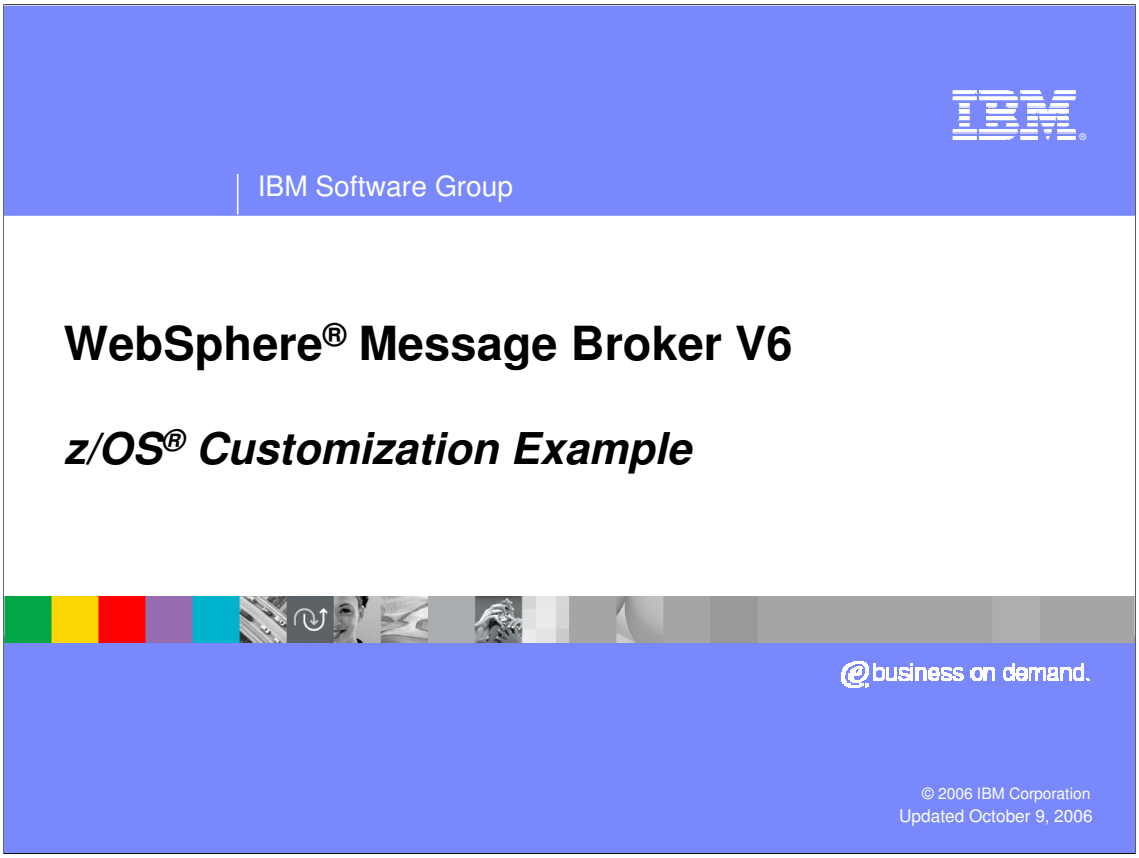


This presentation reviews the steps involved in customizing a broker.

Agenda

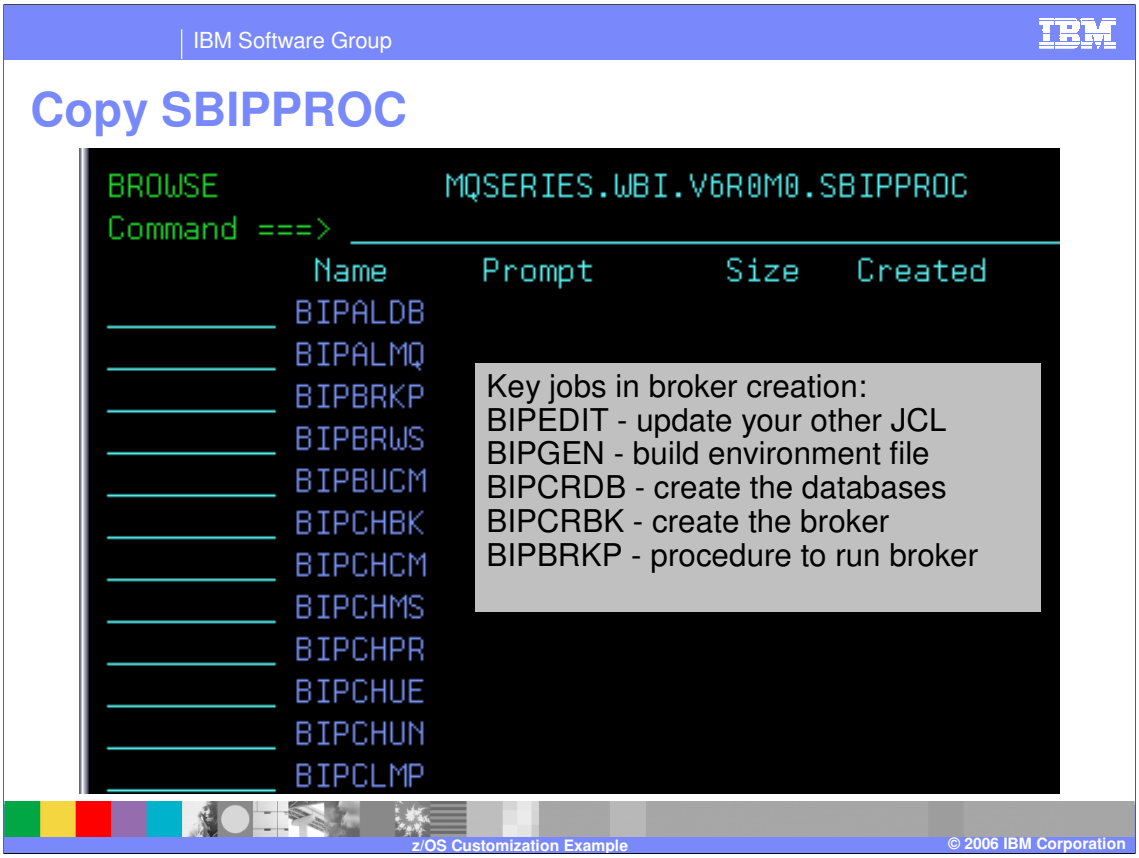
- Customization example
- Runtime output
- Summary and references

