



open source

RPG Makes Friends with Open-Source Apps

Open Your i

Your Presenter

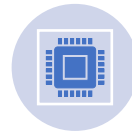


Richard Schoen

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 Native IBM i Git client**
for RDI, SEU and PDM developers

Github: <http://github.com/richardschoen>

Business site: <http://www.mobigogo.net>

Personal site: <http://www.richardschoen.net>

LinkedIn: <https://www.linkedin.com/in/richardschoen>

Twitter: @richardschoen

Blog: <http://blog.richardschoen.net>

iForgit IBM i Git Client: <http://www.iforgit.com>



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 smart devices)**
The greatest computing era !!

Session Topics

Why consider using RPG and Open Source together ?

How to utilize open-source applications from RPG and CL

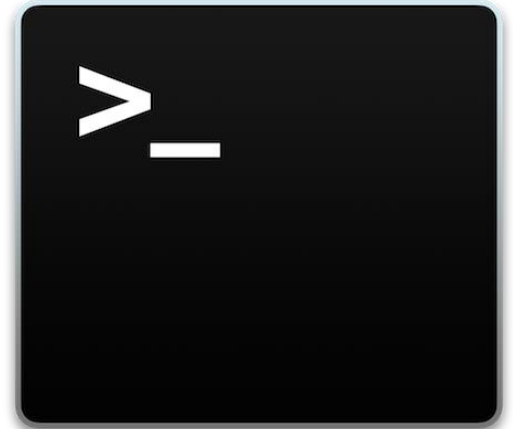
Intro to QShell on i

Review logic flow

A few examples

How to use shells on the IBM i

- Your command line for QShell, PASE and Bash
- A good intro to the various shells on IBM i
- Andy Youens – FormaServe (UK)
- <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

Send and receive email

Crawl IFS directories

Become productive quickly

Integrates or extends existing IBM i apps across process boundaries

QShell on i can be used for all kinds of IBM i utility programs

Run any QShell or PASE code from traditional RPG/CL/COBOL programs

Get The Best of Both Worlds

```
MAIN                                IBM i Main Menu                                System:  SYS1
Select one of the following:
  1. User tasks
  2. Office tasks
  3. General system tasks
  4. Files, libraries, and folders
  5. Programming
  6. Communications
  7. Define or change the system
  8. Problem handling
  9. Display a menu
 10. Information Assistant options
 11. IBM i Access tasks
 90. Sign off
Selection or command
==>
F3=Exit  F4=Prompt  F9=Retrieve
F23=Set initial menu
(C) COPYRIGHT IBM CORP. 1980, 2
MA+ A MJ
```

```
-bash-4.4$ python3 HelloWorld.py
Script Running:HelloWorld.py
Hello World
Your message is:No message passed
-bash-4.4$ python3 HelloWorld.py
```

Traditional 5250
Applications
RPG, CL, COBOL
Applications

QShell, PASE,
Bash Shell
AIX Apps
Java, Python, PHP, Node, C,
C++, Ruby, R, Rust, ODBC,
Postgres, MariaDB,
SQL Server

