

Université IBM i 2017

17 et 18 mai – IBM Client Center de Bois-Colombes

S13 – Du RPG moderne pour des développeurs modernes

Mercredi 17 mai – 16h00-17h30

Nathanaël Bonnet – Gaia



Gaia

- Conseil et formation IBM i depuis 1995
 - Inter et intra entreprise
- Base de connaissance en ligne
 - <http://know400.gaia.fr>
- Organisateur des matinées 400 iday
 - <http://www.gaia.fr/400iday-3>



<http://www.gaia.fr>

<http://twitter.com/GaiaFrance>

Moderne

RPG moderne

■ Définition

- Qui appartient au temps présent ou à une époque relativement récente
- Qui bénéficie des progrès les plus récents
- Qui est fait selon les techniques, les règles et le goût contemporains, par opposition à ancien

■ Etymologie

- (XVe siècle) Du latin modernus, dérivé de modus (« mode »)

RPG moderne

- Pourquoi du RPG moderne ? Quels objectifs ?
 - Lisibilité du code
 - Assurer la maintenance
 - Réduire les temps de maintenance
 - Profitez des évolutions du langage
 - Nouveaux codes opérations, nouvelles fonctions intégrées
 - Nouvelles syntaxes : format libre
 - Intégration des langages (SQL, Java)
 - Se mettre en capacité de répondre à de nouveaux besoins
 - Exposition de traitements sous forme de web services, procédures cataloguées ...
 - Usage d'outils de modernisation
 - Apprentissage facilité
 - Les concepts du RPG moderne sont connus par les jeunes informaticiens, pas le cycle GAP en format fixe !

RPG moderne





RPG moderne

```

H bnddir('ACCRCV') dftactgrp(*no)
Fcustfile          ctl-opt bnddir('ACCRCV');
Freport
D custDs           dcl-f custfile usage(*update);
D today           dcl-ds custDs likerec(custRec);
/copy invoices
C
C
C read custfile custDs;
C dow not %eof;
C     if dueDate > %date(); // overdue?
C     sendOverdueNotice ();
C     write reportFmt;
C/exec sql         exec sql insert :name, :duedate into
C+               mylib/myfile;
C/end-exe         endif;
C read custfile custDs;
C enddo;
C inlr = '1';
C
C snr             dcl-proc sendOverdueNotice;
C               /copy invoices
C               sendInvoice (custDs : IS_OVERDUE);
C               end-proc;
C

```

RPG moderne

- En synthèse, le RPG moderne est
 - Libre
 - Procédural (ILE)
 - Respecte des conventions de nommage
 - Commenté
 - Utilise des DS qualified et template, DS imbriquées
 - Utilise du SQL
 - N'utilise pas d'indicateur
 - Utilise des fonctions intégrées
 - ...

Fonctionnalités modernes ... ou juste cool !

Free form

- Nouvelle syntaxe TOTALEMENT libre !

```
**free
```

```
ctl-opt bnmdir('ACCRCV');
```

```
dcl-f custfile usage(*update);
```

```
dcl-ds custDs likerec(custRec);
```

```
dcl-f report printer;
```

```
read custfile custDs;
```

```
dow not %eof;
```

```
  if dueDate > %date(); // overdue?
```

```
    sendOverdueNotice ();
```

```
    write reportFmt;
```

```
    exec sql insert :name, :duedate into  
      mylib/myfile;
```

```
  endif;
```

```
  read custfile custDs;
```

```
enddo;
```

```
inlr = '1';
```

```
dcl-proc sendOverdueNotice;
```

```
  /copy invoices
```

```
  sendInvoice (custDs : IS_OVERDUE);
```

```
end-proc;
```

Free form

■ Principaux éléments

- `**free` nécessaire en colonne 1 pour avoir une syntaxe totalement libre
 - Vous pouvez alors commencer à écrire en colonne 1
- Les directives de pré-compilation `/free` et `/end-free` ne sont plus nécessaires si vous mixez libre et fixe
 - Le compilateur détecte automatiquement les syntaxes
 - Pas de `**free` dans ce cas
- De nombreuses syntaxes sont simplifiées
 - Particulièrement pour les déclarations de fichiers
- Des fonctionnalités non supportées
 - Cycle

Free form

■ Fichiers

Fcustfile	if	a	e	k	disk
Fqprint	o	f	132		printer
Freport	o	e			printer
Fscreen	cf	e			workstn

```
dcl-f custfile keyed usage(*input : *output);  
dcl-f orders; // usage(*input) par défaut  
dcl-f qprint printer(132);  
dcl-f screen workstn;  
dcl-f year_end_report printer  
      oflind(overflow)  
      extdesc('YERPT')  
      extfile(*extdesc);
```

Free form

■ Variables simples

```
// pour DSPLY
dcl-s msg char(52) ;

// Compteur entier binaire signé sur 1 octet
dcl-s cpt int( 3 ) inz( 0 ) ;

// Indicateur d'erreur ?
dcl-s isError ind inz(*off) ;

// Tableau de 50 messages d'erreur, initialisé par défaut
dcl-s errorList char(7) dim(MAX_ARRAY_SIZE) inz ;
// Pointeur sur le tableau errorList
dcl-s errorListPtr pointer inz( %addr( errorList ) ) ;
// Variable basé sur le pointeur
dcl-s error char(350) based(errorListPtr) ;
```

Free form

■ DS

```
// dcl-subf est facultatif
// POS est équivalent à OVERLAY. Si non renseigné (OVERLAY:*NEXT)
dcl-ds adherent ;
    nom char(25);           // nom
    prenom char(25) pos(26); // prénom
    dcl-subf tel char(10) dim(5) ; // liste des téléphones
end-ds;

// DS modèle qualifiée
dcl-ds dept_t len(50) template qualified ;
    nom char(20) inz('?'); // nom du service
    responsable zoned(10) inz ; // matricule responsable
end-ds;

// DS définie par une DS modèle
dcl-ds dept likeds( dept_t ) inz ;

// DS contenant une autre DS
dcl-ds entreprise qualified;
    nom char(30) inz('Non renseigné');
    siren zoned(9) inz;
    dept_list likeds(dept_t) dim(20) inz(*likeds) ;
end-ds;
```

Indicateurs

- Il est possible de se passer totalement des indicateurs
 - Pour des DSPF/PRTF : INDDS
 - Pour les indicateurs *in : les nommer

```
dcl-s pIndicators Pointer Inz(%Addr(*In));
dcl-ds DspInd Based(pIndicators);
  Exit_03 Ind Pos(3);
  Return_12 Ind Pos(12);
  Error_31 Ind Pos(31);
  StDateError_32 Ind Pos(32);
  EndDateError_33 Ind Pos(33);
end-ds;
```

```
Exit_03 = *on ; // au lieu de *in03 = *on
```

```
dow not Exit_03 and not Return_12 ;
  // ...
enddo ;
```

```
Error_31 = ( StDateError_32 or EndDateError_33 ) and
           not ( Exit_03 or Return_12 ) ;
```

