Running Unix Shell Scripts from the PeopleSoft Process Scheduler

DAVID KURTZ, 24 FEBRUARY 2017

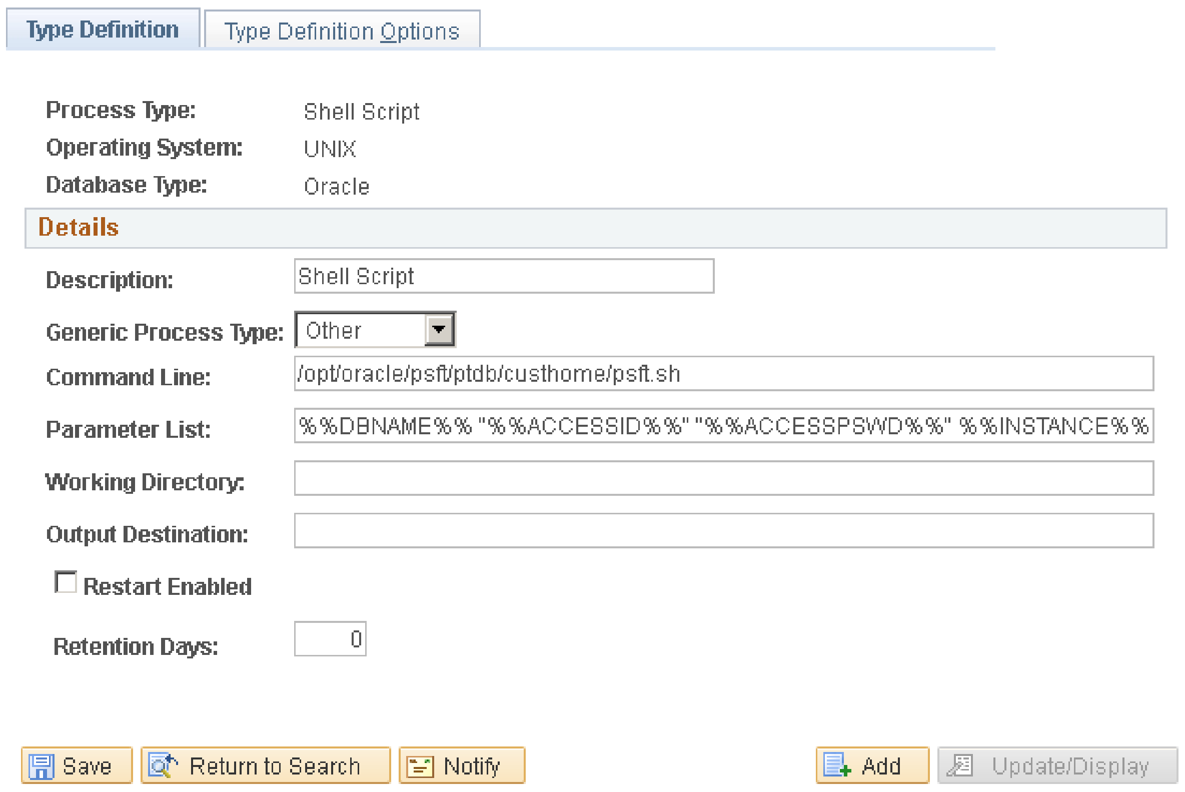
It is nearly 10 years since I wrote the original version of this article[1]. Very little has change, but I have had a number of questions recently, so I thought it was time I checked the script and updated the posting. I have used PeopleTools 8.54 in the preparation of this note.

The Process Scheduler is essentially just a mechanism for running commands on another server. Mostly those commands are to run other PeopleSoft delivered programs. The exception is the Application Engine Tuxedo server (PSAESRV) where the Process Scheduler submits a service request message that is picked up by one of the server processes.

NB: Although the PSAESRV server is configured by default in the Process Scheduler domain, Oracle recommend that you should only use this when you have lots of very short-lived (runtime less than 30 seconds) application engine processes. Typically, this only occurs in CRM.

Process Type Definition

First you need to create a new process type. I chose to call it 'Shell Script'. It runs a named shell wrapper script, psft.sh. The wrapper script calls the script that is to be executed. Note that the command line in the process type definition includes the fully qualified path.



