ILE CL Programming

New possibilities

CLP

- Control language originally from S/38 (1978)
- Trough years … evolutions were mainly new system commands to control new devices, features, possibilities…
- As from V5R3 and beyond… we have many new languages possibilities
  - Subroutines, subprocedures
  - Structured Programming instructions
  - New data types
  - Limitations removed…

ILE CL

- CLP is part of the ILE language family
  - It can be integrated with all ILE languages
    - ILE - RPG
    - ILE - COBLOL
    - ILE – C / C++
  - CRTCLMOD
  - CRTSRVPGM
  - CRPPGM
  - CRTBNDCL

ILE

- Integrated language environment
- Module
- Program
- Service Program
- CALL
- CallProc

- Modules from all ILE languages can interact

File processing

- Dclf OPNID(XXX)
  - Up to 5 files
- Close
  - Reset File Position to start of file
- Rcvf

Source features

- Include
- Dclprcopt

Control execution

- IF COND() THEN()
- ELSE CMD()

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

CL Loops

- Still using GOTO ?
- DOWHILE COND()
- ENDDO
- DOUNTIL COND()
- ENDDO
- DOFOR VAR() FROM() TO() BY()
- ENDDO
- ITERATE
- LEAVE

Subroutines

- SUBR
- ENDSUBR
- RTNSUBR
- CALLSUBR

Pointers

- DCL &point_1 *PTR ADDRESS(&String_1)
- DCL &String_1 *CHAR 10
- DCL &point_2 *PTR
- Built in Functions
- CHGVAR &point_2 %ADDRESS(&String_1)
- %OFFSET()

Data types

- DCL TYPE(*CHAR) 1-5000
- DCL TYPE(*DEC) PACKED DECIMAL
  - 1-15 digits with 0-9 decimal positions
- DCL TYPE(*LGL) ‘1’ or ‘0’
- DCL TYPE(*PTR) Pointer
- DCL TYPE(*INT) 2-4-8 Bytes
- DCL TYPE(*UINT) 8 bytes (>= 0)

Data structures

- DCL &OBJSTRUCT *CHAR 20
- Dcl &OBJNAME *CHAR 10 STG(*DEFINED) DEFVAR(&OBJSTRUCT)
- DCL &LIBNAME *CHAR 10 STG(*DEFINED) DEFVAR(&OBJSTRUCT 11)

Built in functions

- %checkr
- %check
- %scan
- %trim
- %trimr
- %triml

- (needs IBM 7.1 with PTF SI49061)

ILE procedure CALL

- Calling C math functions
  - No support for Floating point (double)
- Calling C IFS API
- Calling any procedure

Samples

- The mythic CHGLIBOWN Command
- The classic way
  - Dspobjd output(*file)
  - Read the file and for each record CHGOBJOWN
- The API way
  - Use QUSLOBJ API to list objects in a user Space
  - Read the user space and process entries

CHGLIBOWN (the classic)

```
PGM        (&LIB &NEWOWNER)
DCL        &LIB *CHAR 10
DCL        &NEWOWNER *CHAR 10
DCLF       QADSPOBJ
/* Create a temporary file with all objects of the library */
OVRDBF     QADSPOBJ QTEMP/QADSPOBJ
DSPOBJD    OBJ(&LIB/*ALL) OBJTYPE(*ALL)  OUTPUT(*OUTFILE)
OUTFILE(QTEMP/QADSPOBJ)
LOOP: RCVF
MONMSG     CPF0864 EXEC(GOTO ENDPG)
/* for each record run the CHGOBJOWN command */
CHGOBJOWN  OBJ(&ODLBNM/&ODOBNM) OBJTYPE(&ODOBTP)
NEWOWN(&NEWOWNER)
/* loop through the entire file */
GOTO       LOOP
 ENDPG: /* at end change the owner of the library object */
CHGOBJOWN  OBJ(&ODLBNM) OBJTYPE(*LIB) NEWOWN(&NEWOWNER)
DLTOVR     QADSPOBJ
ENDPGM
```

CHGLIBOWN - API

- QUSLOBJ API
  - Write a list of objects in a UserSpace
  - Header section (same for all list API's)
    - Contains Offset to list data
    - Contains number of entries
    - Contains the length of an entry
  - List Data Section
    - The format is named
    - OBJL0100
      - Library
      - Name
      - Type

API's

- QUSRCRTUS
  - Create a user space
- QUSPTRUS
  - Get a pointer to a userspace

CHGLIBOWN the code

- Qualified name

```
DCL &QualUsrSpc *char 20
DCL VAR(&USRSPCLIB)          TYPE(*CHAR) STG(*DEFINED) LEN(10) +
      DEFVAR(&QUALUSRSPC 11)
Dcl &UsrspcNam  *char len(10) stg(*defined) +
      defvar(&Qualusrspc 1)
```