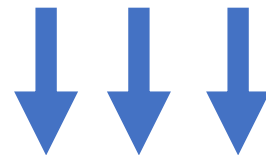
Traditional IBM i Job Program Call Flow

QTEMP
Library

Type CL Command or
Embed in a CL Program

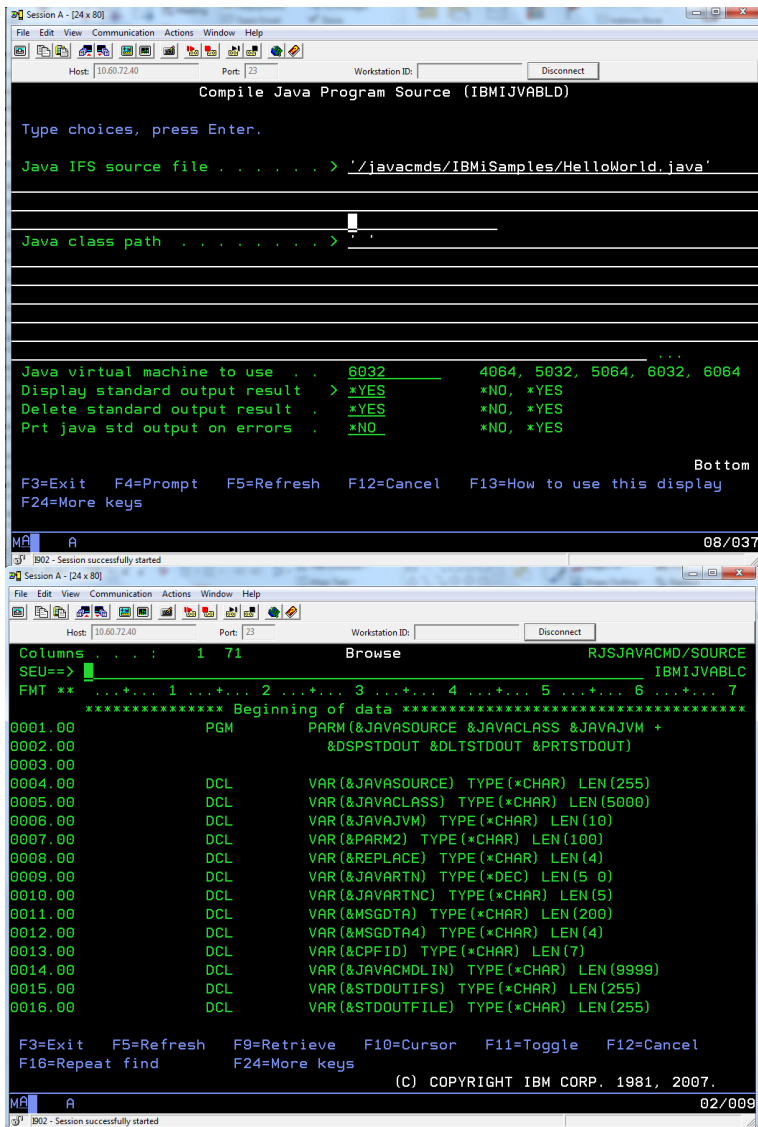
Run CL/RPG Processing Pgm
May pass parameters