Process Type:	Shell Script
Operating System:	UNIX
Database Type:	Oracle

Details

Description:	Shell Script
Generic Process Type:	Other
Command Line:	/opt/oracle/psft/ptdb/custhome/psft.sh
Parameter List:	%%DBNAME%% "%%ACCESSID%%" "%%ACCESSPSWD%%" %%INSTANCE%%
Working Directory:	
Output Destination:	
☐ Restart Enabled	
Retention Days:	0

[1] http://blog.psftdba.com/2007/09/running-unix-commands-and-scripts-from.html

Wrapper Script

This is the wrapper script that will be called by the process scheduler.

```
#!/bin/ksh
# (c) David Kurtz 2007
# Script:  psft.sh
#
# Syntax:  psft.sh DBNAME ACCESSID ACCESSPSWD PRCSINSTANCE
# where
# DBNAME is the name of the PeopleSoft datbase with a corresponding TNS entry
# ACCESSID is the schema containing the PeopleSoft database
# ACCESSPSWD is the password to ACCESSID
# PRCSINSTANCE is the process instance number supplied by PeopleSoft
#
# Purpose: To start Standard UNIX Shell Script from Process Scheduler, and interface with the
PeopleSoft Process Scheduler
# 07.09.2007 Initial Version
# 23.02.2017 Remove unnecessary logfiles section
#set -x

if [ $# -lt 4 ]; then
  echo "Usage $0: <DBNAME> <ACCESSID> <ACCESSPSWD> <PRCSINSTANCE> <command>"
  exit 1
fi

CONNECT=$2/$3@$1[2]
PRCSINSTANCE=$4
shift 4

#
# Function to set status of API aware process instance
#
function prcsapi[3]
{
if [ $# -lt 2 ]; then
 echo "Parameter Error in function $0"
 exit 1
fi

TIMESTAMPCOL=${1}
STATUS=${2}

if [ ${PRCSINSTANCE} -gt 0 ];then
  echo "Setting process request ${PRCSINSTANCE} to status ${STATUS}"
  sqlplus -S /nolog <<!
set termout off echo off feedback off verify off
connect ${CONNECT}
UPDATE psprcsque
SET    runstatus = ${STATUS}
,      sessionidnum = $$[4]
,      lastupddttm = SYSTIMESTAMP
WHERE  prcsinstance = ${PRCSINSTANCE}
;
UPDATE psprcsrqst
SET    runstatus = ${STATUS}
,      prcsrtncd = ${PRCSRTNCD}
,      continuejob = DECODE(${STATUS},2,1,7,1,9,1,0)[5]
,      ${TIMESTAMPCOL} = SYSTIMESTAMP
,      lastupddttm = SYSTIMESTAMP
WHERE  prcsinstance = ${PRCSINSTANCE}
;
COMMIT;
exit
!

  RET=$?
  if [ ! $RET ];then
```

[2] The Oracle user ID, password and TNS name for the PeopleSoft database are supplied in the first three parameters to the wrapper script. The PeopleSoft Process Instance number is the 4th command line parameter. These parameters are then removed with the shift command leaving any other parameters that have been specified.

[3] Function *prcsapi* sets the status on the process request row and updates the appropriate timestamp columns in the Process Scheduler tables. It is this that makes the script API aware.

[4] *PSPRCSQUE.SESSIONIDNUM* holds the operating system process ID of the shell executing the wrapper script.

[5] When the process completes and an end of process status is set (either 9 for success, 3 for failure or 2 for delete) CONTINUEJOB is set to 1, otherwise it is set to 0.

```
    echo "SQL*Plus Error Return Code: $?"
  fi
fi
}

#
# Main Execution Starts Here
#

echo $0:$*
date
uname -a
echo "Current Directory: `pwd`"
echo "Process log files in: ${PSPRCSLOGDIR}"

PRCSRTNCD=0
prcsapi begindttm 7[6]

#Run the command
$*
PRCSRTNCD=$?[7]

if [ ${PRCSRTNCD} -ne 0 ]; then
  prcsapi enddttm 3 # failure
else
  prcsapi enddttm 9 # success
fi

date
```

[6] When the wrapper scripts start it sets the process status on the process request record to 7 indicate that it is processing. This can be seen in the Process Monitor.

