CL V5R3 & V5R4

- Les articles -

Date de mise en ligne : mardi 18 mars 2008

Description :

La V5R3 et la V5R4 apportent des modifications importantes au CLP. Il est maintenant possible d'utiliser des ordres de contrôles complexes tels "FOR, SELECT..", de déclarer des pointeurs et aussi de créer des sous-programmes. Le CLP est devenu un langage évolué...

Environnement iSeries

Possibilité de lire plusieurs fichiers (5 maxi) dans un même CL (V5R3) :

Le nouveau paramètre OPNID permet de manipuler plusieurs fichiers avec les ordres DCLF, RCVF, SNDRCVF et SNDF. Ce paramètre est facultatif pour un seul fichier. Les zones de FICHIER1, y compris les indicateurs, seront préfixées de l'OPNID (ex FIC1_). Par exemple la zone « CLIENT » de FICHIER1 devrait être utilisée ainsi : &FIC1_CLIENT.

- Exemple

```
PGM          DCLF     FILE(FICHIER1) OPNID(FIC1)          DCLF     FILE(ECRAN) OPNID(ECR)          SNDRCVF OPNID(ECR)  LIR :     DOWHILE   COND(*NOT
&ECRAN_IN03)          RCVF     OPNID(FIC1)          MONMSG    MSGID(CPF0864) EXEC(LEAVE CMDLBL(LIR))          CHGVAR &ECR_CLIENT &FIC1_CLIENT
SNDRCVF OPNID(ECR)          ENDDO          ENDPGM
```

Nouveaux ordre de structuration (DOWHILE-DOUNTIL-DOFOR-SELECT) (V5R3) :

Pour les trois boucles (DOxxxx) on peut forcer une sortie anticipée de la boucle par LEAVE ou sauter un tour par ITERATE .

On peut mettre un LABEL devant le DOxxx et indiquer le label lors du LEAVE

- DOWHILE : Le test est réalisé avant d'entrer dans la boucle

```
DOWHILE COND(...)    ...   ENDDO
```

- DOUNTIL : La condition est testée sur le "ENDDO", la boucle sera toujours exécutée au moins une fois.

```
DOUNTIL COND(...)    ...  ENDDO
```

- DOFOR : TO=Départ BY=Pas

```
DOFOR VAR(&VAR) FROM(1) TO(22) BY(3)   ...  ENDDO
```

- SELECT équivalent au SELECT du RPG

```
SELECT    WHEN COND( ) THEN( )      WHEN COND( ) THEN()   OTHERWISE CMD( )  ENDSELECT
```

variables binaires (V5R3)

Nouveau type de variable permettant de ne plus utiliser %BIN

```
DCL        VAR(&VAR) TYPE(*INT)
```

Il existe aussi le format *UINT (binaire non signé) La longueur est de 2 octets et 4 octets

On peut convertir vers du décimal ou caractère avec CHGVAR

Définition de données OVERLAY (V5R4) :

Il est maintenant possible recouvrir une variable avec des autres (même principe que overlay en RPG).

Le nouveau DCL soutient le mot-clé de STG (stockage), qui vous laisse indiquer comment le stockage d'une variable doit être assigné.
— STG (*DEFINED) indique au compilateur ce un ensemble ou une partie de recouvrements de variable d'une autre variable.
— DEFVAR (défini sur la variable) pour dire quelle variable cette nouvelle variable recouvre. DEFVAR prend deux valeurs—le nom de la variable qui est recouverte et la position de départ pour le recouvrement. La position de départ de défaut est "1".

- Exemple => Le cl reçoit un paramètre de 20 caractères de long, les 10 premmiers pour le nom de la bibliothèque et les 10 derniers pour le nom du fichier.

```
pgm (&QualFile) dcl  &QualFile   type(*char) len(20) dcl  &File      type(*char) len(10) +              stg(*defined) defvar(&QualFile 1) dcl  &Lib       type(*char) len(10) +
stg(*defined) defvar(&QualFile 11) chkobj &Lib/&File *file
```

Cette technique est une excellente alternative aux opérations de sous chaine (type SST).

Gestion des pointeurs V5R4

A partir de la V5R4 on peut utilisé les pointeurs dans un CL.

- pour déclarer un pointeur, il ne faut pas renseigner le paramètre LEN

```
DCL &Paddr TYPE(*PTR)
```

- Pour que le pointeur "pointe" une variable on utilise %ADDR (ou %ADDRESS).

```
CHGVAR   VAR(&Paddr)  VALUE(%ADDR(&LIST1))
```

- Déclarer une variable "basée sur un pointeur :

```
DCL  VAR(&Var) TYPE(? ??) STG(*BASED)  BASPTR(&Paddr)
```

Gestion des sous-routines (ou sous programme V5R4) :

Les sous programmes doivent être placés à la fin du pgm (juste avant ENDPGM) et ne doivent pas être imbriqués. Il

n'est pas nécessaire de placer un GOTO ou un RETURN avant la 1ère sous-routine, le compilateur s'en charge. On peu appeler un sous programme à partir d'un autre sous programme dans la limite indiquée par DLCPRCOPT (valeur par défaut 99)

Exemple

```
PGM /* declarations : DCL, DCLF, DCLPRCOPT, COPYRIGHT */ /* global MONMSG commands> */

/* Corps du pgm */


SUBR  /* procedural code */ ENDSUBR


SUBR /*procedural code */ ENDSUBR


ENDPGM
```

Un sous programme peut retourner une valeur numérique (cette valeur doit être une variable de type *INT de lg 4).

```
ENDSUBR RTNVAL(&ret)
```

Cette valeur est alors récupérée lors de l'appel par

```
CALLSUBR RTNVAL(&ret)
```