- Pointer

```
 DCL   VAR(&PTRUSRSPC) TYPE(*PTR)
```

- Format (INCLUDE SRCMBR(OBJL0100))

```
dcl       &ptr0100 *ptr
DCL VAR(&OBJL0100) TYPE(*CHAR) STG(*BASED) +
      LEN(30)BASPTR(&PTR0100)
DCL       VAR(&NAM0100) TYPE(*CHAR) STG(*DEFINED) LEN(10) +
      DEFVAR(&OBJL0100)
DCL       VAR(&LIB0100) TYPE(*CHAR) STG(*DEFINED) LEN(10) +
      DEFVAR(&OBJL0100 11)
DCL       VAR(&TYP0100) TYPE(*CHAR) STG(*DEFINED) LEN(10) +
      DEFVAR(&OBJL0100 21)
```

Generic Header

```
 dcl        &ptrlsthdr *ptr

DCL        VAR(&OBJLSTHDR) TYPE(*CHAR) +
STG(*BASED) LEN(256) BASPTR(&PTRLSTHDR)
DCL        VAR(&USERAREA) TYPE(*CHAR) +
STG(*DEFINED) LEN(64) DEFVAR(&OBJLSTHDR)
DCL        VAR(&OFSList ) TYPE(*INT)  +
STG(*DEFINED)  DEFVAR(&OBJLSTHDR 125)
DCL        VAR(&NbrList) TYPE(*INT)+
STG(*DEFINED) DEFVAR(&OBJLSTHDR 133)
DCL        VAR(&EntrySize) TYPE(*INT)+
STG(*DEFINED)  DEFVAR(&OBJLSTHDR 137)
```

Calling the API's

```
 CALL       QUSCRTUS (&qualUsrspc
'TEMPSPC' +  &SpcSize x'00' '*ALL'  +
     'Temporary Space for list objects'
+
     '*YES' &errstruct)
CALL       QUSLOBJ  (&QualUsrspc +
     &formatName &qualObjNam &ObjType +
     &ErrStruct)

CALL       QUSPTRUS (&QualUsrspc
&PtrUsrSPC)
```

Pointer Processing

```
chgvar   &ptrlsthdr &ptrusrspc
/* now the data structure is set ... we can use the data (number of
entries, offset, etc..) */
chgvar   &ptr0100 &ptrusrspc /* to initialize the pointer */
             /* set the pointer to the first item in the list  */
chgvar   %offset(&ptr0100) (%offset(&ptrUsrSpc) &OfsList )
          /* Now read the list  */
dowhile    (&count < &NbrList )
   chgvar     &count (&count + 1)
   chgobjown  &lib0100/&nam0100 &typ0100 &newOwner
         /* move to the newt entry */
  chgvar  %offset(&ptr0100) (%Offset(&ptr0100) &entrySize)
enddo
```

Call C Library Function

- Exemple abs() : absolute value

```
 pgm
   dcl        &absval *int
   DCL        VAR(&NUMBER) +
    TYPE(*INT) VALUE(-10)
   CALLPRC    PRC('abs') +
    PARM((&NUMBER *BYVAL)) +
    RTNVAL(&ABSVAL)
endpgm
(restriction: double not supported)
```

IFS C Library Function

- Function access() can check if a user has access to an object.

int access(const char *path, int amode);

amode 0 : file exists
4 : read access
2 : write
1 : execute

Check IFS object (STMF or Dir)

```
 pgm        &Stmf   /* check ifs object */
     dcl        &stmf *char 255
     dcl        &null *char 1 value(X'00')
     dcl        &rtcd *int
     dcl        &Mode *int
     dcl        &rtcda *char 10
chgvar     &stmf  (&stmf *tcat &null)
CALLPRC    PRC('access') PARM((&STMF) (&Mode   *BYVAL))+
       rtnval(&rtcd)
  if         (&rtcd *NE 0) do
     chgvar     &rtcda &rtcd
     sndpgmmsg  (&stmf *tcat ' Not Found : ' *cat &rtcda )
  enddo
 endpgm
```

Sample CPYTOSTMF

```
 chgvar     &flag ( &o_create + &O_wronly + &o_codepage)
 chgvar     &code 1252
 chgvar     &stmf ('/temp/common.txt')
 chgvar     &mode &AllMode
 rmvlnk     &stmf
 monmsg     cpf0000
 chgvar     &stmf (&stmf *tcat &null)
           /* call the open API to create the file */
 CALLPRC    PRC('open') PARM((&STMF *BYREF)  +
       (&flag *byval) (&MODE *BYVAL) (&Code *byval)) +
       RTNVAL(&FileHdl)

/* close and reopen to have all options set  conversion,
rights, etc...       */

CALLPRC    PRC('close') PARM((&FILEHDL))
```

