What’s new in RPG 7.5

Barbara Morris

RPG Compiler Development

IBM

IBM i Anywhere
IBM i Everywhere

Agenda

- What's new in 7.5 only
- What's brand new in May 2022 with 7.3, 7.4, and 7.5 PTFs
- Some older enhancements

See the end of this PDF for useful links

What’s new in 7.5 only

Support for the BOOLEAN datatype

IBM i Anywhere
IBM i Everywhere

7.5
- Spring 2022

The RPG compiler supports a database field defined as type BOOLEAN.

The field is treated as an indicator by RPG.

```
create table mylib/booltable
       (boolfld boolean not null with default)
```

DSPFFD shows:

```
Field Level Information
                Data          Field  Buffer    Buffer          Field    Column
  Field         Type         Length  Length  Position          Usage    Heading
  BOOLFLD       BOOLEAN           1       1         1          Both     BOOLFLD
```

Support for the BOOLEAN datatype

IBM i Anywhere
IBM i Everywhere

7.5
- Spring 2022

RPG program using the Boolean field as an indicator:

```
dcl-f booltable;

   read booltable;
   if not boolfld; // It is an indicator, no need to compare to *on or *off
   ...

   boolfld = *on;
```

The cross reference from the compile listing:

```
Global Field References:
   Field                Attributes
   BOOLFLD              N(1)
```

What's brand new with 7.3, 7.4 & 7.5 PTFs in May 2022

SND-MSG opcode

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3, 7.4, 7.5
- Spring 2022

SND-MSG sends a message to the joblog

You can send an informational message to the current procedure

```
SND-MSG 'hello';        // *INFO is the default
SND-MSG *INFO 'hello';
```

You can send an escape message to your calling procedure

```
SND-MSG *ESCAPE 'hello';
```

You can send a specific message ID from a message file

```
SND-MSG %MSG('ABC1234' : 'MYLIB/MYMSGF');
```

Your message can have replacement text

```
SND-MSG *ESCAPE %MSG('ERR1234' : 'MYLIB/MYMSGF' : err1234ReplText);
```

More ...

SND-MSG opcode

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3, 7.4, 7.5 - Spring 2022

- You can send the message explicitly to the current procedure

```
SND-MSG *ESCAPE 'hello' %TARGET(*SELF);
```

- You can send the message explicitly to your caller

```
SND-MSG 'Complete' %TARGET(*CALLER);
```

- You can send the message to a specific program or procedure on the call stack

```
SND-MSG *ESCAPE
        %MSG('ERR1234' : 'MYLIB/MYMSGF' : err1234ReplText)
        %TARGET('myMainProcedure');
```

- You can send the message relative to a specific program or procedure on the call stack

```
SND-MSG *ESCAPE
        %MSG('ERR1234' : 'MYLIB/MYMSGF' : err1234ReplText)
        %TARGET('myMainProcedure': 1);
```

SND-MSG: %MSG

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3, 7.4, 7.5 - Spring 2022

```
%MSG(msgid : msgfile : replacementText)
```

- The message id and message file parameters are required
- The replacement text is optional

For simple replacement text, such as for CPF9898 in QCPFMSG, which just has one replacement value defined as *CHAR length 132, you can just pass a character string.

```
snd-msg %msg('CPF9898' : 'QCPFMSG' : 'Hello ' + firstName);
```

%MSG: complex replacement

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3, 7.4, 7.5 - Spring 2022

Message MSG0002: "Item '&1' price is &2, quantity is &3."

Field	Data Type	Length	Decimal Positions
&1	*CHAR	20	
&2	*DEC	5	2
&3	*DEC	3	0

Define a data structure with a subfield for each value:

```
dcl-ds msg0002 qualified;
   item char(20);
   price packed(5:2);
   quantity packed(3);
end-ds;
```

Set the replacement values in the data structure, and send the message to the joblog.

```
msg0002.item = 'large chair';
msg0002.price = 39.95;
msg0002.quantity = 15;
snd-msg %msg('MSG0002' : 'MYLIB/MYMSGF' : msg0002);
```

Joblog: Item 'large chair' price is 39.95, quantity is 15.'

SND-MSG: %TARGET

IBM i Anywhere
IBM i Everywhere

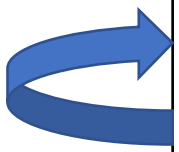
PTFs for 7.3, 7.4, 7.5
- Spring 2022

```
%TARGET(programOrProcedureOnStack : stackOffset)
```

1. A program or procedure on the call stack
2. (optional) How many procedures earlier on the call stack to actually send the message

Example:

- My main procedure calls P1 which calls P2
- I want SND-MSG to send a message from P2 to the caller of my main procedure
- I debug my program and set a breakpoint in P2
- DSPJOB OPTION(*PGMSTK)
- I want to send the message to TEST
- But it might be any program
- I send to procedure SNDTST, offset 2
- %TARGET('SNDTST' : 2)



Program	Library	Statement	Procedure
TEST	BMORRIS		_QRNP_PEP_TEST
TEST	BMORRIS	11	TEST
SNDTST	BMORRIS		_QRNP_PEP_SNDTST
SNDTST	BMORRIS	4	SNDTST
SNDTST	BMORRIS	7	P1
SNDTST	BMORRIS	14	P2