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



```

Session A - [24 x 80]
File Edit View Communication Actions Window Help
Host: [10.60.72.40] Port: [23] Workstation ID: [ ] Disconnect

Compile Java Program Source (IBMIJVABLD)

Type choices, press Enter.

Java IFS source file . . . . . > '/javacmds/IBMiSamples/HelloWorld.java'

Java class path . . . . . >

Java virtual machine to use . . 6032 4064, 5032, 5064, 6032, 6064
Display standard output result > *YES *NO, *YES
Delete standard output result . *YES *NO, *YES
Prt java std output on errors . *NO *NO, *YES

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys

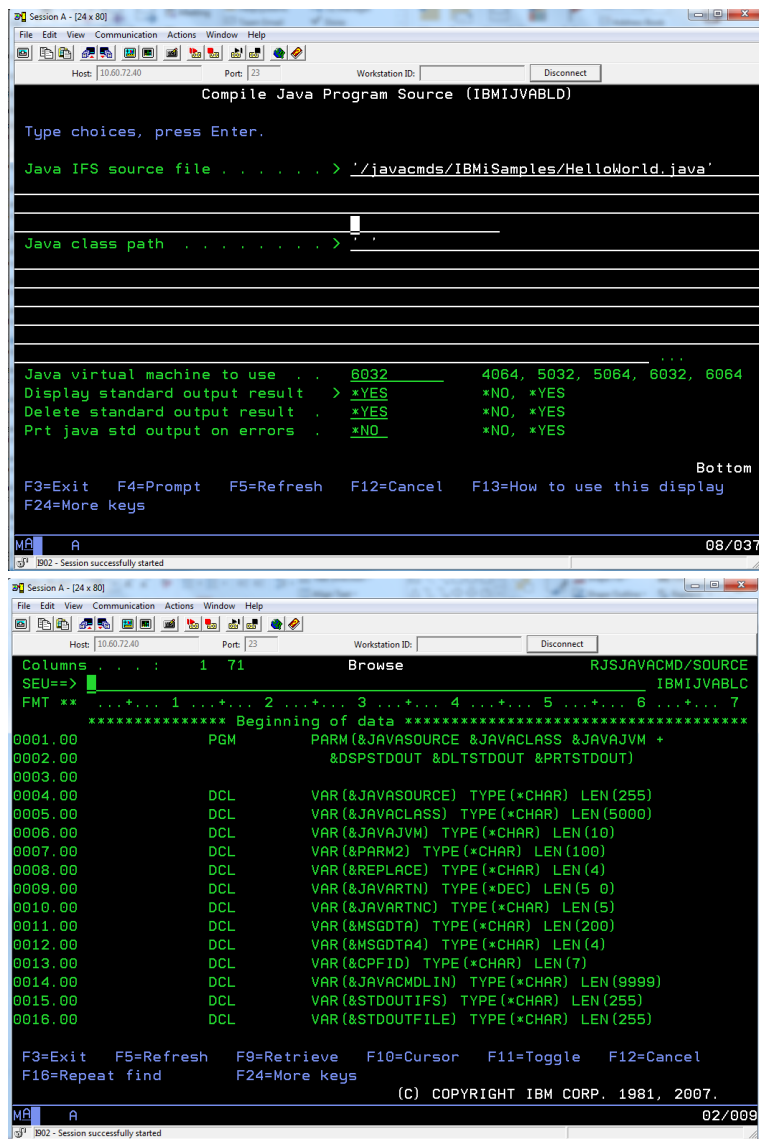
Bottom
08/037

Session A - [24 x 80]
File Edit View Communication Actions Window Help
Host: [10.60.72.40] Port: [23] Workstation ID: [ ] Disconnect

Columns . . . . . 1 71 Browse RJSJAVACMD/SOURCE
SEU=> IBMJVABLD
FMT ** ...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7
***** Beginning of data *****
0001.00 PGM PARM(&JAVASOURCE &JAVACCLASS &JAVAJVM +
0002.00 &DSPSTDOUT &DLTSTDOUT &PRTSTDOUT)
0003.00
0004.00 DCL VAR(&JAVASOURCE) TYPE(*CHAR) LEN(255)
0005.00 DCL VAR(&JAVACCLASS) TYPE(*CHAR) LEN(5000)
0006.00 DCL VAR(&JAVAJVM) TYPE(*CHAR) LEN(10)
0007.00 DCL VAR(&PARM2) TYPE(*CHAR) LEN(100)
0008.00 DCL VAR(&REPLACE) TYPE(*CHAR) LEN(4)
0009.00 DCL VAR(&JAVARTN) TYPE(*DEC) LEN(5 0)
0010.00 DCL VAR(&JAVARTNC) TYPE(*CHAR) LEN(5)
0011.00 DCL VAR(&MSGDTA) TYPE(*CHAR) LEN(200)
0012.00 DCL VAR(&MSGDTA4) TYPE(*CHAR) LEN(4)
0013.00 DCL VAR(&CPFLD) TYPE(*CHAR) LEN(7)
0014.00 DCL VAR(&JAVACMDLIN) TYPE(*CHAR) LEN(9999)
0015.00 DCL VAR(&STDOUTIFS) TYPE(*CHAR) LEN(255)
0016.00 DCL VAR(&STDOUTFILE) TYPE(*CHAR) LEN(255)

F3=Exit F5=Refresh F9=Retrieve F10=Cursor F11=Toggle F12=Cancel
F16=Repeat find F24=More Keys
(C) COPYRIGHT IBM CORP. 1981, 2007.
02/009
  