ILE – procédure

- Code plus lisible, plus autonome, plus facilement réutilisable

```
// Vider le sous-fichier de messages
// -----
dcl-proc viderSfMsg export ;
  dcl-pi *n ind ;
    p_programme char(10) const ;
  end-pi;

  dcl-ds l_error likes( ERRRC0100_t ) inz( *likes ) ;
  // suppression des messages dans la file liée au programme
  QMHRMVP( p_programme :
    0 :
    *blank :
    '*ALL' :
    l_error ) ;
  return l_error.exceptionID <> *blanks ;
end-proc;
```


ILE – export(*dclcase)

- Permet d'imposer la casse
 - Pour des fonctions externes ou internes

```
// proto pour fonction "system" de la bibliothèque standard C
dcl-pr system int(10) extproc(*dclcase) ;
    command pointer value options(*string);
end-pr;

// est équivalent à
dcl-pr system int(10) extproc('system') ;
    command pointer value options(*string);
end-pr;
```

ILE – Java

■ Exemple : base64

```
// java.util.Base64.Encoder :  
// public static Base64.Encoder getEncoder()  
dcl-pr getEncoder object( *java : 'java.util.Base64.Encoder' )  
           extproc( *java : 'java.util.Base64.Encoder' : 'getEncoder' )  
           static  
  
end-pr ;
```

```
// java.util.Base64.Encoder :  
// public byte[] encode(byte[] src)  
dcl-pr encode varchar( 1000000 )  
           extproc( *java : 'java.util.Base64.Encoder' : 'encode' ) ;  
           src    varchar( 1000000 ) const ;  
  
end-pr ;
```

```
dcl-s enc object( *java : 'java.util.Base64.Encoder' ) ;  
dcl-s dst varchar( 1000000 ) ;
```

```
enc = getEncoder() ;  
dst = encode( enc : 'Valeur à encoder' ) ;
```

SQL embarqué – Expressions régulières

- Par exemple tester la validité d'une adresse mail

```
dcl-s mail varchar(50) inz('nbonnet@gaia.fr') ;  
dcl-s valide char(1) inz('N') ;
```

```
exec sql  
  SELECT '0'  
  into :valide  
  FROM sysibm.sysdummy1  
  WHERE REGEXP_LIKE(:mail,  
                    '^[a-z0-9._-]+@[a-z0-9._-]{2,}.[a-z]{2,4}$',  
                    'i');
```

```
if valide = '0' ;  
  // -- traiter le mail  
else ;  
  // -- logger l'erreur  
endif ;
```

SQL embarqué – Result Set

- Permet désormais de récupérer les result set des procédures SQL

```
// pour récupérer le result set
dcl-s resultSet SQLTYPE(RESULT_SET_LOCATOR) ;

// appel procédure
exec SQL
  call MyProc(:Region);
// retrouver le curseur via le result set
exec SQL associate result set locator (:resultSet) with procedure MYPROC;
exec SQL
  allocate C1 cursor for result set :resultSet;

// parcourir le curseur
exec SQL
  fetch next from C1 into :Row;
dow %subst(sqlstt:1:2)='00' or
  %subst(sqlstt:1:2)='01';
  // Faire qq chose ici ...
  exec SQL
    fetch next from C1 into :Row;
enddo;
exec SQL
  close C1;
```

SQL embarqué – HTTP

- Un ensemble de fonctions HTTP est disponible via DB2

```
// déclaration CLOB
dcl-s fichierHTML SQLTYPE( CLOB_FILE ) ;
// créer un fichier .html sur l'IFS
fichierHTML_name = '/home/NB/gaia.html' ;
fichierHTML_n1 = %len( %trim( fichierHTML_name ) ) ;
fichierHTML_fo = SQFCRT ;
// lecture de données HTML
exec sql
  values SYSTOOLS.HTTPGETCLOB('http://www.gaia.fr', '')
        into :fichierHTML ;
if ( SqlCode = 0 ) ; // Erreur ?
  ...
```

SQL embarqué – XML

- Le stockage, décomposition et création de composition XML sont supportés par DB2

```
// déclaration CLOB
dcl-s donneesXML SQLTYPE( XML_CLOB : 10000 ) ;

// lecture de données XML
exec sql select donnees
           into :donneesxml
           from xml_data
           fetch first 1 rows only;

// Vérifier SQLCODE :
if ( SqlCode = 0 ) ;
  w = %subst( donneesXML_data : donneesXML_len ) ;
endif;

// Insérer une valeur XML
exec sql insert into xml_data( donnees )
           values( :donneesXML ) ;
```

DS qualifiées

- Utiliser des DS modèles et DS qualifiées
 - Permet d'avoir plusieurs DS avec les mêmes noms de sous-zones
 - Permet de déclarer plusieurs DS avec des structures identiques
 - Nécessaire pour la définition des prototypes

```
dcl-ds adresse_t template qualified ;  
  ligne1      char(35);  
  ligne2      char(35);  
  cp          char(5);  
  ville       char(25);  
end-ds ;
```

```
dcl-ds client_t template qualified ;  
  id          int(10) ;  
  raisonSociale char(50);  
  adresse     likeds(adresse_t) ;  
end-ds ;
```

DS qualifiées

```
dcl-ds adresse1 likeds( adresse_t ) inz( *likeds ) ;  
dcl-ds adresse2 likeds( adresse_t ) inz( *likeds ) ;  
dcl-ds client likeds( client_t ) ;
```

```
adresse1.ligne1 = '67 rue du Bourbonnais' ;  
adresse2.ligne1 = '17 avenue de l''Europe' ;  
adresse1.ligne1 = adresse2.ligne1 ;
```

```
client.adresse = adresse1 ;  
client.id = 1 ;  
client.adresse.ville = 'Paris' ;
```


DS imbriquées

- Il est désormais possible d'imbriquer dcl-ds dans dcl-ds

```
dcl-ds Universite Qualified;
  nb_session int(5) ;
  debut      date ;
  fin        date ;
  dcl-ds session dim(50) ;
    id        char(3) ;
    titre     varchar(100) ;
    description varchar(1500) ;
    speaker   varchar(50) ;
    debut     timestamp ;
  end-ds ;
end-ds;
```

DS imbriquées

- Préalablement il fallait faire

```
dcl-ds session_t template Qualified ;  
  id          char(3) ;  
  titre       varchar(100) ;  
  description varchar(1500) ;  
  speaker     varchar(50) ;  
  debut       timestamp ;  
end-ds ;
```

```
dcl-ds Universite Qualified;  
  nb_session int(5) ;  
  debut      date ;  
  fin        date ;  
  session    likes( session_t ) dim(50) ;  
end-ds;
```

DS – clé

- En remplacement des KLIST / KLFD

```
// Key list for STOCK file
dcl-ds Key likerec(STOCKR : *key);

// if the record is locked by another job.
key.STOCKNO = stockno;
key.ITEMNBR = itemnbr;
Chain %kds(Key) STOCK;
// ...
key.STOCKNO = stockno;
Chain %kds(Key:1) STOCK;
```

