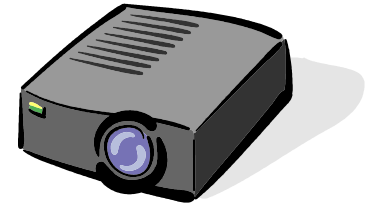


Modernisation, développement d'applications et DB2 sous IBM i
Technologies, outils et nouveautés 2013-2014

13 et 14 mai 2014 – IBM Client Center Paris, Bois-Colombes

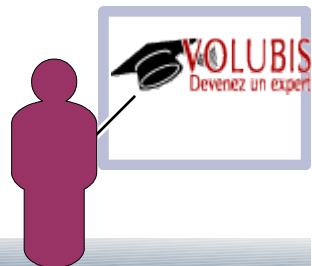
Volubis.fr



Conseil et formation sur OS/400, I5/OS puis IBM *i*
depuis 1994 !

Dans nos locaux, vos locaux ou par Internet

Christian Massé - cmasse@volubis.fr



Avancées RPG en TR7

Il faut aussi la PTF SI51094

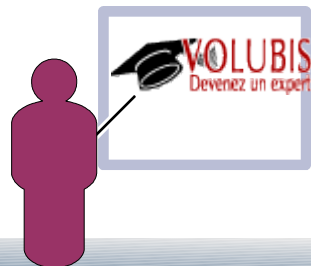
Cette mise à jour apporte un format libre complet : plus de spécifications H, F, D et P.

Plus besoin de /free /end-free, les colonnes 6 et 7 à blanc suffisent.

```
      read fichier;  
C      MOVEA  *ALL'0'          *IN  
      exfmt écran;  
      IF *in03;  
      .....
```

Donc ces dernières (colonnes 6 et 7) sont encore réservées :

6 à la lettre (D, C, ...) si vous les utilisez
7 au / (de /copy par exemple)
vous faites ce que vous voulez, mais à partir de la position 8 !



Avancées RPG en TR7

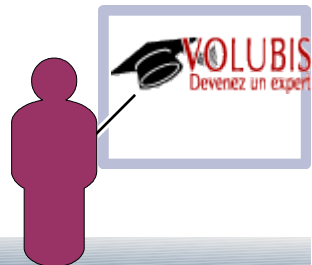
Spécif H

- CTL-OPT mots-clés ;

```
ctl-opt OPTION(*NODEBUGIO : *SRCSTMT) ALWNULL(*USRCTL) ;
```

- On peut mélanger ctl-opt et des spécifs H
(cela sera est aussi vrai pour les spécifs F et D)

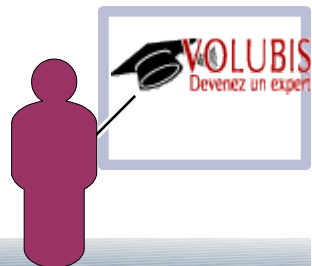
```
ctl-opt OPTION(*NODEBUGIO : *SRCSTMT)  
ALWNULL(*USRCTL) ;  
H DATFMT(*DMY)
```



Avancées RPG en TR7

Spécif F

- DCL-F ***nom-de-fichier*** *unité mots-clés* ;
 - nom-de-fichier, peut faire plus de 10 c., auquel cas EXTDESC est obligatoire
 - Unité
 - DISK , c'est la valeur par défaut , USAGE(*INPUT) par défaut
 - PRINTER , USAGE(*OUTPUT) par défaut
 - WORKSTN, USAGE(*INPUT : *OUTPUT) par défaut
 - USAGE
 - *INPUT (lecture uniquement)
 - *OUTPUT (écriture uniquement)
 - *UPDATE (écriture / mise à jour uniquement, n'autorise plus automatiquement les DELETE)
 - *DELETE (écriture, mise à jour, suppression)



Avancées RPG en TR7

Spécif F

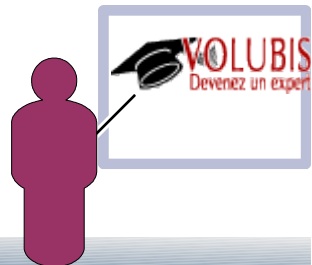
- Mots-clés

KEYED pour indiquer un accès par clé (remplace K en colonne 34)

Les autres mots-clés des spécifs F d'aujourd'hui
(USROPN, EXTFILE,)

- Sauf :

Aucun traitement possible des fichiers tables en format libre
Aucune notion de cycle en format libre



Avancées RPG en TR7

Spécif F

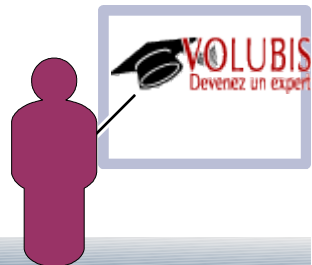
- Exemples

DCL-F clients KEYED ; // *FICHIER BdeD en lecture par clé*

DCL-F personp1 USAGE(*UPDATE); // *Fichier en mise à jour*

DCL-F ecr420 WORKSTN; // *DSPF en lecture/écriture*

DCL-F etat01 PRINTER; // *PRTF en sortie*



Avancées RPG en TR7

Spécif D

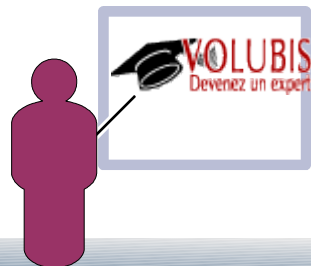
- DCL-C nom-constante 'constante' ;
- DCL-S nom-variable type mots-clés ;