```


Traditional Program Call Flow with QShell/PASE Apps



```

Session A - [24 x 80]
File Edit View Communication Actions Window Help
Host: 10.60.72.40 Port: 23 Workstation ID: Disconnect
Compile Java Program Source (IBMIJVABLD)
Type choices, press Enter.
Java IFS source file . . . . . '/javacmds/IBMiSamples/HelloWorld.java'
Java class path . . . . .
Java virtual machine to use . . 6032 4064, 5032, 5064, 6032, 6064
Display standard output result > *YES *NO, *YES
Delete standard output result . *YES *NO, *YES
Prt java std output on errors . *NO *NO, *YES
F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys
08/037

Session A - [24 x 80]
File Edit View Communication Actions Window Help
Host: 10.60.72.40 Port: 23 Workstation ID: Disconnect
Columns . . . : 1 71 Browse RJSJAVACMD/SOURCE
SEU=> IBMJVABLC
FMT ** ..... 1 ..... 2 ..... 3 ..... 4 ..... 5 ..... 6 ..... 7
***** Beginning of data *****
0001.00 PGM PARM(&JAVASOURCE &JAVACCLASS &JAVAJVM +
0002.00 &DSPSTDOUT &DLTSTDOUT &PRTSTDOUT)
0003.00
0004.00 DCL VAR(&JAVASOURCE) TYPE(*CHAR) LEN(255)
0005.00 DCL VAR(&JAVACCLASS) TYPE(*CHAR) LEN(5000)
0006.00 DCL VAR(&JAVAJVM) TYPE(*CHAR) LEN(10)
0007.00 DCL VAR(&PARM2) TYPE(*CHAR) LEN(100)
0008.00 DCL VAR(&REPLACE) TYPE(*CHAR) LEN(4)
0009.00 DCL VAR(&JAVARTN) TYPE(*DEC) LEN(5 0)
0010.00 DCL VAR(&JAVARTNC) TYPE(*CHAR) LEN(5)
0011.00 DCL VAR(&MSGDTA) TYPE(*CHAR) LEN(200)
0012.00 DCL VAR(&MSGDTA4) TYPE(*CHAR) LEN(4)
0013.00 DCL VAR(&CPFID) TYPE(*CHAR) LEN(7)
0014.00 DCL VAR(&JAVACMDLIN) TYPE(*CHAR) LEN(9999)
0015.00 DCL VAR(&STDOUTIFS) TYPE(*CHAR) LEN(255)
0016.00 DCL VAR(&STDOUTFILE) TYPE(*CHAR) LEN(255)
F3=Exit F5=Refresh F9=Retrieve F10=Cursor F11=Toggle F12=Cancel
F16=Repeat find F24=More keys
(C) COPYRIGHT IBM CORP. 1981, 2007.
02/009
  
```

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

Parameters Returned in STDOUT
IFS file from QShell/Pase call

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

~~QTEMP
Library~~

TMP
Library

QShell/PASE app runs in a
thread job

STDOUT
log info can be
Processed by RPG

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

Integrate QShell/Pase Application via QShell on i

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

QSEXEC, 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

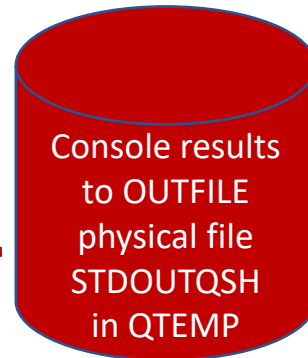
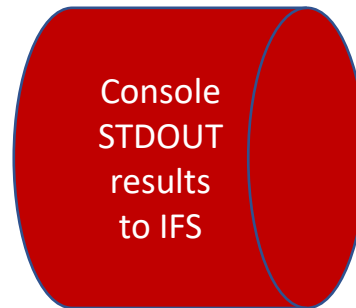
If QShell/PASE command succeeds, exit code = 0. If error, exit code will be $\neq 0$.

