Free RPG

Paul Roy

FREE RPG

- Very easy…
- No more column in C Specs
- C/free

  /end-free

- Instruction ALWAYS ends with a semicolumn ;
- Multiline complex instructions

```
A = 2*B + (X*Y);
presDate = 'Jeudi 3 octobre ';
text = 'ceci est un example pour '  +
     'la presentation du '  + presDate;
```

- Declarations in D specs ONLY

RPG Free

- From RPG IV to RPG free
  - FACTOR1 OPCODE FACTOR2 RESULT
  - Free instruction
    - OPCODE FACTOR1 FACTOR2
    - %BIF
- Automatic conversion in LPEX Editor (Rational)
  - Existing third party tools

Operation codes

- ACQ{(E)} *device-name workstn-file*
- BEGSR *subroutine-name*
- CHAIN{(ENHMR)} *search-arg file-or-record-name* {*data-structure*}
- CLEAR {*NOKEY} {*ALL} *name*
- CLOSE{(E)} *file-name*
- COMMIT{(E)} {*boundary*}
- DEALLOC{(EN)} *pointer-name*
- DELETE{(EHMR)} {*search-arg*} *file-or-record-name*
- DOU{(MR)} *indicator-expression*
- DOW{(MR)} *indicator-expression*
- DSPLY{(E)} {*message* {*message-queue* {*response*}}}

Operation codes

- DUMP{(A)} {*identifier*}
- ELSE
- ELSEIF{(MR)} *indicator-expression*
- ENDDO
- ENDFOR
- ENDIF
- ENDMON
- ENDSL
- ENDSR {*return-point*}
- EVALR{(MR)} *result = expression*
- EVAL-CORR{(EH)} *target-ds = source-ds*

Operation codes

- **EXCEPT** EXCEPT {*except-name*}
- **EXFMT** EXFMT{(E)} *format-name {data-structure}*
- **EXSR** EXSR *subroutine-name*
- **FEOD** FEOD{(EN)} *file-name*
- **FOR** FOR{(MR)} *index* {= *start*} {BY *increment*} {TO|DOWNTO *limit*}
- **FORCE** FORCE *file-name*
- **IF** IF{(MR)} *indicator-expression*
- **IN** 1 IN{(E)} {*LOCK} *data-area-name*
- **ITER** ITER
- **LEAVE** LEAVE
- **LEAVESR** LEAVESR

Operation codes

- **MONITOR** MONITOR
- **NEXT**1 NEXT{(E)} *program-device file-name*
- **ON-ERROR** ON-ERROR {*exception-id1* {:*exception-id2*...}}
- **OPEN** OPEN{(E)} *file-name*
- **OTHER** OTHER
- **OUT**1 OUT{(E)} {*LOCK} *data-area-name*
- **POST** 1 POST{(E)} {*program-device*} *file-name*
- **READ** READ{(EN)} *file-or-record-name* {*data-structure*}
- **READC** READC{(E)} *record-name* {*data-structure*}
- **READE** READE{(ENHMR)} *search-arg*|*KEY *file-or-record-name* {*data-structure*}
- **READP** READP{(EN)} *name* {*data-structure*}
- **READPE** READPE{(ENHMR)} *search-arg*|*KEY *file-or-record-name* {*data-structure*}
- **REL** 1 REL{(E)} *program-device file-name*
- **RESET** 1 RESET{(E)} {*NOKEY} {*ALL} *name*
- **RETURN** RETURN{(HMR)} *expression*
- **ROLBK** ROLBK{(E)}

Operation codes

- **SETGT** SETGT{(EHMR)} *search-arg file-or-record-name*
- **SETLL** SETLL{(EHMR)} *search-arg file-or-record-name*
- **SORTA** SORTA{(AD)} *array-name* or *keyed-ds-array*
- **TEST** 1 TEST{(EDTZ)} {*dtz-format*} *field-name*
- **UNLOCK** 1 UNLOCK{(E)} *name*
- **UPDATE** UPDATE{(E)} *file-or-record-name* {*data-structure*|%FIELDS*(name*{*:name*...})}
- **WHEN** WHEN{(MR)} *indicator-expression*
- **WRITE** WRITE{(E)} *file-or-record-name* {*data-structure*}
- **XML-INTO** XML-INTO{(EH)} *target-or-handler xml-document*
- **XML-SAX** XML-SAX{(E)} *handler xml-document*