Exemples :

DCL-S compteur INT(5);

DCL-S flag IND;

DCL-S message CHAR(30);



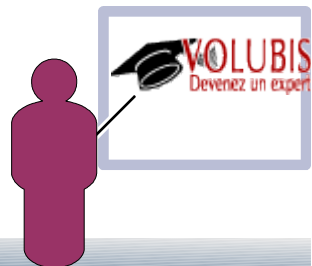
Types

Type	remarque	équivalent D	paramètres
BINDEC	binaire "décimal"	B	Bindec(lg [:décimales])
CHAR	alphanumérique	A	Char(lg)
DATE	date	D	Date(format [séparateur])
FLOAT	notation scientifique	F	Float(lg)
GRAPH	DBCS	G	Graph(lg)
IND	Indicateur	N	
INT	Binaire (compatible C et API)	I	Int(lg)
OBJECT	pour Java (JNI)	O	Object(*JAVA : classe)
PACKED	numérique packé	P	Packed(lg [:décimales])
POINTER	pointeur	*	[*PROC]
TIME	heure	T	Time(format [:séparateur])
TIMESTAMP	horodatage	Z	
UCS2	Unicode	C	UCS2(lg)
UNS	binaire non signé	U	Uns(lg)
VARCHAR	Alphanumérique à taille variable	A + VARYING	Varchar(lg)
VARGRAPH	DBCS à taille variable	G + VARYING	Vargraph(lg)
VARUCS2	Unicode à taille variable	C + VARYING	Varucs2(lg)
ZONED	Numérique étendu	S	Zoned(lg[:décimales])

Avancées RPG en TR7

Spécif D

- Les mots-clés sont en partie les mêmes que sur la spécif D, excepté :
- FROMFILE/TOFILE pour un tableau, qui ne sont pas admis
- CLASS pour un objet (Java), le nom de la classe étant indiqué en argument
- DATFMT pour une date, le format, facultatif, étant indiqué en argument
- TIMFMT pour une heure, le format, facultatif, étant indiqué en argument
- PROCPTR pour un pointeur, l'option *PROC, facultative, étant indiquée en argument
- VARYING puisqu'il y a des types particuliers pour les variables à taille variable



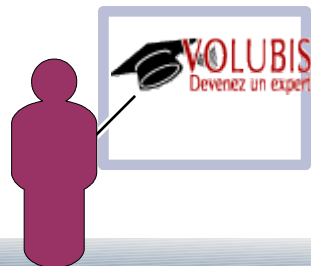
Avancées RPG en TR7

Premier exemple

```
***** Début des données
CTL-OPT  ALWNULL(*USRCTL);
DCL-F PRODUCTEUR KEYED;
DCL-S compteur INT(5);
read producteur;
dow not %eof;
    compteur += 1;
    read producteur;
enddo;
DSPLY (%char(compteur));
*inLR = *on;
***** Fin des données
```

Attention, SEU déclare TOUTES les nouveautés en erreur !

(il faut forcer la sortie)



Avancées RPG en TR7

Data structures

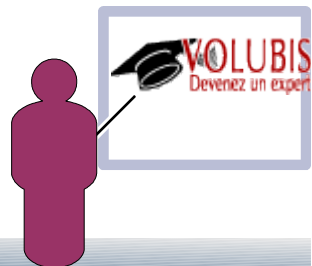
- DCL-DS nom-de-ds [mots-clés] ;
 souszones type mots-clés ;
 END-DS [nom-de-ds] ;

Exemple :

```
DCL-DS Clients QUALIFIED;  
  id INT(10);  
  nom VARCHAR(50);  
  ville VARCHAR(50);  
  cdes LIKEDS(cde_template) DIM(100);  
  nbcdes INT(10);  
END-DS Clients;
```

Dans le code :

```
clients.id += 1 ;  
clients.nom = *blanks ;
```



Avancées RPG en TR7

Data structures *ou bien*

DCL-DS nom-de-ds LIKEDS(autres) ;

DCL-SUBF

quand la sous-zone porte le même nom d'une instruction RPG (Select par exemple)

Les mots-clés sont en partie les mêmes que sur la spécif D, excepté :

OVERLAY où il n'est plus admis de faire référence à la DS, utiliser POS à la place

// exemple INFDS

DCL-F fichier DISK(*EXT) INFDS(fichierInfo);

DCL-DS fichierInfo;

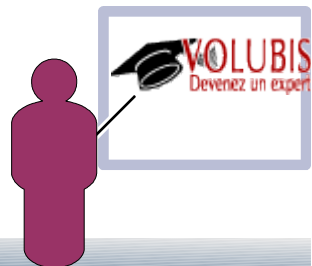
status *STATUS;

opcode *OPCODE;

msgid CHAR(7) POS(46);

END-DS;

DCL-F ecran WORKSTN; // on peut mélanger déclarations de variable et de fichier



Avancées RPG en TR7

Prototypes

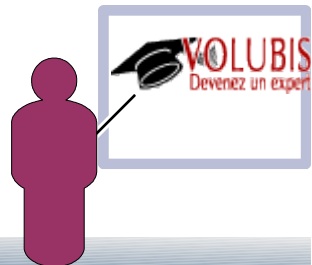
- DCL-PR nom-de-prototype;
 paramètre type mots-clés ;
 END-PR [nom-de-prototype] ;

