IBM WebSphere Development Studio for i5/OS ILE RPG and COBOL Compiler Enhancements V6R1M0

© 2006 IBM Corporation

Overview

- **ILE RPG Enhancements**
- **ILE COBOL Enhancements**
- **One enhancement that affects both ILE RPG and ILE COBOL**

ILE RPG enhancements - overview

Major enhancements for ILE RPG include

- **Ability to run concurrently in multiple threads**
- **Enhancements related to the use of files, including defining files locally in subprocedures, and passing files as parameters**
- **Significantly higher limits for the size of variables (16MB), the number of elements in an array (over 16 million), and the length of string literals**
- **Relaxation of some UCS-2 rules (available starting in V5R3 through PTFs).**

ILE RPG enhancements - threads

Ability to run concurrently in multiple threads:

When THREAD(*CONCURRENT) is specified on the Control specification of a module

- Multiple threads can run in the module at the same time.
- By default, static variables will be defined so that each thread will have its own copy of the static variable.
- Individual variables can be defined to be shared by all threads using STATIC(*ALLTHREAD).
- Individual procedures can be serialized so that only one thread can run them at one time, by specifying SERIALIZE on the Procedure-Begin specification.

ILE RPG enhancements – main without cycle

Ability to define a main procedure which does not use the RPG cycle

- **MAIN keyword on the H specification designates one subprocedure as being the main procedure, that is, the procedure that gets control when the program gets called.**
- **Other than being the program-entry procedure, the main subprocedure is like any other subprocedure. It does not use the RPG cycle.**
- **The prototype for the main subprocedure must have the EXTPGM keyword; the main subprocedure can only be called by a program call.**

ILE RPG enhancements – local files

Files defined in subprocedures

- **Local F specifications follow the Procedure-begin specification and precede the Definition specifications.**
- **I/O to local files can only be done with data structures. There are no I and O specifications for local files.**
- **By default, the storage associated with local files is automatic; the file is closed when the subprocedure returns normally or abnormally.**
- **The STATIC keyword can be used to indicate that the storage associated with the file is static, so that all invocations of the procedure will use the same file. If the file is open when the procedure returns, it will remain open for the next call to the procedure.**

ILE RPG enhancements – qualified formats

Qualified record formats:

- **When a file is defined with the QUALIFIED keyword, the record formats must be qualified by the file name, MYFILE.MYFMT.**
- **Qualified files do not have I and O specifications generated by the compiler; I/O can only be done through data structures.**
- **When files are qualified, the names of the record formats can be the same as the formats of other files. For example, you can have FILE1.FMT1 and FILE2.FMT1.**

ILE RPG enhancements – LIKEFILE

Files defined like other files:

- **Using the LIKEFILE keyword, a file can be defined to use the same settings as another File specification.**
- **If the file is externally-described, the QUALIFIED keyword is implied. I/O to the file can only be done through data structures.**

ILE RPG enhancements – file parameters

Files passed as parameters:

- **A prototyped parameter can be defined as a File parameter using the LIKEFILE keyword.**
- **Any file related by LIKEFILE keywords to the same original File specification may be passed as a parameter to the procedure.**
- **Within the called procedure or program, all supported operations can be done on the file parameter. However, I/O to the file parameter can only be done through data structures.**
- **RPG file parameters are** not **compatible with file parameters in other languages such as C or COBOL.**

ILE RPG enhancements – avoid compile-time overrides (F)

EXTDESC keyword and EXTFILE(*EXTDESC)

- **The EXTDESC keyword identifies the file to be used for an externally-described file** at compile time**.**
- **The filename is specified as a literal in one of the forms 'LIBNAME/FILENAME' or 'FILENAME'.**
- **This removes the need to provide a compile-time override for the file.**
- **EXTDESC can be used to solve the problem where a record format has the same name as the file. When EXTDESC is used to locate the file, then any name can be used as the RPG-internal name for the file.**
- **The EXTFILE keyword is enhanced to allow the special value *EXTDESC, indicating that the file specified by EXTDESC is also to be used at runtime.**

ILE RPG enhancements – avoid compile-time overrides (D)

EXTNAME can specify library and file for an externally-described DS:

- **The EXTNAME keyword is enhanced to allow a literal to specify the library and name for the external file.**
- **EXTNAME('LIBNAME/FILENAME') or EXTNAME('FILENAME') are supported.**
- **This removes the need to provide a compile-time override for the file.**