BIF – Built in Functions

- %ABS absolute value of expression
- %ADDR address of variable
- %ALLOC pointer to allocated storage
- %BITAND bit wise ANDing of the bits of all the arguments
- %BITNOT bit-wise reverse of the bits of the argument
- %BITOR bit-wise ORing of the bits of all the arguments
- %BITXOR bit-wise exclusive ORing of the bits of the two arguments
- %CHAR value in character format
- %CHECK first position of a character that is not in the comparator string, or zero if not found
- %CHECKR last position of a character that is not in the comparator string, or zero if not found
- %DATE the date that corresponds to the specified *value*, or the current system date if none is specified
- %DAYS number of days as a duration

BIF – Built in Functions

- %DEC value in packed numeric format
- %DECH half-adjusted value in packed numeric format
- %DECPOS number of decimal digits
- %DIFF difference between the two dates, times, or timestamps in the specified unit
- %DIV the quotient from the division of the two arguments
- %EDITC string representing edited value
- %EDITFLT character external display representation of float
- %EDITW string representing edited value
- %ELEM number of elements or occurrences
- %EOF '1' if the most recent cycle input, read operation, or write to a subfile (for a particular file, if specified) ended in an end-of-file or beginning-of-file condition; and, when a file is specified, if a more recent OPEN, CHAIN, SETGT or SETLL to the file was not successful '0' otherwise
- %EQUAL '1' if the most recent SETLL (for a particular file, if specified) or LOOKUP operation found an exact match '0' otherwise

BIF – Built in Functions

- %ERROR '1' if the most recent operation code with extender 'E' specified resulted in an error
  '0' otherwise
- %FIELDS list of fields to be updated
- %FLOAT value in float format
- %FOUND '1' if the most recent relevant operation (for a particular file, if specified) found a record (CHAIN, DELETE, SETGT, SETLL), an element (LOOKUP), or a match (CHECK, CHECKR, SCAN
  '0' otherwise
- %GRAPH value in graphic format
- %HANDLER
- %HOURS number of hours as a duration
- %INT value in integer format
- %INTH half-adjusted value in integer format
- %KDS data structure containing keys
- %LEN length in digits or characters
- %LOOKUP array index of the matching element

BIF – Built in Functions

- %MINUTES number of minutes as a duration
- %MONTHS number of months as a duration
- %MSECONDS number of microseconds as a duration
- %NULLIND value in indicator format representing the null indicator setting for the null-capable field
- %OCCUR current occurrence of the multiple-occurrence data structure
- %OPEN '1' if the specified file is open
  '0' if the specified file is closed
- %PADDR address of procedure or prototype
- %PARMS number of parameters passed to procedure
- %PARMNUM number of a procedure-interface parameter
- %REALLOC pointer to allocated storage
- %REM the remainder from the division of the two arguments
- %REPLACE string produced by inserting replacement string into source string, starting at start position and replacing the specified number of characters

BIF – Built in Functions

- %SCAN — first position of search argument in string or zero if not found
- %SCANRPL — string produced by replacing scan string by replacement string in source string, with the scan starting at start position for the specified length
- %SECONDS — number of seconds as a duration
- %SHTDN — '1' if the system operator has requested shutdow
  '0' otherwise
- %SIZE — size of variable or literal
- %SQRT — square root of the numeric value
- %STATUS — 0 if no program or file error occurred since the most recent operation code with extender 'E' specified
  most recent value set for any program or file status, if an error occurred
  if a file is specified, the value returned is the most recent status for that file
- %STR — characters addressed by pointer argument up to but not including the first x'00'
- %SUBARR — array subset
- %SUBDT — an unsigned numeric value that contains the specified portion of the date or time value
- %SUBST — substring
- %THIS — the class instance for the native method
- %TIME — the time that corresponds to the specified *value*, or the current system time if none is specified

BIF – Built in Functions

- %TIMESTAMP — the timestamp that corresponds to the specified *value*, or the current system timestamp if none is specified
- %TLOOKUPxx — '*ON' if there is a match
  '*OFF' otherwise
- %TRIM — string with left and right blanks or specified characters trimmed
- %TRIML — string with left blanks or specified characters trimmed
- %TRIMR — string with right blanks or specified characters trimmed
- %UCS2 — in UCS-2 format
- %UNS — value in unsigned format
- %UNSH — half-adjusted value in unsigned format
- %XFOOT — sum of the elements
- %XLATE — the string with from-characters replaced by to-characters
- %XML
- %YEARS — number of years as a duration