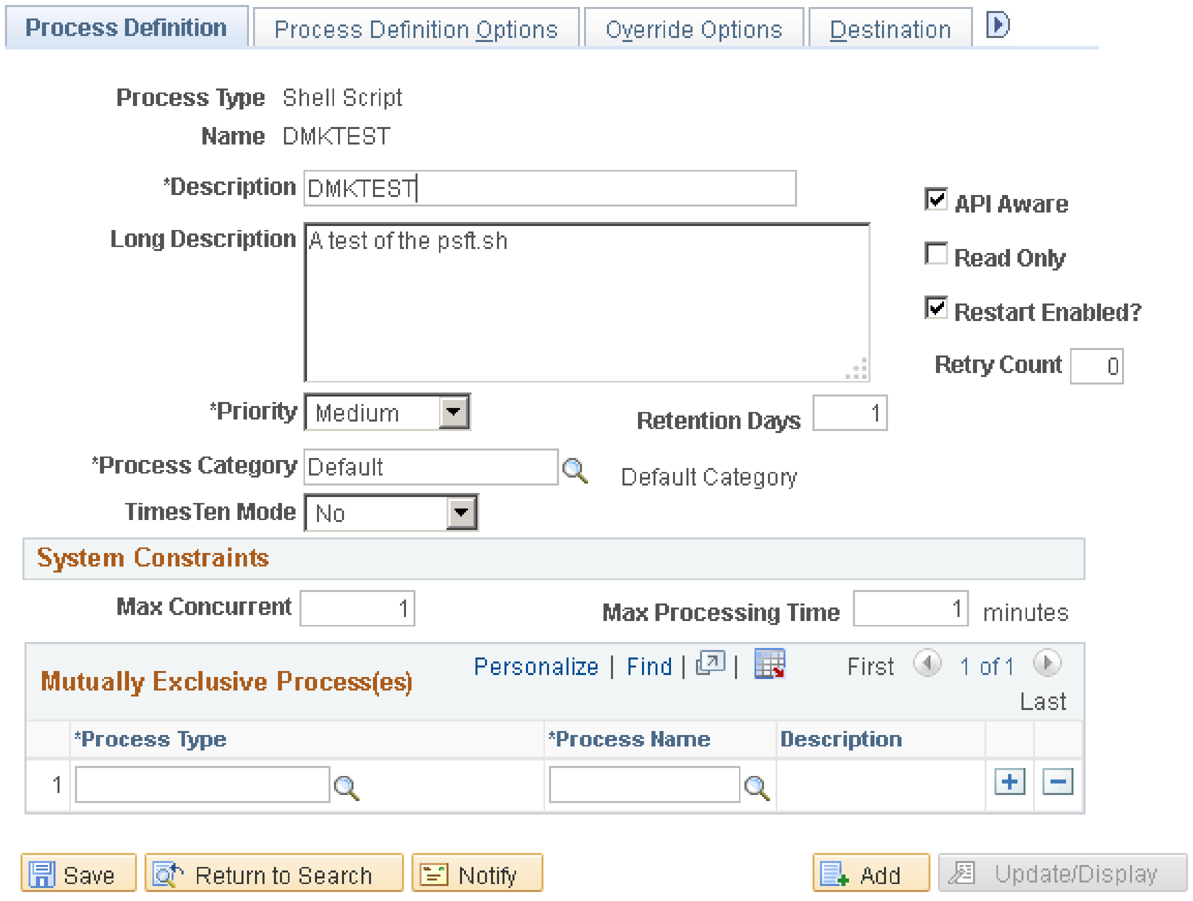
[7] The return code of the executed script is captured. Later it will be recorded on PSPRCSRQST.PRCSRTNCD. A non-zero return code indicates an error and the process request status will be set to error.

Process Definition

Now I can create a process definition that will use the new process type to call the wrapper script to execute another command or script.

The first four parameters passed to the wrapper script are the name of the database, the access ID and password, and the process instance. A string of further parameters will be appended in the individual Process Definition that is the specific command and parameters to be executed.

It is important that this new process type is defined as being API aware. That means the process interacts with the Process Scheduler by updating the process status. You can see how the interaction should be done by looking at procedure *Update-Process-Status* in the delivered SQR library *prcsapi.sqc*. Otherwise, the Process Scheduler cannot determine their status. Consequently, all API-unaware processes have a run status of Success to indicate that they were started successfully.



