

RM/COBOL Version 12 Enhancements

Version 12 Runtime System Features

The RM/COBOL version 12 for Windows and UNIX runtime system has been modified with the following new features:

- New Object Version Level. [Object version 15](#) has been introduced to support the following:
 - the TRAILING adjective in the INSPECT statement,
 - the ability to have more than 65,534 identifiers in a program when the Y (Debug Symbol Table) Compile Command Option is used,
 - a subscripted reference to a variable-length group,
 - a reference to a data item with a length greater than 65,280 characters, other than in a MOVE statement,
 - a SEARCH ALL statement that references a table with more than 65,535 elements, and
 - the JUSTIFIED phrase in reference modification.
- Improved compatibility with Microsoft Windows Server 2008 and Windows Vista.
- The maximum total length of the runtime command line options has been increased to 4095 characters. This modification allows for a longer main program argument value in the A Runtime Command Option and more or longer pathname values in L Runtime Command Option specifications.
- A new value has been added for the ACTION keyword of the TERM-INPUT configuration record. The [ENTER-DEBUGGER value](#) causes the RM/COBOL Interactive Debugger to be entered at the next statement executed after an ACCEPT statement.
- The rmattach utility, used to attach configuration files for the runtime and compiler on Windows, has been removed. Use of an [Automatic Configuration File module](#) is recommended as a replacement for attached configurations.

Version 12 Compiler Features

The RM/COBOL version 12 for Windows and UNIX compiler has been enhanced with the following new features:

- The maximum number of identifiers that can be described in a single separately compiled program has been increased from 65,534 to 840,000. If the [Y Compile Command Option](#) (debugging symbol table) is specified, declaration of more than 65,534 identifiers requires object version 15. In order to allow faster compilation of such large programs, the maximum workspace size, that is, the value for the [W Compiler Command Option](#) or the [WORKSPACE-SIZE configuration keyword](#), has been increased from 16384 to 524288.
- Working-Storage and Linkage Section elementary data items can now be greater than 65,280 characters in length. Also, group and elementary data items greater than 65,280 characters in length can now be referenced in statements other than the just the MOVE statement. File records are still limited to a maximum of 65,280 characters due to file system limitations. References to data items longer than 65,280 characters in length other than in MOVE statements requires object version 15.

RM/COBOL Version 12 Enhancements

- Reference modification now allows data items greater than 65,280 characters in length to be reference modified.
- The OCCURS clause now allows more than 64K occurrences. When a table with more than 64K occurrences is referenced in a SEARCH ALL statement, object version 15 is required.
- The SAME AS clause has been added as a new data description clause. This clause allows declaring another data item that has the same data description as a preceding data item.
- The INSPECT statement has been enhanced with a TRAILING adjective for tallying or replacement of trailing characters in the value of a data item. Use of this feature requires object version 15. This new capability is particularly useful in counting the number of spaces at the end of a nonnumeric data item.
- Reference modification has been enhanced with a JUSTIFIED phrase that allows easy removal of trailing spaces in a sending data item or right justification in a receiving data item. Use of this feature requires object version 15.
- Index-names no longer need to be unique. Qualification of index-names is allowed and must be used to achieve uniqueness of reference when non-unique index-names are defined in a program. As a result of this change, the compiler listing allocation map no longer lists index-names separately, but instead shows where they occur in their hierarchy of possible qualification.
- Elements of a table declared at level-number 01 or 77 are no longer implicitly synchronized. The ability to define tables at level-numbers 01 and 77 was introduced in version 8, but the elements were implicitly synchronized because they inherited the implicit synchronization of level-number 01 and 77 data items. This implicit synchronization of the elements of the table caused unexpected results for users when redefining a set of odd length values as a table. If synchronization of the elements is desired, an explicit SYNCHRONIZED clause is required.
- The ACCEPT statement has been enhanced to allow the BEEP phrase not to be the default by using configuration to specify the default behavior. When NO BEEP is configured to be the default, the BEEP phrase may be specified on those ACCEPT statements where a beep is desired.
- END-COPY and END-REPLACE scope terminators have been added to the COPY and REPLACE statements, respectively. These scope terminators are more visible than the period space separator required in standard COBOL at the end of COPY and REPLACE statements and are clearer specification of structure when the COPY or REPLACE statements are used within other structured statements. Use of these scope terminators is recommended to make error analysis easier for the program author when the scope of a COPY or REPLACE statement is not properly terminated, which is not always obvious, for example, when a lengthy REPLACING phrase is specified in a COPY statement. Various scanning improvements were also made to improve error recovery following a COPY or REPLACE statement syntax error. (See the "Delimited Scope Statements", "COPY Statement", and "REPLACE Statement" topics in Chapter 1: *Language Structure of the RM/COBOL Language Reference Manual*.)
- A period within pseudo-text that is immediately followed by the closing pseudo-text delimiter is now treated as a period space separator in order to match the treatment of a period just before the right margin of a source record. This eliminates a warning that a period space separator was expected that sometimes occurred for pseudo-text in what were otherwise valid COBOL programs.
- Conditional source compilation through the use of strings in the Identification area of source records has been added. The string patterns to include or exclude are specified in the configuration file. Conditionally included lines are forced to not be comment lines and conditionally excluded lines are forced to be comment lines. Configuration of how conditionally included or excluded lines are indicated in the program listing has also been added as part of this feature.
- The rules for the OCCURS clause with the DEPENDING ON phrase were relaxed to allow multiple specifications of variable occurrence data items within a single data record. When this feature is used in such a way that a variable-length group is subscripted, object version 15 is required.
- The REDEFINES and RENAMES clauses (in the data description entry) allow redefinition or renaming, respectively, of variable-length groups. See Chapter 4: *Data Division of the RM/COBOL Language Reference Manual*.
- The compiler now supports the following new special registers (for detailed information, see Chapter 1: *Language Structure of the RM/COBOL Language Reference Manual*):
 - HIGHEST-VALUE and LOWEST-VALUE
These special registers return the highest and lowest values, respectively, for the data item referenced by *identifier-1*. They may be used wherever a literal of the resulting type may be used in the Procedure Division. These special registers are particularly useful when the data item is numeric, in which case, the highest and lowest numeric values are determined by