Unsupported OP CODE

ADD	CAT	GOTO	MOVEA	SETOFF	TESTZ
ADDDUR	CHECK	IFXX	MOVEL	SETON	TIME
ALLOC	CHECKR	KFLD	MULT	SHTDN	WHENXX
ANDXX	COMP	KLIST	MVR	SQRT	XFOOT
BITOFF	DEFINE	LOOKUP	OCCUR	SUB	XLATE
BITON	DIV	MHHZO	ORXX	SUBDUR	Z-ADD
CABXX	DO	MHLZO	PARM	SUBST	Z-SUB
CALL	DOUXX	MLHZO	PLIST	TAG	
CALLB	DOWXX	MLLZO	REALLOC	TESTB	
CASXX	EXTRCT	MOVE	SCAN	TESTN	

UNSUPPORTED OP CODE

BITOFF	Write your own code
BITON	Write your own code
MHHZO	Write your own code
MHLZO	Write your own code
MLHZO	Write your own code
MLLZO	Write your own code
TESTB	Write your own code
TESTN	Write your own code
TESTZ	Write your own code
MOVEA	Write your own code
KLIST	Use data structure
KFLD	Use data structure

Unsupported opcodes

ADD	+ operator
SUB	-
MULT	*
DIV	/
MOVE	EVALR or %BIF

Replaced by BIF

ALLOC	%ALLOC
CHECK	%CHECK
CHECK	%CHECK
DIV	%DIV
EXTRCT	%SUBDT
LOOKUP	%LOOKUP %TLOOKUP
MVR	%REM
OCCUR	%OCCUR
REALLOC	%REALLOC
SCAN	%SCAN
SHTDN	%SHTDN
SQRT	%SQRT
SUBDUR	%DIFF
SUBST	%SUBST
TIME	%DATE %TIME
XFOOT	%XFOOT
XLATE	%XLATE

Unsupported / ALTERNATIVE

-

GOTO	
TAG	
CAB	
CASXX	SELECT/WHEN
IFXX	If
DO	FOR
DOUXX	DOU
DOWXX	DOW
WHENXX	WHEN
ANDXX	AND
ORXX	OR

How to write an RPG Free Program

- H spec
- Contains compile ILE instructions
  - DFTACTGRP(*NO) ACRTGRP(*CALLER) BNDDIR('*LIBL/MYDIRE')

Parameters ?

- Add your program prototype
  (C *ENTRY PLIST)
- Prototype your program as a procedure :

```
D Main            PR                  EXTPGM('UUDR00R')
D Directory                     255A
D extension                       3A
D Pgm2Call                       10A
```

- Declare the procedure interface

```
D Main            PI
D P_dir                         255A
D P_Ext                           3A
D P_Pgm                          10A
```

CALL

- Define the prototype

```
DRunCommand       PR                  EXTPGM('QCMDEXC')
D  Command                    3000    Options(*VarSize)
D                                     CONST
D  Commandlength                15P05 CONST
```

- Use it …

```
C/Free
    Monitor;
        lCommand= 'CLRPFM ' + %trim(gLIB100) + '/' + gOBJ100;
        RunCommand(lCommand:%Len(lCommand));
    On-Error ;
        lMessage = gOBJ100 + ' in ' + gLIB100 + ' Not cleared';
        Dsply lMessage;
    EndMon;
 /End-Free
```

procedures

- Define prototypes

```
DRunCommand        PR                  EXTPGM('QCMDEXC')
D  Command                      3000   Options(*VarSize)
D                                      CONST
D  Commandlength                  15P05 CONST

DSysCommand        PR
D  Command                      3000   Options(*VarSize)
```

- Define the procedure

```
PSysCommand        B
DSysCommand        PI
D  P_Command                    3000   Options(*VarSize)
D  lCommand        S            3000

C/Free
       lCommand= P_Command;
       RunCommand(lCommand:%Len(lCommand));
 /End-Free
PSysCommand        E
```

- Use it in your code

  SysCommand('CLRPFM QTEMP/WORKFILE');

Code

- **Free-Form Syntax**
- To begin a free-form calculation group, specify /FREE in positions 7 to 11 and leave positions 12 to 80 blank. The free-form calculation block ends when you specify /END-FREE.
- In a free-form statement, the operation code does not need to begin in any specific position within columns 8–80. Any extenders must appear immediately after the operation code on the same line, within parentheses. There must be no embedded blanks between the operation code and extenders. Following the operation code and extenders, you specify the Factor 1, Factor 2, and the Result Field operands separated by blanks. If any of these are not required by the operation, you may leave them out. You can freely use blanks and continuation lines in the remainder of the statement. Each statement must end with a semicolon. The remainder of the record after the semicolon must be blank or contain an end-of-line comment.