RPG Makes Friends with Open-Source Apps

Open Your i

April 2020

MobiGoGo
Work at the Speed of Mobility

Your Presenter

Richard Schoen

richard@richardschoen.net

-or-

richard@mobigogo.net

- 30+ yr developer IBM i, Windows, Linux, Mac
- Started RJS Software Systems in 1990
- Created automated report distribution, doc management and app integration software
- Sold business in 2014 when partner retired
- Started MobiGoGo to build multi platform apps
- Provides training and consulting services
- Contributes open source to Github repositories
- Author of iForGit Git client for RDI, SEU and PDM developers

http://github.com/richardschoen http://www.iforgit.com
Business web site: http://www.mobigogo.net
Personal web site: http://www.richardschoen.net
LinkedIn: https://www.linkedin.com/in/richardschoen
Twitter: @richardschoen

Richards Journey with IBM i System Integration

- Integrate S/36 to DOS Turbo Pascal screen -HLLAPI (**1988**)
- Windows 3.1, Microsoft Visual C/C++, Visual Basic 3 (**1993**)
- APPC programming with PC Support to access RPG (**1995**)
- TCP/IP, Email, FTP from Windows to IBM i (**1997**)
- Sockets programming with VB and RPG (**2002**)
- Web development and web services (**1997 –current**)
- IBM i Access ODBC/OLEDB/ADO.Net (**2000-current**)
- JT400 java API converted to .Net with IKVM (**2005-current**)
- HTTP URL API –(AKA REST) (**2004 –current**)
- XMLSERVICE –universal open source IBM i DB access (**2012-current**)
- Editor/IDE choices: RDI, Eclipse, Visual Studio, Visual Studio Code, Notepad++, etc…
- (**1984–today : Green screen to smartphones**)
  The greatest computing era !!

MobiGoGo
Work at the Speed of Mobility

Session Topics

- Why consider using RPG and Open Source together ?
- How to utilize open-source applications from RPG and CL
- Intro to QShell on i
- Review logic flow
- A few examples

How to use shells on the IBM

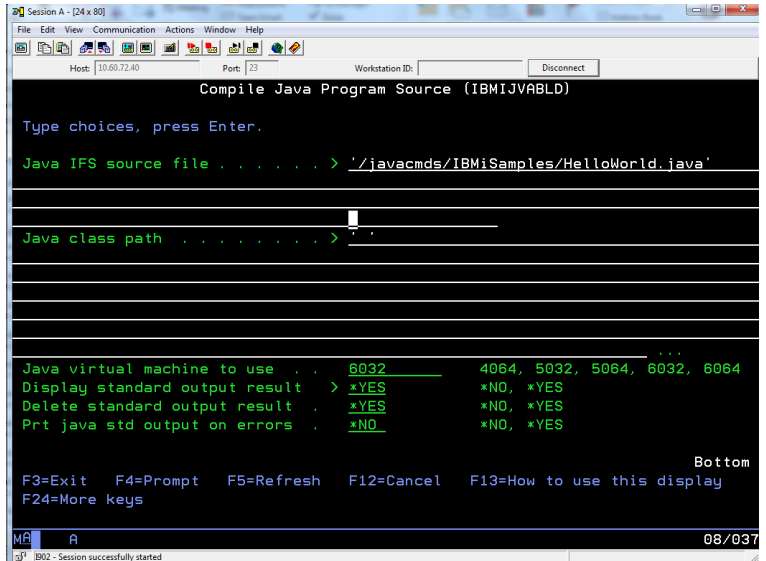
- A good intro to various shells on IBM i
- Andy Youens - FormaServe
- https://www.youtube.com/watch?v=9rL9U8hfIHA
- We will use Qshell, Pase and Bash