This presentation shows screen captures of the various jobs used to create a broker on z/OS and the relevant output.

Section

Customization example

This section explains the steps to take in customizing a broker. A similar process would be done for a configuration manager and a user name server.



Copy all broker JCL from <hlq>.SBIPPROC (the installed library) to your broker's component directory.

Only a few of these members are actually used in a broker creation. The specific jobs used in a broker creation are listed here.

BIPEDIT is the REXX exec that will update your other JCL with system specific configuration parameters.

BIPGEN is the job which builds the environment file in your component file system.

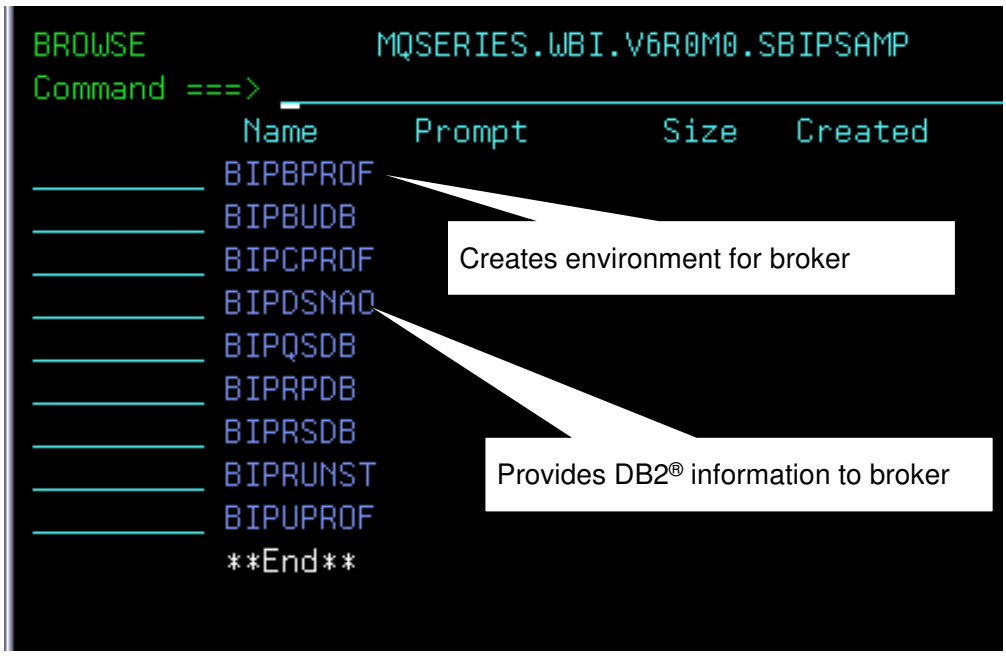
BIPCRDB creates the databases and tables used by the broker.

BIPCRBK creates the broker and its WebSphere MQ queues.

BIPBRKP is the started task procedure needed to run the broker. This must subsequently be copied to your system proclib.

However, it might be well to copy all of them so that the BIPEDIT REXX exec will run against all of them and update your system specific configuration parameters. Some of the other jobs provided are used to create a configuration manager and a user name server. Some are to delete and alter components of the broker and others simply provide JCL to run various mqsi commands.

