of source file MYSRCFILE that have a member type of CLP (Control Language Program). The source file is located using the job library list.

Example 3: Work with a Members by Generic Name

```
WRKMBRPDM   FILE(MYLIB/MYSRCFILE)  MBR('PAY*')
```

This command allows you to work with all members of source file MYSRCFILE in library MYLIB with member names that begin with the letters **PAY**.

Top

Error messages

***ESCAPE Messages**

PDM0055

Error while processing the &1 command.

Top

Work with Objects Using PDM (WRKOBJPDM)

Where allowed to run: Interactive environments (*INTERACT *IPGM *IREXX *EXEC)
Threadsafe: No

Parameters
Examples
Error messages

The Work with Objects Using PDM (WRKOBJPDM) command enables you to work with a list of objects in one library. Using this command, you can bypass the Programming Development Manager (PDM) menu and the Specify Objects to Work With display.

Top

Parameters

Keyword	Description	Choices	Notes
LIB	Library	*Name*, ***PRV**, *CURLIB	Optional, Positional 1
OBJ	Object	*Character value*, ***ALL**, *PRV	Optional, Positional 2
OBJTYPE	Object type	***ALL**, *PRV, *ALRTBL, *AUTL, *BNDDIR, *CFGL, *CHTFMT, *CLD, *CLS, *CMD, *CNNL, *COSD, *CRG, *CRQD, *CSI, *CSPMAP, *CSPTBL, *CTLD, *DEVD, *DOC, *DTAARA, *DTADCT, *DTAQ, *EDTD, *EXITRG, *FCT, *FILE, *FNTRSC, *FNTTBL, *FORMDF, *FTR, *GSS, *IGCDCT, *IGCSRT, *IGCTBL, *IMGCLG, *IPXD, *JOBD, *JOBQ, *JOBSCD, *JRN, *JRNRCV, *LIB, *LIND, *LOCALE, *M36, *M36CFG, *MEDDFN, *MENU, *MGTCOL, *MODD, *MODULE, *MSGF, *MSGQ, *NODGRP, *NODL, *NTBD, *NWID, *NWSD, *NWSCFG, *OUTQ, *OVL, *PAGDFN, *PAGSEG, *PDFMAP, *PDG, *PGM, *PNLGRP, *PRDAVL, *PRDDFN, *PRDLOD, *PSFCFG, *QMFORM, *QMQRY, *QRYDFN, *RCT, *SBSD, *SCHIDX, *SPADCT, *SQLPKG, *SQLUDT, *SRVPGM, *SSND, *SVRSTG, *S36, *TBL, *TIMZON, *USRIDX, *USRPRF, *USRQ, *USRSPC, *VLDL, *WSCST	Optional, Positional 3
OBJATR	Object attribute	*Character value*, ***ALL**, *PRV, *BLANK, BAS, BAS36, BAS38, BSCF38, C, CBL, CBL36, CBL38, CLE, CLP, CLP38, CMD, CMD38, CMNF38, CSPAE, DDMF, DFU, DFUEXEC, DFUNOTEXC, DKTF, DSPF, DSPF36, DSPF38, FTN, ICFF, LF, LF38, MXDF38, PAS, 'PF-DTA', 'PF-SRC', PF38, PLI, PLI38, PRTF, PRTF38, QRY38, RMC, RPG, RPG36, RPG38, RPT, RPT36, RPT38, SAVF, SPADCT, SQLC, SQLCBL, SQLCLE, SQLFTN, SQLPLI, SQLRPG, TAPF, TBL	Optional

Top

Library (LIB)

Specifies the library that contains the objects you want to work with.

***PRV** The library that was used in your previous WRKOBJPDM session will be used.

***CURLIB**
The current library for the job will be used. If no current library is defined, library QGPL will be used.

name Specify the name of the library that contains the objects you want to work with.

Top

Object (OBJ)

Specifies the objects you want to work with. You can use this parameter to work with all the objects or a subset of objects in the specified library.

***ALL** Work with all objects in the library specified for the **Library (LIB)** parameter. You can subset the list of objects by specifying a value other than *ALL for the **Object type (OBJTYPE)** and **Object attribute (OBJATR)** parameters.

***PRV** Work with the same object or objects that you worked with in your previous WRKOBJPDM session.

generic-name
Specify the generic name of the objects to work with.

The generic name can be in one of the following formats:

ABC* Displays a list of all objects that begin with the characters ABC. For example, ABC, ABCD, or ABCTEST.

***ABC** Displays a list of all objects ending with the characters ABC. For example, ABC, DABC, or TESTABC.

B Displays a list of all objects that have the character B anywhere in the name. For example, B, BALL, ABCD.

A*C Displays a list of all objects that begin with the character A and end with the character C. For example, AC, ABC, or AZZZC.

″**a***″ Displays a list of all objects with quoted names that start with the letter **a**. For example, ″a″, ″aB″, ″aD″.