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

Control returned to QSEXEC, 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



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

```
-bash-4.4$ python3 HelloWorld.py
Script Running:HelloWorld.py
Hello World
Your message is:No message passed
-bash-4.4$ python3 HelloWorld.py
```

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 capture
- 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

QShell on i – Use Cases

- Call QShell, Pasa 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/Pasa commands
- Consume web services without HTTPGETCLOB/HTTPPOSTCLOB (Java)
- Compose, send and receive emails
- Talk to other databases (PostgreSQL, MariaDB, SQL Server, SQLite)

IBM i System Prerequisites – V7R2 and Above

- IBM Open Source Package Management
- Install all Python 3 packages (if using Python)
- Install unixODBC and unixODBC-dev
- Install IBMi Access ODBC Driver
- Install freetds if you want to talk to SQL Server
- Install QShell on i (<https://github.com/richardschoen/qshoni>)
- **Uninstall 5733OPS if you can.** Yum packages rule

Install All Python 3 Packages

IBM i Access Client Solutions

File Edit Actions Tools Help

Welcome

System: sysi1

General

- Data Transfer
- 5250 Emulator
- Integrated File System
- Navigator for i
- SSH Terminal
- Printer Output

Database

- Schemas
- Run SQL Scripts
- SQL Performance Center

Console

- 5250 Console
- Virtual Control Panel
- Hardware Management Interface 1

Management

- System Configurations
- 5250 Session Manager
- HMC Probe Utility

Open Source Package Management

File View Connection Utilities

Connection: richard@sysi1:/

Installed packages

Updates available

Available packages

Package	Version	Repository
python3-Pillow	5.0.0-6	@ibm
python3-asn1crypto	0.24.0-1	@ibm
python3-bcrypt	3.1.4-6	@ibm
python3-cffi	1.11.5-3	@ibm
python3-cryptography	2.8-0	@ibm
python3-dateutil	2.8.0-1	@ibm
python3-devel	3.6.12-1	@ibm
python3-ibm_db	2.0.5.12-0	@ibm
python3-idna	2.8-1	@ibm
python3-itookit	1.6.1-1	@ibm
python3-jinja2	2.11.2-1	@ibm
python3-lxml	4.2.1-4	@ibm
python3-markupsafe	1.1.1-1	@ibm
python3-numpy	1.15.4-1	@ibm
python3-pandas	0.22.0-5	@ibm
python3-paramiko	2.6.0-1	@ibm
python3-pip	9.0.1-3	@ibm
python3-psutil	5.5.1-1	@ibm
python3-psycopg2	2.8.5-1	@ibm
python3-pyparser	2.19-2	@ibm
python3-pynad	1.2.1-4	@ibm
python3-pyodbc	4.0.27-0	@ibm
python3-pytz	2018.5-3	@ibm
python3-pyyaml	5.3.1-1	@ibm
python3-pyzmq	17.1.2-0	@ibm
python3-rpm	4.13.1-12	@ibm
python3-sikit-learn	0.19.1-8	@ibm

Done: 444 rows retrieved.

Information

Show files

Reinstall

Remove

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

- **QSEXEC** - Run QShell Command Line
- **QSBASH** - Run Bash Command via QShell
- **QSPYRUN** - Run Python Script via QShell
- **QSHLOGSCAN** - Scan QShell Log File for Values
- **QSPATH** - Set Open Source Package Path Environment Variable
- **QSHQRYTMP** – SQL Query Data to Temp Table
- **QSHIFSCHK** – Check for IFS file existence

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 direct STRQSH call
- Can be called directly from RPG or CL

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 call to STRQSH
- All the same benefits of QSHEXEC because it **uses QSHEXEC**
- Can be called directly from RPG or CL

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 because it uses QSHEXEC
- Can be called directly from RPG or CL

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 via RPG program
- 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