Copy SBIPSAMP



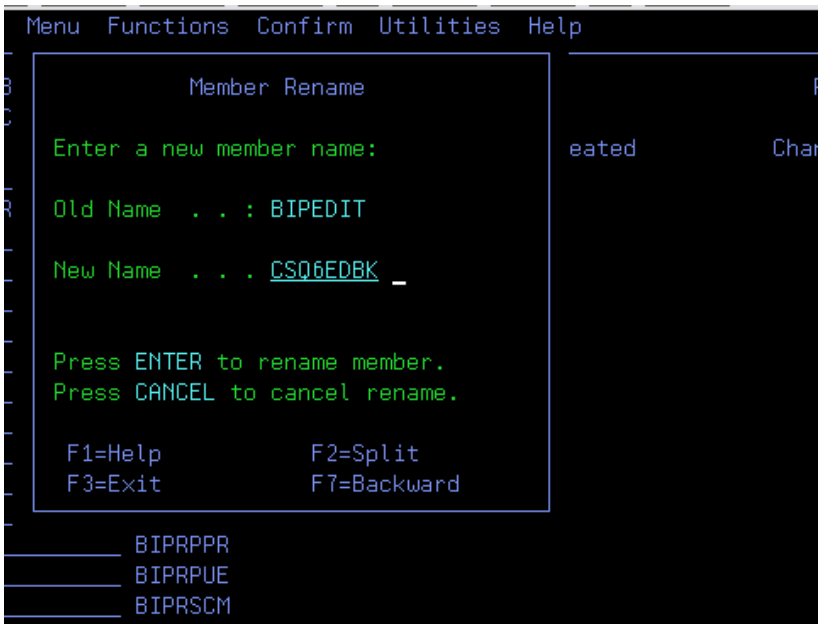
Copy all broker JCL from <hlq>.SBIPSAMP (the installed library) to your broker's component directory. The two that you need are:

BIPBPROF which is used by BIPGEN to create an environment file for your broker

and

BIPDSNAO which provides DB2 specific information to your broker (such as subsystem name)

Rename BIPEDIT



BIPEDIT is an REXX Exec that allows easier customization of the JCL in your component dataset. BIPEDIT should be renamed to avoid any confusion between different BIPEDIT files in different component datasets.

We recommend that you use the following method to rename BIPEDIT to CSQ6EDBK:

CSQ6 is the queue manager name

ED is for Edit

BK is for Broker.

(Change BK to CM for a Configuration Manager and to UN for a User Name Server)

Customize CSQ6EDBK (BIPEDIT)

Go through CSQ6EDBK and change all ++<JCL variables>++ to the value you have determined for your installation. The WebSphere Message Broker V6 Configuration and Customization document has instructions on how to determine your environment information needed.

Do not worry about ++OPTIONS++ for now. This can be changed later.

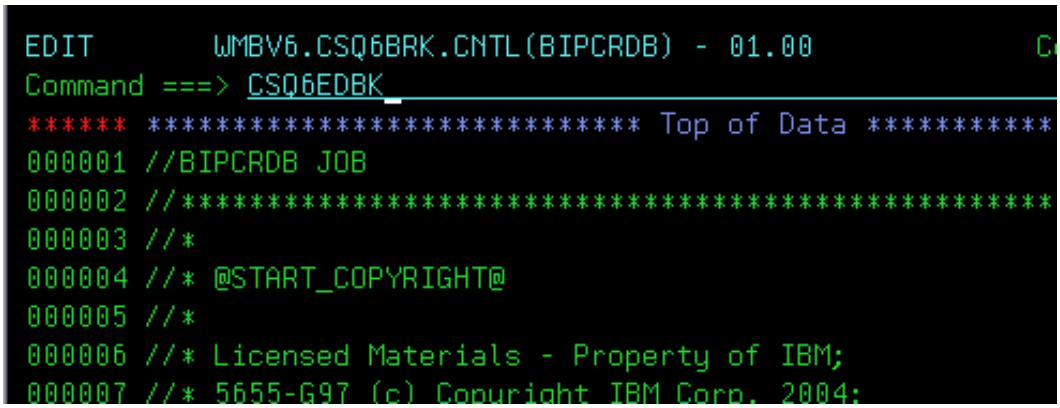
Activate the dataset that contains CSQ6EDBK

Activate the dataset that contains CSQ6EDBK. This allows you to run CSQ6EDBK REXX Exec to customize all the other component JCL. You may also enter the 'TSO ALTLIB ACTIVATE APPLICATION(EXEC)' from the command line in your ISPF session; you will be prompted for the dataset name.

This activation only applies to the current ISPF session. You will need to reactivate the dataset if you exit and restart the ISPF session.

Note: you can also manually edit each member as you use it.

Customize the component JCL



- Running CSQ6EDBK will automatically changed all ++<JCL Variables>++ in the file.

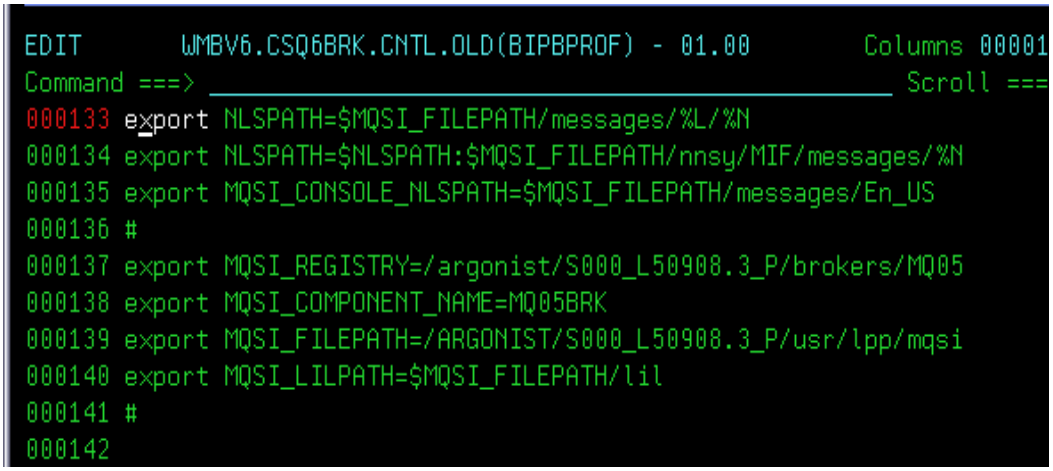
Edit each JCL file in your component dataset (excluding CSQ6EDBK!) by typing CSQ6EDBK on the command line. This will apply the environment specific changes necessary to that file.