ILE RPG enhancements – result DS for EXFMT

EXFMT allows a result data structure:

- **The EXFMT operation is enhanced to allow a data structure to be specified in the result field.**
- **The data structure must be defined with usage type *ALL**

```
EXTNAME(file : fmt : *ALL)
```

or

```
LIKEREC(fmt : *ALL)
```

ILE RPG enhancements – larger fields

Larger limits for data structures, and character, UCS-2 and graphic variables:

- **Data structures can have a size up to 16,773,104.**
- **Character definitions can have a length up to 16,773,104. (The limit is 4 less for variable length character definitions.)**
- **The LEN keyword can be used instead of the Length entry to define the length of a variable or DS**
- **UCS-2 and Graphic definitions can have a length up to 8,386,552 double-byte characters. (The limit is 2 less for variable length definitions.)**
- **The VARYING keyword allows a parameter of either 2 or 4 indicating the number of bytes used to hold the length prefix. A parameter of 4 is required if the defined length is over 65535.**

ILE RPG enhancements – more array elements

Larger limit for DIM and OCCURS

- **There is no arbitrary limit on the number of elements in an array or occurrences in a multiple-occurrence data structure.**
- **The limit on the total size of an array or structure remains the same; it cannot be larger than 16,773,104 bytes.**
- **For example**
  - If the elements of an array are 1 byte in size, the maximum DIM for the array is 16,773,104.
  - If the elements of an array are 10 bytes in size, the maximum DIM for the array is 1,677,310 (16773104/10).

ILE RPG enhancements – longer literals

Larger limits for character, UCS-2 and DBCS literals:

- **Character literals can now have a length up to 16380 characters.**
- **UCS-2 literals can now have a length up to 8190 UCS-2 characters.**
- **Graphic literals can now have a length up to 16379 DBCS characters.**

ILE RPG enhancements - %ADDR for varying data

%ADDR(varying_field : *DATA)

- **The %ADDR built-in function is enhanced to allow *DATA as the second parameter to obtain the address of the data part of a variable length field.**

ILE RPG enhancements – TEMPLATE definitions

TEMPLATE keyword for files and definitions:

- **The TEMPLATE keyword can be coded for file and variable definitions to indicate that the name will only be used with the LIKEFILE, LIKE, or LIKEDS keyword to define other files or variables.**
- **Template definitions are useful when defining parameter types for prototyped calls, since the compiler only uses them at compile time to help define other files and variables, and does not generate any code related to them.**
- **Template data structures can have the INZ keyword coded for the data structure and its subfields, which will ease the use of INZ(*LIKEDS).**

ILE RPG enhancements – new UCS-2 rules

Relaxation of some UCS-2 rules

- **The compiler will perform some implicit conversion between character, UCS-2 and graphic values, making it unnecessary to code %CHAR, %UCS2 or %GRAPH in many cases. This enhancement is also available through PTFs for V5R3 and V5R4.**
- **Implicit conversion is now supported for**
  - Assignment using EVAL and EVALR
  - Comparison operations in expressions and using fixed form operations IFxx, DOUxx, DOWxx, WHxx, CASxx, CABxx, COMP.
  - Note that implicit conversion was already supported for the conversion operations MOVE and MOVEL.
- **UCS-2 variables can now be initialized with character or graphic literals without using the %UCS2 built-in function.**
- **This enhancement is available in V5R3 and V5R4 with PTFs:**
  - `V5R3M0 TGTRLS(*CURRENT):  SI24532`
  - `V5R4M0 TGTRLS(*CURRENT):  SI26312`
  - `V5R4M0 TGTRLS(*PRV)    :  SI25232`

ILE RPG enhancements – reduce module size

Eliminate unused variables from the compiled object:

- **New values *UNREF and *NOUNREF are added to the OPTION keyword for the CRTBNDRPG and CRTRPGMOD commands, and for the OPTION keyword on the Control specification.**
- **The default is *UNREF.**
- ***NOUNREF indicates that unreferenced variables should not be generated into the RPG module. This can reduce program size, and if imported variables are not referenced, it can reduce the time taken to bind a module to a program or service program.**

ILE COBOL enhancements - overview

Major enhancements for ILE COBOL include

- **The ability to use the Unicode CCSID 1200**
- **Support for the system debugger to allow debugging of complex OCCURS DEPENDING ON arrays.**

