Michigan iSeries Technical Education Conference

QShell and the Integrated File System

Presented by

Ryan Technology Resources

Michael Ryan

michael@ryantechnology.com

The Integrated File System

- What is a file system?
  - Webopedia.com defines a file system as “The system that an operating system or program uses to organize and keep track of files.”
- The System i has a number of different file systems
- QSYS.LIB file system uses libraries as containers for other objects
  - Database files
  - Programs
  - Output queues
- QSYS.LIB is a ‘flat’ file system
  - Only one level deep

The Integrated File System

- Other System i file systems include:
  - Root (‘/’ or ‘slash’)
  - QNTC (Windows NT compatible)
  - QDLS (Document Library System) file system
  - QOpenSys (open systems)
  - QSYS.LIB (traditional library)
  - QOPT (optical drive)
  - QFileSvr.400 (iSeries-to-iSeries)
  - UDFS (User Defined)
  - NFS (Network)
  - QNetWare (Novell NetWare)

The Integrated File System

- These different file systems are known collectively on the System i as the Integrated File System (IFS)
- Similar to file systems on Windows or Linux systems
  - A hierarchical file system that's composed of directories, subdirectories and files
  - Files may contain program information, data information or other information
  - No specific object types

The Integrated File System

- File System Commands
  - CPYTOSTMF – Copy to Stream File
  - CPYFRMSTMF – Copy from Stream File
  - CPYTOIMPF – Copy to Import File
  - CPYFRMIMPF – Copy from Import file
  - CPY – Copy a Stream File
  - SAV/RST – Save/Restore an IFS file

The Integrated File System

- Work with Link (WRKLNK) encompasses other IFS-based commands

```
Work with Object Links

Directory  . . . . :   /

Type options, press Enter.
  2=Edit    3=Copy    4=Remove    5=Display    7=Rename    8=Display attributes
  11=Change current directory ...

Opt    Object link             Type      Attribute     Text
       QDLS                    DIR
       QFileSvr.400            DIR
       QIBM                    DIR
       QNetWare                DIR
       QNTC                    DIR
       QOpenSys                DIR
       QOPT                    DIR
       QSR                     DIR
       QSYS.LIB                DIR       PROD          System Library
                                                                    More...
```

Root File System

- Most widely used file system
- Contains important directories such as:
  - QIBM, which contains Client Access, MQ Series, the XML Toolkit and many other IBM products
  - Alo contains ‘UNIX’ like directories
    - bin, dev, etc, home, tmp, usr, var
- Most user-defined and vendor directories reside in root
- Names in the root file system can be mixed case, but the file system is not case sensitive

QOpenSys File System

- Similar to the root file systemwith one important difference – case sensitivity
  - File names may be in mixed case
  - File system **is** case sensitive
- Supports very looong file names
  - Up to 16 megabytes for the object name and all directory names
  - Each component (name) can up to 255 characters
- IBM uses QOpenSys for C++ compiler, InfoPrint, HTTP servers

QNTC File System

- Consists of the NT servers (and associated shares) in a network
- Use WRKLNK '/QNTC/*' to see a list of the NT servers in a network
  - Each server can then be accessed to provide access to the shared directories on that server
  - Building of the list of NT servers can be very slow, especially in a large network
    - Set environment variable QZLC_SERVERLIST to a value of '1' (`ADDENVVAR ENVVAR(QZLC_SERVERLIST) VALUE('1')`) to cause OS/400 to produce faster lists

QNTC File System

- You must have the same userid and password on the System i as on the NT server to be able to access the files on the shared directories
- Names in QNTC can be mixed case, but the file system is not case sensitive
- Files accessed through QNTC are actually on NT servers
  - Most of the IFS commands (CPY*, etc.) may be used
  - QNTC can be a powerful file system if you need to access NT server data from your System i

QDLS File System

- ‘Document Library System’
- Used for folders
- Filenames must be in 8.3 format
- Not case sensitive, all file names are uppercased
- Slower than the other file systems
- Don’t use this file system