You will also need to supply a job card that meets your system standards; the exceptions are BIPDSNAO and BIPBPROF (which are input to other jobs) and BIPBRKP which you will rename and use as you started task.

Note that some JCL may need further editing of either ++OPTIONS++ or mandatory parameters unique to that command.

Your modified BIPEDIT command (CSQ6EDBK in this case) can only be run ONCE on each JCL file since it replaces the parameters; a second run would find nothing to replace.

Broker profile (BIPBPROF)



BIPBPROF is the broker profile. You can add any other environmental changes needed for your environment here. This profile creates the environment for both the commands and the component runtime.

As illustrated here, you can concatenate environment variables in this profile, unlike an ENVFILE.

Broker dsnaoini (BIPDSNAO)

BIPDSNAO is the broker dsnaoini. You can add any other ODBC changes here (such as activating ODBC trace).

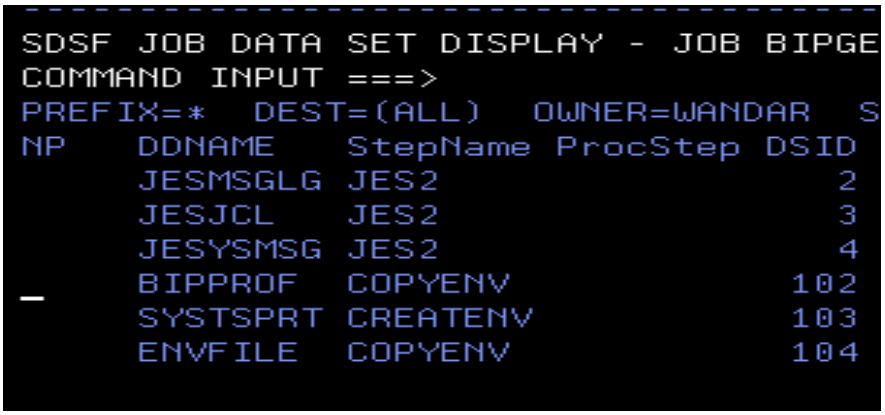
Submit BIPGEN to create ENVFILE

```
EDIT       WMBV6.CSQ6BRK.CNTL(BIPGEN) - 01.01              Columns
Command ===> sub_                                           Scrol
****** ***************************** Top of Data ******************
000001 //BIPGEN  JOB ,WANDAR,MSGLEVEL=(1,1),NOTIFY=&SYSUID,
000002 //           MSGCLASS=H,USER=WANDAR,TIME=1440,REGION=0M
000003 //*************************************************************
000004 //*************************************************************
000005 //*
```


BIPGEN takes the component profile (BIPBPROF for a broker) and converts it to an ENVFILE in your component HFS file system. It is this ENVFILE that actually creates the environment for commands run by JCL and the component runtime.

NOTE: Changes made to the component profile will NOT take effect until BIPGEN is run!

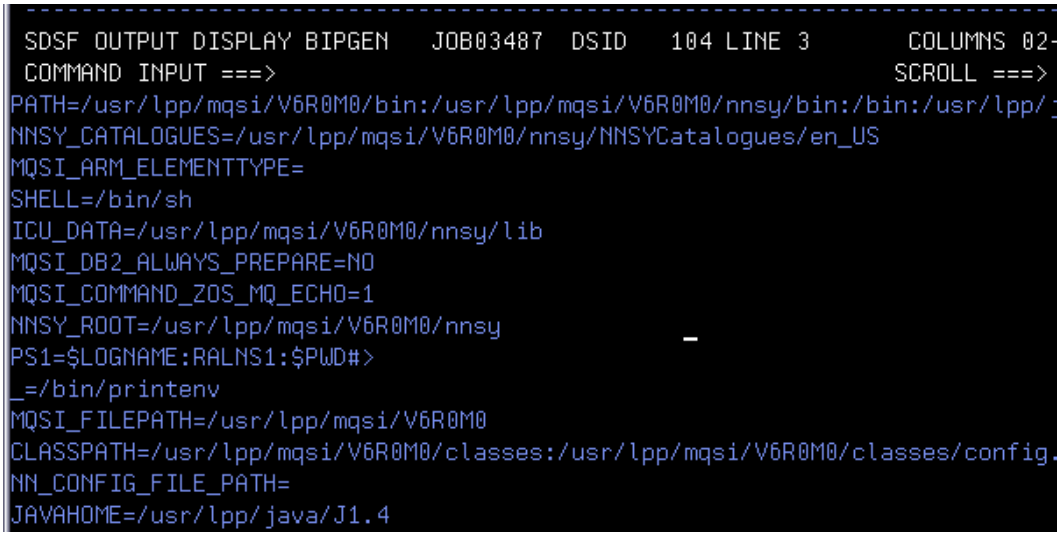
Check BIPGEN results



- It is important to check that all steps in BIPGEN ran successfully.
- If there are any errors then check the rest of the JOBLOG for specifics..