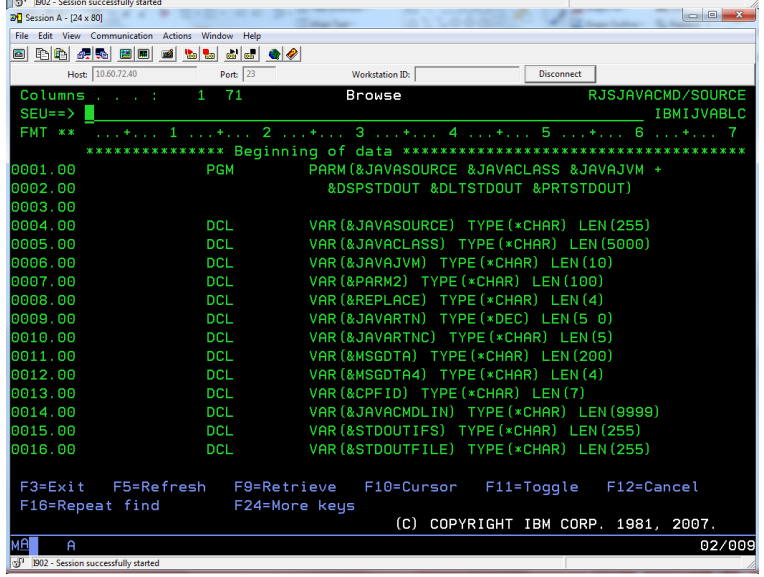
Why consider using RPG and Open Source together ?

- Functionality not easily available in CL or RPG by themselves
- Tons of Python and other language example code, libraries and tutorials
- Access web services without using HTTPGETCLOB etc.
- Send and receive email
- Crawl IFS directories
- Become productive quickly
- Integrates or extends existing IBM i apps across boundaries
- Qshell on I can be used for all kinds of IBM i utility programs
- Run any Qshell or PASE code from traditional programs

MobiGoGo
Work at the Speed of Mobility

Traditional CL Command/Program Call Flow





Type CL Command or
Embed in a CL Program

Run CL/RPG Processing Pgm
May pass parameters

Processing Program
CL, RPG, COBOL

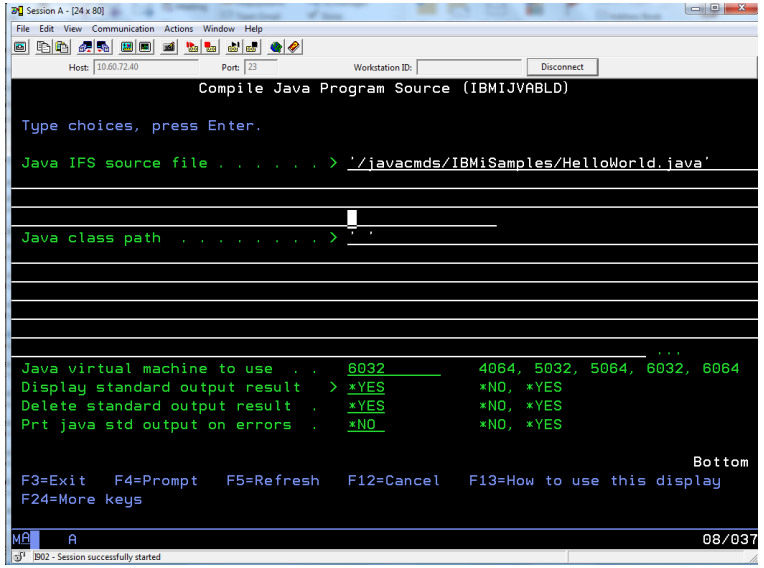
Parameters or Data Returned to
Calling CL Program or to
Command Line as CPF Messages

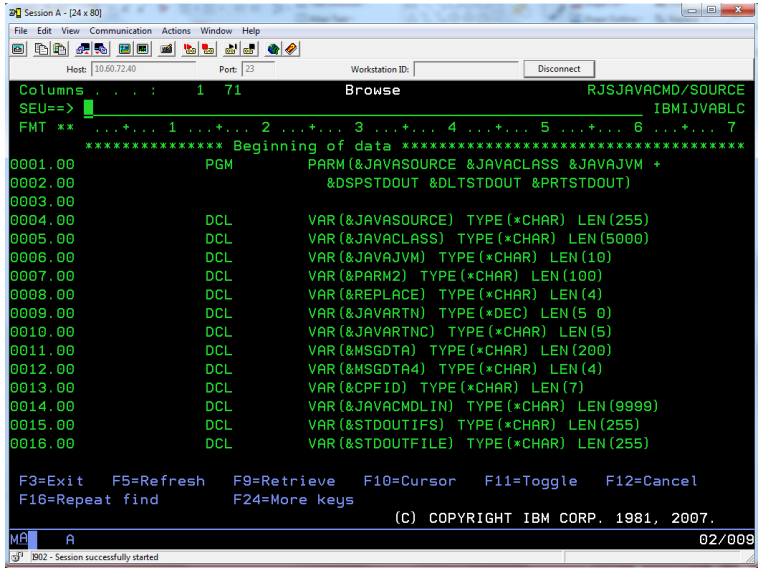
Log info written to job log,
physical file or IFS file