QSYS.LIB File System

- Traditional file system accessed through IFS
- Syntax when using IFS commands:
- /QSYS.LIB/library.LIB/file.FILE/member.MBR
  - The extensions (.LIB, .FILE, .MBR) are used to identify the type of objects being accessed
- QSYS.LIB file system uppercases names, so case sensitivity is not supported

QSYS.LIB File System

- Can access traditional objects

```
CPYFRMSTMF
   FROMSTMF('/qntc/NTServer/transfer/myfile.txt')
TOMBR('/qsys.lib/michael.lib/myfile.file/myfile.mbr')
```

  - Copies the myfile.txt file from the transfer directory on the NTServer server to the physical file MICHAEL/MYFILE

```
RMVDIR DIR('/qsys.lib/aplib.lib')
```

  - Deleted library APLIB

QShell

- Different systems have a different ‘look and feel’
  - System i is a different environment than a UNIX, Linux or Windows system
- The different environments can be called shells
    - Primarily a *NIX term
    - Korn shell, the Bourne Again Shell (BASH), the C shell and others

QShell

- System i has its own shell - QShell
  - A command interpreter and environment that looks and feels like a *NIX shell
- This is important for a few reasons
  - Allows programmers and administrators from other systems to use familiar tools on iSeries
  - Assists in the implementation of Java and C or C++ based systems (including environment variables)
  - Enables easier manipulation and management of Integrated File System (IFS) based resources

QShell

- QShell is available on all iSeries systems (V4R3 and later)
  - Optional part of the operating system
    - May not be installed
- The QShell Interpreter is option 30 of OS/400

```
5722SS1    30       OS/400 - QShell Interpreter
```

  - Use GO LICPGM to determine if QShell is installed on your system

QShell Invocation

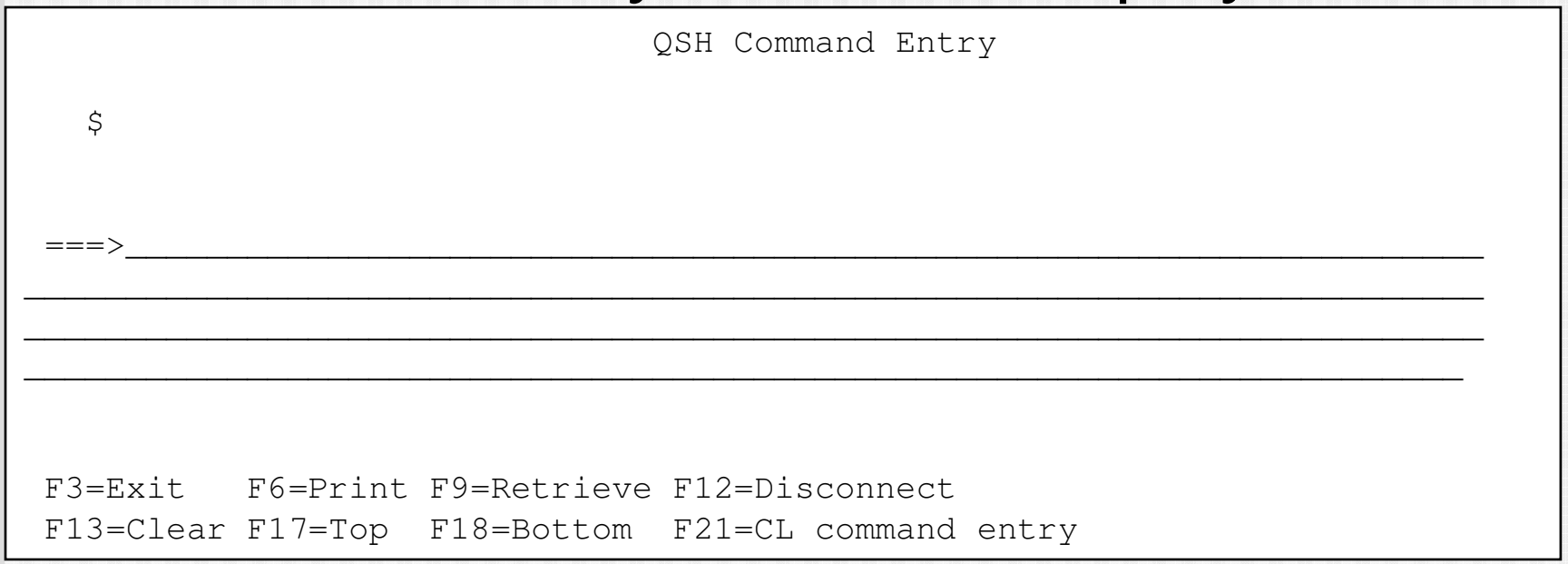
- Invoke the QShell interpreter with:
  - Start QShell Interpreter (STRQSH) command
  - Or simply with the QSH command
- Both commands call the same command-processing program
- One parameter (CMD) is available for the QSH command
  - An optional command to be executed
    - This is an optional parameter