Ensure that all steps in the BIPGEN job ran successfully. If there are errors, check the JOBLOG and SYSTEM LOG for specifics.

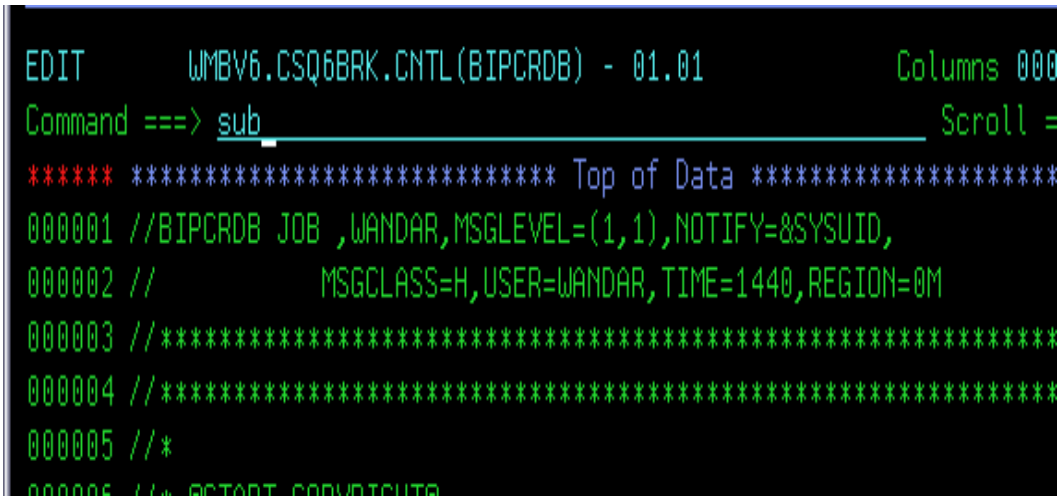
Check BIPGEN results (cont.)



Check the ENVFILE output stream to ensure that it contains the correct values.

A quick scan through the ENVFILE should be enough to ensure it contains the right information.

Prime DB2



BIPCRDB creates the DB2 storage group and database. Note that the proper database authorities are required to run this job.

Having run CSQ6EDBK, BIPCRDB should be ready to submit.

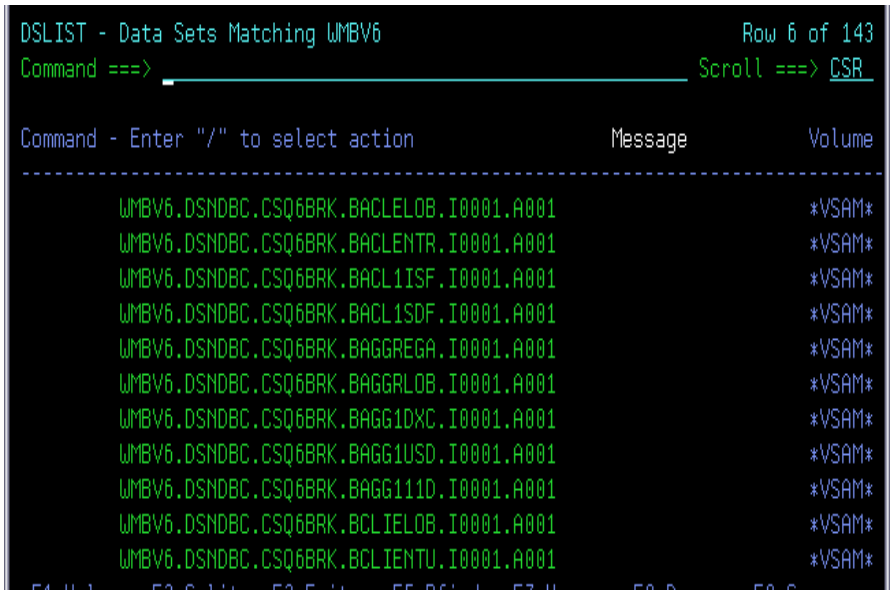
Check BIPCRDB results

```
$HASP373 BIPCRDB  STARTED - INIT 1    - CLASS A - SYS CAP1
IEF403I BIPCRDB - STARTED - TIME=16.14.48
    BIPCRDB     STORGRP     IKJEFT01    0000
    BIPCRDB     DATABASE    IKJEFT01    0000
    BIPCRDB     TABLESPC    IKJEFT01    0000
IEF404I BIPCRDB - ENDED - TIME=16.14.54
$HASP395 BIPCRDB  ENDED
```


It is important to check that all steps in BIPCRDB ran successfully.

If there are any errors then check the output of the JOBLOG and the SYSTEM LOG for specifics.

Check BIPCRDB results (cont.)



The database components will appear on the volume you selected.