****ALL** Displays a list of all objects ending with ALL. For example, ALL, BALL, or TESTALL. The double asterisk is needed in this case, since *ALL is defined as the special value to display a list of all objects.

name Specify the name of the object or objects to work with. Multiple objects will be listed if there are objects with different object types with this name and *ALL is specified for the OBJTYPE parameter.

Top

Object type (OBJTYPE)

Specifies the object type for objects you want to work with. You can use this parameter to work with all object types or a subset of objects.

***ALL** Work with objects that match the object name specified for the **Object (OBJ)** parameter.

***PRV** Work with objects that are the same object type you worked with in your previous WRKOBJPDM session.

object-type
Specify the system object type of object you want to work with.

You can choose from the following object types:

***ALRTBL**
Alert Table

***AUTL**
Authorization List

***BNDDIR**
Binding Directory

***CFGL**
Configuration List

***CHTFMT**
Chart Format

***CLD** C Locale Description

***CLS** Class

***CMD** Command

***CNNL**
Connection List

***COSD**
Class of Service Definition

***CRG** Cluster Resource Group

***CRQD**
Change Request Description

***CSI** Communication Side Information

***CSPMAP**
Cross-System Product Map

***CSPTBL**
Cross-System Product Table

***CTLD**
Control Description

***DEVD**
Device Description

***DOC** Document

***DTAARA**
Data Area

***DTADCT**
Data Dictionary

***DTAQ**
Data Queue

***EDTD**
Edit Description

***EXITRG**
Exit Registration

***FCT** Forms Control Table

***FILE** File

***FNTRSC**
Font Resource

***FNTTBL**
Font Mapping Table

***FORMDF**
Form Definition

***FTR** Filter

***GSS** Graphic Symbol Set

***IGCDCT**
Ideographic Character Dictionary

***IGCSRT**
Ideographic Character Sort

***IGCTBL**
Ideographic Character Table

***IMGCLG**
Optical Image Catalog

***IPXD** Internet Packet Exchange Description

***JOBD**
Job Description

***JOBQ**
Job Queue

***JOBSCD**
Job Schedule

***JRN** Journal

***JRNRCV**
Journal Receiver

***LIB** Library

***LIND**
Line Description

***LOCALE**
Locale Space

***M36** Advanced 36 Machine

***M36CFG**
Advanced 36 Machine Configuration

***MEDDFN**
Media Definition

***MENU**
Menu

***MGTCOL**
Management Collection

***MODD**
Mode Description

***MODULE**
Module

***MSGF**
Message File

***MSGQ**
Message Queue

***NODGRP**
Node Group

***NODL**
Node List

***NTBD**
NetBIOS Configuration Data

***NWID**
Network Interface Description

***NWSCFG**
Network Server Configuration

***NWSD**
Network Server Description

***OUTQ**
Output

***OVL** Overlay

***PAGDFN**
Page Definition

***PAGSEG**
Page Segment

***PDFMAP**
Portable Document Format Map

***PDG** Print Manager CPI: Logical Print Descriptor

***PGM** Program

***PNLGRP**
Panel Group

***PRDAVL**
Product Availability

***PRDDFN**
Product Definition

***PRDLOD**
Product Load

***PSFCFG**
Print Services Facility Configuration

***QMFORM**
Query Manager Form

***QMQRY**
Query Manager Query

***QRYDFN**
Query Definition

***RCT** Remote Control Table

***SBSD**
Subsystem Description

***SCHIDX**
Search Index

***SPADCT**
Spelling Aid Dictionary

***SQLPKG**
DB2/400 Query Manager Package

***SQLUDT**
SQL User Defined Type

***SRVPGM**
Service Program

***SSND**
Session Description

***SVRSTG**
Server Storage Space

***S36** System/36 Machine Description

***TBL** Table

***TIMZON**
Time Zone Description

***USRIDX**
User Index

***USRPRF**
User Profile

***USRQ**
User Queue

***USRSPC**
User Space

***VLDL**
Validation List

***WSCST**
Workstation User Customization

Top

Object attribute (OBJATR)

Specifies the object attribute for objects you want to work with. You can use this parameter to work with a subset of objects that match the object name and object type values specified for the **Object (OBJ)** and **Object type (OBJTYPE)** parameters.

***ALL** Work with all objects that match the values specified for the OBJ and OBJTYPE parameters.

***PRV** Work with objects that have the same object attribute you worked with in your previous WRKOBJPDM session.

***BLANK**
Work with all objects that have no attribute value.

value Specify the object attribute of objects you want to work with. If you specify an object attribute, you do not have to specify the object type.