Exemple :

```
DCL-PR QCMDEXC EXTPGM;  
  cde CHAR(50) CONST;  
  cdl PACKED(15 : 5) CONST;  
END-PR;
```

Dans le code :

```
QCMDEXC('WRKSPLF' : 7) ;
```



Avancées RPG en TR7

Prototypes

- remarques

S'il s'agit d'une fonction, indiquer le type retour sur la déclaration

S'il n'y a pas de paramètre en entrée, indiquer END-PR sur la même ligne

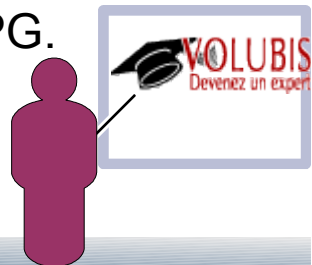
```
DCL-PR MaFonction PACKED(5:0) END-PR;
```

```
EXTPROC(*DCLCASE)
```

pour imposer un respect absolu de la casse (Api systèmes, par ex)

```
DCL-PARM
```

permet de déclarer un paramètre qui se nomme comme une instruction RPG.



Avancées RPG en TR7

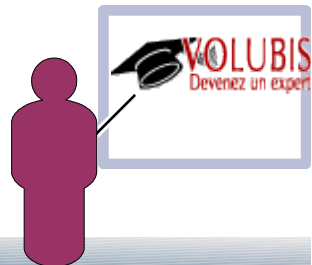
Procédures

- DCL-PROC nom-de-procédure
DCL-PI interface de procédure;
...
END-PROC [nom-de-procédure] ;

DCL-PI nom-de-procédure | *N [type de retour]
paramètre type mots-clés ;
END-PI [nom-de-procédure] ;

Exemple

```
////////////////////////////////////  
DCL-PROC PLUSUN;  
DCL-PI *N IND;  
PR_CODE INT(10) CONST;  
END-PI;  
if compteur < 32767;  
compteur += 1;  
return *ON;  
else;  
return *off;  
endif;  
END-PROC;
```



Avancées RPG en TR7

Procédures

- S'il n'y a pas de paramètre en entrée, indiquer END-PI sur la même ligne

```
// fonction, retourne un booléen  
DCL-PI *N IND END-PI;
```

- *N fait référence à la procédure ou **au programme en cours**

```
// Pgm avec un paramètre en entrée  
ctl-opt dftactgrp(*no) ;
```

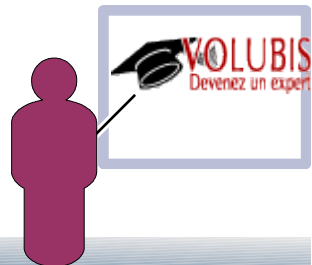
```
·DCL-PI *N;  
  nom CHAR(10) CONST;  
END-PI;
```

```
CALL PGM(FREE_PGMPI)  
Pointeur non défini pour position mémoire référencée.
```

```
dsply ('bonjour ' + nom) ;
```

```
CALL PGM(FREE_PGMPI) PARM('Christian')  
DSPLY  bonjour Christian
```

```
*INLR = *ON;
```



Avancées RPG en TR7

Procédures

- Préciser EXTPGM pour un pgm sans cycle (mot-clé MAIN , **nouveauté V6**)

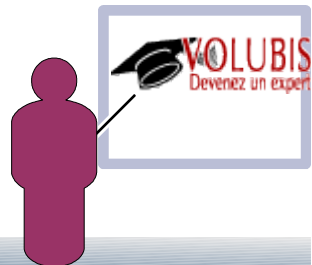
```
CTL-OPT MAIN(Bonjour)
```

```
DCL-PROC Bonjour;
```

```
DCL-PI *N EXTPGM;  
  nom CHAR(10) CONST;  
END-PI;
```

```
  dsply ('bonjour ' + nom) ;
```

```
END-PROC;
```



Avancées RPG en TR7

Procédures

- EXTPROC(*DCLCASE), permet de demander un respect de la casse

DCL-PROC getCdeSuivante;

dans cet exemple :

```
DCL-PI *N IND EXTPROC(*DCLCASE);  
  commandeDS LIKEDS(commande_t);  
END-PI;
```

```
DCL-F cdes STATIC;
```

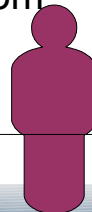
```
READ cdes commandeDS;  
RETURN %eof(cdes);
```

```
END-PROC getCdeSuivante;
```

la procédure se nomme "getCdeSuivante",
exactement !
l'interface de procédure indique :
-que la casse doit être respectée quant au nom
de la procédure
-une valeur retour de type indicateur
-un paramètre de type Data Structure en entrée

Cette procédure possède une déclaration locale d'un
fichier :

- La lecture du fichier doit se faire impérativement
dans une DS (fichier local => pas spécifs I)
- l'instruction END-PROC, contient le nom
de la procédure, *ceci est facultatif.*



Avancées RPG en TR7

Comparaisons

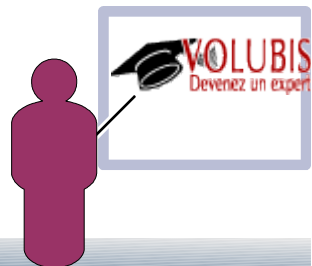
Dwp_date	S	D	
Dwp_jour	S	1 0	
Dundimanche	S	D	inz(D'1980-01-06')
Dnbjours	S	6 0	
C	*entry	plist	
C		parm	wp_date
C		parm	wp_jour