DS – I/O

- Manipulation d'enregistrements dans des DS

```
dcl-f Parts Disk(*ext) usage(*update: *output) Keyed qualified;  
dcl-ds partsDS likerec(PARTS.partr);
```

```
Chain PartNo parts partsDS;  
partsDS.PartNum = PartNo;
```

```
Select;  
When Action = UpdRecord And %FOUND(Parts);  
    Update parts.partr partsDS;  
When Action = AddRecord And Not %FOUND(Parts);  
    Write parts.partr partsDS;  
Other;  
    Action = Error;  
EndSl;
```

XML-INTO

- xml-into permet d'alimenter une DS depuis un flux XML
 - Également de façon événementielle

```
dcl-s xml char(4096) ;
```

```
dcl-ds script qualified ;  
  dcl-ds qsh ;  
    rows char(3) ;  
    row char(50) dim(99) ;  
  end-ds ;  
end-ds;
```

```
xml = '<?xml version="1.0"?>' +  
      '<script>' +  
        '<qsh rows="on">' +  
          '<row>error.log</row>' +  
          '<row>error20170502.log</row>' +  
          '<row>indicateurs.csv</row>' +  
        '</qsh>' +  
      '</script>' ;
```

```
xml-into script %XML( xml : 'doc=string path=script allowmissing=yes' ) ;
```

Gestion des erreurs

- Via un bloc MONITOR ... ON-ERROR ... ENDMON

```
dcl-s dateCar char(10) inz('2017-05-17') ;  
dcl-s date      date ;
```

```
test(ed) *iso dateCar ;  
if not %error() ;  
    date = %date( dateCar : *iso ) ;  
else ;  
    date = *hival ;  
endif ;
```

```
monitor ;  
    date = %date( dateCar : *iso ) ;  
on-error ;  
    date = *hival ;  
endmon ;
```

Déclaration des fichiers et variables

- Plus d'ordre de déclaration entre fichiers et variables

```
dcl-f orders usage (*update : *output) keyed;  
dcl-ds orders_dsi likerec (ordersR:*input);  
dcl-ds orders_dso likerec (ordersR:*output);  
dcl-s num_ororders int(10);
```

```
dcl-f report printer;  
dcl-ds report_ds likerec (reportR:*output);
```

CCSID

- RPG dispose de nouveaux mots-clés pour affiner la gestion des CCSID

```
CTL-OPT CCSID(*CHAR : *UTF8) CCSID(*GRAPH : 835) ;
```

```
DCL-S char1 char(10);
```

```
DCL-S graph1 graph(10);
```

```
/SET CCSID(*CHAR : 37) CCSID(*UCS2:1200)
```

```
DCL-S char2 char(10);
```

```
DCL-S char3 LIKE(char1) CCSID(*DFT);
```

```
/RESTORE CCSID(*CHAR)
```

```
DCL-S greeting varchar(20) CCSID(*UTF8) inz('Hello') ;
```

```
DCL-S message UCS2(30) inz('Successful operation') ;
```

```
DCL-S result varchar(100) inz('Successful operation') ;
```

```
result = 'The customer's name is ' + %CHAR(char1) + '.';
```


```
result = %CHAR(greeting : *JOB RUN);
```




```
result = %CHAR(message : *JOB RUN);
```



ALIAS – DS




- Permet l'utilisation des noms longs SQL

```
dcl-ds client1 extname('CLIENTS')          qualified end-ds;  
dcl-ds client2 extname('CLIENTS') alias  qualified end-ds;
```

▲  client1 : QUALIFIED

-  ID : Binaire (9,0)
-  NOM : Caractère (25)
-  NOMJF : Caractère (25)

▲  client2 : QUALIFIED ALIAS

-  ID : Binaire (9,0)
-  NOM : Caractère (25)
-  NOM_JEUNE_FEMME : Caractère (25)

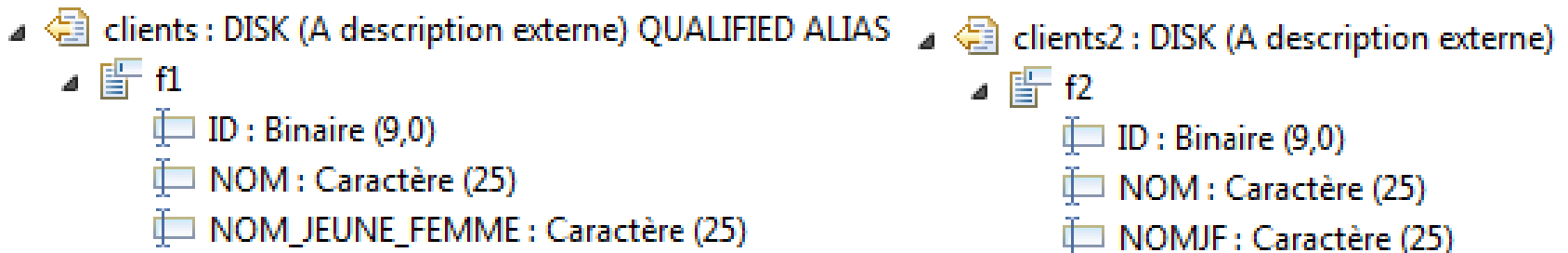
ALIAS – Fichiers

- Permet l'utilisation des noms longs SQL

```
dcl-f clients qualified rename(clients:f1) alias ;  
dcl-f clients2 extdesc('CLIENTS') rename(clients:f2) ;
```

```
dcl-ds enregCli likerec(clients.f1) ;
```

```
read clients.f1 enregCli ;  
read f2 ;
```



Valeurs nulles

- Evolutions de NULLIND

```
ctl-opt alwnull(*usrctl) ;
```

```
dcl-s noemp char(6) nullind ;
```

```
// mettre à null :
```

```
%nullind( noemp ) = *on ;
```

```
// Valoriser
```

```
%nullind( noemp ) = *off ;
```

```
noemp = '000012' ;
```

```
// tester la valeur
```

```
if not %nullind(noemp) and noemp <> '00010' ;
```

```
    dsply ( 'n° employé : ' + noemp ) ;
```

```
endif ;
```

Valeurs nulles

- Avec une variable en tant qu'indicateur de nullité

```
ctl-opt alwnull(*usrctl) ;

// variables locales
dcl-s numemp      char(6) nullind(numemp_null) ;
dcl-s numemp_null ind ;

// Mettre à null
%nullind( numemp ) = *on ; // ou numemp_null = *on ;

// valoriser
%nullind( numemp ) = *off ; // ou numemp_null = *off ;
numemp = '00010' ;

// tester la valeur
if not %nullind(numemp) and numemp <> '00010' ;
    dsply ( 'n° employé : ' + numemp ) ;
endif ;
```

Faites le avec style !