Process Type Shell Script
Name DMKTEST
*Description DMKTEST
Long Description A test of the psft.sh
☑ API Aware
☐ Read Only
☑ Restart Enabled?
Retry Count 0
*Priority Medium
Retention Days 1
*Process Category Default Default Category
TimesTen Mode No

System Constraints

Max Concurrent 1
Max Processing Time 1 minutes

Mutually Exclusive Process(es)

	*Process Type	*Process Name	Description
1			

I have written a silly test script called *mybanner.sh* that I want to be executed by the process scheduler. It just prints out the command line parameters as banner text to both standard output and a file called *mybanner.log*. This script will be called by psft.sh.

The Process Scheduler creates a working directory for each process request. It sets the variable *$PSPRCSLOGDIR* to the fully qualified location of this directory. Note that *mybanner.sh* changes the current directory to the location of this variable so that it writes *mybanner.log* there, and thus it is picked up by the distribution agent and made available via the report repository. You may wish to do this in your scripts.

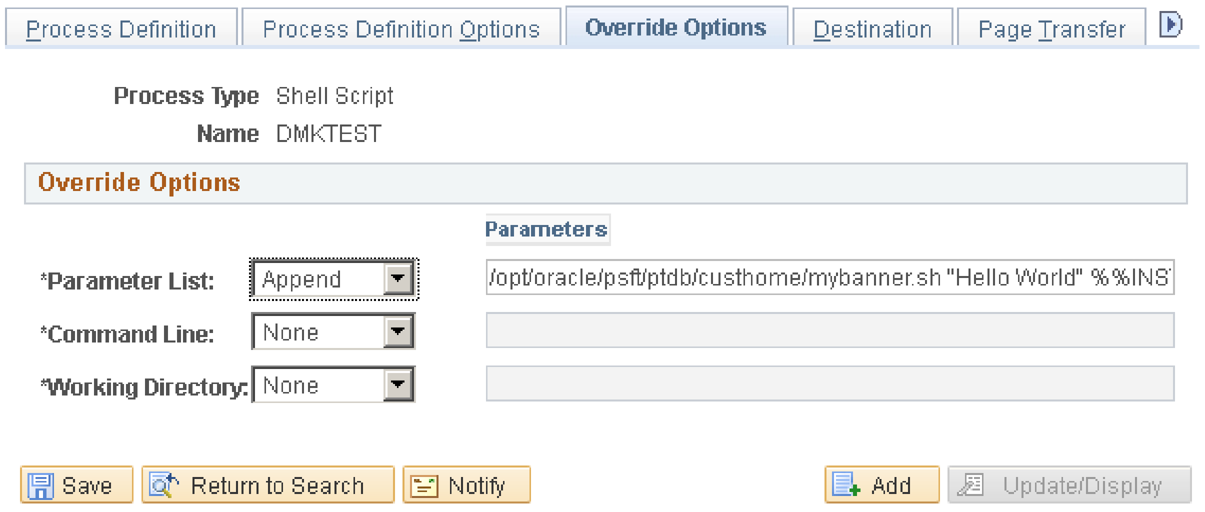
Current working directory can be specified at Process Type or Process definition. However, during my testing, I found that these settings had no effect. The working directory of the script did not change, and the value was not found in any environmental variable.

```
#!/bin/ksh
#A silly script to test psft.sh
#(c) David Kurtz 2017
#banner function from http://stackoverflow.com/questions/652517/whats-the-deal-with-the-banner-command

if [ "$PSPRCSLOGDIR" ] ; then
  cd $PSPRCSLOGDIR
fi

(
while [ $# -gt 0 ]
 do
  /opt/oracle/psft/ptdb/custhome/banner $1
  shift
done
) | tee mybanner.log
exit $?
```

I can now create a Process Definition that uses the Shell Script process type that will execute *mybanner.sh*. Note that this command line is appended to the existing command line specified in the Process Type definition.