SND-MSG opcode

PTFs for 7.3, 7.4, 7.5 - Spring 2022

The joblog shows

- The procedure and statement number with the SND-MSG
- The procedure and statement number for the target

```
dcl-s status packed(5);
snd-msg 'Message 1, info, default target';
p1();
return;
dcl-proc p1;
   callp(e) p2();
   status = %status();
   snd-msg *escape 'Message 3, procedure p2 failed, status ' + %char(status)
                   %target(*caller : 2); // allow for PEP proc
end-proc;
dcl-proc p2;
   snd-msg *escape 'Message 2, escape, default target';
end-proc;
```

SND-MSG opcode

PTFs for 7.3, 7.4, 7.5 - Spring 2022

```
snd-msg 'Message 1, info, default target';
```

```
Message . . . . :   Message 1, info, default target

From program . . . . . . . . . :   SNDMSGXMP
  From library . . . . . . . . :     BMORRIS
  From module  . . . . . . . . :     SNDMSGXMP
  From procedure . . . . . . . :     SNDMSGXMP
  From statement . . . . . . . :     3

To program . . . . . . . . . . :   SNDMSGXMP
  To library . . . . . . . . . :     BMORRIS
  To module  . . . . . . . . . :     SNDMSGXMP
  To procedure . . . . . . . . :     SNDMSGXMP
  To statement . . . . . . . . :     3
```

The message was sent to the same statement as the SND-MSG.

SND-MSG opcode

PTFs for 7.3, 7.4, 7.5
- Spring 2022

```
dcl-proc p1;
   callp(e) p2();
   ...
dcl-proc p2;
   snd-msg *escape 'Message 2, escape, default target';
end-proc;
```

```
      Message . . . . :   Message 2, escape, default target.

      From program . . . . . . . . . :   SNDMSGXMP
        From library . . . . . . . . :     BMORRIS
        From module  . . . . . . . . :     SNDMSGXMP
        From procedure . . . . . . . :     P2
        From statement . . . . . . . :     14

      To program . . . . . . . . . . :   SNDMSGXMP
        To library . . . . . . . . . :     BMORRIS
        To module  . . . . . . . . . :     SNDMSGXMP
        To procedure . . . . . . . . :     P1
        To statement . . . . . . . . :     7
```

The message was sent
- from statement 14 of p2
- to statement 7 of p1 (the statement with the call to p2)

SND-MSG opcode

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3, 7.4, 7.5
- Spring 2022

```
dcl-proc p1;
   ...
   snd-msg *escape 'Message 3, procedure p2 failed, status ' + %char(status)
                   %target(*caller : 2); // allow for PEP proc
end-proc;
```

```
      Message . . . . :   Message 3, procedure p2 failed with status 202.

      From program . . . . . . . . . :   SNDMSGXMP
        From library . . . . . . . . :     BMORRIS
        From module  . . . . . . . . :     SNDMSGXMP
        From procedure . . . . . . . :     P1
        From statement . . . . . . . :     9

      To program . . . . . . . . . . :   QUOCMD
        To library . . . . . . . . . :     QSYS
        Instruction  . . . . . . . . :     03B3
```

The message was sent
- from statement 9 of p1
- to the command line

ON-EXCP opcode

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3, 7.4, 7.5 - Spring 2022

ON-EXCP lets you monitor for specific messages

```
MONITOR;
   do_something ();
ON-EXCP 'ABC1234' : 'DEF2345';
   error_occurred = *ON;
   process_error (psds.errmsgid);
ENDMON;
```

You can code more than one ON-EXCP section:

```
MONITOR;
   do_something ();
ON-EXCP 'ABC1234' : 'DEF2345';
   ...
ON-EXCP(C) 'GHI1234';  // only monitor for messages sent to this procedure
   ...
ENDMON;
```

ON-EXCP opcode

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3, 7.4, 7.5 - Spring 2022

ON-EXCP and ON-ERROR can be in the same MONITOR group.

ON-EXCP must be coded before ON-ERROR.

```
MONITOR;
   do_something ();
ON-EXCP 'ABC1234' : 'DEF2345';
   error_occurred = *ON;
   process_error (psds.errmsgid);
ON-ERROR 100 : 202;
   snd-msg *escape 'Unexpected error!';
ENDMON;
```

ON-EXCP opcode: (C) extender

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3, 7.4, 7.5 - Spring 2022

Code (C) if you only want to handle messages sent to the current procedure.

```
MONITOR;
   doSomething ();
ON-EXCP(C) 'GHI3456' : 'DEF2345';
   snd-msg 'escape msg to me';
ON-EXCP 'ABC1234' : 'DEF2345';
   snd-msg 'escape msg to something I called';
ON-ERROR 100 : 202;
   snd-msg *escape 'Unexpected error!';
ENDMON;
```

```
dcl-proc doSomething;
   someOtherProc();
end-proc;

dcl-proc someOtherProc;
   if x > 10;
      snd-msg *escape %msg('DEF2345':'MYMSGF');
   elseif y > 20;
      snd-msg *escape %msg('DEF2345':'MYMSGF' )
                      %target(*caller : 1);
   endif;
end-proc;
```

- If x > 10, it sends DEF2345 to its caller, someOtherProc.
  - ✓ ON-EXCP catches it

- If x < 10 and y > 20, it sends DEF2345 to its caller's caller, which is the procedure with the MONITOR.
  - ✓ ON-EXCP(C) catches it

Runtime PTF for the new built-ins

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3, 7.4, 7.5 - Spring 2022

Warning

You need a runtime PTF on any system where you run a program using SND-MSG or ON-EXCP.