You can choose from the following object attribute values:

BAS BASIC

BAS36
BASIC System/36

BAS38
BASIC System/38

BSCF38
Binary Synchronous Communication File System/38

C C Language

CBL COBOL

CBLLE
Integrated Language Environment COBOL/400

CBL36 COBOL System/36

CBL38 COBOL System/38

CLLE Control Language Integrated Language Environment

CLP Control Language

CLP38 Control Language System/38

CMD Command

CMD38
Command System/38

CMNF38
Communications File

CSPAE
Cross-System Product Application Execution

DDMF
Distributed Data Management

DFU Data File Utility

DFUEXEC
Data File Utility Executable File

DFUNOTEXC
Data File Utility Non-Executable File

DKTF Diskette File

DSPF Display File

DSPF36
Display File System/36

DSPF38
Display File System/38

FTN FORTRAN/400

ICFF Inter-System Communications Function File

LF Logical File

LF38 Logical File System/38

MXDF38
Mixed File System/38

PAS Pascal

PF-DTA
Physical File - Data

PF-SRC
Physical File - Source

PF38 Physical File System/38

PLI PL/I

PLI38 PL/I System/38

PRTF Printer File

PRTF38
Printer File System/38

QRY38
System/38 QUERY

RMC RM/COBOL-85**

RPG RPG/400

RPGLE
Integrated Language Environment RPG/400

RPG36
RPG System/36

RPG38
RPG System/38

RPT RPG Auto Report

RPT36 RPG Auto Report System/36

RPT38 RPG Auto Report System/38

SAVF Save File

SPADCT
Spelling Aid Dictionary

SQLC DB2/400 Query Manager C

SQLCBL
DB2/400 Query Manager COBOL

SQLCBLLE
DB2/400 Query Manager Integrated Language Environment COBOL/400

SQLFTN
DB2/400 Query Manager FORTRAN

SQLPLI
DB2/400 Query Manager PL/I

SQLRPG
DB2/400 Query Manager RPG

SQLRPGLE
DB2/400 Query Manager Integrated Language Environment RPG/400

TAPF Tape File

TBL Table

generic-value
Specify the generic attribute of the objects to work with.

The generic value can be in one of the following formats:

RPG* Displays a list of all objects whose attribute type begins with the characters RPG. For example, RPG, RPG36, and RPG38.

***C** Displays a list of all objects whose attribute type ends with the character C. For example, C and SQLC.

I Displays a list of all objects that have the character I anywhere in the attribute type. For example, ICFF, PLI, PLI38, and SQLPLI.

P*38 Displays a list of all objects whose attribute type begins with the character P and ends with the characters 38. For example, PLI38 and PRTF38.

″**a***″ Displays a list of all objects whose attribute type contains the letter **a** within quotation marks. For example, ″a″, ″aB″, ″aD″.

****ALL** Displays a list of all objects whose attribute type ends with ALL. For example, ALL, BALL, or TESTALL. The double asterisk is needed in this case, since *ALL is defined as the special value to display objects regardless of the object attribute.

Top

Examples

Example 1: Work with All Objects in a Library

```
WRKOBJPDM   LIB(MYLIB)  OBJ('*ALL')
```

This command allows you to work with all of the objects in library MYLIB.

Example 2: Work with Objects of One Type

```
WRKOBJPDM   LIB(*PRV)  OBJ('*ALL')  OBJTYPE(*CMD)
```

This command allows you to work with all command (*CMD) objects in the same library you worked with in the previous WRKOBJPDM session.

Top

Error messages

***ESCAPE Messages**

PDM0055
Error while processing the &1 command.

Top

Appendix. Notices

This information was developed for products and services offered in the U.S.A.

IBM may 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 described 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 Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY8809
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 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 may 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.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation

Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

Advanced Function Printing
AFP
AS/400
CICS
COBOL/400
C/400
DataPropagator
DB2
IBM
Infoprint
InfoWindow
iSeries
LPDA
OfficeVision

i5/OS
Print Services Facility
RPG/400
SystemView
System/36
TCS
WebSphere

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

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

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

Other company, product, and service names may be trademarks or service marks of others.

Terms and conditions for downloading and printing publications

Permissions for the use of the publications you have selected for download are granted subject to the following terms and conditions and your indication of acceptance thereof.

Personal Use: You may reproduce these Publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative works of these Publications, or any portion thereof, without the express consent of IBM.

Commercial Use: You may reproduce, distribute and display these Publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these Publications, or reproduce, distribute or display these Publications or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the Publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the Publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations. IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED ″AS-IS″ AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE

All material copyrighted by IBM Corporation.

By downloading or printing a publication from this site, you have indicated your agreement with these terms and conditions.

Code disclaimer information

This document contains programming examples.

IBM grants you a nonexclusive copyright license to use all programming code examples from which you can generate similar function tailored to your own specific needs.

All sample code is provided by IBM for illustrative purposes only. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

All programs contained herein are provided to you ″AS IS″ without any warranties of any kind. The implied warranties of non-infringement, merchantability and fitness for a particular purpose are expressly disclaimed.

Printed in USA