Copy DB File to IFS

```
chgvar     &flag (&O_textdata + &o_create + &O_wronly )
CALLPRC    PRC('open') PARM((&STMF *BYREF)  +
       (&flag *byval) (&MODE *BYVAL) (&Code *byval)) +
       RTNVAL(&FileHdl)
CallSubr   ReadFile
DoWhile    ( ^ &EndOfFile)
 chgvar     &line (&Myfile_fldTxt *Tcat &crlf *tcat &null)
 callprc    ('strlen') PARM((&line *byref)) rtnval(&size)
 callprc    PRC('write') PARM((&FileHdl *Byval) +
        (&line *BYREF) (&size *byval)) rtnval(&nbW)
 callsubr   ReadFile
Enddo
CALLPRC    PRC('close') PARM((&FILEHDL))

subr     ReadFile
         RCVF       opnid(MyFile)
         monmsg     cpf0864 exec(chgvar &EndOfFile '1')
endsubr
```

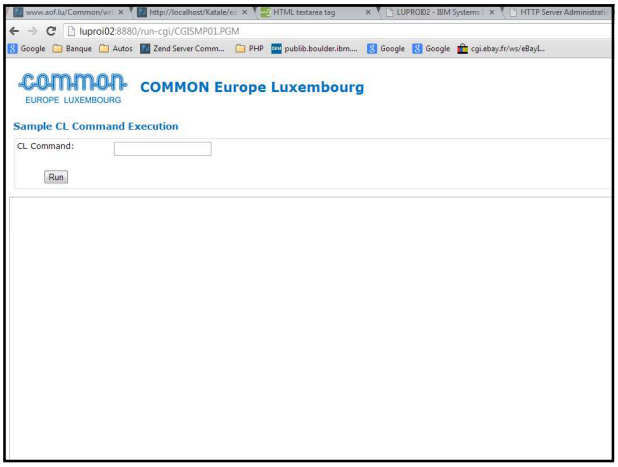
Web Program in ILE CL

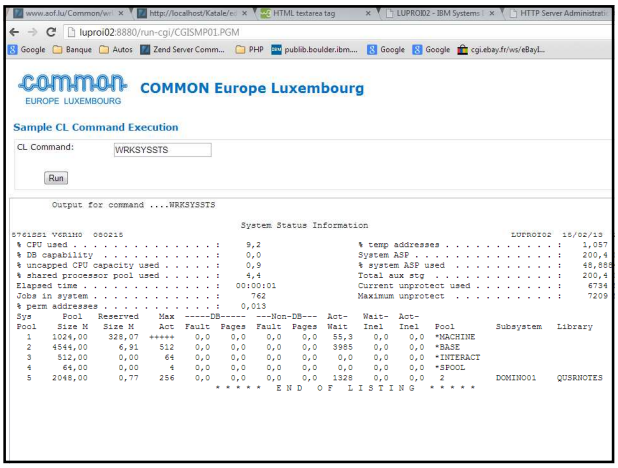
- CGI program are executed by Apache server
- CGI programs read their input from stdin
- CGI programs write their output to stdout(html)
- Apache config directive : httpd.conf

```
<Directory /QSYS.LIB/COMMON.LIB/>
   Allow From all
   Options +ExecCGI
</Directory>
ScriptAliasMatch /run-cgi/(.*)
/QSYS.LIB/COMMON.LIB/$1
```

HTML

```
<html><body>
<h2> Sample CL Command Execution </h2>
<form method="POST">
<dl><dt>
<label for="CLCMD">CL Command:</label></dt>
<dd><input type="text" id="CLCMD" name="CLCMD"
maxlength="60" value=&&CLCMD&& /></dd><br>
<dt/> <dd>
<input type="submit" id="action" name="action"
value="Run" /></dd>   </dl>
</form>
<textarea readonly rows="100" cols =
"132">&&CMDOUTPUT&&</textarea>
</body></html>
```





CGI Principe

- POST Read the data from stdin
- QUERY-STRING :

CLCMD=xxxxxxx&action=Run

- HTTP Server API : Service Program QZHBCGI
- QtmhRdStin
- QtmhGetEnv
- QtmhWrStout
- Write the HTML to Stdout → the browser

CGI Sample

- IF POST
  - Read the Query string and set the variable
  - Run the CL command + CPYSPLF
- Read the IFS stream file HTML
- Replace &&xxxx&& by the value of the variable &xxxx
- Write the output of the cl command to stdout