Objectif

■ Lisibilité

```

Colon
SEU=
FMT C
0002.1
_0002.1
0002.1
0002.1
init() ;
// List of invoices to process
// -----
dow getInvoice( curInvoice ) ;

    // Calculate each line for this invoice
    if ( not calculateItems( curInvoice : p_date ) ) ;
        // if an error occurs : continue with next invoice
        iter ;
    endif ;

    // calculate invoice discount
    calculateInvoiceDiscount( curInvoice : p_date ) ;

    // update invoice
    setInvoice( curInvoice ) ;

enddo ;

// exit !
return ;

end-proc ;

```

```

PGLESRC
505004B
+HiLoEq

```

```

rifier'

```

Objectifs

- 1 unique objectif
 - Lisibilité !



Généralités

■ Commentaires

- Un code lisible n'a besoin que de peu de commentaires
 - But du programme et principales étapes
 - Résumé d'une procédure, détail des paramètres
 - Expliciter une technique, un algorithme particulier

■ Proscrire le code obsolète

- Ne plus utiliser les codes opération non supportés en format libre
 - MOVE, GOTO, ADD ... CALL
- A équivalence, utiliser la fonction intégrée plutôt que le code opération
 - XLATE vs %xlate
- Utiliser des vraies dates, l'ensemble des types de données
 - %date, %subdt, %diff ...
- Bannir les fichiers décrits en interne
- ...



Casse

- Eviter de tout saisir en majuscule

```
CTL-OPT BNDDIR('ACCRCV');

DCL-F CUSTFILE USAGE(*UPDATE);
DCL-DS CUSTDS LIKERECL(CUSTREC);
DCL-F REPORT PRINTER;

READ CUSTFILE CUSTDS;
DOW NOT %EOF;
  IF DUEDATE > %DATE(); // OVERDUE?
    SENDOVERDUENOTICE ();
    WRITE REPORTFMT;
    EXEC SQL INSERT :NAME, :DUEDATE INTO
      MYLIB/MYFILE;
  ENDIF;
READ CUSTFILE CUSTDS;
ENDDO;
INLR = '1';
```

```
ctl-opt bnmdir('ACCRCV');

dcl-f custfile usage(*update);
dcl-ds custDs likerecl(custRec);
dcl-f report printer;

read custfile custDs;
dow not %eof;
  if dueDate > %date(); // overdue?
    sendOverdueNotice ();
    write reportFmt;
    exec sql insert :name, :duedate into
      mylib/myfile;
  endif;
read custfile custDs;
enddo;
inlr = '1';
```

– Plusieurs possibilités

```
CHAIN PostalCode CityMaster;
chain postalCode cityMaster;
Chain PostalCode CityMaster;
```

Indentation

- Indenter les déclarations et le code exécutable

```
Dcl-ds NumSS;  
  sexe char(1);  
  anneeNaiss zoned(2);  
  moisNaiss zoned(2);  
  casA char(5) pos(6); // Naissance en métropole  
    deptNaissMetropole char(2) pos(6);  
    codeOfficielCommuneNaiss char(3) pos(8);  
  casB char(5) pos(6); // Naissance en outre-mer  
    deptNaissOutreMer char(3) pos(6);  
    code2CommuneNaiss char(2) pos(9);  
  casC char(5) pos(6); // Naissance hors de France  
    naissHorsFrance char(2) pos(6);  
    idPaysNaissance char(3) pos(8);  
  rangNaiss char(3) pos(11);  
  cleControle char(2) pos(13);  
End-ds;
```

Indentation

- Indenter les déclarations et le code exécutable

```
select;

when function = 8 ; // création
  // Suivant notre chiffrage, le type retournée est le même, donc copie de la définition
  encoded_attr = decoded_attr ;

when function = 0 ; // INSERT => encodage
  invBytes(decoded_Data : encoded_Data : decoded_attr.sqlfpLength);

when function = 4 ; // SELECT => decodage
  // Seulement sur l'utilisateur est QSECOFR
  if user = 'QSECOFR';
    invBytes(encoded_Data : decoded_Data : encoded_attr.sqlfpLength);
  else;
    %subst(decoded_Data : 1 : encoded_attr.sqlfpLength) =
      %subst(encoded_Data : 1 : encoded_attr.sqlfpLength) ;
  endif;

other ;
  SQLSTATE = '38001';
  message = 'Demande non gérée';
endsl ;
```

Aligner

- Dans le cas d'instructions sur plusieurs lignes

```
dcl-proc datetimeToAxis export ;
  dcl-pi *n ind ;
    p_datetime      timestamp const ;
    p_xsddatetime   likeds(xsd_datetime) ;
  end-pi;

  // Traitement
  clear p_xsddatetime ;
  monitor ;
    // YYYY-mm-dd-HH.MM.SS => 19 caractères
    if strptime( %subst( %char(p_datetime) : 1 : 19 ) :
      '%G-%m-%d-%H.%M.%S' :
        p_xsddatetime.value) = *null ;
      clear p_xsddatetime ;
      return *off ;
    endif ;
  on-error ;
    clear p_xsddatetime ;
    return *off ;
  endmon ;
  return *on ;

end-proc;
```

Constantes

- A la place des valeurs littérales

```
// -- Gestion du clavier
dcl-c F1          const( x'31' ) ;
dcl-c F2          const( x'32' ) ;
dcl-c F3          const( x'33' ) ;
...
dcl-c ENTREE     const( x'F1' ) ;
dcl-c PAGEAR     const( x'F4' ) ;
dcl-c PAGEAV     const( x'F5' ) ;
dcl-c RETOUR     const( x'F8' ) ;

// -- Divers
dcl-c QUOTE const(''') ;

// -- Codes retour usuels
dcl-c CD_ERREUR  const('0') ;
dcl-c CD_OK      const('1') ;
```

Tableau chargé à la compilation

- Ne pas utiliser CTDATA
 - Sépare la déclaration des données
 - Préférer la syntaxe suivante

```
Dcl-ds JoursData;  
  *N Char(8) Inz('Lundi');  
  *N Char(8) Inz('Mardi');  
  *N Char(8) Inz('Mercredi');  
  *N Char(8) Inz('Jeudi');  
  *N Char(8) Inz('Vendredi');  
  *N Char(8) Inz('Samedi');  
  *N Char(8) Inz('Dimanche');  
  Jours Char(8) Dim(7) Pos(1);  
End-ds;
```

Au lieu de

```
dcl-s  jours char(8) dim(7) ctdata perrcd(7) ;
```

```
**ctdata jours  
Lundi   Mardi   MercrediJeudi   VendrediSamedi   Dimanche
```

Options de compilation SQL

■ SET OPTIONS

- Permet de mémoriser les options de compilation dans le source
- Attention, doit être la première instruction SQL dans le source

```
// Options SQL
exec sql set option naming = *sys ,
                    commit = *none ,
                    datfmt = *iso ,
                    dftrdbcol = nb ;
```

