Enterprise COBOL for z/OS 6.4

Data Sheet

June 2024

This edition applies to Version 6.4 of IBM® Enterprise COBOL for z/OS® (program number 5655-EC6) and to all subsequent releases and modifications until otherwise indicated in new editions. Make sure that you are using the correct edition for the level of the product.

You can view or download softcopy publications in Enterprise COBOL for z/OS library. Enterprise COBOL for z/OS supports the continuous delivery (CD) model and publications are updated to document the features delivered under the CD model, it is a good idea to check for updates every couple of months.

Contents

Chapter 1. Summary of changes

This section lists the key changes that have been made to this document since Enterprise COBOL for z/OS 6.4 was released in May 2022.

June 2024

- Runtime APAR PH61133 (V2R4 or later): You can use ddname `IGZPROUT` at the run step of your JCL to generate a report of all COBOL 5 or later programs that are called dynamically, statically, and via DLL linkage. (Generating a report of COBOL programs)
- PH61700: The cjbuild utility in the COBOL/Java interoperability framework is simplified and improved. A "methods" file is not longer required. You can process all files in the COBOL artifact directory or just specify COBOL programs directly with the new `-i/--progid` option on the cjbuild command line. (cjbuild command reference)

April 2024

- Runtime APAR PH59864 (V2R5 or later): When PTFs for LE APAR PH56800 and COBOL Runtime APAR PH59864 are installed on z/OS 2.5 or later systems where COBOL programs run, the AMODE 31 subprograms called by the caller program can be either a DLL or a non-DLL. (Dynamic call between AMODE 31 and AMODE 64 programs)

February 2024

- PH59733: You can generate and parse JSON null values by using the `JSON GENERATE` and `JSON PARSE` statements. (JSON GENERATE statement and JSON PARSE statement)

December 2023

- PH58384: You can use the `NAME IS OMITTED` phrase to parse an anonymous JSON array in addition to an anonymous JSON object. (JSON GENERATE statement and JSON PARSE statement)

October 2023

- PH57297: You can use UTF-8 (`PIC U`) data items as the arguments to the `STRING` and `UNSTRING` statements. (STRING statement and UNSTRING statement)

  Note: COBOL Runtime LE APAR PH57264 (for AMODE 31) or APAR PH57265 (for AMODE 64) must also be applied on all systems where programs that make use of this new feature are linked or run.
- PH57397: You can use a function prototype to define the function name, parameters, and returning value of a user-defined function or other non-COBOL external functions such as C functions and invoke these functions. This is part of the 2014 COBOL Standard. (Using function prototypes)
- PH57398: You can use the `ENCODING` phrase of the `JSON GENERATE` and `JSON PARSE` statements to specify the encoding of the JSON document. (JSON GENERATE statement and JSON PARSE statement)

  Note: COBOL Runtime LE APAR PH57152 must also be applied on all systems where programs that make use of this new feature are linked or run.
- PH57400: You can use dynamic-length and UTF-8 (`PIC U`) data items as the arguments to the `JSON GENERATE` and `JSON PARSE` statements. (JSON GENERATE statement and JSON PARSE statement)

  Note: COBOL Runtime LE APAR PH57152 must also be applied on all systems where programs that make use of this new feature are linked or run.

August 2023

- PH56036 (AMODE 31) and PH56037 (AMODE 64): Added an optional alternate logic path for those who use dynamic access for VSAM files so that they can take advantage of zHyperLink. (VSAM dynamic access optional logic path)
- PH56142: For compiling under z/OS UNIX, added a new `cob2 -M` option to generate a make dependency file, `file.u`. This file contains entries for each copybook file that resides in a z/OS UNIX file system and is referenced in the COBOL source file. (cob2 syntax and options)

April 2023

- PH53631: Enhanced the `ON EXCEPTION` phrase support to deal with exceptions in the non-OO COBOL/Java interoperability framework. (Handling errors in Java-interoperable COBOL applications)

February 2023

- PH51752: A new sample JCL is provided to demonstrate how a non-OO COBOL/Java interoperable application can be built and run entirely using JCL. (Sample JCL for building and running the COBPROD application)