Process Type Shell Script
Name DMKTEST

Override Options

Parameters

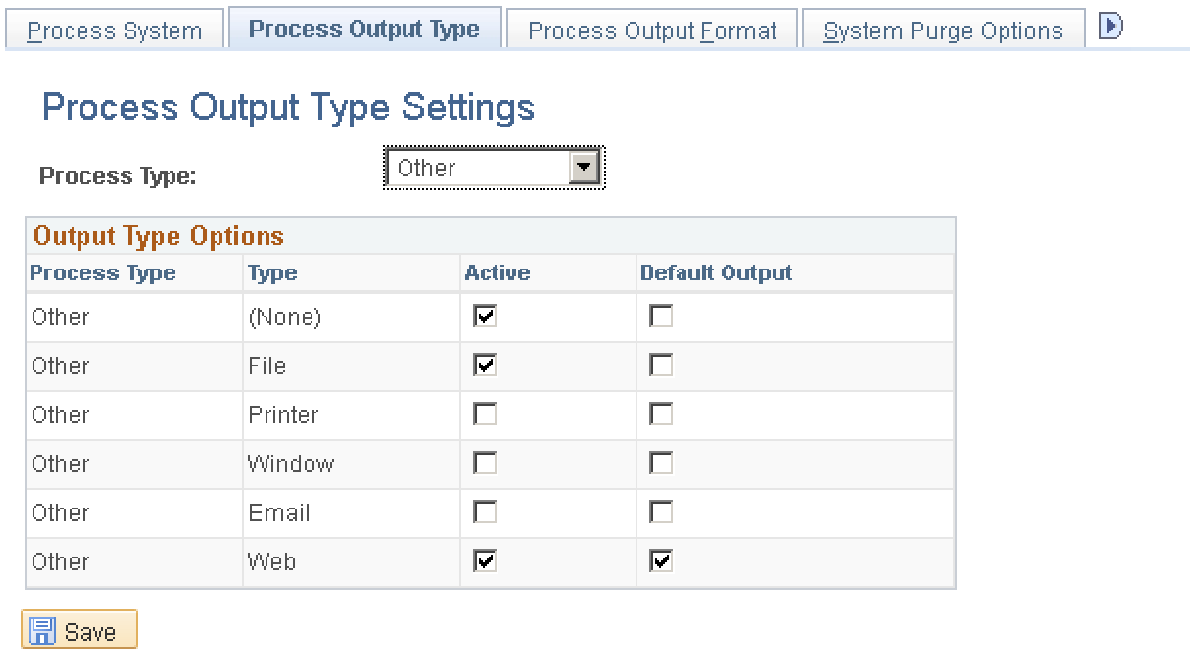
*Parameter List: Append /opt/oracle/psft/ptdb/custhome/mybanner.sh "Hello World" %%INS
*Command Line: None
*Working Directory: None

You can't quite see it in the screen shot, but the parameter list includes the process instance number:

```
/opt/oracle/psft/ptdb/custhome/mybanner.sh "Hello World" %%INSTANCE%%
```

Process Scheduler System Settings

During my testing, I found that it was necessary to specify output type settings for process type other in the Process Scheduler System Settings; otherwise the output files were not posted to the report repository.