RM/COBOL Version 12 Enhancements

the decimal precision and scale specified in the PICTURE character-string.

- INITIAL-VALUE

This special register returns the initial value of the data item referenced by *data-name-1*. It may be used wherever a literal of the resulting type may be used in the Procedure Division. The initial value is defined as the value that would be placed in the data item referenced by *data-name-1* if it were initialized using the INITIALIZE statement with the VALUE and DEFAULT phrases specified.

- MAX-VALUE and MIN-VALUE

These special registers return the maximum or minimum values, respectively, for the data item referenced by *identifier-1*. They may be used wherever a literal of the resulting type may be used in the Procedure Division. These special registers are particularly useful when the data item is numeric, in which case, the maximum and minimum numeric values are determined by the storage format (from the combination of the USAGE and PICTURE clauses) for the data item. (Differences between the maximum and minimum values versus the highest and lowest values, respectively, are the result of issues with BINARY and PACKED-DECIMAL storage for decimally-defined numeric data items.)

- PROCEDURE-NAME

This special register exists for any paragraph or section in the Procedure Division of a program. The value is a nonnumeric literal determined as follows: if PARAGRAPH is specified, this special register returns the name of the paragraph in which it is specified; if PROCEDURE is specified, this special register returns the qualified name of the current paragraph in which it is specified; if SECTION is specified, this special register returns the name of the section in which it is specified.

- The [COMPILER-OPTIONS configuration record](#) now supports the following new keywords:

Note: Two additional keywords, used specifically with XML Extensions, are described in [Features Added to Support XML Extensions](#).

- ACCEPT-BEEP-DEFAULT

This keyword can be set to a value of NO (or OFF) to reverse the RM/COBOL default behavior for ACCEPT statements to match the Micro Focus COBOL default for ACCEPT statements.

- EXTERNAL-INDEX-NAMES

This keyword controls whether index-names declared within an external record area are external or not external. If the value of this keyword is set to YES, then index-names are external when declared within an external record area. If the value of this keyword is set to NO, then index-names are never external. The default value of this keyword is YES to match prior RM/COBOL behavior. (The COBOL language was changed to not have external index-names after RM/COBOL was implemented. Thus the NO setting matches this later definition of COBOL.) The NO setting is recommended when using the new qualified index-names feature in version 12 because external items do not support qualification.

- LISTING-CONDITIONAL-INCLUSION-INDICATOR and LISTING-CONDITIONAL-EXCLUSION-INDICATOR

These keywords allow the configuration of the indicator column value in the listing for conditionally included or excluded source lines. This configuration applies to the new conditional compilation feature in version 12 based on strings in the Identification area of source records.

- SOURCE-PATTERN-INCLUDE and SOURCE-PATTERN-EXCLUDE

These keywords specify a list of pattern strings to be matched against strings in the Identification area of source records for the purpose of conditionally including or excluding a source record in the compilation. The patterns are simple strings (not regular expressions).