QTEMP
Library

MobiGoGo
Work at the Speed of Mobility

Command/Program Call Flow using QShell/PASE





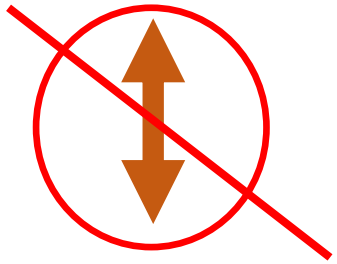
Type CL Command or
Embed in a CL Program

Run CL/RPG Processing Pgm
May pass parameters

Processing Program
CL, RPG, COBOL

Run QShell/PASE
Program or Command

QTEMP
Library



Qshell/PASE app runs in a
thread job

Parameters Returned in STDOUT
IFS file from Qshell/Pase call

STDOUT
log info can be
Processed by RPG

Parameters or Data Returned to
Calling CL Program or to
Command Line as CPF Messages

STDOUT log info written to job
log, physical file or IFS file

What is Standard Output (STDOUT) ?

STDOUT – Standard Output

- Console output from a PASE/QShell application
- Similar to IBM i joblog
- Allows PASE/Qshell apps to pass back data and messages
- All Qshell/PASE languages can generate STDOUT console messages
- STDOUT data can be used from CL/RPG/COBOL apps
- Great way to communicate between PASE/Qshell and traditional IBM i apps such as CL/RPG/COBOL

MobiGoGo
Work at the Speed of Mobility

QShell on i – What is it ?

- Easy to use wrapper around QShell and PASE commands
- Simplifies running QShell or PASE commands including bash
- Sets up multithreading environment
- Handles command output logging - STDOUT
- Allows QShell/PASE commands to be submitted
- Don’t need an SSH session or STRQSH to run QShell/PASE commands
- Can be called directly from RPG or CL

MobiGoGo
Work at the Speed of Mobility

QShell on i – Use Cases

- Call QShell, Pase or Bash commands directly from RPG or CL
- Run any Python, PHP, Node, shell scripts, etc
- Submit nginx, gunicorn or other background server/web service jobs
- Run interactively, submit or job schedule Qshell/Pase commands
- Consume web services without HTTPGETCLOB/HTTPPOSTCLOB (Java)
- Compose, send and receive emails
- Talk to other databases (PostgreSQL, MariaDB, SQLite, others)