S'écrit maintenant

```
Dcl-S undimanche Date inz(D'1980-01-06');
Dcl-S nbjours Packed(6:0);

// Procedure interface (remplace *ENTRY PLIST)

Dcl-Pi FREE01 ExtPgm('FREE01');
  wp_date      Date;
  wp_jour      Packed(1:0);
End-Pi;
```



Avancées RPG en TR7

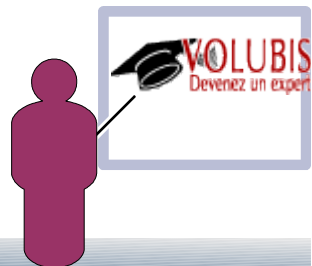
Pour une fonction

(joursemaine, fonction dans un *SRVPGM qui retourne le numéro du jour dans la semaine [1-7])

h nomain				
Djoursemaine	pr	1	0	
Dwp_date			D	
Pjoursemaine	b			export
D	pi	1	0	
Dwp_date			D	
Dwp_jour	s	1	0	
Dundimanche	s		D	inz(D'1980-01-06')
Dnbjours	s	6	0	

```
/FREE  
nbjours = %diff(wp_date : undimanche : *DAYS);  
wp_jour = %rem(nbjours : 7);  
if wp_jour < 1;  
    wp_jour += 7;  
endif;  
  
return wp_jour;  
/END-FREE
```

Pjoursemaine	e			
--------------	---	--	--	--



Avancées RPG en TR7

Pour une fonction

(joursemaine, fonction dans un *SRVPGM qui retourne le numéro du jour dans la semaine [1-7])

```
Ctl-Opt nomain;
```

```
Dcl-Pr joursemaine Packed(1:0);  
  wp_date          Date;  
End-Pr;
```

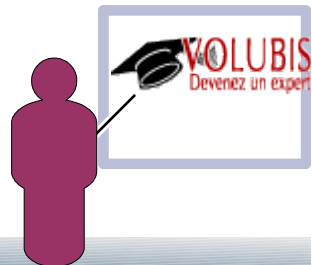
```
Dcl-Proc joursemaine export;  
  Dcl-Pi *n Packed(1:0);  
    wp_date          Date;  
  End-Pi;  
  Dcl-S wp_jour      Packed(1:0);
```

```
  Dcl-S undimanche  Date          inz(D'1980-01-06');  
  Dcl-S nbjours     Packed(6:0);
```

```
  nbjours = %diff(wp_date : undimanche : *DAYS);  
  wp_jour = %rem(nbjours : 7);  
  if wp_jour < 1;  
    wp_jour += 7;  
  endif;
```

```
  return wp_jour;  
End-Proc joursemaine;
```

```
h nomain  
Djoursemaine pr 1 0  
Dwp_date D  
  
Pjoursemaine b 1 0 export  
D pi 1 0  
Dwp_date D  
Dwp_jour s 1 0  
  
Dundimanche s D inz(D'1980-01-06'  
Dnbjours s 6 0  
  
/FREE  
nbjours = %diff(wp_date : undimanche : *DAYS);  
wp_jour = %rem(nbjours : 7);  
if wp_jour < 1;  
  wp_jour += 7;  
endif;  
  
return wp_jour;  
/END-FREE  
Pjoursemaine e
```



RDI 9.0.1

La version 9.0.1 de RDI du 10 Décembre 2013 reconnaît toutes ces nouveautés :

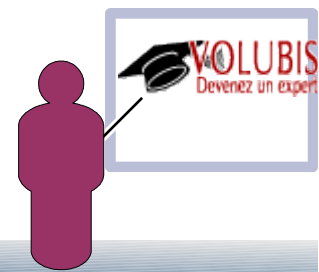
Notre premier test
(où SEU déclarait tout en erreur)

Ligne 1	Colonne 1	Replacer
	Free-Form+++++
000100		CTL-OPT ALWNULL(*USRCTL);
000200		DCL-F PRODUCTEUR KEYED;
000300		DCL-S compteur INT(5);
000400		read producteur;
000500		dow not %eof;
000600		compteur += 1;
000700		read producteur;
000800		enddo;
000900		DSPLY (%char(compteur));
001000		*inLR = *on;

ici, ctrl+espace ->

Dcl-Ds path fmt TEMPLATE qualified;

<ul style="list-style-type: none">Dcl-dsDcl-subfDcl-sDcl-prDcl-piDcl-parmDcl-fDcl-c	DCL-DS Define a data structure inz(*ALLx'00'); inz(*ALLx'00'); inz(0); in, le sérateur
--	--



RDI 9.0.1

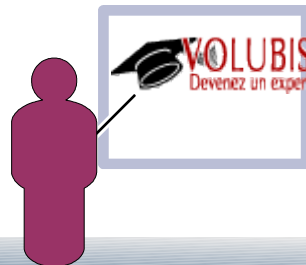
La fenêtre structure analyse bien ces nouvelles déclarations