- On a 7.3 system, you need PTF 5770SS1 SI79656
- On a 7.4 system, you need PTF 5770SS1 SI79655
- On a 7.5 system, you need PTF 5770SS1 SI79642

The PTF will already be on the system where you compile your program, but if you save it and restore it on another system, you will have to ensure the runtime PTF is on that system.

What’s new with 7.3 & 7.4 PTFs in September 2021

%MAXARR and %MINARR

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Fall 2021

%MAXARR returns the index of the maximum element in the array

```
arr1 = %LIST('Jupiter' : 'Neptune' : 'Mars' : 'Earth');
index = %MAXARR(arr1);
maxvalue = arr1(index);
// index = 2
// maxvalue = 'Neptune'
```

%MINARR returns the index of the minimum element in the array

```
minvalue = arr1(%MINARR(arr1));
// minvalue = 'Earth'
```

You can use %MINARR directly to index the array

%MAXARR and %MINARR: DS subfield

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2021

Find the DS array element with the max or min value for a subfield:

- Use the DSNAME(*).SUBFIELD_NAME syntax

```
info(1).item = 'chair';
info(1).quantity = 20;
info(2).item = 'candle';
info(2).quantity = 25;
info(3).item = 'table';
info(3).quantity = 1;

index = %MAXARR(info(*).quantity);
max_quantity = info(index).quantity;
max_item = info(index).item;
// max_quantity = 25
// max_item = 'candle'
```

%MAXARR and %MINARR

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2021

- The optional second parameter is the first element to check

```
arr1 = %LIST('Earth' : 'Neptune' : 'Jupiter' : 'Mars');
index = %MAXARR(arr1 : 3); // Check elements 3 ...
maxvalue = arr1(index);
// index = 4
// maxvalue = 'Mars'
```

It does not consider "Earth" or "Neptune" in elements 1 and 2

- The optional third parameter is the number of elements to check

```
index = %MINARR(arr1 : 2 : 2); // Check elements 2 - 3
minvalue = arr1(index);
// index = 3
// minvalue = 'Jupiter'
```

It does not consider "Earth" or "Mars" in elements 1 and 4

Sort a DS array by multiple subfields

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2021

Specify the subfields for sorting a DS array using %FIELDS

```
info(1).item = 'chair';
info(1).quantity = 20;
info(1).code = 'K';

info(2).item = 'candle';
info(2).quantity = 20;
info(2).code = 'X';

info(3).item = 'table';
info(3).quantity = 1;
info(3).code = 'P';

SORTA info %FIELDS(quantity : item);
```

After the SORTA operation:

```
info(1).item = 'table';
info(1).quantity = 1;
info(1).code = 'P';

info(2).item = 'candle';
info(2).quantity = 20;
info(2).code = 'X';

info(3).item = 'chair';
info(3).quantity = 20;
info(3).code = 'K';
```

- If the values of the "quantity" subfield are equal, it compares the "item" subfield

Debug named constants

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Fall 2021

Specify DEBUG(*CONSTANTS) to enable debugging named constants

```
ctl-opt debug(*constants);

dcl-c MAX_ELEMS 5;

... Thousands of lines ...

if n > MAX_ELEMS;
  ...
```

What is MAX_ELEMS?

With DEBUG(*CONSTANTS), you can see the value of MAX_ELEMS in the debugger

- with F11 or EVAL, in the system debugger
- by hovering, in RDi

Debugger PTF for debugging constants

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Spring 2021

Warning

You need a debugger PTF on any system where you debug a program compiled with DEBUG(*CONSTANTS).

- On a 7.3 system, you need PTF 5770SS1 SI76807
- On a 7.4 system, you need PTF 5770SS1 SI76808

The PTF will already be on the system where you compile your program, but if you save it and restore it on another system, you will have to ensure the debugger PTF is on that system.

What’s new with 7.3 & 7.4 PTFs in April 2021

%LOWER and %UPPER

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Spring 2021

- %LOWER converts to lower case
- %UPPER converts to upper case
- Alphanumeric and UCS-2 strings are supported

```
string = %LOWER('HELLO WORLD');
// string = 'hello world'
string = %UPPER('Héllo Wörld');
// string = 'HÉLLO WÖRLD'
```

It handles accented characters like é and ö

%LOWER and %UPPER

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Spring 2021

The second parameter is the start position for conversion

```
string = %LOWER('HELLO world' : 2); // From position 2
// string = 'Hello world'
```

The third parameter is the length of conversion

```
string = %UPPER('hello world' : 1 : 1); // Only position 1
// string = 'Hello world'
```

%SPLIT

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Spring 2021

%SPLIT splits a string into an array of substrings

- The default is to split at blanks
- Alphanumeric, UCS-2, and Graphic strings

```
array = %SPLIT('One two three');
// array(1) = 'One'
// array(2) = 'two'
// array(3) = 'three'
```

```
for-each x in %SPLIT('a b c');
   ... // x will be 'a', then 'b', then 'c'
endfor;
```