QShell Invocation

- If QShell is invoked without specifying a parameter, a specific hierarchy is followed to execute a command:
  - QShell executes commands that are specified in the IFS file /etc/profile (if the file exists);
  - QShell executes commands that are specified in the IFS file .profile in the users home directory (if the file exists);
  - QShell executes commands that are specified in the file whose name is specified in the ENV environment variable

QShell Invocation

- If a command is not specified, any profile files are executed and the QShell command entry screen is displayed



QShell Commands

- . – Execute a command in the local directory (slashdot ‘/.’)
- cat – Display a file
- cd – Change directory
- chmod – Change mode (file mode)
- chown – Change ownership (file)
- cp – Copy files
- export – Set a variable in the environment
- file – Identify type of file
- find – Find a file on disk
- grep – Search a file for specified string
- kill – End a process

QShell Commands

- ls – List contents of a directory
- mkdir – Make a directory
- mv – Move (delete, rename) files
- pax – Portable archive interchange (tar files)
- pwd – Print working directory
- rm – Delete files
- rmdir – Remove directory
- system – Run an OS/400 CL command
- tail – Display the last part (tail) of a file
- touch – Create an empty file

Grep Command Example

- Grep will find a string within a file
- First, let's examine the contents of a file
- Syntax: cat <file to display>

```
> cat /iSeries/filea
  Now is the time for all good men to come to the
   aid of their party
  The quick brown fox jumped over the lazy dog
  Wherever you go, there you are
  A stitch in time saves nine
  $
```

Grep Command Example

- I need to find every instance of the string ‘the’ within the file. Here’s an example of using the grep command:
- Syntax: grep <search string> <file to search>
  - I’ll search file /iSeries/filea for the string ‘the’

```
> grep the /iSeries/filea
Now is the time for all good men to come to the aid of
 their party
The quick brown fox jumped over the lazy dog
Wherever you go, there you are
$
```

  - grep found every instance of the string ‘the’, including in the word ‘there’ in the last line.

Piping Example

- A common *NIX technique is piping
  - Sending the output of one command to the input of another
    - In other words, using the information created by executing a command as the input stream to a different command
    - For example, if I needed to identify every file in the /iSeries directory that had the letter ‘a’ in the name, I could use the grep command with piping
  - I’ll pipe the output from the ls command (list contents of a directory) to the grep command

Piping Example

```
> ls
  filea                    fileb
   filewithalongname
  $
> ls | grep a
  filea
  filewithalongname
  $
```

- The pipe operator (“|”) is used to pipe the output of the ls command (the names of the files in the directory) to the grep command
- Grep examined each line in the input and found two filenames that contained the letter ‘a’

Redirection Example

- Redirection is the technique of changing the input to a command or output from a command from the standard location (keyboard and display respectively) to a different location
- This different location will in many cases be a file – not unlike using an OS/400 display (DSP*) command and sending the output to an outfile
- The redirection operators include:
  - “>” to redirect output to a new file
  - “>>” to append output to an existing file
  - “<” to receive input from a file