- The compiler listing allocation map now indicates table data items in the Order column with an "*" for a fixed-occurrence table and a "#" for a variable-occurrence table.
- The compiler listing allocation map now includes the level-number for data-names and the level indicator FD, SD or CD for file-names, sort-merge-file-names and cd-names, respectively.
- The figurative constant ZERO (ZEROS, ZEROES) may now be used with its nonnumeric meaning in literal concatenation expressions. Prior to this change, this figurative constant was disallowed in concatenation expressions because concatenation of numeric literals is not supported and the figurative constant was treated as numeric in this context.
- CodeWatch now has the ability to debug Xcentrity Business Information Server (BIS) service programs, running under Internet Information Server on a Microsoft Windows host.

RM/COBOL Version 12 Enhancements

A new option in the New Workspace Wizard allows you to select the type of program that you are working on—either a traditional COBOL program or a BIS service program. If you choose the latter, additional options allow you to specify the URL of the starting page of your BIS application (useful for browser-driven applications) or the URL of the virtual BIS directory for web services debugging.

In addition, CodeWatch stores relative paths in a saved workspace file (.CWF) wherever possible, making it much easier to move saved workspace files between machines. Since the paths are relative to the location of the .cwf file itself, Liant recommends that a workspace file be stored in a directory that encompasses your working directory and source directories. Storing the .cwf file in the previous default location (“My Documents”) is still possible if your entire source tree is located here.

This release also corrected numerous minor problems and improved compatibility with Windows Vista and Windows Server 2008.

Features Added to Support XML Extensions

Version 12 of XML Extensions includes an XML Extensions - enabled RM/COBOL compiler that will generate and embed an XML-format symbol table in the COBOL object file. This feature eliminates the need to run a separate utility after compilation to create an XML Extensions model file. When the XML-format symbol table exists in the object, XML Extensions can use that symbol table at runtime and the symbol table is guaranteed to match the currently-running program. Several enhancements have been made to the RM/COBOL compiler to support this feature:

- The [COMPILER-OPTIONS configuration record](#) supports the following new keywords:
 - **KEEP-TEMP-XML-SYMBOL-TABLE-FILE**
This keyword specifies a path where the temporary file that contains the XML-format symbol table should be kept. By default, the temporary file is deleted after compression into the object file.
 - **SUPPRESS-XML-SYMBOL-TABLE**
With the value YES, this keyword specifies that the XML-format symbol table is not to be created; with the value NO, the keyword specifies that the XML-format symbol table is to be created. The default value is NO.

- The compiler supports two new environment variables:

- **RM_ENCODING**
This environment variable, available only on UNIX, specifies the encoding of characters in the source for purposes of translating them to Unicode in the XML symbol table. See [Environment Variables for UNIX](#) and the “UNIX Character Encoding” topic in the XML Extensions User’s Guide.
- **RM_KEEP_XML_SYMTAB_FILE**
This environment variable, available on both UNIX and Windows, specifies the path of the directory where the temporary XML-format symbol table file from the compiler should be preserved. (For further information, see [Environment Variables for UNIX](#) and [Environment Variables for Windows](#).) This environment variable provides an alternative to using the KEEP-TEMP-XML-SYMBOL-TABLE-FILE keyword of the COMPILER-OPTIONS configuration record (discussed previously in this section). If both the environment variable and the configuration keyword specify a pathname, the keyword value is used.

Note: The creation of the XML-format symbol table is controlled by the license. Without a license for XML Extensions, the XML-format symbol table is not created regardless of the setting of the SUPPRESS-XML-SYMBOL-TABLE keyword and cannot be preserved regardless of the setting of the SUPPRESS-XML-SYMBOL-TABLE keyword or RM_KEEP_XML_SYMTAB_FILE environment variable.

About Micro Focus

Micro Focus provides innovative software that allows companies to dramatically improve the business value of their enterprise applications. Micro Focus Enterprise Application Modernization and Management software enables customers' business applications to respond rapidly to market changes and embrace modern architectures with reduced cost and risk.

For additional information please visit: www.microfocus.com

www.microfocus.com

Micro Focus Worldwide

Australia.....	1800 632 626
Austria.....	0800 293 535
Belgium.....	0800 11 282
Canada.....	877-772-4450x1123
France	0800 835 135
Germany.....	0800 182 5443
Italy.....	800 784 420
Japan.....	+81 3 5793 8550
Luxembourg.....	800 23743
Netherlands	+31 23 5689 138
Norway	+47 22 91 07 20
Spain.....	+349 15 72 6699
Switzerland.....	0800 564 247
Sweden	+46 8 545 13 390
United Kingdom.....	0800 328 4967
United States	1 877 772 4450
Other Countries ...	+44 1635 32646

© 2008 Micro Focus. All Rights Reserved.
Micro Focus is a registered trademark.
Other trademarks are the property of
their respective owners.

DSRMCB1108-US