CGI Sample

```
crtpf      qtemp/webfile rcdlen(200)
      monmsg     cpf0000
   clrpfm     qtemp/webfile
   chgvar     &envName 'REQUEST_METHOD'
   callprc    'QtmhGetEnv'    (&EnvRec &EnvRecLen +
      &EnvLen &EnvName &EnvNameLen &QUSEC)
   chgvar     &rqsMethod &EnvRec
   if         (&rqsMethod = 'POST') do
      chgvar     &envName 'CONTENT_LENGTH'
      callprc    'QtmhGetEnv'    (&EnvRec &EnvRecLen +
       &EnvLen &EnvName &EnvNameLen &QUSEC)
      chgvar     &contLen %sst(&envrec 1 &envlen)
      chgvar     &bufInpLen &contLen
      callprc    'QtmhRdStin' (&BufInput &BufInpLen +
             &StdInLen &QUSEC)
```

```
 /*CLCMD=DSPNETA+*PRINT&action=Run */
chgvar     (&i)  7
chgvar     &o 1
dowhile    (%sst(&BufInput &i 1) ^= '&')
   if  (%sst(&BufInput &i 1) = '+' ) +
            chgvar %sst(&Bufinput &i 1) ' '
    chgvar     %sst(&clcmd &o 1) %sst(&bufinput &i 1)
    chgvar     &o (&o+1)
    chgvar     &i (&i+1)
enddo
chgvar     &cmdLen &o
OVRPRTF    FILE(*PRTF) HOLD(*YES) SPLFNAME(WEBSPOOL)
call       QCMDEXC (&clcmd &cmdlen)
  monmsg     cpf0000 exec(do )
    chgvar     &CmdOutput ('Error found on ' *cat &clcmd )
    goto       noOutput
  enddo
CPYSPLF    FILE(WEBSPOOL) TOFILE(QTEMP/WEBFILE)
JOB(QTMHHTP1/QPRTJOB) SPLNBR(*LAST)
DLTSPLF FILE(WEBSPOOL) JOB(QTMHHTP1/QPRTJOB) SPLNBR(*LAST)
```

```
noOutPut:   chgvar     &BufOutPut 'Content-type: text/html'
            chgvar     &BufOutPut (&BufOutput *tcat &nl *tcat &nl)
            chgvar     &o 25
chgvar     &stmfName  ('/www/common/htdocs/Sample1.html' *tcat &null)
    chgvar     &flag (&O_rdonly + &O_TextData )
 CALLPRC    PRC('open') PARM((&STMFName *BYREF)  (&flag *byval) ) RTNVAL(&StmfHdl)
 CALLPRC ('read') ((&stmfHdl *Byval) (&data *byref) (&dataLen *byval)) rtnval(&Rdlen)
 CALLPRC    'close' PARM(&stmfHDL)
     DOFOR      VAR(&I) FROM(1) TO(&RdLEN)
      if         (%sst(&data &i 2) = '&&')   do
          chgvar     &s  (&i+2)
          chgvar     &i &s
          dowhile    (%sst(&data &i 2) ^= '&&')
              chgvar     &i (&i + 1)
          enddo
          chgvar     &j (&i - &s)
          chgvar     &varName %sst(&data &s &j)
          chgvar     &i (&i + 2)
          Select
            When       (&varName = 'CLCMD' )  Do
               chgvar     &BufOutPut (&BufOutPut *Tcat '"' *tcat  &clcmd *tcat '"')
               chgvar     &O (&O +100)
            Enddo
            When       (&varName = 'CMDOUTPUT') Do
               CALLSUBR   GETOUTPUT
            enddo
          EndSelect
       enddo
       chgvar     &o (&o + 1)
       chgvar     %sst(&BufOutPut &o 1)    %SST(&data &i 1)
    enddo
 CALLSUBR   WrtSTDOUT
```

```
SUBR       WrtSTDOUT
     chgvar     &o 5000
     DoWhile    (%SST(&BufOutput &o 1)= ' ')
        chgvar     &O (&O - 1)
     enddo
     chgvar     &BufOutLen &o
     Callprc    'QtmhWrStout' (&BufOutput &BufOutLen &QUSEC)
     chgvar     &o 0
     chgvar     &BufOutput ' '
endsubr

SUBR       GETOUTPUT
 chgvar     &BufOutPut (&BufOutPut *Tcat 'Output for command ....' *cat &clcmd +
        *cat &nl *cat &nl *cat &cmdOutPut )
 ovrdbf     webfile qtemp/webfile
 chgvar     &eof '0'
 callsubr   ReadSpl
 Dowhile    (^ &eof)
    chgvar     &BufOutput (&BufOutput *Tcat &WebFile_webfile *tcat &nl)
    CALLSUBR   WrtSTDOUT
    callsubr   ReadSpl
 Enddo
 close      opnId(WebFile)
endsubr

Subr       ReadSpl
   rcvf       opnId(WebFile)
     monmsg     cpf0864 exec(do)
        chgvar     &eof '1'
     enddo
endsubr
```

Questions ?

- References
  - http://publib.boulder.ibm.com/infocenter/iseries
  - paul.roy@real.lu
  - Presentation and samples to download from
    - www.common.lu

THANK YOU