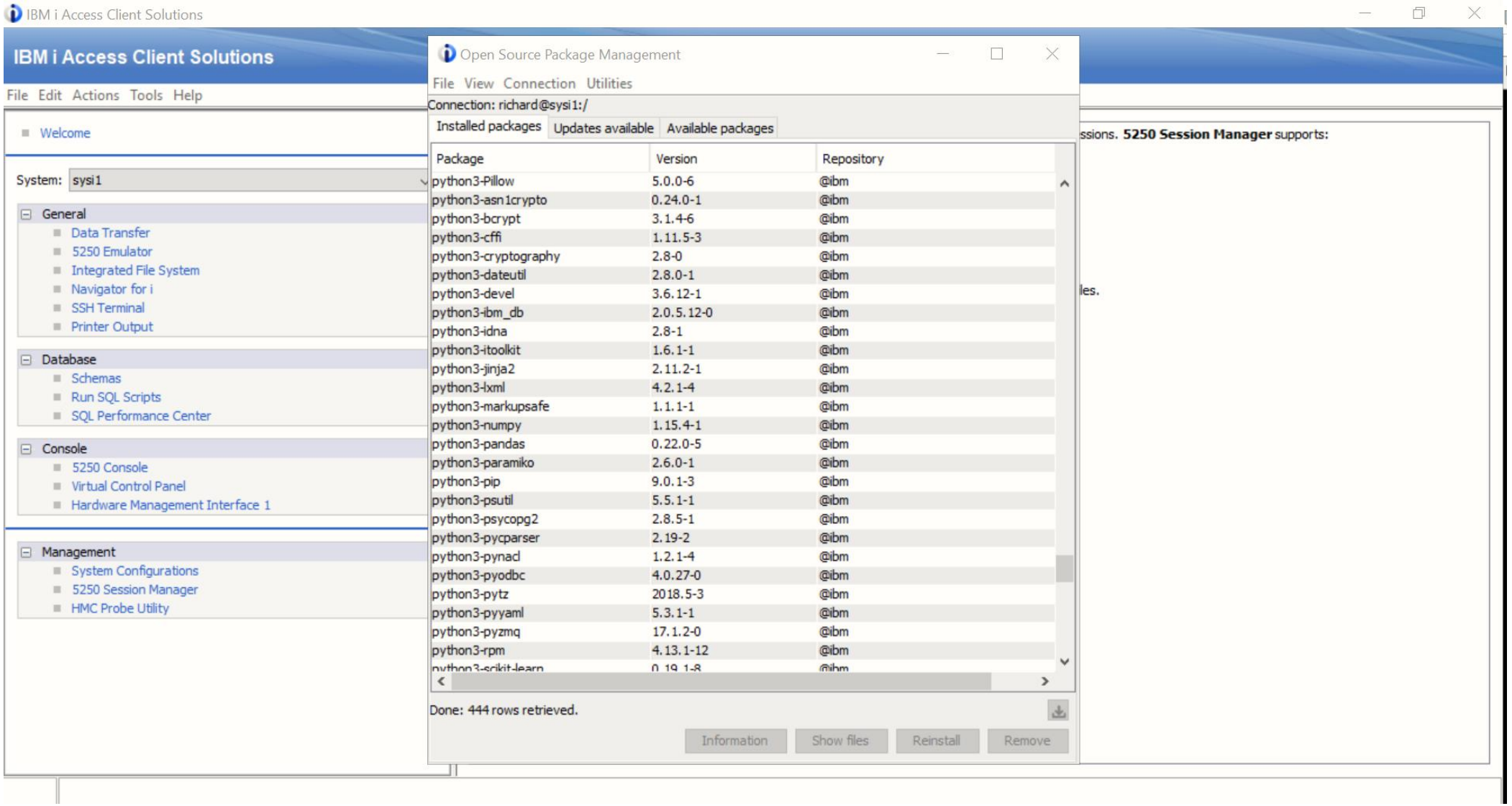
MobiGoGo
Work at the Speed of Mobility

System Prerequisites

- IBM Open Source Package Management
- Install all Python 3 packages (if using Python)
- Install unixODBC
- Install IBMi Access ODBC Driver
- Install Qshell on I
- Uninstall 5733OPS if you can. Yum packages rule

MobiGoGo
Work at the Speed of Mobility

Install All Python Packages



MobiGoGo
Work at the Speed of Mobility

Installing QShell on i

- Visit the Github site, download and build

  http://www.github.com/richardschoen/qshoni

- Command documentation in main **readme.md** page
- Library name: QSHONI
- Installing and Building QSHONI via getrepo-qshoni.sh
- Installing and Building QSHONI via Git Clone
- Installing QSHONI via Save File

QShell on i Commands

- **QSHEXEC** - Run QShell Command Line
- **QSHBASH** - Run Bash Command via Qshell
- **QSHPYRUN** - Run Python Script via Qshell
- **QSHLOGSCAN** - Scan Qshell Log File for Values
- **QSHPATH** - Set Open Source Package Path Environment Variable
- **QSHIFSCHK** – Check for IFS file existence

MobiGoGo
Work at the Speed of Mobility

QSHEXEC - Run Qshell Command Line

- Easy to use wrapper around Qshell and PASE commands
- Simplifies running Qshell or PASE commands including bash
- Sets up multithreading environment
- Handles command output logging - STDOUT
- Allows Qshell/PASE commands to be submitted
- Don’t need an SSH session or STRQSH
- Can be called directly from RPG or CL

MobiGoGo
Work at the Speed of Mobility

QSHBASH - Run Bash Command via QShell

- Simplifies bash script calls
- Convenience wrapper over bash calls
- No need to type: **bash –c** to run your bash command.
- Don’t need an SSH session or STRQSH
- All the same benefits of QSHEXEC
- Can be called directly from RPG or CL

MobiGoGo
Work at the Speed of Mobility

QSHPYRUN - Run Python Script via QShell

- Convenience wrapper over Python calls
- Simplifies Python 2 or Python 3 script calls
- No need to type **python2** or **python3** command
- Script path and script name passed individually
- Up to 40 parameters can be passed to script
- All the same benefits of QSHEXEC
- Can be called directly from RPG or CL