October 2022

- PH49715: A new cjbuild command reference section is added to address various usability and stability issues relating to the non-OO Java/COBOL interoperability. (cjbuild command reference)
- PH49967: The `NUMCHECK` compiler option is updated to avoid generating runtime checking code of zoned-decimal senders in MOVE statements when the receiver is an alphanumeric data item and NUMCHECK(ZON(LAX)) is in effect. (NUMCHECK)
- PH50296: The new `CONDCOMP` option is introduced to control how conditional code will be displayed in the listing. (CONDCOMP)

August 2022

- PH48453: New sample files demonstrating a COBOL/Java interoperable application and how to build it are provided in the demo subdirectory of the zFS install path. (Example: COBPROD application)
- PH48667: A problem is fixed for using figurative constant HIGH-VALUES with fixed byte-length UTF-8 data items of a length not a multiple of 4 bytes. (Using UTF-8-character figurative constants)

Chapter 2. Enable your COBOL applications to utilize new features of IBM z16

Enterprise COBOL is a premier enterprise class COBOL compiler for IBM z/OS. It delivers innovation for modernizing business-critical applications, programming features to increase programmer productivity, and bolsters the overall benefits of transactional and data systems such as IBM CICS®, IBM IMS, and IBM Db2®.

Enterprise COBOL for z/OS 6.4 delivers advanced compiler support to allow you to fully benefit from hardware advancements. The Enterprise COBOL for z/OS compiler is capable of unleashing the full power of IBM processors that are delivered in the various models of IBM Z hardware. Developers only need to focus on the logic of the applications and let the compiler determine the best way to transform and optimize the code generation for the IBM Z hardware on which the application will run.

With its enhanced capabilities, simplified programming, and increased programmer productivity features, you can use Enterprise COBOL for z/OS to modernize existing business-critical applications. You can deliver new enhancements quicker, with less cost and with lower risks. You can add modern graphical user interfaces to business-critical COBOL applications or extend them to work with web, cloud, or mobile infrastructures. With the investment in new compiler technology and the continued delivery of new features, Enterprise COBOL for z/OS 6.4 reaffirms IBM's commitment to COBOL on z/OS. You gain the benefit of new investments that are combined with more than 60 years of IBM experience in compiler innovation and development.

Chapter 3. Highlights

Enterprise COBOL for z/OS 6.4 delivers the following new and improved features:

- Support of IBM z16 to maximize your hardware investment, reduce CPU usage, and improve performance of critical COBOL applications
- Improved Java™/COBOL interoperability to easily extend the capabilities of your COBOL applications with Java
- Interoperability between AMODE 31 (31-bit) and AMODE 64 (64-bit) COBOL programs to handle your growing COBOL program data without converting the entire application to run in AMODE 64
- Support for user-defined functions to enable you to write your own functions and invoke them like intrinsic functions, improving code modularity and maintainability
- Support for function prototypes to enable you to define the function name, parameters, and returning value of a user-defined function or other non-COBOL external functions such as C functions and invoke these functions
- Improved integration with IBM Automatic Binary Optimizer for z/OS to invest in your future so that modules you compile today can take advantage of future IBM Z hardware enhancements, without having to be recompiled

Support of IBM z16 to maximize your hardware investment, reduce CPU usage, and improve performance of critical COBOL applications

Enterprise COBOL for z/OS 6.4 incorporates leading-edge code generation and optimization technology to maximize hardware utilization and to help improve application performance.

- Enterprise COBOL for z/OS 6.4 adds support for the new Vector Packed Decimal Enhancement Facility 2 in IBM z16 through the new ARCH(14) compiler option. No source changes are required to take advantage of this new facility; just recompile with ARCH(14) to target IBM z16.
- This new facility adds performance improvements for COBOL programs that contain one or more of the following types of statements:
  - Exponentiation operations on packed or zoned decimal data items where the exponent is declared with one or more fractional digits
  - Arithmetic statements involving mixed decimal and floating-point data items
  - Statements using numeric-edited data items

Improved Java/COBOL interoperability to easily extend the capabilities of your COBOL applications with Java

Enterprise COBOL for z/OS 6.4 helps simplify interoperability between your COBOL and Java applications so that you can easily extend your COBOL applications with Java. Java is a popular language that is familiar to your enterprise-wide developers, including those newly hired.