Edit BIPCRBK (mqsicreatebroker)

```
EDIT       WMBV6.CSQ6BRK.CNTL(BIPCRBK) - 01.01            Columns 00
Command ===> _____________________________________________ Scroll =
000109 //STDOUT   DD SYSOUT=*
000110 //STDERR   DD SYSOUT=*
000111 //SYSTSPRT DD SYSOUT=*
000112 //SYSTSIN  DD *
000113 BPXBATSL PGM -
000114   /usr/lpp/mqsi/V6R0M0/bin/-
000115 mqsicreatebroker -
000116   CSQ6BRK -
000117   -q CSQ6 -
000118   -u CSQ6BRK -
000119   -n MVS227D1
000120 /*
000121 //
```


Edit BIPCRBK which runs the mqsicreatebroker command. Verify that the WebSphere MQ queue manager name (-q) and the data source userid (-u parameter; CSQ6BRK in this case) and data source name (-n parameter; MVS227D1 in the example here) are correct. If no other options are entered, as in the example shown, all three passes of mqsicreatebroker command will be executed.

Additional options (-1, -2, and -3) give you the capability to create the broker in separate passes: registry, WebSphere MQ, and database. This allows different users to run specific portions of the mqsicreatebroker command.

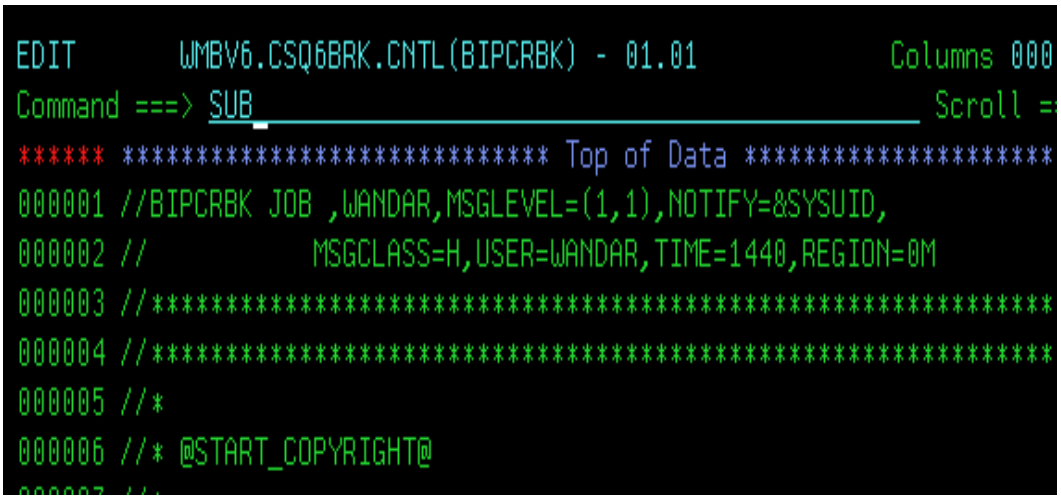
Here are the available options and the authorities required to run them.

Option -1 creates the files and directories which are placed in the default storage group. NOTE: This job must run first, and to do this you need authority to access the broker root directory.

Option -2. creates the WebSphere MQ queues. If you do not have the requisite authority, ask your WebSphere MQ system administrator to run the job.

Option -3. creates the DB2 tables. If you do not have the requisite authority, ask your DB2 system administrator to run the job.

Submit BIPCRBK

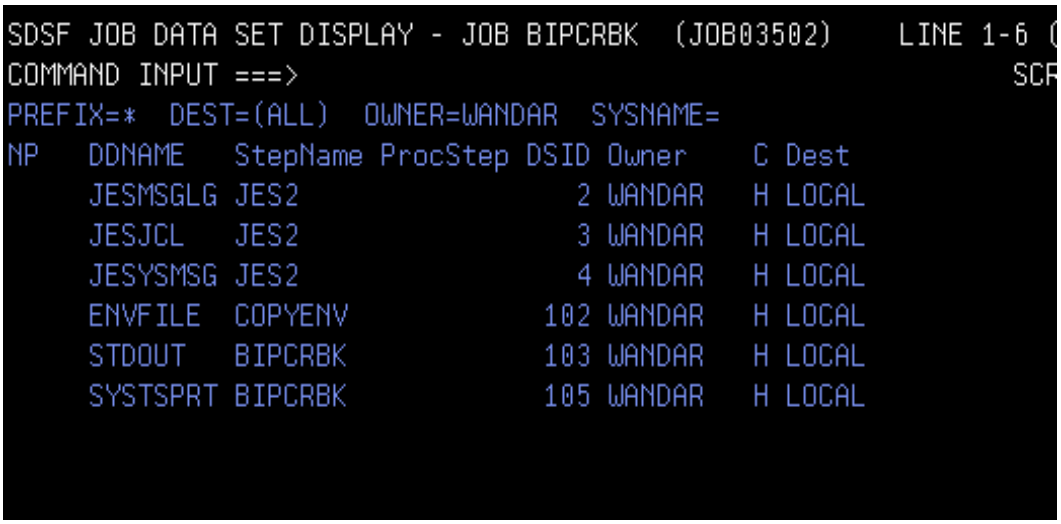


Submit BIPCRBK to create the broker component.

