User Guide

Version 2.0.0

mqdocument@mqsystems.com

Contents

What Is MQDocument?

MQDocument extracts configuration and security information[1] about WebSphere MQ objects from running Queue Manager(s) and writes this information to an XML file.

Presentation

Using a style sheet (mqdocument.xsl) included with MQDocument the organised information within the XML file can be displayed in a browser, and an XSLT processor[2] to generate HTML files using the XML output and style sheet. The HTML files can in turn be edited using Microsoft Word or other HTML-supporting word processor, to include configuration and security information about your WebSphere MQ objects in existing documentation.

Benefits

- Ease Of Use
  - ✓ rapid production of WebSphere MQ object configuration and security information
- Save Time, Improve Quality
  - ✓ generate quality WebSphere MQ object configuration and security documentation
  - ✓ produce on-line, shareable documentation
- Flexibility And Customisation
  - ✓ Using XML/XSL
    - re-use the WebSphere MQ configuration information
    - customise layouts to fit corporate standards. The MQDocument style sheets provides the framework *you* can extend to fit the need
    - customise to local language if needed, all text is stored in a separate file and can be customised.
- Secure
  - ✓ Does NOT require the command server to run, uses local `runmqsc` and `amqoamd`

Platform Support

Windows, Linux, AIX, Solaris and other UNIX (including QSH mode on iOS) systems providing the `runmqsc` and `amqoamd` WebSphere MQ commands. For other platforms like z/OS you can either use the native output of runmqsc equivalent (like MAKEDEF for z/OS) or use the client connection / intermediate queue manager route using for example MS03 (SaveQmgr or SaveQmgrc), MO71 or MO72 supportpacs (see also MQDocument supportpac support below).

Minimum Configuration

- ✓ WebSphere MQ Version 5.2 with CSD 4
- ✓ Java 1.3.1 runtime
- ✓ Internet Explorer with MSXML 3.0

[1] Distributed platforms
[2] MQDocument has been tested with nxslt , nxslt2 and instant SAXON

Windows Quick Start!

1. On the Server/Workstation hosting the Queue Manager(s) create a directory named `MQDocument` (**Note:** this is for the trial version only, with the full product you can choose your own directory)

2. Unzip the supplied MQDocument file to the above directory

3. From a command line within the above directory, use

   ```
   java -cp .;NetRexxR.jar;MQDocument.jar mqdocument /h
   ```

   for the 'Help' screen

4. If you are part of the `mqm` group or have access to the `runmqsc` and `amqoamd` commands, run `mqdocument` using

   ```
   java -cp .;NetRexxR.jar;MQDocument.jar mqdocument -m QueueManagerName
   OR
   java -cp .;NetRexxR.jar;MQDocument.jar mqdocument -m ALL
   ```

   and view the results using Internet Explorer

To use `mqdocument` without specifying `-cp .;NetRexxR.jar;MQDocument.jar` add these settings to the `CLASSPATH` variable statement.

AIX, Solaris & Linux (including QSH mode on iOS) Quick Start!

1. On a suitable workstation create a directory named `mqdocument` (**Note:** this is for the trial version only, with the full product you can choose your own directory)

2. Unzip the supplied MQDocument file to the above directory

3. `ftp` the `MQDocument.jar`, `mqdocument.lic` and `NetRexxR.jar` files[3] to a suitable directory on the Server/Workstation hosting the Queue Manager(s)

4. From within the above directory, use

   ```
   java -cp .:NetRexxR.jar:MQDocument.jar mqdocument /h
   ```

   for the 'Help' screen

5. If you are part of the `mqm` group or have access to the `runmqsc` and `amqoamd` commands, run `mqdocument` using

   ```
   java -cp .:NetRexxR.jar:MQDocument.jar mqdocument -m QueueManagerName
   OR
   java -cp .:NetRexxR.jar:MQDocument.jar mqdocument -m ALL
   ```

To use `mqdocument` without specifying `-cp .:NetRexxR.jar:MQDocument.jar` add the settings to your `CLASSPATH` variable statement.