Imbrication

- Limiter l'imbrication des IF

```
// création de la BD
if not creerDB() ;
    return ;
endif ;

// population de la BD
if not peuplerDB( p_nbart : p_nbCde : p_nbLig ) ;
    return ;
endif ;

// calcul
if not calculer() ;
    return ;
endif ;

// sortie
return ;
```

```
// création de la BD
if creerDB() ;
    // population de la BD
    if peuplerDB( p_nbart : p_nbCde : p_nbLig ) ;
        // calcul
        if calculer() ;
            else ;
                return ;
            endif ;
        else ;
            return ;
        endif ;
    else ;
        return ;
    endif ;
endif ;

// sortie
return ;
```


Cycle GAP

- Utiliser le mot-clé main
 - Le compilateur ne génère plus le code nécessaire au cycle

```
// options
ctl-opt actgrp(*new)
      datfmt(*iso) timfmt(*iso)
      usrprf(*owner)
      pgminfo(*pcml:*module)
      option(*nodebugio)
      main(grosbatch);

// ce programme
dcl-pr grosbatch extpgm ;
  p_nbArt char(10) const ;           // nombre d'articles
  p_nbCde char(10) const ;           // nombre de commandes
  p_nbLig char(10) const ;           // nombre de lignes moyen par commande
end-pr;

// Corps principal
dcl-proc grosbatch ;
  dcl-pi *n ;
    p_nbArt char(10) const ;         // nombre d'articles
    p_nbCde char(10) const ;         // nombre de commandes
    p_nbLig char(10) const ;         // nombre de lignes moyen par commande
  end-pi ;
```

Cycle GAP

– Ou avec un nom différent

```
ctl-opt main(main)
      actgrp(*new)
      pgminfo(*pcml:*module) ;
```

```
dcl-pr main extpgm('WS_ACC_BD') ;
      parmout char(25) ;
end-pr;
```

```
dcl-proc main ;
      dcl-pi *n ;
      parmout char(25) ;
end-pi ;
```

ILE

- Construisez des applications modulaires
 - Programmation procédurale
 - Favorise la réutilisation du code
 - Permet d'organiser le code
 - Facilite la maintenance
 - Masque la complexité
- Programmes de service
 - Pour toutes vos procédures réutilisables
- Procédure
 - Utiliser au maximum les variables locales : réduire le nombre de variables globales
 - Transmettre toutes les valeurs en paramètres : lisibilité
 - Utiliser les valeurs de retour
- Prototyper tous les appels
 - Appels de programme : fiabilise le code par le contrôle de l'adéquation des paramètres à la compilation
 - Utiliser const plutôt que value
- Centraliser vos prototypes dans des membres à inclure par /include

Et encore ...

- Structures de contrôle
 - Utiliser SELECT ... WHEN ... OTHER ... ENDSL

- DS
 - Utiliser DIM et non OCCURS

- Fichiers
 - Utiliser la qualification des fichiers
 - Utiliser les DS pour les IO
 - Toujours indiquer le nom du fichier lors de l'utilisation des fonctions %error, %found, %eof, %equal


- ...

Outils

RDi !

- SEU est stabilisé en 6.1
 - Aucun support des nouveautés RPG depuis le 7.1
 - Anomalies syntaxiques
 - La compilation fonctionne

```
Colonnes . . . : 1 71 Edition                               NB/QRPGLESRC
SEU=>                                                       AXISTOOLS
FMT C .....CL0N01Factor1+++++0opcode&ExtFactor2+++++Result+++++Len++D+H
0001.21 // Déclarations des prototypes des procédures exportées du module
0001.22 // -----
0001.23 // pour inclusion des définitions AXIS
0001.24 /define AxisTools_AxisDefinitions
0001.25 /include AXISTOOLSP ;
0001.26
0001.27
0001.28 // Prototypes internes : C standard library (time.h)
0001.29 // -----
0001.30
0001.31 // typedef
0001.33 dcl-s size_t int(10) template ;
0001.34
0001.35 // strftime : Convert Date/Time to String
0001.36 // size_t strftime(char *s, size_t maxsize, const char *format,
0001.37 //                  const struct tm *timeptr);
0001.39 dcl-pr strftime like(size_t) extproc(*cwiden : *dclcase) ;

F3=Exit  F4=Invite  F5=Réafficher  F9=Rappel  F10=Curseur  F11=Basculer
F16=Répéter recherche  F17=Répéter remplacement  F24=Autres touches
L'entrée Niveau de contrôle est incorrecte. Niveau précédent pris par défaut. +
MA A MW 04/015 
```



RDi

Explorateur de systèmes distants - RemoteSystemsTempFiles\10.2.0.1\QSYS.LIB\NB.LIB\QRPGLSRC.FILE\AXISTOOLS.RPGLE - IBM Rational Developer for i

Fichier Éditer Source Compiler(N) Naviguer Recherche Projet Data Exécuter Fenêtre Aide

Activité : Autre activité

Accès rapide

Systèmes distants Equippe

- NB:com.ibm.etools.iseries.subsystems.qsys.objects
 - CN-neptune-com.ibm.etools.iseries.subsystems.qsys.objects
 - NB/*
 - NB/Fichiers source
 - EVFTMPF01:file.pf-src
 - EVFTMPF02:file.pf-src
 - QCLSRC:file.pf-src
 - QCLSRC2:file.pf-src
 - QCMDSRC:file.pf-src
 - QODDSRC:file.pf-src
 - QMNUSRC:file.pf-src
 - QPNLSRC:file.pf-src
 - QRPGLSRC:file.pf-src
 - AXISDATE.rpgle
 - AXISDATE2.rpgle
 - AXISDATE3.rpgle
 - AXISREST1.rpgle
 - AXISREST1S.sqlrpgle
 - AXISREST10.sqlrpgle
 - AXISREST2S.sqlrpgle
 - AXISREST3S.sqlrpgle
 - AXISREST4.rpgle
 - AXISREST5S.sqlrpgle
 - AXISREST6S.sqlrpgle
 - AXISREST7.rpgle
 - AXISREST8.sqlrpgle
 - AXISREST9.sqlrpgle
 - AXISREST1.rpgle
 - AXISTOOLS.rpgle
 - AXISTOOLS2.rpgle
 - AXISTOOLY.rpgle
 - AXISTOOLYP.rpgle
 - AXISTOOLZ.rpgle
 - AXISTOOLZP.rpgle
 - CHIFFRER.rpgle
 - CIIFNT.rpgle

```

Ligne 16      Colonne 1      Remplacement
000121 // Déclarations des prototypes des procédures exportées du module
000122 // -----
000123 // pour inclusion des définitions AXIS
000124 //define AxisTools_AxisDefinitions
000125 //include AXISTOOLS.P ;
000126
000127
000128 // Prototypes internes : C standard library (time.h)
000129 // -----
000130
000131 // typedef
000132 //
000133 //dcl-s size_t int(10) template ;
000134
000135 // strftime : Convert Date/Time to String
000136 // size_t strftime(char *s, size_t maxsize, const char *format,
000137 //                 const struct tm *timeptr);
000138 //
000139 //dcl-pr strftime like(size_t) extproc(*cwidenc : *dclcase) ;
000140 //      s          char(255)          options(*varsize) ;
000141 //      maxsize   like(size_t)       value ;
000142 //      format    pointer             options(*string) value ;
000143 //      tm        likeds(xsd_tm)     const ;
000144 //end-pr;
000145
000146
000147 // strftime : Convert String to Date/Time
000148 // char *strftime(const char *buf, const char *format, struct tm *tm);
000149 //
000150 //dcl-pr strftime pointer extproc(*cwidenc : *dclcase) ;
000151 //      charDate  pointer             options(*string : *trim) value ;
000152 //      format    pointer             options(*string : *trim) value ;
000153 //      tm        likeds(xsd_tm)     ;
000154 //end-pr;
000155
000156 // Procédures exportées du module
000157 // -----
000158
000159
000160
000161 // dateToAxis
000162 // =====
000163
000164
000165
  