Redirection Example

- Here's an example of redirecting the output of the find command into a file:

```
➢ find /iseries -name fileb >myfilelist
   $
> cat myfilelist
  /iseries/fileb
   $
```

- The redirection operator (">") is used to send the output of the find command (the location of the name I specified) to the file myfilelist
  - Displaying the contents of myfilelist shows the output produced by find

Shell Scripts

- A shell script is similar to a Control Language (CL) program on an iSeries system
  - Can accept input in the form of parameters passed to the script
  - Process information
  - Produce results
- Shell scripts are often used for system management and job control activities, such as controlling a jobstream of programs.

Shell Script Example

- Controls jobs and produces output in multiple directories:

```
# Any line starting with a '#' is a
  comment…
system "call michael/settlecl"
cp /finoutdir/settfile /iSeries/testfile
rm /finoutdir/settfile
echo "Settlement file produced"
  >/iSeries/msgfile
return
```

Shell Script Example

- The first action in this shell script is to execute the system command
  - This command executes an OS/400 command
- In this case, the command is CALL, and I'm calling a program in library MICHAEL called SETTCL
  - Note:Case is not important
- The next step is to copy (using cp) the file settfile in directory /finoutdir to file testfile in directory iSeries

Shell Script Example

- The file /finoutdir/settfile is deleted using the rm command
- A message is placed into file /iSeries/msgfile using the echo command and a little redirection
- The last step is to return from the shell script back to where the script was called – another shell script, the QSH command line, or the OS/400 command line if the shell script was specified as the CMD parameter on the QSH command

Shell Script Example

- Here is the result of executing this shell script:

```
> /michael/settshell
  CPCA082:  Object copied.
  $
> cat /iseries/msgfile
  Settlement file produced
  $
```

Shell Script Example

- The MICHAEL/SETTLECL program is a simple one – it simply copies a file from the traditional file system to the IFS

```
PGM
CPYTOSTMF  +
FROMMBR('/qsys.lib/michael.lib/settfile.fil
  e/settfile.mbr') +
  TOSTMF('/finoutdir/settfile')
ENDPGM
```

Shell Script Example

- The CPCA082 message in the QSH session came from OS/400 indicating the file was copied (using the CPYTOSTMF command)
- When I display the /iSeries/msgfile file using the cat command, you can see the message that was placed in that file with the echo command

Shell Scripts

- You can create and edit shell scripts stored in the IFS in a couple of different ways:
  - Editing them using traditional Source Entry Utility and then moving them to the IFS
  - Using a client editor and a NetServer mapped drive to edit the script
  - The Edit File (EDTF) command
    - EDTF enables you to edit IFS files as well as QSYS data files (internally described) and source physical files

QSH and the IFS

- QSH can be used for tasks that interact with the Integrated File System
  - You can also use WRKLNK or iSeries Navigator
- Invoke QSH and use the file and directory commands
- For instance, I'll use the mkdir and rmdir from a QSH command rather than using the same commands from an OS/400 command line
  - The reason is simple – I don't have to be concerned about quoting the argument to the command

QSH and the IFS

- Here's what I mean: if I want to create a new IFS directory and I'm currently using the OS/400 command line, I would need to do this:

```
mkdir '/michael/newdir'
```

- The problem is that I will usually do this, at least the first time:

```
mkdir /michael/newdir
```

QSH and the IFS

- And the problem is?
- I'll get the message "/ not in expression enclosed in parentheses"
- However, if I invoke QSH, I don't need to worry about quoting
- That can reduce frustration is you do a lot of IFS work

QSH and the IFS

- There are over 30 different file and directory oriented commands available within QSH, including commands to:
  - Create, modify and delete files and directories
  - Work with file and directory ownership and authorization
  - Display files
  - Archive files
  - Other functions

Getting More Information

- The Information Center
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/index.jsp?topic=/rzahz/intro.htm
- Several Redbooks that discuss QShell as part of another product
- QShell for iSeries by Ted Holt