- Removes the need to write object-oriented COBOL and reduces the number of manual JNI calls required compared to COBOL 6.3.
  - Compared to COBOL 6.3, COBOL 6.4 automatically handles a wider variety of interoperation scenarios between COBOL and Java, so the COBOL/Java interoperability feature in COBOL 6.4 reduces the need for users to manually make JNI calls from their programs.
- Three COBOL/Java communication features are provided:
  - Enabling a COBOL program to be callable from Java
  - Enabling the CALL statement in COBOL to call a static Java method
  - Enabling Java applications to easily access the working-storage memory of a COBOL program

- Interoperate existing AMODE 31 (31-bit) COBOL applications and Java applications without converting the entire COBOL application to run in AMODE 64 (64-bit) or using AMODE 31 (31-bit) Java.
  - Interoperate existing AMODE 31 COBOL applications and AMODE 64 Java applications.
  - No need to convert your entire COBOL application to run in AMODE 64.
  - Use of this feature requires IBM Language Environment® (LE) for z/OS 2.3 or 2.4 with APAR PH28966, or LE for z/OS 2.5.
  - Client applications that previously used the AMODE 31 Java SDK might need to be modified to run in AMODE 64 mode. Only AMODE 64 versions are available with the Java SDK version 11, known as IBM Semeru Certified Edition for z/OS 11.
  - Use of these features requires IBM SDK for z/OS, Java Technology Edition 8.0.6.36 (JVM), IBM Semeru Certified Edition for z/OS 11.0.14.1 or later.

PH48453: New sample files demonstrating a COBOL/Java interoperable application and how to build it are provided in the demo subdirectory of the zFS install path.

PH49715: A new cjbuild command reference section is added to address various usability and stability issues relating to the non-OO Java/COBOL interoperability.

PH51752: A new sample JCL is provided to demonstrate how a non-OO COBOL/Java interoperable application can be built and run entirely using JCL.

PH61700: The cjbuild utility is enhanced to allow users to specify the names of COBOL programs in the command line and process all files in the COBOL artifact directory.

Interoperability between AMODE 31 (31-bit) and AMODE 64 (64-bit) COBOL programs to handle your growing data without converting the entire application to run in AMODE 64

Enterprise COBOL for z/OS 6.4 provides support for creating AMODE 64 COBOL applications that can interoperate with your existing AMODE 31 COBOL applications. AMODE 64 COBOL applications can access data items greater than the existing AMODE 31 data size limits, without changes to the program logic.

- An AMODE 64 COBOL program can access data stored in the address space above 2 GB (up to 16 EB), extending the available space for your growing data. With AMODE 31 COBOL, this storage is limited to the address space below 2 GB.
- Removes the need to convert the entire COBOL application to run in AMODE 64 immediately. Gradually convert AMODE 31 applications to AMODE 64.
- AMODE 31 COBOL programs can call AMODE 64 COBOL programs and AMODE 64 COBOL programs can call AMODE 31 COBOL program using dynamic calls. When PTFs for LE APAR PH56800 and COBOL Runtime APAR PH59864 are installed on z/OS 2.5 or later systems where COBOL programs run, the AMODE 31 subprograms called by the caller program can be either a DLL or a non-DLL.
- The compiler and the COBOL runtime library take advantage of a new Language Environment feature to manage AMODE switching.
- This LE feature allows a DLL subprogram to have a different AMODE from its caller. Existing applications running in AMODE 31 using dynamic calls can take advantage of this feature with minimal or no changes.

Support for user-defined functions to enable you to write your own functions and invoke them like intrinsic functions, improving code modularity and maintainability

Write your own functions using the new Enterprise COBOL construct, the user-defined function definition, and invoke them like intrinsic functions. As with many popular programming languages, COBOL 6.4 supports user-defined functions, which gives new COBOL programmers a familiar structure.

- User-defined functions can only be invoked statically and must be defined within the same sequence of programs as the programs that invoke them.

- User-defined functions is a COBOL 2002 standard feature. Support of programming language standards provides you with additional functionality so that you can modernize your application. It also allows for maximum portability of your source code among a variety of compiler implementations.