MobiGoGo
Work at the Speed of Mobility

QSHLOGSCAN – Scan Qshell Log for Value

- Scans the QTEMP/STDOUTQSH outfile for an anticipated value
- Great way to check your results file for information
- Scans line by line
- Looks for a specific value or a partial match if desired
- Returns CPF9898 Escape message if no value found
- Completes normally if value was found

QSHIFSCHK – Check for IFS file existence

- Check if IFS file or directory exists
- Send CPF9898 escape message if no file/dir
- Send CPF9897 escape message if file/dir found
- You should monitor for both CPF9897 and CPF9898 when using

MobiGoGo
Work at the Speed of Mobility

QSHPATH – Set Open-Source Package Path

- Adds /QOpenSys/pkgs/bin to PATH environment variable
- Used by QSHEXEC, QSHBASH and QSHPYRUN SETPKGPATH = *YES

MobiGoGo
Work at the Speed of Mobility

QSHEXEC - Run Qshell Command Line

```
QSHEXEC CMDLINE('ls /tmp')
        SETPKGPATH(*YES)
        DSPSTDOUT(*YES)
        LOGSTDOUT(*NO)
        PRTSTDOUT(*NO)
        DLTSTDOUT(*YES)
        IFSSTDOUT(*NO)
        IFSFILE(' ')
        IFSOPT(*REPLACE)
        CCSID(*SAME)
        PRTSPLF(QSHEXECLOG)
        PRTUSRDTA(*NONE)
        PRTTXT(*NONE)
```

QSHEXEC Command Temporary Objects

- **/tmp/qsh** (Temporary log file IFS location)
- Use **ERASE '/tmp/qsh/*'** CL command to clear IFS dir periodically
- **QSHONITMP library**
- Can use QSHONITMP library for temporary files or objects
- QSHONITMP auto-created by QSHell on i commands
- **CLRLIB QSHONITMP** periodically or after system IPL to keep clean

MobiGoGo
Work at the Speed of Mobility

CMDLINE and SETPKGPATH parameters

- **CMDLINE** - Takes any Qshell or PASE command line
- **SETPKGPTH** adds /QOpenSys/pkgs/bin to search path
- Ensures IBM Open-Source Package Management Yum packages found

MobiGoGo
Work at the Speed of Mobility

Logging Command Output

- Outfile QTEMP/STDOUTQSH is always created
- There are several additional options for handling STDOUT logs

Displaying Standard Output - *NO

DSPSTDOUT - *NO – DO NOT Display standard output result

- Set this setting to *NO if you don't want to display console results
- After running, the stdout log output from /tmp/qsh IFS file for job is always captured to outfile QTEMP/STDOUTQSH so it can be used.
- After running the command, the STDOUT log outfile can be used to selectively process log information returned from Qshell/PASE program call via STDOUT.
- **You would use this mode for production where you might want to selectively process only certain messages in the log file.**
- **We utilize this OUTFILE to write the STDOUT messages to joblog as well if that option is enabled.**

Displaying Standard Output - *YES

DSPSTDOUT - *YES – Display standard output result

- Set this setting to *YES if you want to display console results
- After running, the stdout log output from /tmp/qsh IFS file for job is always captured to outfile QTEMP/STDOUTQSH so it can be used.
- After running Qshell/PASE program or command, the STDOUT log data is displayed interactively from the outfile.
- **You would Only use this mode for testing and debugging when you need to see Qshell/PASE console output logs.**

Log Standard Output - *YES

LOGSTDOUT – Log standard output to job log

- Set this setting to *YES if you want to write STDOUT to the main job log
- After running the command, the STDOUT log data is written to the calling jobs job log.
- Each console message will have a CPF message id of: **QSS9898**
- Job log data can be used for debugging
- Job log data can be captured for use in subsequent job steps
- **You would normally use this option in production perhaps when you want to capture QShell/PASE output to your job log.**
- **If you log LOTS of messages to STDOUT, you probably want to leave this setting to *NO or you could overload your joblog with messages.**

Print Standard Output - *YES

PRTSTDOUT – Print standard output result

- Set this setting to *YES if you want to print your command log
- After running the command, the STDOUT log data is written to a spool file
- **Spool file name, user data and print text** can be specified
- **You would normally use this option in production perhaps when you want to capture STDOUT log output to a spool file for auditing rather than an OUTFILE or the job log.**