BIPCRBK runs mqsicreatebroker which will create the following :

- Component directory (Including registry)
- MQ queues
- DB2 Tables and Indexes

Check BIPCRBK results



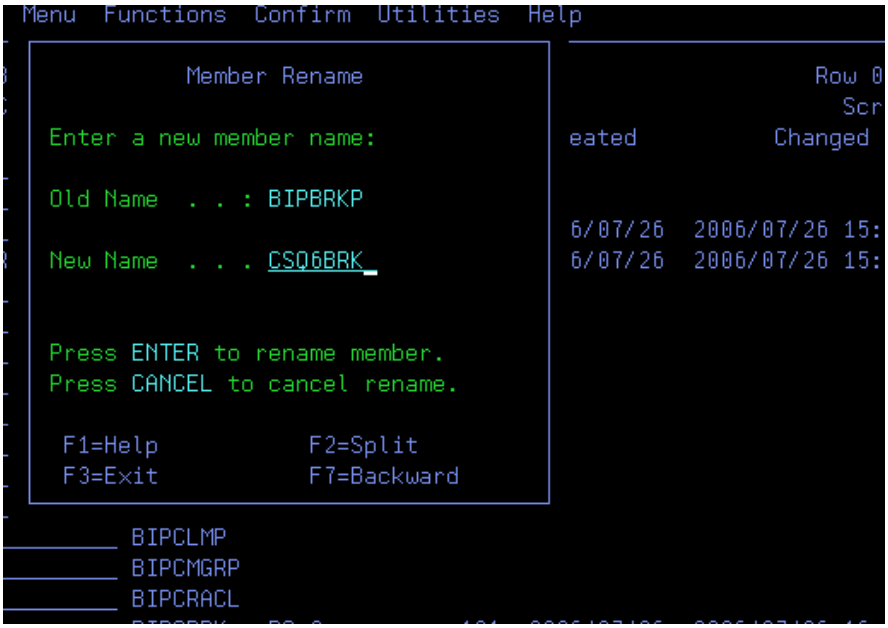
It is important to check that all steps in BIPCRBK ran successfully.

Check the STDOUT stream in the BIPCRBK joblog for specifics. All output from running the mqsicreatebroker command is written here.

Also check SYSTEM LOG for any possible messages. If there are errors, correct and resubmit BIPCRBK.

You should see “BIP8071I: Successful command completion.” at the bottom of the STDOUT stream in the BIPCRBK joblog.

Rename BIPBRKP (started task)



BIPBRKP contains the broker started task JCL. It should have already been customized when you ran your BIPEDIT (CSQ6EDBK in this sample) . Rename the file to the same as ++STARTEDTASKJCL++ that you defined in CSQ6EDBK (BIPEDIT).

Copy this started task procedure to the procedures library defined for your installation.

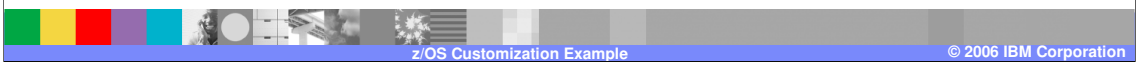
Start the broker

You are now ready to start your broker! However, be sure that you have started DB2 and WMQ before starting the broker.

This command may be entered from the system console or from SDSF, as shown.

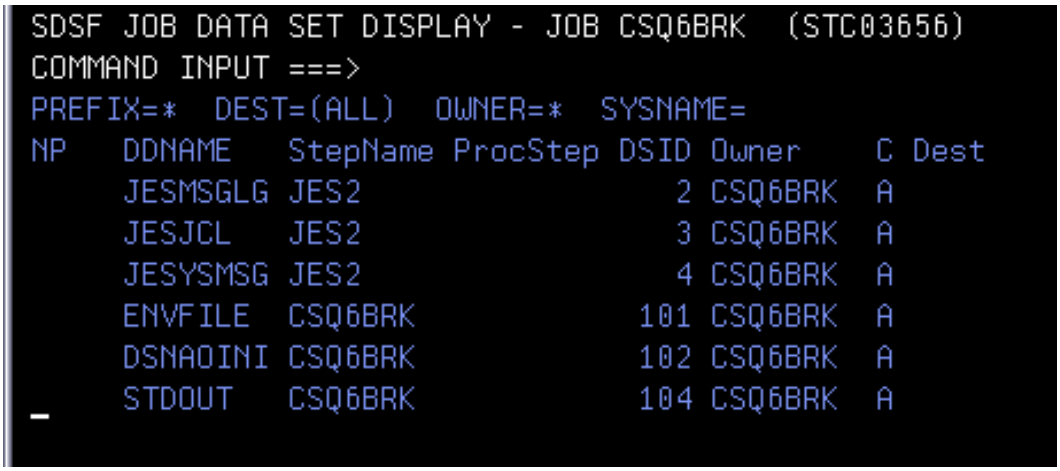
Section

Runtime output



This section reviews information provided when running a broker.

Check verification output

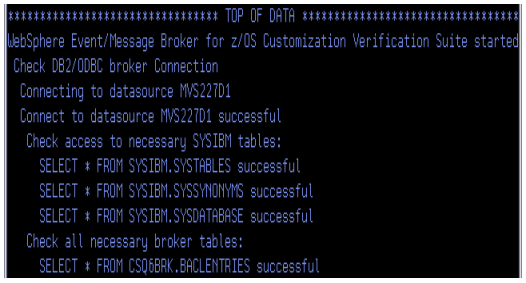


When a broker is started, one of the initial steps to run is VFYDB2MQ. This makes certain verification checks regarding the broker's environment, DB2 database and MQ queues.