Support for function prototypes to enable you to define the function name, parameters, and returning value of a user-defined function or other non-COBOL external functions such as C functions and invoke these functions

As a COBOL 2014 standard feature, function prototypes enable you to invoke COBOL user-defined functions in the following ways:

- Enable you to invoke COBOL user-defined functions or non-COBOL external functions such as C functions using the `FUNCTION` keyword.
- Enable you to invoke user-defined functions whose definitions are written in other files such as compilation groups.
- Enable you to invoke user-defined functions with a linkage interface (such as STATIC, DYNAMIC, or DLL) that was specified on the prototype using the `ENTRY-INTERFACE` phrase of the `FUNCTION-ID` paragraph.

Improved integration with IBM Automatic Binary Optimizer for z/OS to invest in your future so that modules you compile today can take advantage of future IBM Z® hardware enhancements, without having to be recompiled

- Automatic Binary Optimizer for z/OS (sold separately) improves the performance of already-compiled COBOL program modules without recompiling, source code migration, or performance tuning.
- COBOL 6.4 generates binary metadata that is designed to allow modules compiled with COBOL 6.4 today to be easily optimized in the future by Automatic Binary Optimizer for z/OS.
- Use Enterprise COBOL for z/OS 6.4 for new development, modernization, and maintenance. Use Automatic Binary Optimizer for z/OS to improve the performance of COBOL modules that are stable and do not need any source changes.

Chapter 4. Other Enterprise COBOL for z/OS features

Additional enhancements

- Enterprise COBOL for z/OS 6.4 adds support for building and running COBOL applications for the z/OS 2.5 operating system.
- Default ARCH changed to ARCH(10) [IBM zEC12, IBM zBC12] and support removed for ARCH(8) [IBM z10EC, IBM z10BC] and ARCH(9) [IBM z196, IBM z114].
- The following enhancement is introduced in Enterprise COBOL 6.4 via the service stream:
  - PH48667: A problem is fixed for using figurative constant HIGH-VALUES with fixed byte-length UTF-8 data items of a length not a multiple of 4 bytes.
  - PH49967: The `NUMCHECK` compiler option is updated to avoid generating runtime checking code of zoned decimal senders in MOVE statements when the receiver is an alphanumeric data item and NUMCHECK(ZON(LAX)) is in effect.
  - PH56036 (AMODE 31) and PH56037 (AMODE 64): An optional alternate logic path is added for those who use dynamic access for VSAM files so that they can take advantage of zHyperLink.
  - PH56142: For compiling under z/OS UNIX, a new `cob2 -M` option is added to generate a make dependency file, `file.u`. This file contains entries for each copybook file that resides in a z/OS UNIX file system and is referenced in the COBOL source file.
  - PH57297: You can use UTF-8 (`PIC U`) data items as the arguments to the `STRING` and `UNSTRING` statements.

    Note: COBOL Runtime LE APAR PH57264 (for AMODE 31) or APAR PH57265 (for AMODE 64) must also be applied on all systems where programs that make use of this new feature are linked or run.
  - PH57398: You can use the `ENCODING` phrase of the `JSON GENERATE` and `JSON PARSE` statements to specify the encoding of the JSON document.

    Note: COBOL Runtime LE APAR PH57152 must also be applied on all systems where programs that make use of this new feature are linked or run.
  - PH57400: You can use dynamic-length and UTF-8 (`PIC U`) data items as the arguments to the `JSON GENERATE` and `JSON PARSE` statements.

    Note: COBOL Runtime LE APAR PH57152 must also be applied on all systems where programs that make use of this new feature are linked or run.
  - PH58384: You can use the `NAME IS OMITTED` phrase to parse an anonymous JSON array in addition to an anonymous JSON object.
  - PH59733: You can generate and parse JSON null values by using the `JSON GENERATE` and `JSON PARSE` statements.
  - Runtime APAR PH61133 (V2R4 or later): You can use ddname `IGZPROUT` at the run step of your JCL to generate a report of all COBOL 5 or later programs that are called dynamically, statically, and via DLL linkage.