IBM Support Pac Support

The MQDocument modules can be used with the MO71, MO72 and MS03 Support Pac's.

[3] `mqdocument.xsl` is only required to view the output using a browser, therefore you may need to `ftp` the output file to a suitable environment if there is no facility to view the output on the machine on which the output was generated

MQDocument modules

mqdocument

```
mqdocument -m QMGR [-o OutFile] [-s] [-a] [-t] [-noSec]|[-secOnly] [-noXSL]
                   [-reportonly tags] [-savecopy dateformat] [-log] [-version]
                   [-schedule] [-queue queuename] [-queuemgr queuemanagername]
```

`-m` `QMGR` is the name of the local queue manager to record, or to simultaneously record multiple queue managers separate their names with a colon, e.g. `-m QMGR1:QMGR3` or for all queue managers use `-m ALL`

Optional parameters:

`-o`	`OutFile` is the name of the output file to be generated, defaults to `<qmgrname>.SNAPSHOT.XML`
`-s`	Suppress recording of `SYSTEM` objects
`-a`	Include recording of `QLOCAL` attributes for `IPPROCS`, `OPPROCS`, and `CURDEPTH`
`-t`	Include recording of `QLOCAL` definitions for `TEMPDYN` queues
`-noSec`	Do **not** record ACL/Security settings (`amqoamd` output)
`-secOnly`	Record ACL/Security settings **only** (`amqoamd` output)
`-noXSL`	Do **not** include XSL style sheet for easy viewing of the XML files
`-reportonly`	:AttributeTags: report only objects and attributes that match the attribute tags i.e. :PUT:GET:CONNAME: will only report objects that contain attributes PUT, GET or CONNAME and only report the values of these attributes.
`-savecopy`	yyyyMMddHHmmss makes a copy of the generated file and inserts the given string values after the first qualifier of the name of the file, so QMGRA.XML becomes QMGRA.yyyyMMddHHmmss.XML, you could also use yyyyMMdd or yyyyDDMM or any other combination.
`-log`	Record Log Settings as Attributes of QMGR
`-version`	Record Version information as Attributes of QMGR
`-schedule`	0,hh:mm,hh:mm start mqdocument and run in a daily schedule on listed times for example 0,08:00,12:00,18:00
`-queue`	If this option with queuename is specified, a copy of the generated file will be put to the specified queue. *This allows you to setup distributed queuing and collect the files using a central queue manager.*
`-queuemgr`	Queue manager name is the name of the queue manager you want the files to be put using the -queue option. *If a default queue manager is present on the system and this is the queue manager you want to use, then there is no need to specify a -queuemgr option.*

Note: using the above options to send requires the addition of *com.ibm.mq.jar* to your CLASSPATH (-cp .;NetRexxR.jar;[YourMQDirectory]\Java\lib\com.ibm.mq.jar;MQDocument.jar)

If you have setup the send queue to be a remote queue with transmission queue and channels to a central location you can use this option to have **mqdocument send** the file to a central location using MQ…

To **receive** the files on the central location you use:
java -cp .;NetRexxR.jar;*[YourMQDirectory]*\Java\lib\com.ibm.mq.jar;MQDocument.jar mqdocqutil -queue *Queuename* [-queuemgr *QueueManagerName*]

http://www-01.ibm.com/support/docview.wss?rs=171&context=SSFKSJ&dc=DB560&dc=DB520&uid=swg21316673&loc=en_US&cs=UTF-8&lang=en&rss=ct171websphere

mqdocclient (MO71 and MO72 Support Pac)

```
mqdocclient -m QMGR [MQSC Client parms] [-o OutFile] [-s] [-a] [-t]              [-
                 noXSL] [-i InputFile]
```

`-m` `QMGR` is the name of the remote queue manager to record

Optional parameters:

`-o` `OutFile` is the name of the output file to be generated, defaults to `<qmgrname>.SNAPSHOT.XML`

`-s` Suppress recording of `SYSTEM` objects

`-a` Include recording of `QLOCAL` attributes for `IPPROCS`, `OPPROCS`, and `CURDEPTH`

`-t` Include recording of `QLOCAL` definitions for `TEMPDYN` queues

`-noXSL` Do **not** include XSL style sheet for easy viewing of the XML files

`-i` `InputFile` is the name of the `runmqsc` output of `DIS QMGR`, `DIS QL(*) ALL`, etc.

