

IMQ09 - IBM MQ V9 for z/OS Wildfire Workshop



L05 – Using CAPEXPY on z/OS

Version V6.0

October 2018



Table of Contents

Table of Contents	1
History and Tradition	3
Lab Objectives	3
General Lab Information and Guidelines	3
Step 1 – Defining and priming the queues	4
Step 2 – Enabling CAPEXPY on the queue manager	7
Step 3 – Setting the CAPEXPY on the queue	8
Step 4 – Test the new value - Putting new messages on the queue	11
Step 5 – Stop and Start the queue manager	12

History and Tradition

Setting message expiration has always been the prerogative of the application program and programmer. This has been decried by generations of systems administrators, because of an application tendency to not set message expiration (expiry = unlimited) or to set it to an unreasonably high value. This can mean that messages destined for a specific application can remain on a queue forever, taking up valuable space and potentially causing elongated recovery times if the messages are persistent.

Please note that this is a departure from the normal MQ way of doing things – where the application has the final word on the message disposition.

Lab Objectives

This lab has the following objectives:

- To allow users to implement and test the CAPEXPY feature as it is implemented in z/OS.

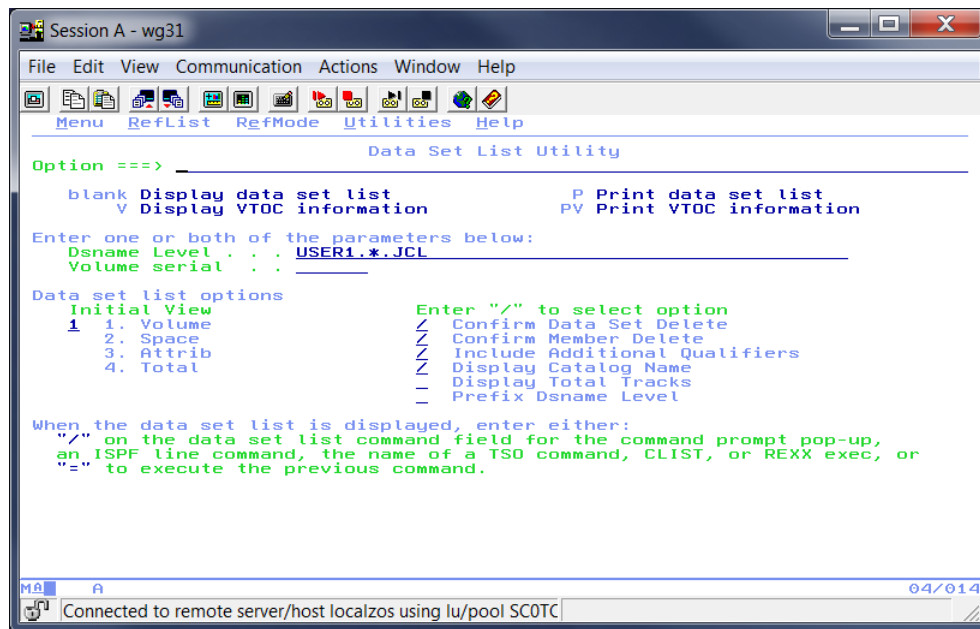
General Lab Information and Guidelines

- Information required to complete this exercise will be provided on a ‘worksheet’ prior to the start of this exercise. Refer to this worksheet for which user identity and password are to be used and for other values, for example:
 - ✓ This exercise requires using TSO user *USER1* on the wg31.washington.ibm.com system
 - ✓ As a reminder, when a value from your worksheet should be used, the values in the instructions will be in **red** rather than black.
 - ✓ ***Bold italicized*** text indicates values that need to be entered on a screen.
 - ✓ *Italicized* text indicates values that are constants or names that appear on a screen.
 - ✓ **Bold** text indicates the name of buttons or keyboard keys that need to be pressed.