Ease into migration

You can visit the COBOL Migration Portal for all Enterprise COBOL for z/OS migration-related information, including case studies, COBOL experts interview videos, the cloud-based COBOL Migration Assistant for a navigation through the migration process, COBOL Migration and Performance Tuning Webinars, FAQs, other IBM products to support your migration, and many other resources, which help ease your migration efforts from COBOL 4 or earlier to COBOL 6 compiler.

If you have programs compiled with OS/VS COBOL, VS COBOL II, COBOL/370, IBM COBOL for MVS™ & VM, or IBM COBOL for OS/390® & VM, or if your Enterprise COBOL for z/OS programs are compiled with

the CMPR2 compiler option, they may be written in 1968 and 1974 COBOL Standard. You need to convert them into 1985 COBOL Standard programs before they can be compiled with Enterprise COBOL for z/OS 6.

IBM File Manger for z/OS includes View Load Module that can help identify which of your programs were compiled with the older compilers. You can convert 1968 and 1974 COBOL Standard programs to 1985 COBOL Standard programs using the COBOL conversion tool (CCCA) included in IBM Debug for z/OS 14.2 or earlier.

Support for modern development tools

IBM Developer for z/OS (formerly IBM Developer for z Systems® and Rational® Developer for z Systems) supports Enterprise COBOL and helps improve the productivity of COBOL developers. IBM Developer for z/OS provides an interactive, workstation-based environment to help you create, maintain, and reuse applications. IBM Developer for z/OS includes support for traditional development using COBOL, but also has the ability to generate web services interfaces from COBOL constructs to ease creation of web services from existing COBOL applications.

IBM Developer for z/OS provides a workstation interface to IBM Debug for z/OS, and is also integrated with IBM File Manager for z/OS and IBM Fault Analyzer for z/OS. File Manager integration enables you to access Keyed Sequence Data Set (KSDS) files from the IBM Developer for z/OS workbench, and gives you the ability to browse and update data sets. By integrating with Fault Analyzer, IBM Developer for z/OS enables you to browse Fault Analyzer ABEND reports on CICS, IMS, batch, Java, WebSphere®, and other run times.

COBOL across platforms

Enterprise COBOL for z/OS is part of a family of compatible compilers, application development tools, and maintenance tools.

Chapter 5. System requirements

The following table presents the system requirements for Enterprise COBOL for z/OS 6.4.

Table 1. System requirements for Enterprise COBOL for z/OS 6.4

Software	**Hardware**
Enterprise COBOL for z/OS 6.4 runs under the control of, or in conjunction with, the currently supported releases of the following programs and their subsequent releases or their equivalents. For more information, see the Program Directory and the preventive service planning (PSP) bucket. • z/OS V2.3 (5650-ZOS), or later is required. • For installation on z/OS, z/OS SMP/E is required. • For customization during or after installation, z/OS High Level Assembler is required. • Enterprise COBOL XML PARSE statements in programs, which are compiled with the XMLPARSE(XMLSS) compiler option, require z/OS XML System Services 2.3 (5650-ZOS), or later. • The new COBOL/Java interoperability feature available in Enterprise COBOL for z/OS 6.4 requires IBM SDK for z/OS, Java Technology Edition 8.0.6.36 (JVM), IBM Semeru Certified Edition for z/OS 11.0.14.1 or later.	Enterprise COBOL for z/OS 6.4 runs on and generates code that runs on the following IBM Z servers: • IBM z16 • IBM z15® Models T01 and T02 • IBM z14® Models M01-M05 • IBM z14 Model ZR1 • IBM z13® • IBM z13s® • IBM zEnterprise® EC12 • IBM zEnterprise BC12

Optional licensed programs

Depending on the functions used, you might require other software products such as CICS, Db2, or IMS. For a list of compatible software, see the Software Product Compatibility Reports (SPCR) website.

From the SPCR website, click Create a Report under in-depth reports, search for Enterprise COBOL for z/OS, choose Version 6.4, and then click submit.

Chapter 6. For more information

To learn more about IBM Enterprise COBOL for z/OS 6.4, contact your IBM representative or IBM Business Partner, or visit the Enterprise COBOL for z/OS product page.

Chapter 7. Notices

References in this document to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this publication is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Product Number: 5655-EC6