```

Structure

entrer le texte du filtre

- Instructions de contrôle
- Définitions globales
- Structures de données
- Zones
- Prototypes
- Sous-procédures
 - dateToAxis : Indicateur (1) EXPORT
 - axisToDate : Indicateur (1) EXPORT
 - timeToAxis : Indicateur (1) EXPORT
 - axisToTime : Indicateur (1) EXPORT
 - datetimeToAxis : Indicateur (1) EXPORT
 - axisToDateTime : Indicateur (1) EXPORT

Détails du système distant

neptune_colruyt neptune

CHGCURLIB CURLIB(NB)
Bibliothèque en cours modifiée en NB.
Cause : La bibliothèque en cours de la liste des bibliothèques a été modifiée en NB.

Commande Normale

Inviter... Exécuter

Remplacement 16:1

RDi

- Rational Developer for i
 - Version actuelle 9.5.1 fix 2
 - Editeur graphique
 - Compatible Windows et Mac
 - Basé sur Eclipse



RDi

- Principales fonctionnalités
 - Aide en ligne (F1)

```

p_xsddate likeds(xsd_date)
p_date date ;
end-pi;

// Variables locales
dcl-ds l_tm likeds(xsd_tm)
dcl-s l_dateA char(11) inz;

// Traitement
clear p_date ;
if p_xsddate.isNil = *on ;
    return *off ;
endif ;
monitor ;
    l_tm = p_xsddate.value ;
    strftime(l_dateA : %len(1
p_date = %date( %subst( l
on-error ;
    clear p_date ;
    return *off ;
endmon ;
return *on ;

end-proc;

// timeToAxis
// =====
dcl-proc timeToAxis export ;
dcl-pi *n ind ;
            
```

Aide - IBM Rational Developer for i
 Recherche : OK Portée: Toutes les rubriques

Contenu

- ⊕ Rational Developer for i 9.5.1
- ⊕ Data Tools Platform User Documentation
- ⊕ Eclipse Marketplace User Guide
- ⊕ EGIt Documentation
- ⊕ Guide d'utilisation de la plateforme des ou
- ⊕ IBM Data Studio
- ⊕ ILE RPG Reference
 - ⊕ ILE RPG Reference
 - PDF file for ILE RPG Reference
 - ⊕ About ILE RPG Reference
 - ⊕ What's New
 - ⊕ RPG IV Concepts
 - ⊕ Definitions
 - ⊕ Specifications
 - ⊕ Operations, Expressions, and Functions
 - ⊕ Operations
 - ⊕ Expressions
 - ⊕ Built-in Functions
 - ⊕ %ABS (Absolute Value of Expres
 - ⊕ %ADDR (Get Address of Variabl
 - ⊕ %ALLOC (Allocate Storage)
 - ⊕ %BITAND (Bitwise AND Operati
 - ⊕ %BITNOT (Invert Bits)
 - ⊕ %BITOR (Bitwise OR Operation)
 - ⊕ %BITXOR (Bitwise Exclusive-OR)
 - ⊕ %CHAR (Convert to Character D
 - ⊕ %CHECK (Check Characters)
 - ⊕ %CHECKR (Check Reverse)
 - ⊕ %DATE (Convert to Date)
 - ⊕ %DAYS (Number of Days)
 - ⊕ %DEC (Convert to Packed Decir

ILE RPG Reference > ILE RPG Reference > Operations, Expressions, and Functions > Built-in Functions

%DATE (Convert to Date)

%DATE{(expression{:date-format})}

%DATE converts the value of the expression from character, numeric, or timestamp data to type date. The converted value remains unchanged, but is returned as a date.

The first parameter is the value to be converted. If you do not specify a value, %DATE returns the current system date.

The second parameter is the date format for character or numeric input. Regardless of the input format, the output is returned in *ISO format.

»For information on the input formats that can be used, see [Date Data Type](#). If the date format is not specified for character or numeric input, the default format is *ISO. For more information, see [DATFMT\(fmt{separator}\)](#).

If the first parameter is a timestamp, *DATE, or UDATE, do not specify the second parameter. The system knows the format of the input in these cases.

For more information, see [Information Operations](#) or [Built-in Functions](#).

Figure 1. %DATE Example

```

*..1...+...2...+...3...+...4...+...5...+...6...+...7...+...
/FREE

    string = '040596';
    date = %date(string:*MDY0);
    // date now contains d'1996-04-05'
/END-FREE
            
```



RDi

- Coloration
- Personnalisable

```

015200
015300  /* Parcourir l'ensemble des résultats */
015400  /* ----- */
015500
015600  /* Alimenter le pointeur vers la Key 1 ou
015700  CHGVAR      &L_KEY1P %ADDR(&L_KEY)
015800  CHGVAR      %OFFSET(&L_KEY1P) (%OFFSET(&L_
015900  /* Alimenter le pointeur vers le 1er récé
016000  CHGVAR      &L_JRNRCVP &L_KEY1P
016100  CHGVAR      %OFFSET(&L_JRNRCVP) (%OFFSET(&
016200
016300  DOFOR      VAR(&L_CPT) FROM(1) TO(&L_NBR
016400
016500  /* Suppression du récepteur si détaché
016600  IF          COND( &L_JRNRCVS ^= '1' &
016700  DLTJRNRCV  JRNRCV(&L_JRNRCVL/&L_JRN
016701  MONMSG     MSGID(CPA0000 CPD0000 CP
016800  ENDDO
016900
017000
017100  /* Récepteur suivant */
017200  CHGVAR      &L_JRNRCVP &L_JRNRCVP
017300  CHGVAR      %OFFSET(&L_JRNRCVP) (%OFFSET(&L_JRNRCVP) + +
017400  %SIZE(&L_JRNRCV))
017500
017600  ENDDO
017700
017800  ENDPGM