Delete Standard Output - *YES

DLTSTDOUT – Delete standard output result

- Set this setting to *YES to delete the IFS log file after processing
- After running, the stdout log output from /tmp/qsh IFS file for job is always captured to an outfile so it can be used.
- STDOUT data is optionally written to the joblog or printed to a spool
- Finally the temporary STDOUT log file in /tmp/qsh is deleted if this setting is set to *YES which is the default
- **This parameter should normally ALWAYS be set to *YES unless you're debugging an unknown problem. Normally you always want to clean up these files since the data gets captured to an OUTFILE automatically before cleanup anyway**

Copy STDOUT to IFS File - *YES

IFSSTDOUT – Copy standard output result to IFS file

- Set this setting to *YES to copy or append STDOUT to IFS file
- Allows /tmp/qsh temporary IFS log file to be copied or appended to IFS
- Finally the temporary STDOUT log file in /tmp/qsh is deleted if this setting is set to *YES which is the default
- Useful to aggregate log info to a single IFS log file.
- **This parameter should normally ALWAYS be set to *NO unless you want to copy or aggregate STDOUT data to a single IFS file location or directory before deleting the temporary IFS stdout log.**

MobiGoGo
Work at the Speed of Mobility

Integrating QShell/PASE Calls with IBM i Jobs

- Use the **QSHEXEC, QSHBASH or QSHPYRUN** command in **QSHONI** lib
- Allows Qshell/PASE calls to be embedded in CL, RPG and COBOL
- Called via standard QCMDEXC mechanism
- Pass complete commands with parameters in to QShell/Pase calls
- Receive return parameters from calls via Console/STDOUT log
- Pipeline STDOUT directly back to job so RPG/CL/COBOL can process any response information and check for errors in the logs.
- Send STDOUT to IFS file, outfile, job log or print file

MobiGoGo
Work at the Speed of Mobility

Integrate QShell/Pase Application via QShell on i

QSHEXEC, QSHBASH, QSHPYRUN CL command
executed from CL, RPG or COBOL application with parameters

QSHEXEC, QSHBASH or QSHPYRUN CL command calls Qshell or PASE command

New Qshell/PASE process starts to run command or program

PASE Pgm/Cmd Runs until Completion – Console logs captured to /tmp IFS log and outfile in QTEMP

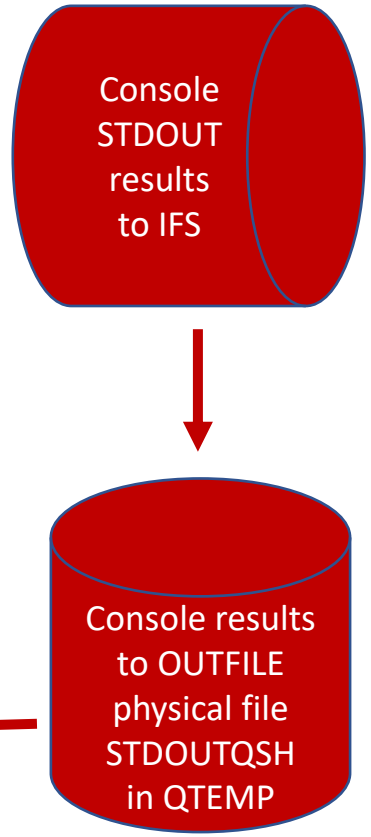
If Qshell/PASE command succeeds, exit code = 0. If error, exit code will be <> 0.

Qshell/PASE process ends after call to run command or app

Control returned to QSHEXEC, QSHBASH or QSHPYRUN CL
command which sends escape or completion message

Control returned to original CL, RPG or COBOL application

Original application reads and processes /tmp/qsh IFS log file or outfile – QTEMP/STDOUTQSH



VS Develop Demo

Demo

MobiGoGo
Work at the Speed of Mobility

Hello World Python Sample

- Simple Hello world example
- Illustrates the plumbing and how it works

MobiGoGo
Work at the Speed of Mobility

Python Directory Crawler Example

- Crawl Directory Tree
- Capture output to a tilde delimited IFS text file
- CPYFRMIMPF of data from IFS text file to PF – DIRCRAWL
- RPG program to read and process the PF

MobiGoGo
Work at the Speed of Mobility

Next Steps

- Install QShell on i Library

  https://github.com/richardschoen/Qshoni

- Start using it

- Try example Python scripts

  https://github.com/richardschoen/RpgOpenSource

- Give me feedback or example of how you are using

  richard@richardschoen.net