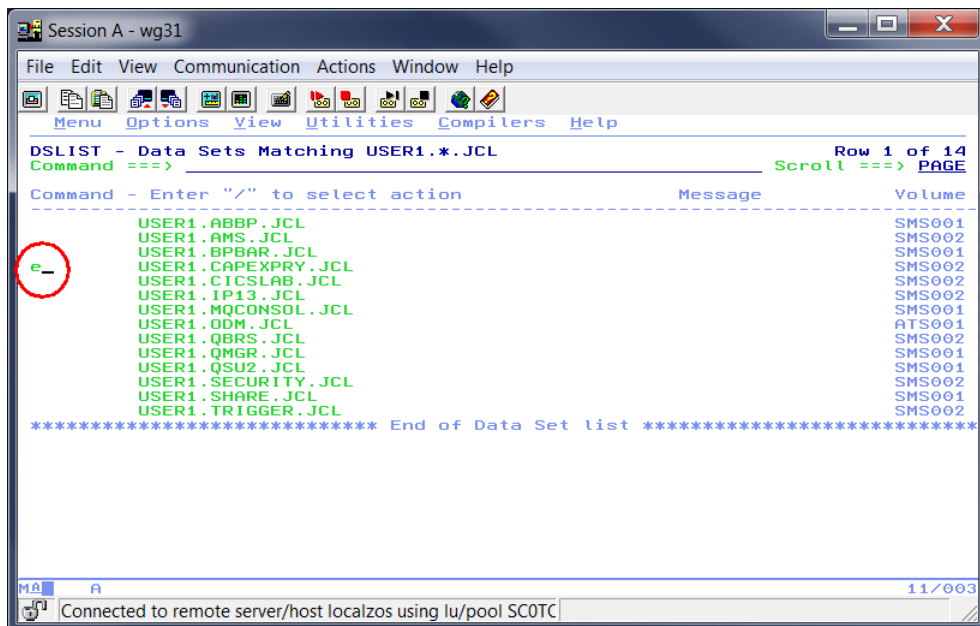
Step 1 – Defining and priming the queues

Use the MQ utility CSQUTIL job to define your queues for this lab.

- ____1. Logon to TSO and use the ISPF command =3.4 to navigate to the dataset selection panel.
- ____2. Enter *USER1.*.JCL* in the *Dsname Level* field as shown below.



- ____3. From the list of JCL PDS libraries, edit the *USER1.CAPEXPYR.JCL* data set as shown below.



4. Select the member *DEFQUES*. This job creates three queues that will be used in this test. The first two have the same characteristics, with one difference – the default persistence. The third queue is a target for report messages.