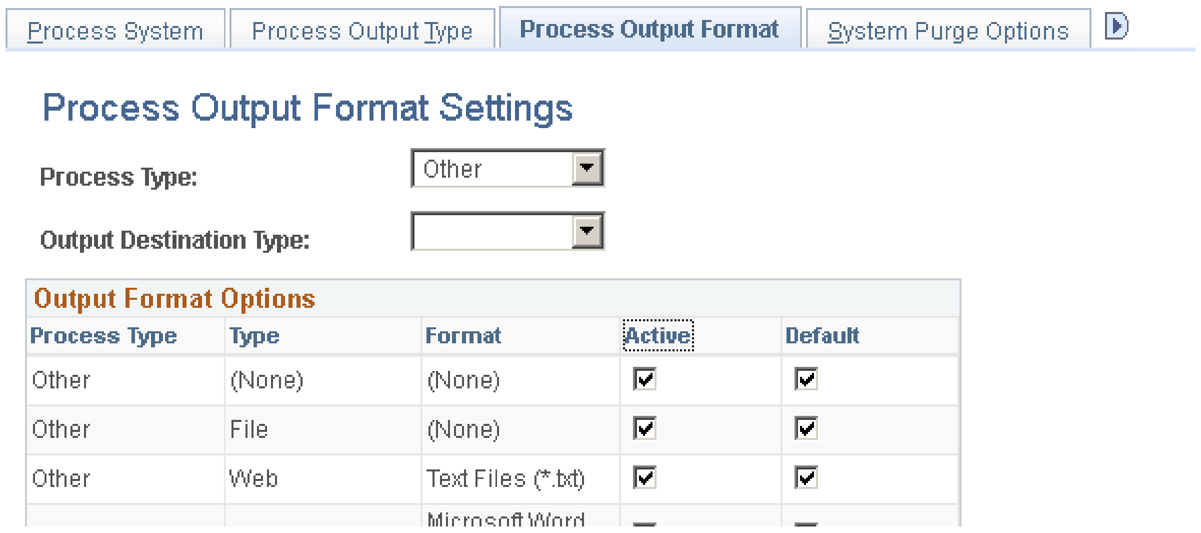


Process Output Type Settings

Process Type: Other

Output Type Options

Process Type	Type	Active	Default Output
Other	(None)	✓	
Other	File	✓	
Other	Printer		
Other	Window		
Other	Email		
Other	Web	✓	✓



Process Output Format Settings

Process Type: Other

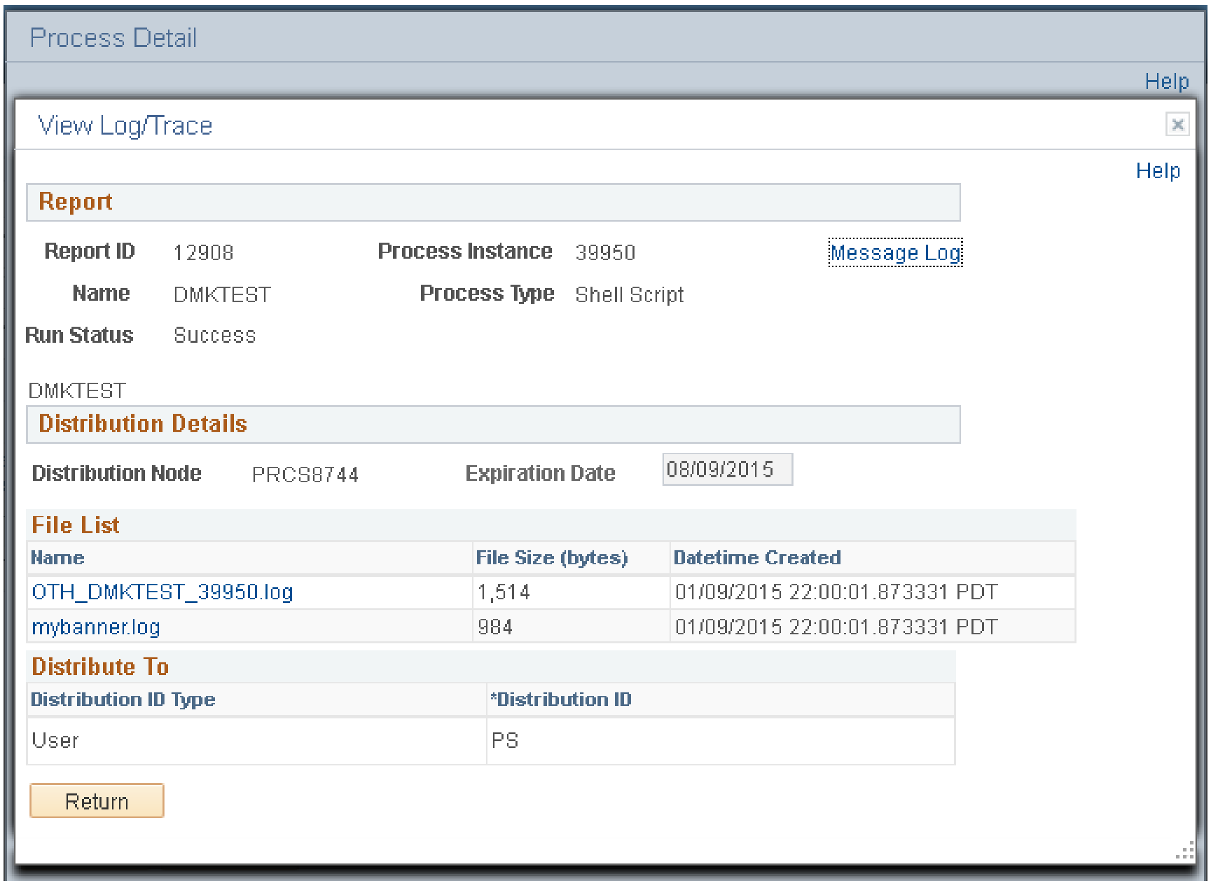
Output Destination Type:

