Beyond Variables & Parms Part Too/Also

Beyond Variables And Parms

- Deep dive into dynamic variables for distributed Unix and some Windows.
- Command line based presentation, no DWC GUI involved.
- You will learn how to use variables in some new ways
  - Executable and DB job types
  - OS vars mixed with parms
- Go over the gotchas

Beyond Variables And Parms Agenda

- OS level TWS Variables
- FTA Parms (quick review)
- Dynamic Parms
- Examples
- Gotchas
- Questions

Beyond Variables And Parms

OS Variables (Discover via a test job):

Unix:	Windows
.profile .bashrc	System/User Environment
jobmanrc .jobmanrc	jobmanrc djobmanrc
$VARIABLE	%VARIABLE%
env	set
echo $?	%ERRORLEVEL%

Beyond Variables And Parms

Why 2 sets of parms:

- Tried to use FTA type ^^s and parms in dynamic job definitions.

- ^^s and parms **cannot* be read in dynamic job definitions**

- However you can use the **${} type** variables set by jobprop and param

- So if you need to set values in dynamic jobs, you are likely to need to use the new syntax.

* exceptions exist, aka in OPENS in jobstreams, and setting/reading parms, however do not expect them to work everywhere.

Beyond Variables And Parms

FTA level Parms/Vars:

parms: Unix exe, everyone can run, scope is each CPU (or workstation)

^^ "hats" with tables. In DB, scope is everything.

Dynamic Level parms/vars

params Unix exe, everyone can run, scope is each CPU (or workstation)

jobprop Unique to whatever scope you set, can pass data from job to job

Beyond Variables And Parms

Environments Used

- MDM/DWC OS RHEL 6/7
- FTA/DYN AIX 6/7 RHEL 6/7 Windows 2008/2012
- TWS level 9.3 FP2/FP3 UNIX – 9.4 Windows
- DWC non-root install / MDM root install
- FTA/DYN installed as a pair

Beyond Variables And Parms

Hats (also called parms and variables)

Set in TWS DB.

No problem in FTA jobs/jobstreams

nonreferencable via normal syntax in dynamic jobs

Loadable via the $PARMS syntax

Limited to 16 characters in name

Ex:

^MYVAR^

parms

Called on the command line

Unique to each workstation

Can change dynamically easily during TWS plan.

Limited to 16 characters in name

Manage security in dumpsec/makesec

Ex:

`parms -c OTHER-VAR value`

Beyond Variables And Parms

jobprop

Can only be called from a dynamic job

Set's a value that other dynamic jobs can reference to a scope you set

Set Command (inside dynamic executable job called MYFTA#MYJOB)

#call your tws_env with .

jobprop var1 value1
Use on DYN Job (inside XML in ANOTHER job)
${job:MYJOB.var1}

Note the exact usage, including the word job, where the jobname goes and the : . And so on

params

Works similar to jobprop however stored on agent side in flat file.

Examples in the docs I used did not work for me.
Stored locally on each dynamic agent (like parms)

Display Command and output

param VARNAME
jm_variables..VARNAME = thevalue

Set Command
param -c VARNAME thevalue

Use on a DYN job where it is set:
${agent :VARNAME}

Note the word agent and that param works on command line.

Beyond Variables And Parms

stdlist dynamic variable

Can ref stdlist in dyn jobs, however it is limited to the 1st 2-4kb and thus if your output!

In executable file do this if the job you need to see is called JOBNAME:

echo "${job:JOBNAME.stdlist}" >> output-stdlist.txt

Beyond Variables And Parms

Applications used: conman and DYN jobs

- Submitting dynamic job on the fly with conman (can we do it????)
- Passing what variables in dynamic jobs? What works?

Beyond Variables And Parms

Dynamic Job submission (allowed but fails).

Submit a dynamic job via sbd.

How?

Problem, there is a size limit on sbd!

So like OPENS, let's create many variables.

Beyond Variables And Parms

parms -c pmla "<?xml verson=\"1.o\" encoding=\"UTF-8\"?><jdsl:jobDefinition xmlns:jsdl"

parms -c pmlb "=\http://www.ibm.com/xlns/prod....

and so on .. the simplest job takes 16 parms!

Then submit:

sbd DYNAGT#"`parms pmla``parms pmlb`` ..... ";alias="JOBALIAS"

And we get an error message even when uncompressed this way under the limits!

sbd checks the uncompressed length!

Beyond Variables And Parms

How to use DYN VARIABLES example with more details:

1. USE THE USERS TABLE IN TWS for usernames and passwords.
2. (Most dynamic jobs require USERS and PASSWORDS. If you set the password values securely in the TWS DB, you can refer to those values from there for dynamic workload instead of encrypting and embedding in every individual job!)
3. **${password:USERNAME}**
4. where username is the actual username in the TWSDB.
5. Set jobprop variables in one job, and then can read them from a 2nd job.
6. Submit dynamic jobs in schedules using the traditional sbs syntax from conman.