```

```

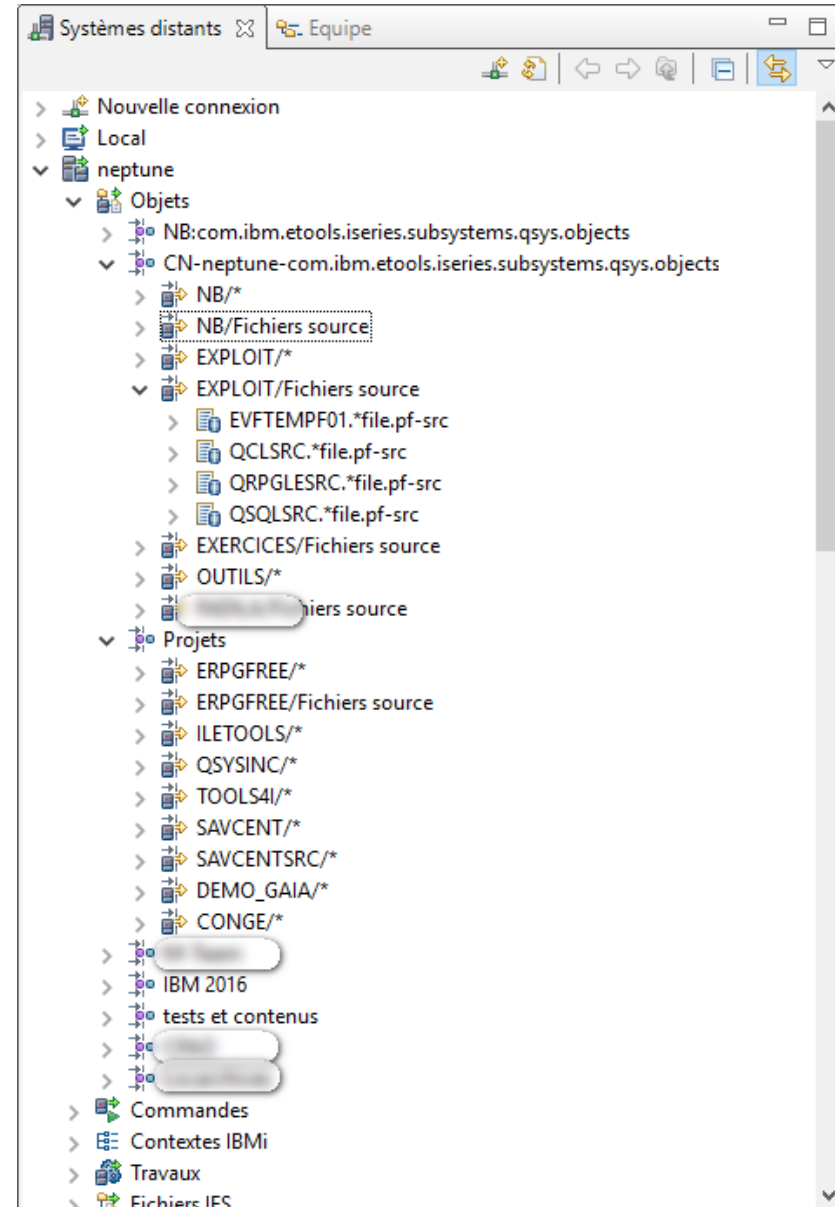
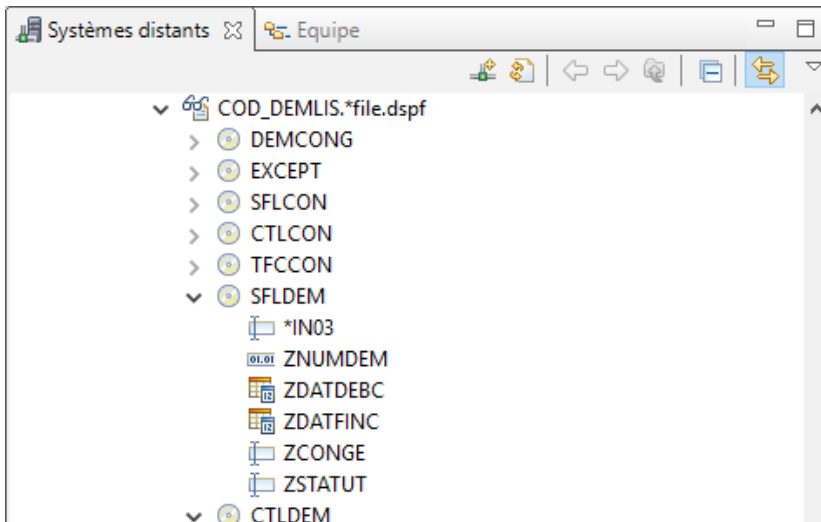
000258 // axisToDate
000259 // =====
000260
000261 dcl-proc axisToDate export ;
000262   dcl-pi *n ind ;
000263   p_xsddate likeds(xsd_date) const ;
000264   p_date     date ;
000265   end-pi;
000266
000267 // Variables locales
000268   dcl-ds l_tm     likeds(xsd_tm) ;
000269   dcl-s  l_dateA char(11) inz;
000270
000271 // Traitement
000272
000273   clear p_date ;
000274   if p_xsddate.isNil = *on ;
000275       return *off ;
000276   endif ;
000277   monitor ;
000278       l_tm = p_xsddate.value ;
000279       strftime(l_dateA : %len(l_dateA) : '%G-%m-%d' : l_tm);
000280       p_date = %date( %subst( l_dateA : 1 : 10 ) : *iso ) ;
000281   on-error ;
000282       clear p_date ;
000283       return *off ;
000284   endmon ;
000285   return *on ;
000286
000287
000288 end-proc;

```



RDi

- Explorateur de systèmes distants
 - Gestion des bibliothèques, objets, membres
 - Exécution de commandes
 - Accès aux travaux
 - IFS
 - Accès aux spoules
 - Interpréteur Qshell
 - Introspection des objets
 - ...





RDi

- Vue Structure
 - Géniale !
 - Représentation arborescente de la structure du module
 - Mise à jour dynamiquement au cours de la frappe
 - Indique la décomposition des DS, des formats
 - Introspection des références externes (fichiers, include, copy ...)
 - Positionnement dans le source
 - ...

The screenshot shows the 'Structure' window of an IDE. The window title is 'Structure' and it contains a search bar at the top with the text 'entrer le texte du filtre'. The main area displays a tree view of the module's structure. The tree is expanded to show the 'axisToDate' indicator and its parameters.