The screenshot displays the RDI 9.0.1 interface. The main window shows a code editor for 'QZIPP.RPGLE' with the following content:

```
Ligne 9      Colonne 15  Replacer
..... Free-Form+++++
000100      Dcl-Ds path_fmt TEMPLATE qualified;
000200          CCSID          Int (10)      inz;
000300          // 0 = CCSID du JOB
000400          pays            Char (2)       inz (*ALLx'00');
000500          // x'0000' = pays du job
000600          langage        Char (3)       inz (*ALLx'00');
000700          // x'000000' = langue du job
000800          reserve1       Char (3)       inz (*ALLx'00');
000900          typ_indicateur  Int (10)      inz (0);
001000          // 0 => path_name contient un chemin, le sérateur est sur 1
001100          // 1 => path_name contient un pointeur, le séparateur est sur 1
001200          // 2 => path_name contient un chemin, le séparateur est sur 2
001300          // 3 => path_name contient un pointeur, le séparateur est sur 2
001400          path_len        Int (10);
001500          // lg du chemin
001600          path_delimiter  Char (2)       inz ('/');
001700          // si le séparateur est sur 1, c'est le premier caractère
001800          reserve2       Char (10)      inz (*ALLx'00');
001900          path_name       Char (1024);
001901      End-Ds;
```

The right-hand side of the image shows the 'Structure' window, which displays the following data structure definition:

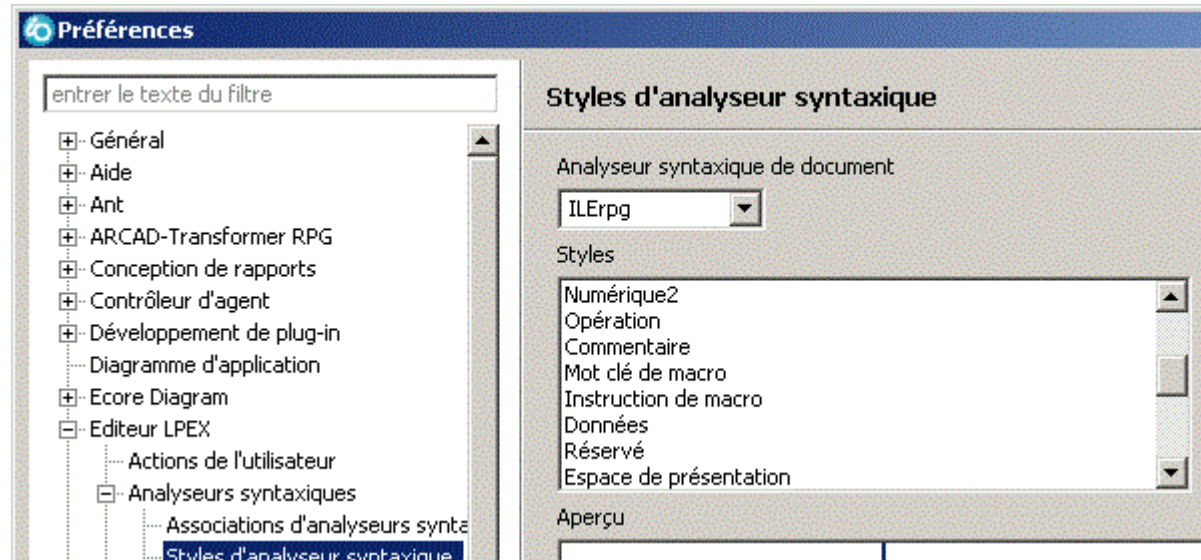
- Définitions globales
 - Structures de données
 - path_fmt : QUALIFIED TEMPLATE
 - CCSID : Entier (10,0)
 - pays : Caractère (2)
 - langage : Caractère (3)
 - reserve1 : Caractère (3)
 - typ_indicateur : Entier (10,0)
 - path_len : Entier (10,0)
 - path_delimiter : Caractère (2)
 - reserve2 : Caractère (10)
 - path_name : Caractère (1024)
 - 30
 - 31
 - 53
 - 54
 - chemin_in : LIKEDS(path_fmt)
 - chemin_out : LIKEDS(path_fmt)
 - Zipoption : QUALIFIED
 - errcodeDS : QUALIFIED
 - Indicateurs



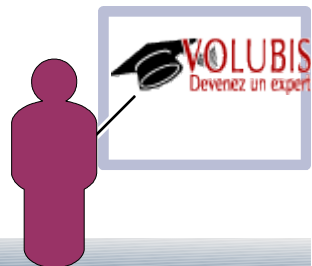
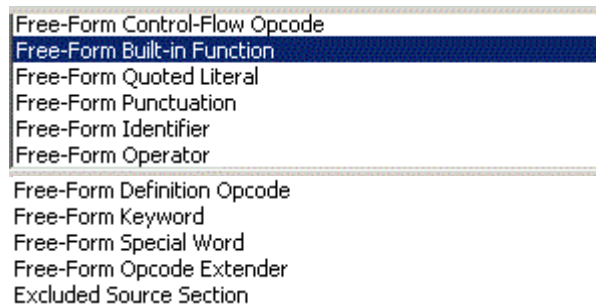
RDI 9.0.1