Beyond Variables And Parms

Gotcha List of Warnings.

1. sbd limits are too short for dynamic jobs
2. stdlist variable ref limits 2kb - 4kb
3. ^^s do not work in dynamic jobs anywhere
4. parms are not as useful for dynamic agents.
5. jobprop does not work on command line by design.
6. param variables is stored locally on agent
   in $TWSHOME/ITA/cpa/config/jm_variables_files

Addendum

```
MYLINUX#THISJOB
 TASK
   <?xml version="1.0" encoding="UTF-8"?>
   <jsdl:jobDefinition
        xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
        xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
   <jsdl:application name="executable">
   <jsdle:executable interactive="false">
   <jsdle:script>
  . /home/twsuser/.profile
  jobprop testserver `cat /tmp/testserver`
   jobprop thiscmd “df -kh”
 </jsdle:script>
   </jsdle:executable>
   </jsdl:application>
   </jsdl:jobDefinition>
 DESCRIPTION "Test variables."
 RECOVERY STOP
```

Addendum

MYLINUX#REMCMD
TASK

```
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
 xmlns:jsdlremotecommand="http://www.ibm.com/xmlns/prod/scheduling/1.0/
  jsdlremotecommand" name="REMOTECOMMAND">
 <jsdl:application name="remotecommand">
  <jsdlremotecommand:remotecommand>
                                    <jsdlremotecommand:RemoteCommandParameters>
                                          <jsdlremotecommand:taskPanel>
                                                <jsdlremotecommand:command>.
./home/twsuser/.profile;${job:THISJOB.thiscmd} </jsdlremotecommand:command>
                                          </jsdlremotecommand:taskPanel>
                                          <jsdlremotecommand:environmentPanel>
                                          <jsdlremotecommand:standardOutput/>
                                                <jsdlremotecommand:standardError/>
                                          </jsdlremotecommand:environmentPanel>
                                          <jsdlremotecommand:serverPanel>
```

Addendum

```
                                                    <jsdlremotecommand:serverInfo>

        <jsdlremotecommand:serverName>${job:THISJOB.testserver}</jsdlremotecommand:serverName>
                                                                <jsdlremotecommand:port/>

        <jsdlremotecommand:protocol>ssh</jsdlremotecommand:protocol>
                                                    </jsdlremotecommand:serverInfo>
                                                    <jsdlremotecommand:credentials>

        <jsdl:userName>twsuser</jsdl:userName>

        <jsdl:password>${password:twsuser}</jsdl:password>
                                                    </jsdlremotecommand:credentials>
                                                    <jsdlremotecommand:certificates>

        <jsdlremotecommand:keystoreFilePath/>
                                                    </jsdlremotecommand:certificates>
                                                </jsdlremotecommand:serverPanel>
                                            </jsdlremotecommand:RemoteCommandParameters>
                                        </jsdlremotecommand:remotecommand>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "remote command."
 RECOVERY STOP
```

Beyond Variables And Parms

Database Job Type (one solution of several):

- Oracle Job Type did not work by default.
- 1. Instead choose custom
- 2. Then for Oracle Jar file choose the parent folder of the jar file
- 3. Finally you need to put in your **correct jdbc string for your database install**.
- 4. Oracle SQL commands must drop the trailing **;** (however PL/SQL needs ;)
- 5. DWC will sometimes read in the SQL incorrectly so **you need to generate the xml on the commandline outside the DWC.**
- 6. Driver class is you use jdbc6 is oracle.jdbc.driver.OracleDriver
- 7. All xml-unsafe characters like < and > need to be changed to > < in the SQL because GUI cannot be safely used to do it for you.
- 8. For PL/SQL remove the outer BEGIN and END from the block

Beyond Variables And Parms

- Getting around stdlist limitations:
- 1. Use OS stdlist variable $UNISON_STDLIST to find the directory and name of the stdlist file
- 2. Once you know name of file, you can use cat or type (and so on) to dump the contents into a variable to be read and processed with grep, find and so on.
- 3. TWS always stores the jobnumber and stdlist file in variables for your usage on the fly
- 4. However TWS uses Unix filepaths even on windows (at least when you have windows agents and a Unix MDM) so you need to handle this properly by transforming the unix separators to windows separators to make use of these variables on windows.

What's your take?

Your feedback helps to set the course for future ASAP UNIVERSITY training initiatives.

- How will you use what you learned? Share your insights with users between sessions on the **ASAP User Forum** at twsuser.org.
- Remember to stay connected to ASAP. Tweet! **@ASAPUniversity**