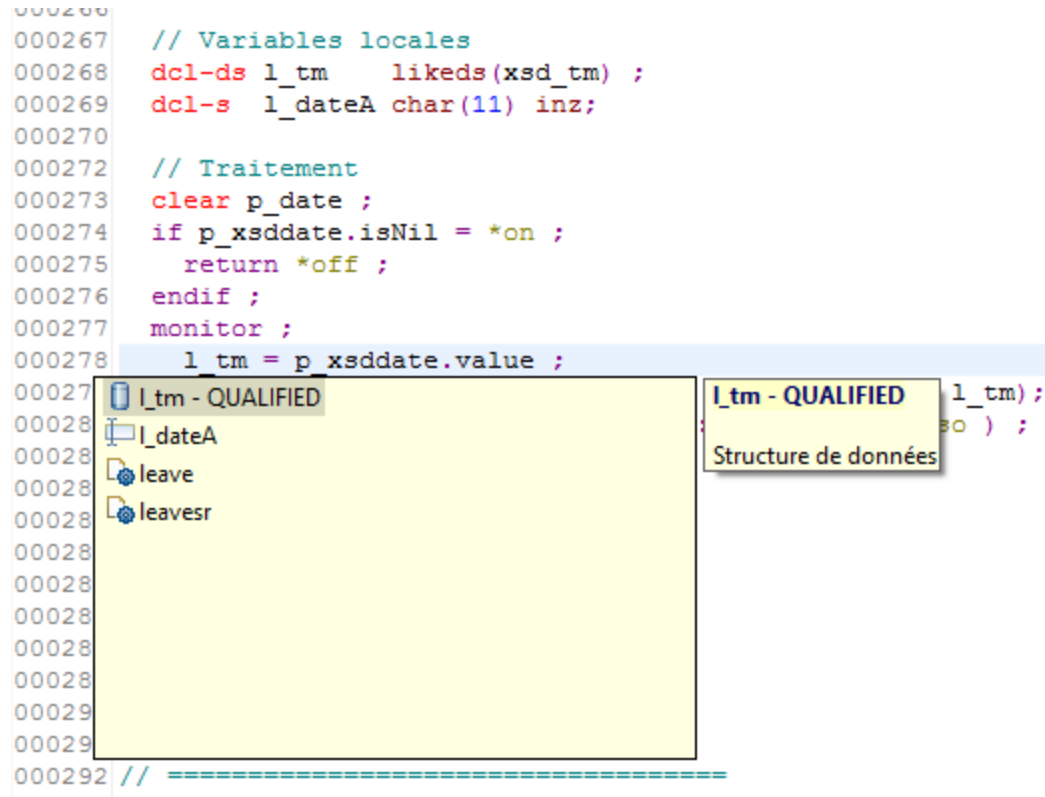
- Instructions de contrôle
- Definitions globales
 - Structures de données
 - Zones
 - Prototypes
- Sous-procédures
 - dateToAxis : Indicateur (1) EXPORT
 - axisToDate : Indicateur (1) EXPORT**
 - Paramètres
 - p_xsddate : LIKEDS(xsd_date)
 - isNil
 - value : LIKEDS(xsd_tm)
 - sec : Entier (10,0)
 - min : Entier (10,0)
 - hour : Entier (10,0)
 - mday : Entier (10,0)
 - mon : Entier (10,0)
 - year : Entier (10,0)
 - wday : Entier (10,0)
 - yday : Entier (10,0)
 - isdst : Entier (10,0)
- 93
- 97
- p_date : Date (10)

- Definitions locales
- Structures de données
 - p_xsddate : LIKEDS(xsd_date)
 - l_tm : LIKEDS(xsd_tm)
- Zones
 - 20 (D)
 - 81
- timeToAxis : Indicateur (1) EXPORT
- axisToTime : Indicateur (1) EXPORT
- datetimeToAxis : Indicateur (1) EXPORT
- axisToDateTime : Indicateur (1) EXPORT

RDi

- Content assist / complétion
 - Code opération et fonction du langage
 - Template de code
 - Variables !

```
000266
000267 // Variables locales
000268 dcl-ds l_tm      likeds(xsd_tm) ;
000269 dcl-s  l_dateA  char(11) inz;
000270
000272 // Traitement
000273 clear p_date ;
000274 if p_xsddate.isNil = *on ;
000275     return *off ;
000276 endif ;
000277 monitor ;
000278 l_tm = p_xsddate.value ;
000279
000280
000281
000282
000283
000284
000285
000286
000287
000288
000289
000290
000291
000292 // =====
```



The screenshot shows a code editor with a line of code highlighted: `l_tm = p_xsddate.value ;`. A tooltip is displayed over this line, showing a tree view of the variable `l_tm - QUALIFIED`. The tree view includes a folder icon for `l_tm - QUALIFIED`, a folder icon for `l_dateA`, and two leaf nodes: `leave` and `leavesr`. To the right of the tooltip, a small window shows the text `l_tm - QUALIFIED` and `Structure de données`.

RDi

- Screen Designer et Report Designer
- Projets IBM i
- Messages de compilation dans le source
- Formatage de code
- Conversion en format libre (cartes C)
- Plusieurs sources ouverts en même temps
- Plusieurs vues de l'éditeur pour un même source
- Restructurer (renommer)
- Débogage
- Couverture de code
- Personnalisations possibles
- Extensions via des plugins Eclipse
- ...

Autres outils

- CVTRPGSRC
 - Conversion RPG 3 -> RPG IV

- Conversion format (totalement) libre
 - Des éditeurs proposent des outils
 - Il ne s'agit pas de conversion ligne à ligne mais de refactoring : transformation des sous-routines en procédures etc ...

- Boite à outil
 - Littéralement
 - Produits vous proposant des outils pour le développement et pour vos programmes

- Référentiels de source et gestion des changement
 - Versionning, build, déploiement, ...

Conclusion

Et maintenant

- Appliquer les dernières PTF pour profitez des nouveautés !
- Utiliser les nombreuses ressources en ligne pour vous guider dans vos débuts
 - ILE RPG Reference :
https://www.ibm.com/support/knowledgecenter/fr/ssw_ibm_i_73/rzasd/rzasdmain.htm
 - SQL embarqué :
https://www.ibm.com/support/knowledgecenter/fr/ssw_ibm_i_73/rzajp/rzajpkickoff.htm
 - DeveloperWorks RPG Cafe :
<https://www.ibm.com/developerworks/community/forums/html/forum?id=11111111-0000-0000-0000-000000002284>
 - Technology Updates :
<https://www.ibm.com/developerworks/community/wikis/home?lang=en#/wiki/IBM%20i%20Technology%20Updates/page/IBM%20i%20Technology%20Updates>
- RPG Open Source
 - XMLService
 - <https://bitbucket.org/inext/xmlservice-rpg>
 - OSS ILE
 - <https://github.com/OSSILE/OSSILE>

Et après ?

- Quel futur pour le RPG IV
 - Faciliter l'intégration avec les nouvelles interfaces et les nouveaux langages
 - D'autres fonctionnalités RPG à venir
 - ...
 - Surtout de nouveaux programmeurs

- Demandez vos évolutions
 - Via des RFE (Request for Enhancement)
 - Exemple :
https://www.ibm.com/developerworks/rfe/execute?use_case=viewRfe&CR_ID=90712



display 'MERCI !' ;