The output from the verification step can be seen in the STDOUT stream.

.

Verification output samples



The STDOUT from the Broker job shows the verification steps some examples are seen here. The verification step checks database connections, database table authorizations, APF authorization of bipimain, Java™ level, directory permissions, and MQ connections and queues. If verification fails then the broker will not be started.

System log messages at broker startup

```
0281  $HASP373 CSQ6BRK  STARTED
0090  IEF403I CSQ6BRK - STARTED - TIME=12.38.12
0090      CSQ6BRK     COPYENV     IKJEFT01    0000
0290  IEF170I 1 CSQ6BRK      CSQ6BRK     COPYENV     IKJEFT01    0000
0090      CSQ6BRK     COPYDSN     IKJEFT01    0000
0290  IEF170I 1 CSQ6BRK      CSQ6BRK     COPYDSN     IKJEFT01    0000
0090      CSQ6BRK     CHECKDFE    IKJEFT01    0000
0290  IEF170I 1 CSQ6BRK      CSQ6BRK     CHECKDFE    IKJEFT01    0000
0090      CSQ6BRK     VFYDB2MQ    BPXBATSL    0000
0290  IEF170I 1 CSQ6BRK      CSQ6BRK     VFYDB2MQ    BPXBATSL    0000
0090      CSQ6BRK     BROKER      BPXBATCH    0000
0290  IEF170I 1 CSQ6BRK      CSQ6BRK     BROKER      BPXBATCH    0000
0090  +BIP9141I CSQ6BRK 0 THE COMPONENT WAS STARTED. : ImbControlService(744)
0090  +BIP9108I CSQ6BRK 0 BROKER SERVICE VALUE IS IMBSERV.V6R0M00.FP1..... :
      ImbControlService(749)
0090  +BIP2001I CSQ6BRK 0 THE WEBSPHERE MESSAGE BROKERS SERVICE HAS STARTED AT
       VERSION 6001; PROCESS ID 81. : ImbControlService(773)
```


When the broker is started, the system log will show messages as shown here.

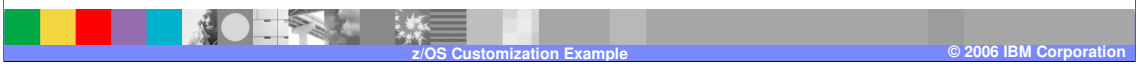
OMVS display of running broker

```
0290  D OMVS,U=CSQ6BRK
0090  BPXO040I 12.38.55 DISPLAY OMVS 845
0090  OMVS     000E ACTIVE          OMVS=(SL)
0090  USER     JOBNAME  ASID       PID       PPID STATE   START    CT_SECS
0090  CSQ6BRK  CSQ6BRK  0037   50331727          1 1W---- 12.38.18     1.591
0090    LATCHWAITPID=          0 CMD=/usr/lpp/mqsi/V6R0M0/bin/bipimain
0090  CSQ6BRK  CSQ6BRK  0037   16777296   50331727 HR---- 12.38.18     1.591
0090    LATCHWAITPID=          0 CMD=bipservice CSQ6BRK AUTO
0090  CSQ6BRK  CSQ6BRK  0037         81   16777296 HR---- 12.38.20     1.591
0090    LATCHWAITPID=          0 CMD=bipbroker CSQ6BRK
0090  CSQ6BRK  CSQ6BRK  0037   33554514         81 HR---- 12.38.22     1.591
0090    LATCHWAITPID=          0 CMD=biphttplistener CSQ6BRK
```


An OMVS display of a running broker shows the processes. As message flows are deployed and execution groups are started, this display will show the additional processes running.

Debugging – relevant documentation

- JOBLOG
  - JESMSGLG for any BIP runtime messages
  - STDOUT for any command messages or output
  - STDERR for any error text
  - JESYSMSG for any JCL customization errors
  - ENVFILE – Created by BIPGEN
- SYSLOG
  - Some errors (such as from BPXBATCH) are only written to the SYSLOG
- ++HOME++ directory
  - Files are written here when submitting BIPGEN



If you have problems in the broker, there are various places to look for indications of the error: the JOBLOG of the broker job, the JES message logs, command message output, error text and the environment file. The system log also contains messages that could be helpful in determining the cause of errors. The home directory (for example, /u/CSQ6BRK) may also have files with relevant information.

Section

Summary and references

This section summarizes the z/OS customization steps.

Summary

- Customize jobs to create broker database, registry and WMQ queues
- Run jobs and verify output
- Set up started task procedure
- Start broker
- Check verification step
- Confirm that the broker is running

Screen captures of the steps required to create a broker, with relevant output, were provided. Similar steps should be followed to create Configuration Manager or User Name Server. There is a PDF document provided which lists details of authorities needed and specifics on customizing the jobs for both creating a broker and a configuration manager. It also includes information on how to set up the configuration manager in the toolkit and how to deploy a message flow to a z/OS configuration manager.

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	CICS	IMS	MQSeries	Tivoli
IBM(logo)	Cloudscape	Informix	OS/390	WebSphere
e(logo)business	DB2	iSeries	OS/400	xSeries
AIX	DB2 Universal Database	Lotus	pSeries	zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.