La colorisation syntaxique aussi

en version 8,5,1 vous étiez
Limités aux options suivantes



Voici ce que vous propose en plus la version 9



RDI 9.0.1

D'ailleurs, vous avez plus d'exemples de code dans le source affiché lors du paramétrage (et actualisé en fonction de vos choix.)

```
DCL-F myfile DISK(*EXT) USAGE(*UPDATE)
      EXTDESC('MYLIB/MYFILE');
DCL-DS *n; // This is a comment
      message CHAR(100) INZ('hello');
      dateDue DATE(*ISO) INZ(D'2013-01-02');
END-DS;
IMYFMT          01
I
C              MOVE          'Hello'          MYFLD          101112
C              SETON          LR
/COPY LIBRARY/FILE(MEMBER)
  eval x = 5;
  VALUE = 10;
  /IF DEFINED(condition)
    d1 = d2 + %DAYS(1);
  /ELSE
    d1 = d2 - %MONTHS(1);
  /ENDIF
READ(E) myfile;
IF NOT %EOF(myfile) AND NOT %ERROR;
  process ('234' : 5 : name : %SUBST(addr : 2)
          : d1 > d2);
ENDIF;
```



RDI 9.0.1

Divers

IBM annonce une version 9.1.0 disponible début juin

la liste des messages d'erreur dans l'éditeur (en rose) est automatiquement effacée lors d'une compilation.

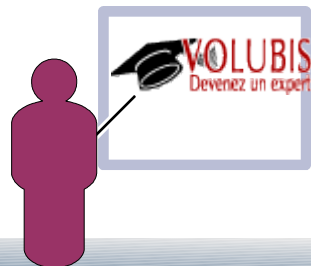
Mise en évidence des ELSE et des WHEN lors de l'affichage de l'imbrication (*Ctrl+MAJ+O*)

```
If *IML1 = *ON;  
    TOTC = 0;  
    QTTOTC = 0;  
    Chain CODE FICH2F1;  
    *IN50 = not %Found;  
    Write FCODE;  
else;  
    TOTC = 0;  
EndIf;
```

Rappel, il est possible de faire apparaître un lien vers une déclaration de fonction/procédure, en 8.5 (touche ctrl enfoncée, on peut cliquer sur le nom)

C'est désormais possible pour les procédures externes et les sous programmes

```
exsr sflsaisie;  
enddo; sflsaisie
```



RDI 9.0.1

Démonstration

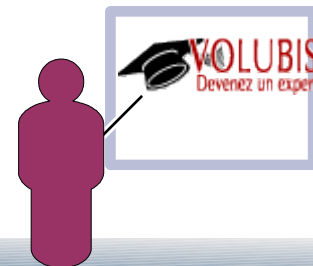
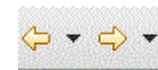
On clique sur traitement

```
exsr traitement;  
  
exfmt f2;  
enddo;  
*inlr = *on;  
  
// gestion d'une saisie, pré  
// =====  
begsr traitement;
```

le curseur se déplace sur begsr

```
eval chemin2 = 'toto';  
exsr traitement;  
  
exfmt f2;  
enddo;  
*inlr = *on;  
  
// gestion d'une saisie, préparation  
// =====  
begsr traitement;
```

il est désormais possible de revenir à la situation précédente par ces flèches
(ou *Alt+flèche gauche*, *Alt+flèche droite*)



Conversion de code existant

Deux produits proposent la conversion de votre patrimoine applicatif existant

Linoma



RPG Toolbox - Modernize your RPG programs



RPG Toolbox will greatly improve the productivity of developers who write and maintain software on the IBM i (iSeries). The Toolbox allows you to modernize your RPG programs, write applications faster and maintain source code more effectively.

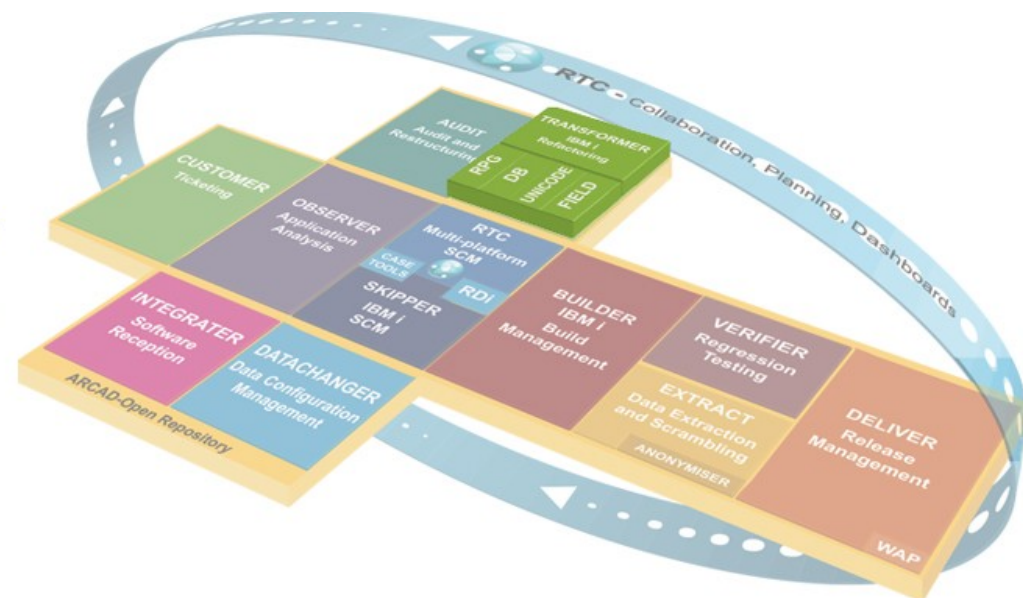
Now Available! Also convert H, F, D and P specifications to Totally Free Form RPG.