%SPLIT

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Spring 2021

The optional second parameter lists the separator characters

```
array = %SPLIT('Hello! How are you?' : '!.,? ');
   // array(1) = 'Hello'
   // array(2) = 'How'
   // array(3) = 'are'
   // array(4) = 'you'
```

The string is split at any of the characters in the second parameter:

! . , ? or blank

%SPLIT

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Spring 2021

- Leading and trailing separators are ignored
- If there are multiple separators together, they are treated as a single separator

```
array = %SPLIT('  Hello! !  there ! ' : '! ');
   // array(1) = 'Hello'
   // array(2) = 'there'
```

Runtime PTF for the new built-ins

PTFs for 7.3 and 7.4
- Spring 2021

Warning

You need a runtime PTF on any system where you run a program using %LOWER, %UPPER, or %SPLIT.

- On a 7.3 system, you need PTF 5770SS1 SI76098
- On a 7.4 system, you need PTF 5770SS1 SI76099

The PTF will already be on the system where you compile your program, but if you save it and restore it on another system, you will have to ensure the runtime PTF is on that system.

EXPROPTS(*STRICTKEYS)

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Spring 2021

New parameter *STRICTKEYS for the EXPROPTS keyword

With *STRICTKEYS, the rules for %KDS and a list of keys are more strict

- CCSID conversion is not supported
- The search argument cannot be longer than the key
  - For numeric keys, this applies to both integer positions and decimal positions
  - For date keys, this applies to the number of digits in the year
- If either the search argument or the key is float, both must be float

EXPROPTS(*STRICTKEYS)

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Spring 2021

```
ctl-opt expropts(*strictkeys);

dcl-f myfile keyed;
// The keys are
//     char(2)
//     packed(5 : 2)

chain ('abc' : 1234) myfile;
```

With *STRICTKEYS:

- 'abc' is too long, since the length of the key is only 2
- 1234 has too many integer positions, since the key only has 3 integer positions

Earlier PTF enhancements for 7.3 and 7.4

OPTIONS(*EXACT) for data structures

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2019

```
dcl-pr p1;
   parm likeds(custInfo_t);
end-pr;

dcl-pr p2;
   parm likeds(orderinfo_t);
end-pr;

dcl-ds order likeds(orderinfo_t);

p1 (order); // Bug! The procedure expects custInfo
p2 (order); // Ok
```

The compiler allows the call to p1 even though p1 expects a `custInfo_t` data structure and it is passed an `orderInfo_t` data structure.

OPTIONS(*EXACT) for data structures

IBM i Anywhere
IBM i Everywhere

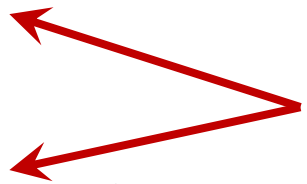
PTFs for 7.3 and 7.4 - Fall 2019

```
dcl-pr p1;
   parm likeds(custInfo_t) options(*exact);
end-pr;

dcl-pr p2;
   parm likeds(orderinfo_t) options(*exact);
end-pr;

dcl-ds order likeds(orderinfo_t);

p1 (order); // Compile-time error: Defined with the wrong LIKEDS
p2 (order); // Ok
```



With OPTIONS(*EXACT), the compiler does not allow the call to p1

%RANGE

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Fall 2020

Check whether a value is in the range of two other values using %RANGE

```
IF num IN %RANGE(1 : 10); // num >= 1 AND num <= 10
 . . .
ENDIF;
```

All data types are supported for %RANGE

```
IF dueDate IN %RANGE(orderDate : orderDate + %DAYS(31));
 . . .
ENDIF;
```

%LIST

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Fall 2020

Define a temporary array using built-in function %LIST

%LIST is allowed in calculations anywhere an array is allowed except SORTA, %LOOKUPxx, %MAXARR, %MINARR, or %SUBARR.

```
dcl-s libraries char(10) dim(5);

libraries = %LIST('QGPL'
                  : currentUserProfile
                  : 'QTEMP'
                  : getDevLib());
```

IN operator

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2020

The 'IN' operator can be used to check whether a value is in an array or %LIST

```
if 3 in myArray;
   // one element of myArray has the value 3
endif;

if info.filename in %SUBARR(filenames : 1 : numFilenames);
   // one element of filenames has the same value as info.filename
endif;

if orderStatus in %LIST(OVERDUE : PENDING : CANCELLED);
   // the value of orderStatus is one of the values in the list
endif;
```

FOR-EACH opcode

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Fall 2020

FOR-EACH iterates through an array, sub-array or %LIST

(It also uses the IN operator)

```
for-each myItem in myArray;
   // myItem has the value of myArray(1), myArray(2) ...
endif;

for-each filename in %SUBARR(filenames : 1 : numFilenames);
   // filename has the value of filenames(1), filenames(2) ...
endif;

for-each orderStatus in %LIST(OVERDUE : PENDING : CANCELLED);
   // orderStatus has the value OVERDUE, then PENDING, then CANCELLED
endif;
```

Allow blanks for %DEC, %INT etc.