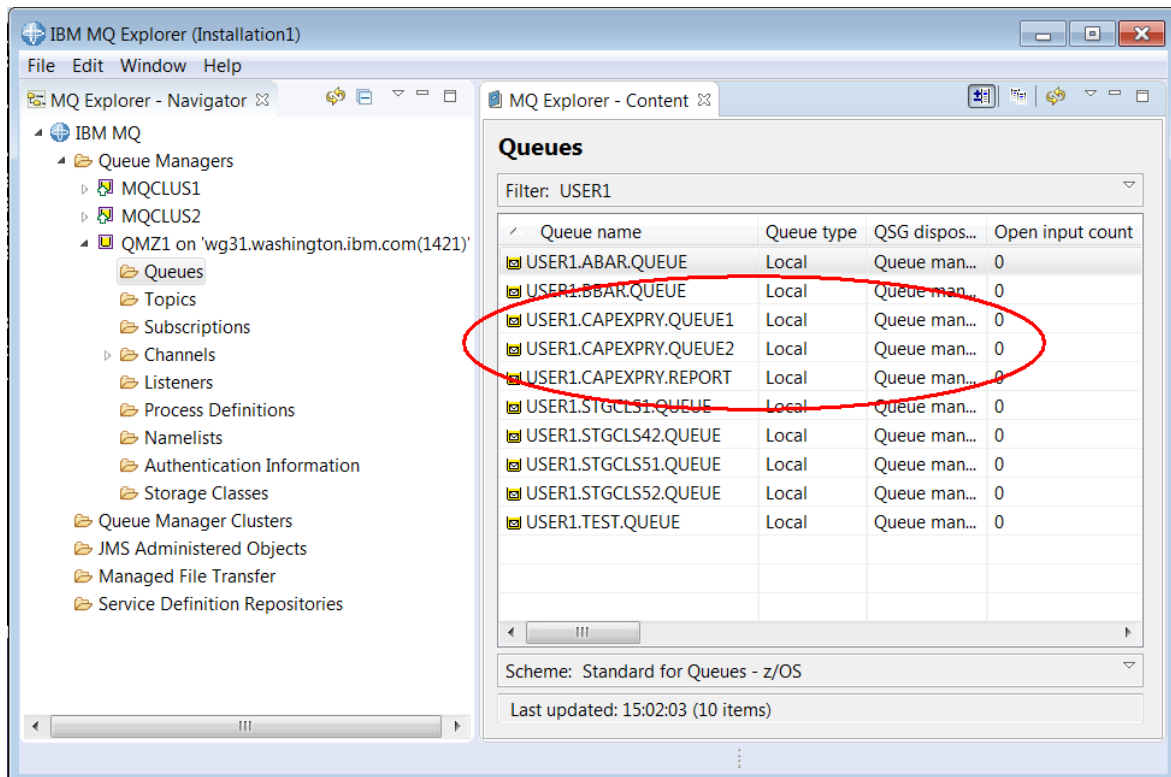
```

Session A - wg31
File Edit View Communication Actions Window Help
File Edit Edit_Settings Menu Utilities Compilers Test Help

EDIT      USER1.CAPEXPYR.JCL(DEFQUES) - 01.01      Columns 00001 00072
Command ==>                                     Scroll ==> PAGE
***** ***** Top of Data *****
000001 //USER1Q JOB      NOTIFY=&SYSUID,MSGCLASS=H
000002 //CSQUTIL EXEC  PGM=CSQUTIL,REGION=4096K,PARM='QMZ1'
000003 //STEPLIB DD DSN=MQ900.SCSQANLE,DISP=SHR
000004 //          DD DSN=MQ900.SCSQAUTH,DISP=SHR
000005 //SYSPRINT DD SYSOUT=*
000006 //SYSIN   DD *
000007 COMMAND DD *
000008 //CSQUCMD DD *
000009 DEFINE QLOCAL(USER1.CAPEXPYR.QUEUE1) +
000010 DESCR('CAPEXPYR TEST QUEUE 1') +
000011 QSGDISP(QMGR) +
000012 PUT(ENABLED) +
000013 GET(ENABLED) +
000014 DEFPERSIST(YES) +
000015 MSGDLVSQ(FIFO) +
000016 NOTRIGGER +
000017 USAGE(NORMAL) +
000018 REPLACE +
000019 SHARE
000020
000021 DEFINE QLOCAL(USER1.CAPEXPYR.QUEUE2) +
000022 DESCR('CAPEXPYR TEST QUEUE 2') +
000023 QSGDISP(QMGR) +
000024 PUT(ENABLED) +
000025 GET(ENABLED) +
000026 DEFPERSIST(NO) +
000027 MSGDLVSQ(FIFO) +

```

5. Submit the JCL to create the queues and make sure the job gets a zero return code.
6. Verify the queues were created by using the MQExplorer:



- ___7. Return to the member list of the *USER1.CAPEXPYR.JCL* and select member *PUTMSGSGS*.

```

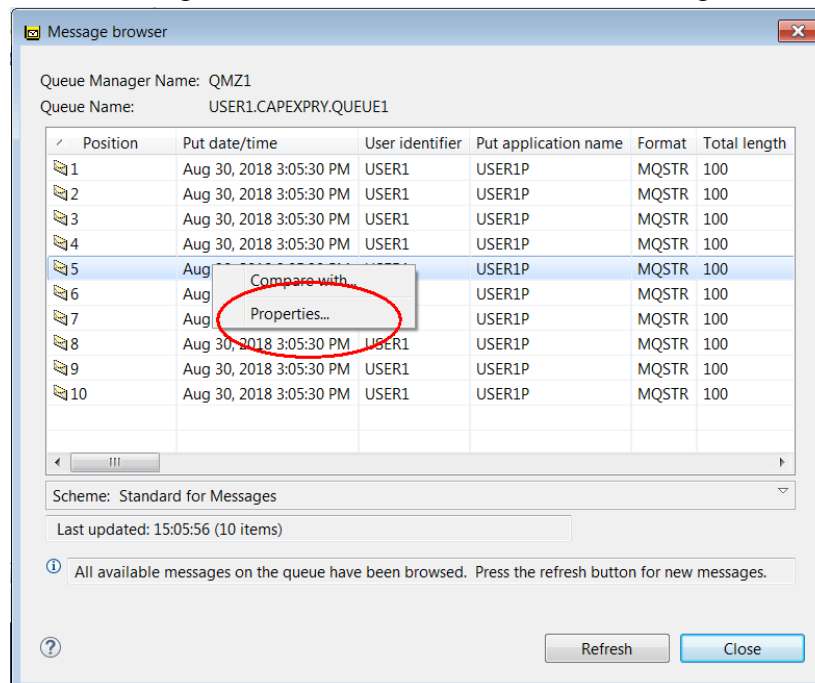
Session A - wg31
File Edit View Communication Actions Window Help
File Edit Edit_Settings Menu Utilities Compilers Test Help

EDIT      USER1.CAPEXPYR.JCL(PUTMSGSGS) - 01.01      Columns 00001 00072
Command ==>      Scroll ==> PAGE
***** ***** Top of Data *****
000001 //USER1P JOB NOTIFY=&SYSUID,MSGCLASS=H
000002 // SET M=QMZ1
000003 // SET Q=USER1.CAPEXPYR.QUEUE1
000004 //S1 EXEC PGM=DEMPUT,REGION=0M,
000005 // PARM=(' -m&M. -n10 -q&Q -s100 -crlf -p -fileDD:MSGIN ')
000006 //SYSIN DD *
000007 *
000008 //STEPLIB DD DISP=SHR,DSN=SYSP.MP1B.V81.LOAD
000009 // DD DISP=SHR,DSN=MQ900.SCSQLOAD
000010 // DD DISP=SHR,DSN=MQ900.SCSQAUTH
000011 // DD DISP=SHR,DSN=MQ900.SCSQANLE
000012 //SYSPRINT DD SYSOUT=*
000013 //MSGIN DD DISP=SHR,DSN=USER1.CAPEXPYR.JCL(TSTMSG)
000014 ***** Bottom of Data *****

MA A
Connected to remote server/host localzos using lu/pool SCOTC 27/035

```

- ___8. This job will put 10 messages of 100 bytes on the queue specified by the *Q* parameter. In this case it is *USER1.CAPEXPYR.QUEUE1*. Submit the job and make sure it runs with a zero return code.
- ___9. Verify that messages were put to the queue, by using the MQ Explorer by browsing the messages on queue *USER1.CAPEXPYR.QUEUE1*. (Hint: select the queue and right mouse button click and select the *Browse Messages* option).
- ___10. Select one of the messages (it does not matter which one) and right mouse button click.



- ____11. Select *Properties* (see above) to view the message expiration value, which should be *Unlimited*. These 10 messages will never expire.

The screenshot shows a window titled "Message 3 - Properties". On the left is a sidebar with a tree view containing "General", "Report", "Context", "Identifiers", "Segmentation", and "Data". The "General" tab is selected. The main area displays various message properties:

- Position: 3
- Message type: Datagram
- Priority: 0
- Persistence: Persistent
- Put date/time: Apr 28, 2016 11:43:56 AM
- Expiry: Unlimited (circled in red)
- Reply-to queue:
- Reply-to queue manager: QMZ1
- Backout count: 0

At the bottom right is a "Close" button.

Step 2 – Enabling CAPEXPY on the queue manager

Enabling CAPEXPY for a queue manager requires using the MQ *RECOVER* command.