QSHQRYTMP – SQL Query Data to Temp Table

- Convenience command to run SQL queries to output table
- Can be used to select and sort data
- Wrapper over RUNSQL
- Use for SELECT queries only
- Should work with IBM i DB2 Services
- Outputs to selected OUTFILE
- OUTFILE file results can be consumed from RPG/CL/COBOL

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

QSHPATH – Set Open-Source Package Path

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

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

CMDLINE and SETPKGPATH parameters

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

Logging Command Output

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

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.**

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.**

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.**

Integrating QShell/PASE Calls with IBM i Jobs

- Use the **QSEXEC**, **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

Qshell on i Demo

Demo

Wrap Simple Qshell/Pase Commands

- List files in IFS directory with ls command

```
QSHEXEC CMDLINE('cd /rpgopensource; ls -l') DSPSTDOUT(*YES)
```

- Locate files starting with **Send***

```
QSHEXEC CMDLINE('find /rpgopensource -name Send*')  
DSPSTDOUT(*YES)
```

- Call shell script to ping an IP address – **PASE/Qshell has no PING cmd**

```
QSHEXEC CMDLINE('cd /rpgopensource; pingi.sh "8.8.8.8"')  
DSPSTDOUT(*YES)
```

Hello World Python Sample

- Simple Hello world example
- Illustrates the plumbing and how it works a couple ways
- Python script – **helloworld.py**
- Calling via **QSHEXEC** or **QSHPYRUN**
- Calling via convenience wrapper command: **HELLOWORLD**

Use Python 3 Qsh/Pase/Bash Command Line

- Python 3 command line.
- Call from pase or bash command line or shell script

HelloWorld.py “This is a test”

python3 HelloWorld.py “This is a test”

Use QSHPYRUN Command

- Call Python via **QSHPYRUN** or **QSHEXEC** Command
- Call interactively from RPG/CL, schedule or submit to batch

```
QSHONI/QSHEXEC CMDLINE('python3 /rpgopensource/helloworld.py  
"This is a test from IBM i") DSPSTDOUT(*YES)
```

```
QSHONI/QSHPYRUN SCRIPTDIR('/rpgopensource')  
  SCRIPTFILE(helloworld.py)  
  ARGS('This is a test from IBM i') DSPSTDOUT(*YES)
```


Use HELLOWORLD Wrapper Command

- Make calling Python more user friendly
- Provides context to the actual function being performed
- Call interactively from RPG/CL, schedule or submit to batch

HELLOWORLD MESSAGE('This is my Hello World Message')
DSPSTDOUT(*YES)

Send Authenticated Email via Office 365

- Company has moved mail from On-Prem Exchange to Office 365
- Need new way to send authenticated emails via O365
- Python script with O365 functionality – **SendO365.py**
- **QSHPYRUN** to run Python script from standard job
- **SNDMAIL365** command to wrap call to Python/QSHPYRUN
- Wrapper command makes the Python much more user-friendly

Use Python 3 Qsh/Pase/Bash Command Line

- Python 3 command line.
- Call from pase or bash command line or shell script

```
SendMail365.py "richard@richardschoen.net"  
"richard@richardschoen.net" "O365 Test from IBM i" "This is a test  
from IBM i" "/tmp/excel.xlsx" "richard@richardschoen.net" ""
```

```
python3 SendMail365.py "richard@richardschoen.net"  
"richard@richardschoen.net" "O365 Test from IBM i" "This is a test  
from IBM i" "/tmp/excel.xlsx" "richard@richardschoen.net" ""
```

Use QSHPYRUN Command

- Call Python via QSHPYRUN Command
- Call interactively from RPG/CL, schedule or submit to batch

```
QSHONI/QSHPYRUN SCRIPTDIR('/rpgopensource')  
SCRIPTFILE(SendMail365.py)  
ARGS('richard@richardschoen.net'  
'richard@richardschoen.net'  
'O365 Test from IBM i - QSHPYRUN'  
'This is a test from IBM i'  
'/rpgopensource/excel.xlsx'  
'richard@richardschoen.net' ' ')
```