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Fall 2020

When a string is blank for %DEC, %INT, %FLOAT etc:

- Normally, it is an error
- With EXPROPTS(*ALWBLANKNUM), these built-in functions return zero

```
ctl-opt expropts(*alwblanknum);

str = *blanks;
num = %dec(str : 63 : 15);  // Error, without *alwblanknum
```

With *ALWBLANKNUM, num = 0

Allow separators for %DEC, %INT etc.

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2020

%DEC, %INT, %FLOAT etc. normally treat both comma and period as a decimal point

- '1.2' and '1,2' both mean 1.2
- '1,234,567.89' is invalid because it has "3 decimal points"

With EXPROPTS(*USEDECEDIT), the decimal point and digit separator are determined by the DECEDIT setting.

```
ctl-opt expropts(*usedecedit);

str = '1,234,567.89';
num = %dec(str : 63 : 15);  // Error, without *usedecedit
```

With *USEDECEDIT, num = 1234567.89

DECEDIT('.') means that period is the decimal point and comma is the separator.

DECEDIT(',') means that comma is the decimal point and period is the separator.

DECEDIT(*JOBRUN) means that the job DECFMT setting is used.

DECEDIT('.') is the default.

Optionally require prototypes

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4
- Fall 2020

Normally, RPG allows any procedure to be defined without a prototype

- Ok for local non-exported procedures
- Might be a problem for exported procedures
- Might be a problem for a main procedure

New command parameter and H spec keyword REQPREXP lets you get either a sev10 warning or sev30 error if a prototype is missing for an exported procedure or for the main procedure.

REQPREXP(*NO | *WARN | *REQUIRE)

Optionally require prototypes

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2020

Sometimes a particular program or procedure will never be called from another RPG program

- A command-processing program for a command
- A Java native method
- …

You can indicate that it's ok not to have a prototype using REQPROTO(*NO).

```
ctl-opt reqprexp(*require);
ctl-opt main(myCmdPgm);

dcl-proc myCmdPgm reqproto(*no);
   ...
end-proc;
```

Normally a prototype is required for the MAIN procedure if you have REQPREXP(*REQUIRE).

It is optional if you code REQPROTO(*NO) on the DCL-PROC statement.

Optionally require prototypes

PTFs for 7.3 and 7.4 - Fall 2020

You might consider the prototype to be optional for a cycle-main procedure.

For a cycle-main procedure, code REQPROTO(*NO) on the Procedure Interface.

```
ctl-opt reqprexp(*require);

dcl-pi *n reqproto(*no);
   ...
end-pi;
```

There is no MAIN or NOMAIN keyword in the CTL-OPT statements, so the module has a "cycle main" procedure.

A Procedure Interface in the global definitions is for the cycle-main procedure.

Debug the return value from a procedure

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2020

With H spec keyword DEBUG(*RETVAL), you can see or change the return value of a procedure while at the breakpoint at the end of the procedure

In the debugger, work with _QRNU_RETVAL on the final statement of the procedure

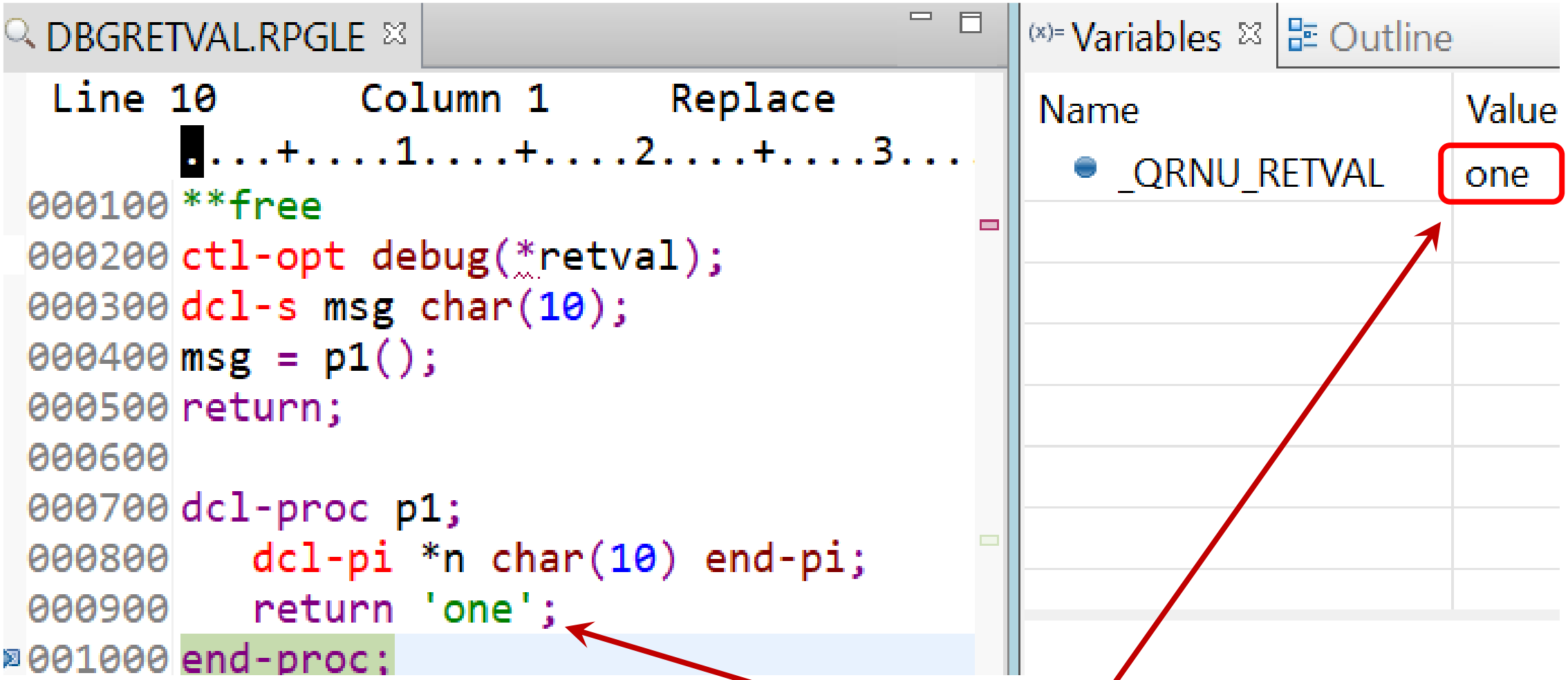
- The END-PROC statement in free-form
- The P spec with the 'E' type in fixed-form