Et Arcad



ARCAD
Transformer

IBM i Refactoring Tools

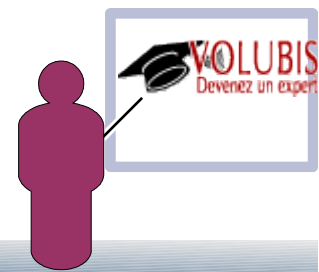
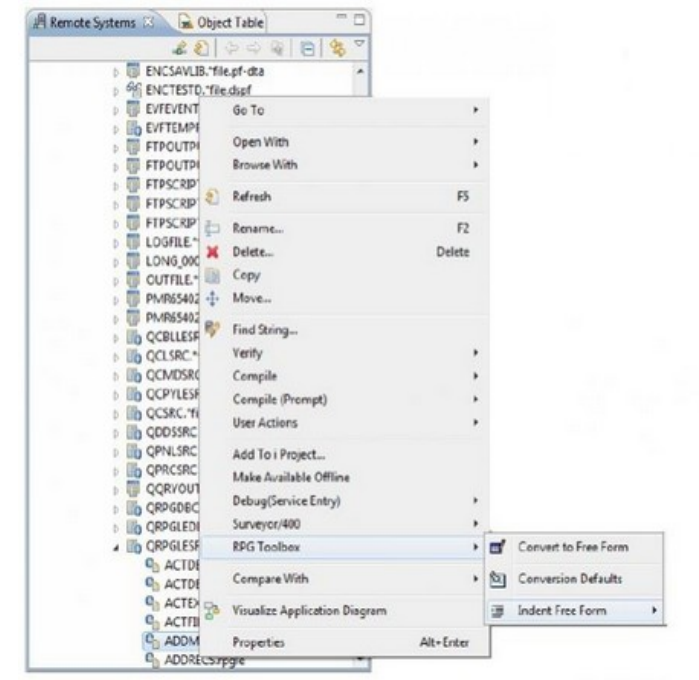
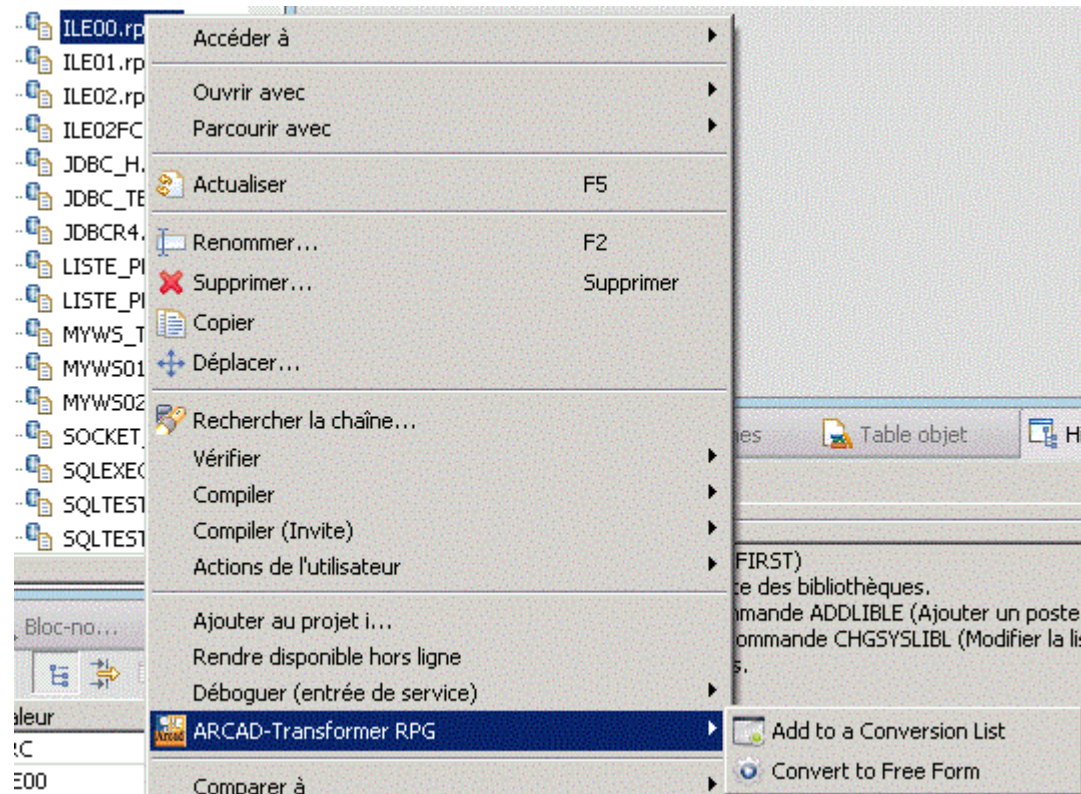


Conversion de code existant

Les deux fonctionnent en 5250 ou sous RDI

Linoma RPG Toolbox


Arcad Transformer



Conversion de code existant

Quelques exemples de code transformé (ici par Arcad)

```
c 50          add 1          p2
c N50        add 2          p2
```