ILE COBOL enhancements – more Unicode CCSIDs

National UCS-2 CCSID enhancement

- **The NTLCCSID parameter has been added to the CRTCBLMOD and CRTBNDCBL commands, and to the PROCESS statement, to allow you to specify the UCS-2 CCSID to be used for National data items.**
- **With this parameter you can specify a CCSID other than the default 13488, such as CCSID 1200, to be used for National items.**

ILE COBOL enhancements – debugging ODO arrays

Debugging complex OCCURS DEPENDING ON arrays

- **The system debugger and the WDSC debugger will now be able to debug complex OCCURS DEPENDING ON arrays.**

ILE COBOL enhancements – allow larger modules

Support for a very large number of data items in a COBOL module

- **The ILE COBOL compiler can now support more data items than it could in previous releases.**
- **This support is also available in V5R3M0 and V5R4M0 with PTFs:**
  - `V5R3M0:  SI27834`
  - `V5R4M0:  SI26591`

Store parameter information in the program

Both the ILE RPG and ILE COBOL compiler are enhanced to allow information about the parameters for the program or procedures to be stored in the program.

- The information is in the form of Program Call Markup Language (PCML). Starting in V5R2, the ILE RPG and ILE COBOL compilers were able to output PCML to a stream file.
- Starting in V6R1, the PCML can also be placed directly in the module.
- The information can later be retrieved with the new QBNRPII API.

Store parameter information in program - commands

- **The PGMINFO command parameter for the CRTRPGMOD, CRTCBLMOD, CRTBNDRPG and CRTBNDCBL commands is enhanced to specify where the PCML is to go.**
  - The default location is the stream file specified by the INFOSTMF parameter
  - PGMINFO(*PCML:*MODULE) says to place the PCML information directly in the module. The PCML information becomes part of the program or service program containing the module.
  - PGMINFO(*PCML:*ALL) says to place the PCML information both in the module and in the INFOSTMF stream file.

Store parameter information in program – source files

H spec keyword for RPG or PROCESS option for COBOL

- **The PGMINFO command parameter can be augmented or overridden by an H spec keyword (ILE RPG) or PROCESS option (ILE COBOL).**
  - RPG: `PGMINFO(*PCML:*MODULE) or PGMINFO(*NO)`
  - COBOL: `PGMINFO(PCML MODULE) or PGMINFO(NOPGMINFO)`
- **If the source keyword specifies “module”, then it augments the PGMINFO command parameter. For example, if the command requested *STMF, and the source keyword specifies *MODULE, then the PCML will be generated both to the stream file and into the module.**
- **If the keyword specifies “no”, then it overrides the PGMINFO command parameter. No matter what was specified by the command parameter, no PCML information will be generated by the compiler.**

Store parameter information in program – V5R4 PTFs

PTF support for V5R4

- **Part of this support is available in V5R4 with PTFs.**
  - The H specification keyword for ILE RPG
  - The PROCESS option for ILE COBOL
  - The QBNRPII API
- **The following V5R4M0 PTFs will provide the various parts of this function**
  - `5722SS1 SI23544 (QBNRPII API)`
  - `5722SS1 SI27064 (Support for compilers)`
  - `5722WDS SI27061 (ILE RPG compiler PTF 1)`
  - `5722WDS SI27065 (ILE RPG compiler PTF 2)`
  - `5722WDS SI27154 (ILE COBOL compiler)`

Trademarks and Disclaimers

© IBM Corporation 1994-2007. All rights reserved.

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.
Trademarks of International Business Machines Corporation in the United States, other countries, or both can be found on the World Wide Web at http://www.ibm.com/legal/copytrade.shtml.

Adobe, Acrobat, PostScript and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.
UNIX is a registered trademark of The Open Group in the United States and other countries.
Cell Broadband Engine and Cell/B.E. are trademarks of Sony Computer Entertainment, Inc., in the United States, other countries, or both and are used under license therefrom.
Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Other company, product, or service names may be trademarks or service marks of others.

Information is provided "AS IS" without warranty of any kind.

The customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Some information addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in IBM product announcements. The information is presented here to communicate IBM's current investment and development activities as a good faith effort to help with our customers' future planning.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Prices are suggested U.S. list prices and are subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.