Prior to the final statement, the value of _QRNU_RETVAL is meaningless

Debug session – at END-PROC

IBM i Anywhere
IBM i Everywhere

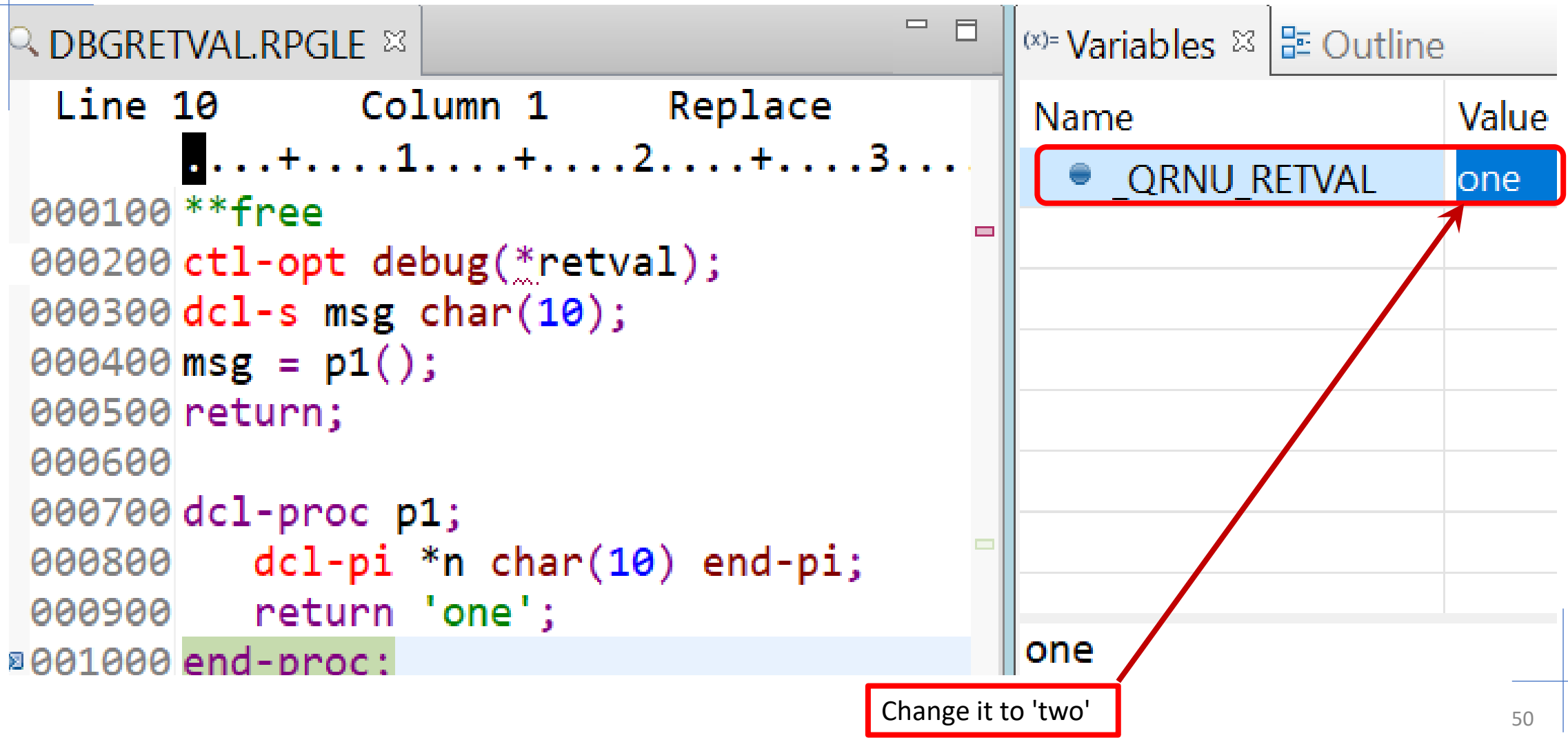
PTFs for 7.3 and 7.4 - Fall 2020



Debug session – at END-PROC

IBM i Anywhere
IBM i Everywhere

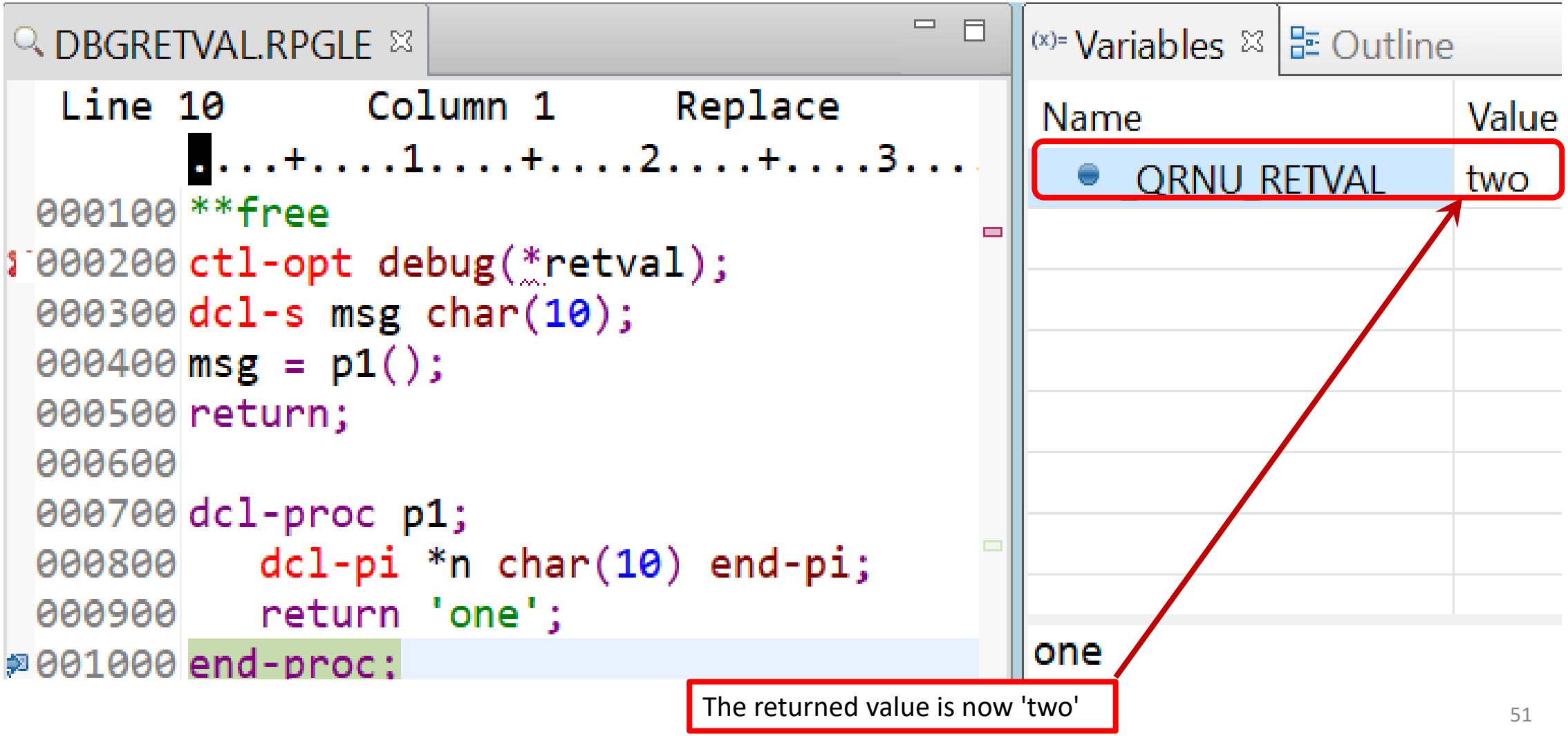
PTFs for 7.3 and 7.4 - Fall 2020



Debug session – at END-PROC

IBM i Anywhere
IBM i Everywhere

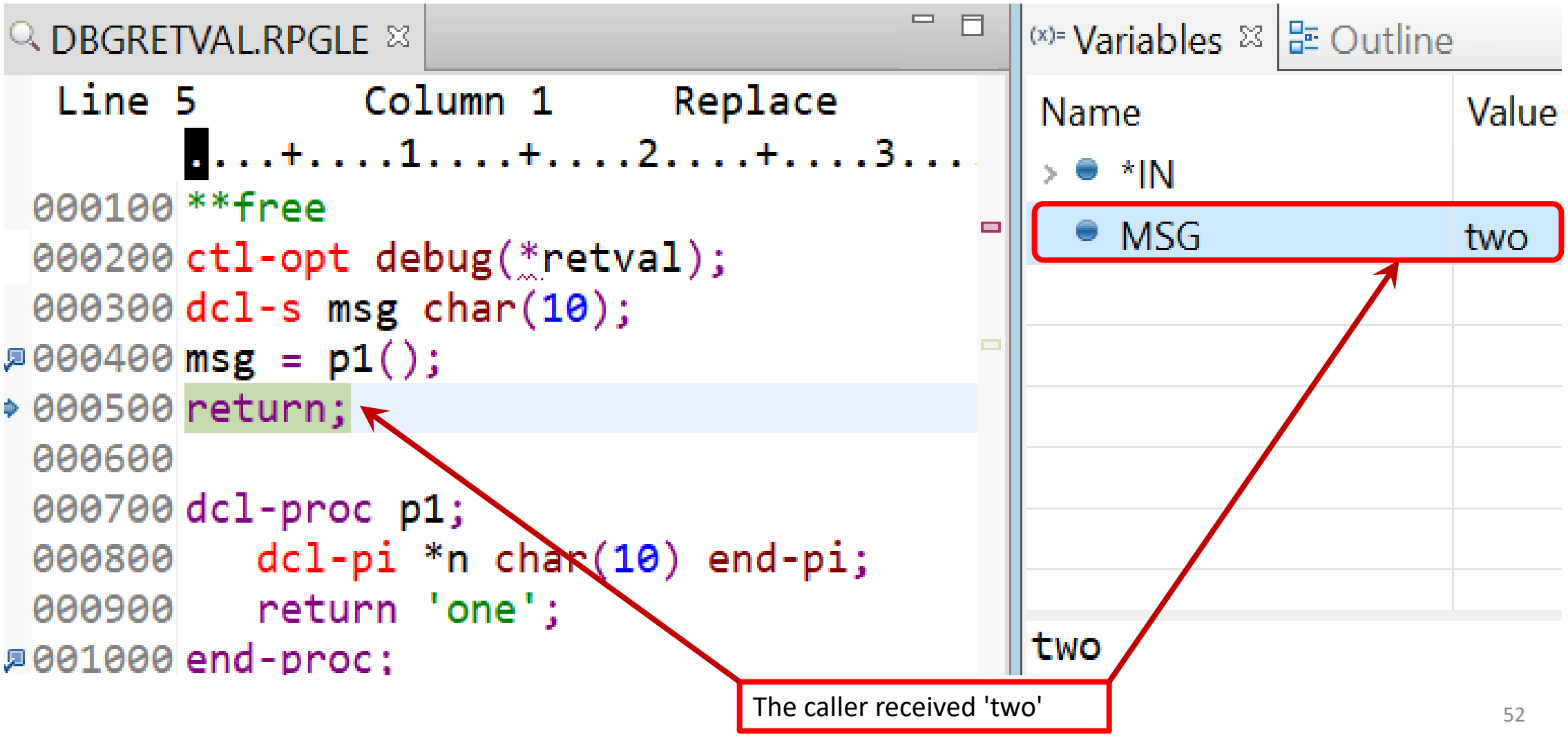
PTFs for 7.3 and 7.4
- Fall 2020



Debug session – after the call

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2020



%TIMESTAMP to return microseconds

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2020

%TIMESTAMP formerly only returned milliseconds

Now, it returns microseconds

If for some reason you only want milliseconds:

Code this

```
%timestamp(*sys : 3)
```

%TIMESTAMP to return a unique value

IBM i Anywhere
IBM i Everywhere

PTFs for 7.3 and 7.4 - Fall 2020

Use %TIMESTAMP(*UNIQUE)