- ____1. Use the ISPF command **=D.LOG** to go to the SDSF log panel and enter the command below:

QMZ1 RECOVER QMGR (TUNE CAPEXPY ON)

Tech-Tip: *QMZ1* is the *command prefix* string of this queue manager. A command prefix (cpf) string is used to direct commands to a queue manager. On this system the queue manager's *cpf* string is the same as the queue manager's name.

The leading slash is required when entering MVS commands at the SDSF prompt.

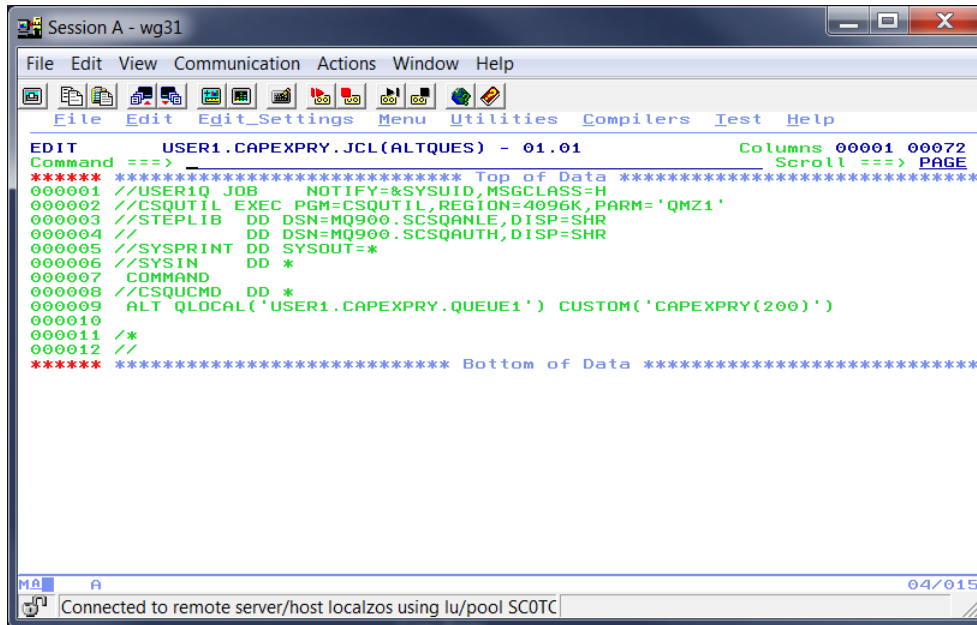
- ____2. You should see the messages below in the system log.

CSQI834I QMZ1 PARM CAPEXPY = ON (Default OFF SET)
CSQ9022I QMZ1 CSQIRECP 'RECOVER QMGR' NORMAL COMPLETION

Step 3 – Setting the CAPEXPY on the queue

Currently enabling CAPEXPY for a queue requires the use of the queue's *CUSTOM* attribute.