Use SNDMAIL365 Wrapper Command

- Make calling Python more user friendly
- Provides context to the actual function being performed
- Call interactively from RPG/CL, schedule or submit to batch

```
SNDMAIL365 FROMEMAIL('richard@richardschoen.net')  
TOEMAIL('richard@richardschoen.net')  
SUBJECT('O365 Test from IBM i - SNDMAIL365')  
BODYTEXT('This is the body text')  
ATTACHFILE('/rpgopensource/excel.xlsx')  
O365USER(richard@richardschoen.net)  
O365PASS(' ')  
DSPSTDOUT(*YES)
```

Python Directory Crawler Example

- Crawl IFS Directory Tree (Including IFS files and Libraries)
- Capture output to a tilde delimited IFS text file
- CPYFRMIMPF of data from IFS text file to PF – DIRCRAWL
- Use RPG program or SQL to read and process the PF

Python Directory Crawler Example

- Crawl selected dir tree
**QSHONI/QSHEXEC CMDLINE('/rpgopensource/pydircrawl.py
"/python" "/tmp/dircrawl.txt" "true" "false" "~" "true")
DSPSTDOUT(*YES)**
- Copy contents of IFS text file to PF
**CPYFRMIMPF FROMSTMF('/tmp/dircrawl.txt')
TOFILE(QSHONI/DIRCRAWL)
MBROPT(*REPLACE)
RCDDL(*LF)
FLDDL('~')
FROMRCD(2)**

Python Directory Crawler Example

- Create SQL Table for Directory Crawler Data

```
create table qshoni.dircrawl  
(IFSFULL VARCHAR(256),  
IFSFIL FILE VARCHAR(256),  
IFSPREFIX VARCHAR(256),  
IFSEXT VARCHAR(10),  
IFSSIZE DECIMAL(15,2),  
IFSTYPE VARCHAR(5),  
IFSSYMLNK VARCHAR(5));
```


Use DIRCRAWL Wrapper Command

- Make calling Python more user friendly
- Provides context to the actual function being performed
- Call interactively from RPG/CL, schedule or submit to batch
- Creates OUTFILE in library TMP automatically

```
DIRCRAWL TOPDIR('/rpgopensource')  
  OUTPUTFILE('/tmp/dircrawl.txt')  
  REPLACE('True')  
  FOLLOWLNK('False')  
  DELIM(',') OFILE(DIRCRAWL) OLIB(TMP)
```

Python Read Web Page Example

- Read web page or web service URL
- Output to IFS file

```
QSHONI/QSHPYRUN SCRIPTDIR('/rpgopensource')  
    SCRIPTFILE(pyreadwebpage.py)  
    ARGS('/tmp/wegpageinfo.txt')  
    PYVERSION(3)  
    DSPSTDOUT(*YES)
```

Python Read Web Page RPG Example

- Put it all together
- Reads web page or web service URL via Python
- Output JSON to IFS file
- Consume JSON via YAJL functions from RPG program,
- Insert in to DB2 table RPGOPNSRC/

CALL RPGOPNSRC/JSONYAJL1

Python Learning Links

- There are tons of free and paid learning resources
- Seiden Group offers paid Python, PHP and other training classes
<http://www.seidengroup.com>
- Learn Python - Full Course for Beginners [Tutorial]
<https://www.youtube.com/watch?v=rfscVS0vtbw>
- Freecodecamp
<https://www.freecodecamp.org/news/search/?query=python>
- Google
<https://www.google.com/search?q=python+free+training>

Next Steps

- Install QShell on i Library

<https://github.com/richardschoen/Qshoni>

- Start using it to call: python, bash, node, php and other scripts
- Try example Python scripts and IBM i RPG Samples

<https://github.com/richardschoen/RpgOpenSource>

- Give me feedback or examples of how you are using this

richard@richardschoen.net