The timestamp has 12 "fractional digits"

```
Z'2020-03-24-01.02.03.123456123456'
```

- The first six fractional seconds provide microsecond precision
- The final six digits are used to create a unique timestamp
- Each *UNIQUE timestamp is guaranteed to be greater than the previous one

Don't use the "uniqueness" digits in calculations.

When getting the time between unique timestamps, use %TIMESTAMP(uniqueTs : 6) so you only the use the microseconds.

Runtime PTF for *UNIQUE

PTFs for 7.3 and 7.4 - Fall 2020

Warning

You need a runtime PTF on any system where you run a program using TIMESTAMP(*UNIQUE).

- On a 7.3 system, you need PTF 5770SS1 SI73189
- On a 7.4 system, you need PTF 5770SS1 SI73191

The PTF will already be on the system where you compile your program, but if you save it and restore it on another system, you will have to ensure the runtime PTF is on that system.

Useful links

Where to find PTF'd RPG enhancements

IBM i Anywhere
IBM i Everywhere

.

Regularly check the “welcome” page in the RPG Cafe wiki

http://ibm.biz/rpg_cafe

There is a section for “Enhancements delivered through PTFs”, and the “Announcement” section at the top will have information about the most recent enhancements

Regularly check the “What’s New Since …” section in the ILE RPG Reference

Where to request RPG enhancements

.

RPG is part of the IBM Ideas" process (formerly the RFE (Request for Enhancements) process

- You can submit requirements
- You can vote on requirements that others have requested
  - Votes aren’t the only consideration when IBM decides which Ideas to work on, but they are important
  - Be careful not to vote for too many Ideas. Just vote on the ones that you need the most
- Add comments to existing Ideas, to clarify the request, or add additional reasons why the enhancement is important

Here is a link to the current Ideas for the RPG compiler: http://ibm.biz/rpg_rfe

They are under

- IBM i
  - Languages – RPG

Thank you

IBM i Anywhere
IBM i Everywhere