Output Format Options

Process Type	Type	Format	Active	Default
Other	(None)	(None)	✓	✓
Other	File	(None)	✓	✓
Other	Web	Text Files (*.txt)	✓	✓
		Microsoft Word		

The newly defined Process can be run just as any other process is usually run. Any output from the script on the standard output channel is captured by the Process Scheduler and written to a log file that can then be viewed from the View Log/Trace facility within Process Monitor.

In this case the standard output was written to *OTH_DMKTEST_<process_instance>.log*, and I also get the *mybanner.log* that was written to *$PSPRCSLOGDIR* in the list of available files.



Report

Report ID	12908	Process Instance	39950	Message Log
Name	DMKTEST	Process Type	Shell Script	
Run Status	Success			

DMKTEST

Distribution Details

Distribution Node	PRCS8744	Expiration Date	08/09/2015

File List

Name	File Size (bytes)	Datetime Created
OTH_DMKTEST_39950.log	1,514	01/09/2015 22:00:01.873331 PDT
mybanner.log	984	01/09/2015 22:00:01.873331 PDT

Distribute To

Distribution ID Type	*Distribution ID
User	PS

mybanner.log just contains the three words passed as parameters

```
H     H         ll      ll
H     H          l       l
H     H  eeee    l       l       oooo
HHHHHHH e    e   l       l      o    o
H     H eeeeee   l       l      o    o
H     H e        l       l      o    o
H     H  eeee   lll     lll      oooo

W     W                 ll           d
W     W                  l           d
W     W  oooo   rr rr    l           d
W     W o    o   rr   r   l       ddddd
W  W  W o    o   r       l      d    d
W W W W o    o   r       l      d    d
 W   W   oooo   rr      lll      dddd d

 33333   99999   99999  5555555  00000
3     3 9     9 9     9 5       0    00
      3 9     9 9     9 5       0   0 0
  3333  999999  999999  55555   0  0  0
      3       9       9      5 0 0    0
3     3       9       9 5    5 00     0
 33333   99999   99999   55555   00000
```

OTH_DMKTEST_39950.log contains the standard output of the entire command - including the additional messages emitted by psft.sh (in bold).

Note that the current directory is reported as being the location of the process scheduler Tuxedo domain.

```
/opt/oracle/psft/ptdb/custhome/psft.sh:/opt/oracle/psft/ptdb/custhome/mybanner.sh Hello World 39950
Tue Sep  1 21:59:46 UTC 2015
Linux hcm.london.go-faster.co.uk 2.6.39-400.215.10.el5uek #1 SMP Tue Sep 9 22:51:46 PDT 2014 x86_64 x86_64
x86_64 GNU/Linux
Current Directory: /home/psadm2/psft/pt/8.54/appserv/prcs/PRCSDOM
Process log files in: /home/psadm2/psft/pt/8.54/appserv/prcs/PRCSDOM/log_output/OTH_DMKTEST_39950
Setting process request 39950 to status 7
H     H         ll      ll
H     H          l       l
H     H  eeee    l       l        oooo
HHHHHHH e    e   l       l       o    o
H     H eeeeee   l       l       o    o
H     H e        l       l       o    o
H     H  eeee   lll     lll       oooo

W     W                 ll            d
W     W                  l            d
W     W  oooo   rr rr    l            d
W     W o    o   rr  r   l        ddddd
W  W  W o    o   r       l       d    d
W W W W o    o   r       l       d    d
 W   W   oooo   rr      lll       dddd d

 33333   99999   99999  5555555  00000
3     3 9     9 9     9 5       0    00
      3 9     9 9     9 5       0   0 0
  3333   999999  999999  55555  0  0  0
      3       9       9      5 0 0   0
3     3       9       9 5    5 00    0
 33333   99999   99999   55555   00000

Setting process request 39950 to status 9
Tue Sep  1 21:59:46 UTC 2015
```