1. Return to the *USER1.CAPEXPY.JCL* member list and select member *ALTQUES*. The *ALTER* command in this job updates the expiration for each message put to the queue to 20 seconds, the expiration value is in tenths of seconds. Submit the job and make sure there is a zero return code.



```

Session A - wg31
File Edit View Communication Actions Window Help
File Edit Edit_Settings Menu Utilities Compilers Test Help

EDIT      USER1.CAPEXPY.JCL(ALTQUES) - 01.01      Columns 00001 00072
Command ==> Scroll ==> PAGE

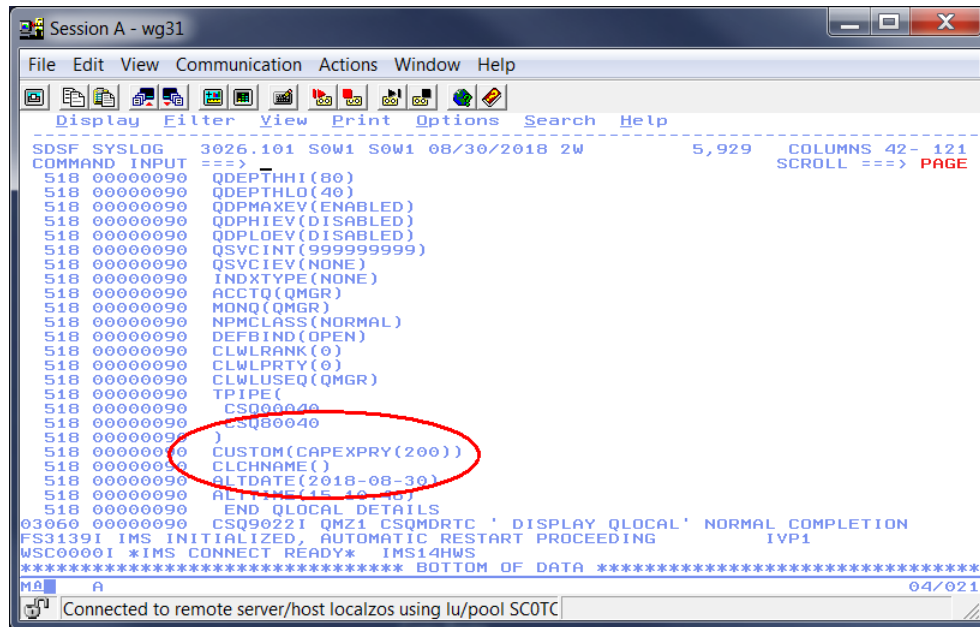
***** Top of Data *****
000001 //USER1Q JOB      NOTIFY=&SYSUID,MSGCLASS=H
000002 //CSQUTIL EXEC   PGM=CSQUTIL,REGION=4096K,PARM='QMZ1'
000003 //STEPLIB DD    DSN=MQ900.SCSQANLE,DISP=SHR
000004 //          DD    DSN=MQ900.SCSQAUTH,DISP=SHR
000005 //SYSPRINT DD    SYSOUT=*
000006 //SYSIN DD *
000007 COMMAND
000008 //CSQUCMD DD *
000009 ALT QLOCAL('USER1.CAPEXPY.QUEUE1') CUSTOM('CAPEXPY(200)')
000010 /*
000011 /*
000012 /*
***** Bottom of Data *****

04/015
Connected to remote server/host localzos using lu/pool SC0TC

```

3. Use the ISPF command *=D.LOG* to go to the SDSF log panel and enter the MQ command below:

QMZ1 DISPLAY QL (USER1.CAPEXPY.QUEUE1) ALL



The value should also be shown on the *Extended* tab of the queue properties from the MQ Explorer.

The screenshot shows the 'Properties' dialog for the queue 'TEAMZZ.CAPEXPYR.QUEUE1'. The 'Extended' tab is selected in the left-hand pane. The right-hand pane displays various queue properties with their current values:

Property	Value
Max queue depth:	999999999
Maximum message length (bytes):	4194304
Shareability:	Shareable
Default input open option:	Input exclusive
Message delivery sequence:	FIFO
Retention interval (hours):	999999999
Definition type:	Predefined
Pipe name:	CSQ00031
Index type:	None
Page set ID:	4
Default read ahead:	No
Default put response type:	Synchronous
Property control:	Compatibility
Custom:	CAPEXPYR(200)
Cluster channel names:	

At the bottom of the dialog are buttons for 'Apply', 'Cancel', and 'OK'. A help icon (?) is located in the bottom-left corner.

Step 4 – Test the new value - Putting new messages on the queue

- ____1. Return to the JCL library and select member *PUTMSG*S and submit the job for execution.
- ____2. When the job completes, browse the new messages on the queue. Note that you may need to refresh the browse list to display the new messages.

Select one of the new messages and display the properties. They should look something like what is shown, but the *Expiry* value may be different depending on how quickly you get to the messages.

The screenshot shows a window titled "Message 19 - Properties" with a close button (X) in the top right corner. On the left is a vertical sidebar with a list of tabs: "General" (selected), "Report", "Context", "Identifiers", "Segmentation", and "Data". The main area of the window is titled "General" and contains several labeled text input fields:

Property	Value
Position:	19
Message type:	Datagram
Priority:	0
Persistence:	Persistent
Put date/time:	Apr 28, 2016 2:42:23 PM
Expiry:	182
Reply-to queue:	
Reply-to queue manager:	QMZ1
Backout count:	0

At the bottom left of the window is a help icon (question mark in a circle), and at the bottom right is a "Close" button.