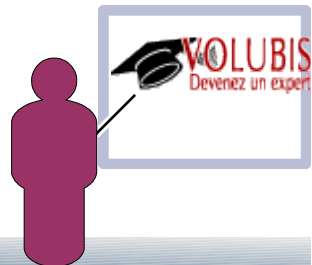


```
If *In50;
  p2 += 1;
Endif;
If not *In50;
  p2 += 2;
Endif;
```

```
C      CLE      CHAIN      fichier      90
```



```
CHAIN CLE fichier;
*IN90 = %EOF(fichier);
```



Conversion de code existant

C

seton

LR



```
*INLR = '1';
```


c

p2

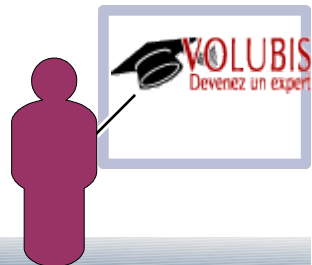
comp

0

50




```
*IN50 = (p2 = 0);
```



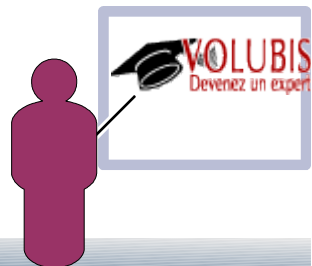
Conversion de code existant

c	debut	tag			
C		add	1	I	
C	i	comp	100		82
c	82	goto	debut		



```
Dcl-S ATag          Char(14);


DoU ATag <> 'DEBUT';
  // branch when ATag = 'DEBUT'
  ATag = *Blanks;
  I += 1;
  *IN82 = (i < 100);
  If *In82;
    ATag = 'DEBUT';
  Iter;
  Endif;
EndDo;
```



Conversion de code existant

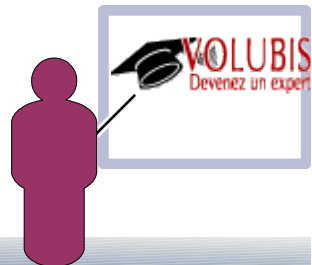
Attention !

```
C      debut          tag
C      * I passe de 99 à 00 (2 dt 0) sans broncher ==> 1e pgm boucle
C      add            1          I          2 0
C      i              comp      100
C 82      goto        debut          82
```



```
Dcl-S ATag          Char(14);
Dcl-S I             Packed(2:0);


DoU ATag <> 'DEBUT';
  // branch when ATag = 'DEBUT'
  ATag = *Blanks;
  // dépassement de capacité, le pgm plante -> RNQ0103
  // Cible pour opération numérique trop petite pour contenir résultat
  I += 1;
  *IN82 = (i < 100);
  If *In82;
    ATag = 'DEBUT';
  Iter;
Endif;
EndDo;
```



Conversion de code existant


Autres exemples

D	anmois	s	4	
C		move1	*year	anmois
C		move	*month	anmois

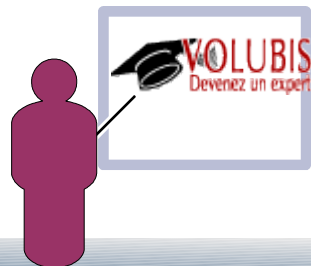


```
Dcl-S anmois Char(4);  
  
anmois = %EditC(*year:'X');  
%Subst(anmois:3:2) = %EditC(*month:'X');
```

C		MOVEA	'010'	*IN(80)
---	--	-------	-------	---------



```
Dcl-S pAToArrStr Pointer;  
  
Dcl-S AToArrStr Char(65535)  
Based(pAToArrStr);  
  
pAToArrStr = %Addr(*in(80));  
%Subst(AToArrStr:1:3) = '010';
```



Conversion de code existant


KLIST, PLIST

C	CLE1	KLIST	
C		KFLD	societe
C		KFLD	NOCLI
C	CLE1	CHAIN	fichier



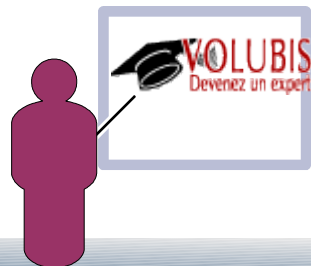
CHAIN (societe : nocli) fichier

C	CALL	'QCMDEXC'		
c	PARM		cmd	50
c	PARM		cmdl	15 5



Dcl-Pr Pgm_QCMDEXC EXTPGM('QCMDEXC');
 Cmd Char(50);
 Cmdl Packed(15,5);
End-Pr;

Pgm_QCMDEXC(cmd : cmdl);



Conversion de code existant

Divers

C BITON '0123' X



```
X = %BitOr(X:X'F0');
```

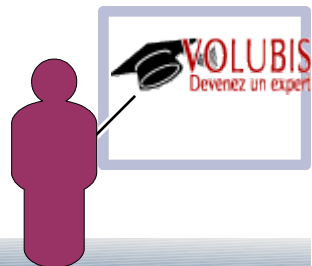
C MLLZO 'D' X



```
C MLLZO 'D' X
```

Reste en Spécif C

Qui utilise encore cela ?



Conclusions

Profitez en pour « faire bouger les lignes»

- Nouvelle syntaxe
 - Plus proche des autres langages
 - Plus simple pour intégrer de jeunes recrues
- Nouveaux outils
 - Plus puissants
 - Plus ouverts (RDI est basé sur eclipse)
- A vous de voir si vous prenez position pour les nouveaux développements uniquement ou si vous convertissez l'existant
- Dans tous les cas... formez vous ! ;-)

→ *Voyez notre cours en ligne du 16 Juin*