- ____3. Wait a minute or so, to allow the message to expire. Re-browse the queue; the new messages should have disappeared while the original 10 messages should remain.

Step 5 – Stop and Start the queue manager

- ____1. From the SDSF DA (=D.DA) panel enter the stop queue manager command.

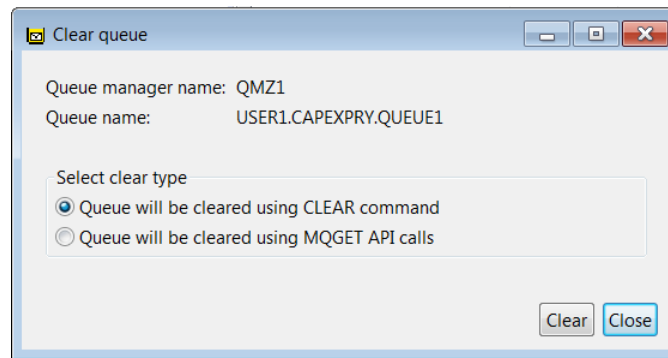
QMZ1 STOP QMGR

- ____2. Once *QMZ1MSTR* address space has stopped, restart the queue manager by entering that start command.

QMZ1 START QMGR

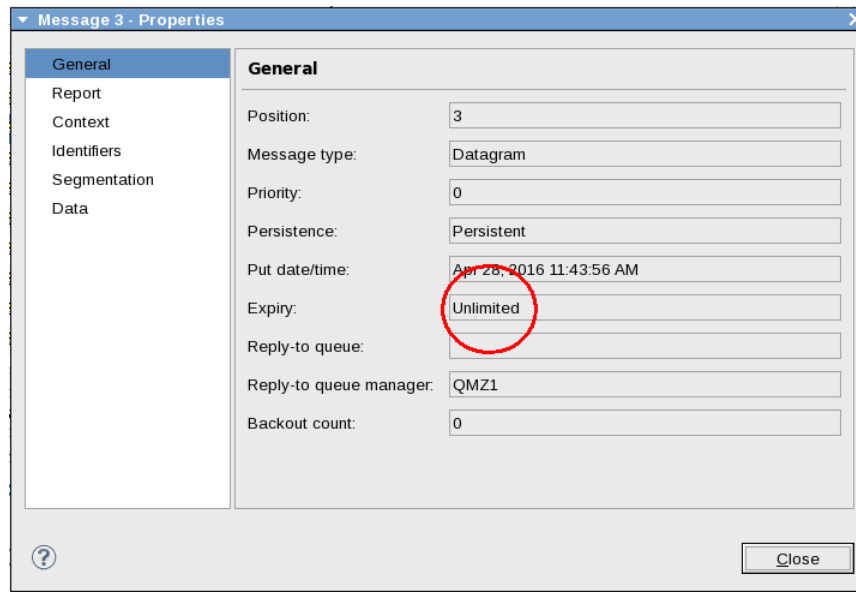
- ____3. Wait until the two address spaces (*QMZ1MSTR* and *QMZ1CHIN*) are started before continuing.

- ____4. Any messages on the *USER1.CAPEXPYR.QUEUE1* should be cleared using MQ Explorer.



- ____5. From the *USER1.CAPEXPYR.JCL* data set, submit the *PUTMSGs* job again.

___6. Browse the new messages and check the properties of one of the messages.



___7. The reason the expiry is back to *Unlimited* is that the command used to enable the option on the queue manager is temporary. It is a 'tuning' option that is only valid for the duration until the queue manager is stopped. In the next step, the queue manager startup will be modified to make this permanent.

Step 6 – Set up the queue manager for CAPEXPY all the time

- ____1. From the *Dataset List Utility* panel (=3.4), enter *USER.PROCLIB* in the *Dsname Level* field and press **ENTER**.
- ____2. Edit member *QMZIMSTR* in data set *USER.PROCLIB*.
- ____3. At the end of the *CSQINP2* concatenation, insert a line to include the command to turn on CAPEXPY for the queue manager from the *USER1.CAPEXPY.JCL* library as shown. Save the *QMZIMSTR* member with these changes.

```

EDIT      USER.Z22D.PROCLIB(QMZIMSTR) - 01.01      Columns 00001 00072
Command ==>                                         Scroll ==> PAGE
000017 //*****
000018 //CSQINP1 DD DSN=MQ900.SCSQPROC(CSQ4INP1),DISP=SHR
000019 // DD DSN=WMQ900.QMZ1.SCSQPROC(WMQ9INP1),DISP=SHR
000020 //CSQINP2 DD DSN=MQ900.SCSQPROC(CSQ4INYS),DISP=SHR
000021 // DD DSN=MQ900.SCSQPROC(CSQ4INSX),DISP=SHR
000022 // DD DSN=MQ900.SCSQPROC(CSQ4INSG),DISP=SHR
000023 // DD DSN=WMQ900.QMZ1.SCSQPROC(QMZ#STGC),DISP=SHR
000024 //* DD DSN=MQ900.SCSQPROC(CSQ4INSS),DISP=SHR
000025 // DD DSN=MQ900.SCSQPROC(CSQ4INSJ),DISP=SHR
000026 // DD DSN=MQ900.SCSQPROC(CSQ4INSR),DISP=SHR
000027 // DD DSN=MQ900.SCSQPROC(CSQ4INSA),DISP=SHR
000028 // DD DSN=MQ900.SCSQPROC(CSQ4INSM),DISP=SHR
000029 // DD DSN=WMQ900.QMZ1.SCSQPROC(CSQ4INYG),DISP=SHR
000030 // DD DSN=MQ900.SCSQPROC(CSQ4INYC),DISP=SHR
000031 //* DD DSN=MQ900.SCSQPROC(CSQ4INYD),DISP=SHR
000032 // DD DSN=MQ900.SCSQPROC(CSQ4INIS),DISP=SHR
000033 // DD DSN=WMQ900.QMZ1.SCSQPROC(CSQ4CHIN),DISP=SHR
000034 // DD DSN=WMQ900.QMZ1.SCSQPROC(QMZ1DEFS),DISP=SHR
000035 // DD DSN=USER1.CAPEXPY.JCL(SETQMGR),DISP=SHR
000036 //CSQINPT DD DSN=MQ900.SCSQPROC(CSQ4INPT),DISP=SHR
000037 // DD DSN=MQ900.SCSQPROC(CSQ4INPT),DISP=SHR
000038 //CSQOUT1 DD SYSOUT=*
000039 //CSQOUT2 DD SYSOUT=*
000040 //CSQOUTT DD SYSOUT=*
000041 //CSQP0000 DD DSN=QMZ1.QMGR.PSID00,DISP=SHR
000042 //CSQP0001 DD DSN=QMZ1.QMGR.PSID01,DISP=SHR
000043 //CSQP0002 DD DSN=QMZ1.QMGR.PSID02,DISP=SHR
000044 //CSQP0003 DD DSN=QMZ1.QMGR.PSID03,DISP=SHR
  
```

- ____4. Stop and restart the queue manager as described in the previous step.
- ____5. Using the *PUTMSG* job in the *USER1.CAPEXPY.JCL* PDS, put 10 more messages on the queue.

- ____6. Using the MQ Explorer, browse the queue – note that you shall probably have to reconnect as the queue manager has been cycled since your last connection. Pick one of the recent messages, and the expiration should be a value other than unlimited.

Message 17 - Properties

General

Report
Context
Identifiers
Segmentation
Data

General

Position: 17

Message type: Datagram

Priority: 0

Persistence: Persistent

Put date/time: Apr 28, 2016 3:11:32 PM

Expiry: 142

Reply-to queue:

Reply-to queue manager: QMZ1

Backout count: 0

?

Close

- ____7. Congratulations! You have completed the implementation and testing of CAPEXPY!