The `InputFile` can be generated by the chaining the relevant commands in an input file and directing the output to a file, e.g. `runmqsc QMGR < commands.txt > QMGR.runmqscout`, and output from the ACL file generated by the `amqoamd -s` command must be provided as a second file, e.g. `-i QMGR.runmqscout:QMGR.acl`

`[MQSC Client parms]` will be passed 'as is' to the MO71/MO72 Client making the remote connection to set up the communication for `mqdocclient`

Valid parameters:

`-c` channel name

`-f` path to `MQMON.CFG` file

`-g` use `MQMON.CFG` configuration file

`-h` Host Name of the Server hosting the queue manager

`-z` Identifies the queue manager as z/OS-based

sample commands to run on Windows:

Note: QMGRA setup using MQMON.CFG and MQMON.CFG in C:\WMQ

```
java -cp .;NetRexxR.jar;MQDocument.jar mqdocclient -m QMGRA -f C:\WMQ -g
```

Note: QMGRA setup to run remote on host 192.168.171.2, port 14141

```
java -cp .;NetRexxR.jar;MQDocument.jar mqdocclient -m QMGRA -c SYSTEM.DEF.SVRCONN -
h 192.168.171.2(14141)
```

or the **don't install any software on my system at all** option: (!!!)

create a file called **qmgr_input.txt** with:

```
DIS QMGR ALL
DIS QUEUE(*) ALL
DIS CHANNEL(*) ALL
DIS PROCESS(*) ALL
DIS NAMELIST(*) ALL
DIS AUTHINFO(*) ALL
DIS SERVICE(*) ALL
DIS LISTENER(*) ALL
```

for V7 you can add:

```
DIS TOPIC(*) ALL
DIS SUB(*) ALL
```

runmqsc QueueManagerName < qmgr_input.txt > QueueManagerName.scr

amqoamd –m QueueManagerName –s > QueueManagerName.acl

copy/ftp the files *QueueManagerName.scr and QueueManagerName.acl to your own workstation*

java -cp .;NetRexxR.jar;MQDocument.jar mqdocclient –m QMGRA -I QMGRA.SCR:QMGRA.ACL

mqdocmqsc (MS03 Support Pac)

```
mqdocmqsc -i InputFile [-m QMGR] [-o OutFile] [-s] [-t]
                       [-noSec] [-secOnly] [-noXSL]
```

`-i` `InputFile` is the name of the output of `saveqmgr`

The ACL information generated by the `amqoamd -s` command must be provided as a second file, e.g. `-i QMGR.MQS:QMGR.acl`, unless the `-z` option[4] is used by `saveqmgr`

Optional Parameters:

`-m` `QMGR` is the name of the queue manager recorded which if not clear from the `InputFile` name, `QMgr="noname"` will be set in the `OutFile`.

`-o` `OutFile` is the name of the output file to be generated, defaults to `<qmgrname>.SNAPSHOT.XML`

`-s` Suppress recording of `SYSTEM` objects

`-t` Include recording of `QLOCAL` definitions for `TEMPDYN` queues

`-noSec` Do **not** record ACL/Security settings (`amqoamd` output)

`-secOnly` Record ACL/Security settings **only** (`amqoamd` output)

`-noXSL` Do **not** include XSL style sheet for easy viewing of the XML files

The `InputFile` can also be generated by the chaining the relevant commands in an input file and directing the output to a file, e.g. `runmqsc QMGR < commands.txt > QMGR.runmqscout`, and output from the ACL file generated by the `amqoamd -s` command must be provided as a second file, e.g. `-i QMGR.runmqscout:QMGR.acl`

sample commands to run on Windows:

Note: QMGRB setup to run locally

```
saveqmgr -m QMGRB -f QMGRB_ms03.mqsc -o -z QMGRB_ms03.acl

java -cp .;NetRexxR.jar;MQDocument.jar mqdocmqsc -m QMGRB -i
QMGRB_ms03.mqsc:QMGRB_ms03.acl
```

Note: QMGRA setup to run remote on host 192.168.171.2, port 14141 and using saveqmgrc

```
saveqmgrc -m QMGRA -f QMGRA_ms03_c.mqsc -o -z QMGRA_ms03_c.acl -a
192.168.171.2(14141)

java -cp .;NetRexxR.jar;MQDocument.jar mqdocmqsc -m QMGRA -i
QMGRA_ms03_c.mqsc:QMGRA_ms03_c.acl
```

[4] Applicable to WebSphere MQ v6.0 only

mqdocmakedef (z/OS-based Queue Managers)

```
mqdocmakedef -i InputFile [-o OutFile] [-migrate]
```

`-i` `InputFile` is the name of the `MAKEDEF`/`JCL` file

Optional Parameters:

-o is the name of the outputfile to be used for output
(default name will be [QMgrname].MAKEDEF.MQSC
This file can then be used as input for the mqdocmqsc utility
to generate the [QMgrname].SNAPSHOT.XML or
to migrate object definitions from z/OS to other platforms such as
Windows or Unix, see also the -migrate switch

-migrate Suppress z/OS specific objects and attributes to migrate object definitions to non-z/OS platforms

An output file, *[QMGRName]*.MAKEDEF.MQSC, will be produced against which `mqdocmqsc` must be run to produce the XML file

mqdocmerge

```
mqdocmerge -i InputFileNames -m MergeFileName [-noXSL]
```

`-i` `InputFileNames` is the name of the input files to be merged, which must be separated with a colon, e.g. `QMGR1.SNAPSHOT.XML:QMGR2.SNAPSHOT.XML`

`-m` `MergeFileName` is the name of the output file containing the merged files

Optional Parameters:

`-noXSL` Do **not** include XSL style sheet for easy viewing of the XML files

mqdocdiff

```
mqdocdiff -i InputFileName -p PreviousFileName [-s] [-a] [-t]
                                                [-random]
                                                [-nodates]
                                                [-defaults]
                                                [-noSec]
```

`-i` `InputFileName` is the name of the newest XML snapshot file

`-p` `PreviousFileName` is the name of the XML snapshot file to be compared with the `InputFileName`

Optional Parameters:

`-s` Suppress comparing of objects that begin with ”SYSTEM”

`-a` Include comparing of `QLOCAL` attributes for `IPPROCS`, `OPPROCS` and `CURDEPTH` etc.

`-t` Include comparing of `QLOCAL` definitions for `TEMPDYN` queues

`-random` Enable random comparison of two ‘clone’ queue managers and can also be used to ignore/suppress DATE/TIME changes

`-nodates` Can be used to ignore/suppress DATE/TIME changes

`-defaults`	Can be used to compare object settings to the defaults and report differences
`-noSec`	Ignore ACL/Security settings

mqdocgen (generate MQSC and ACL command files from your SNAPSHOT XML Files)

```
        mqdocgen -i InputFileName [-c CurrentFileName] or [-m GenQMgr]
                                  [-r Reference] [-tag LeadTagName]
                                  [-extMQSC MQSCextension] [-extACL ACLextension]
                                  [-Replace] [-defaults]
-i          is the name of the Input (Change) XML/MQSC file.
            This is the file that contains the desired "state" of object's
```

Optional Parameters:

-c is the name of the Current Version XML/MQSC file.
This is the file you created on your live system OR
is the export file from your current configuration database.

Note: If NO Current Version file is specified, definitions are generated as if there were no prior definitions

-m is the name of the QueueManager for which you want to generate MQSC definitions if NO CurrentFileName is specified

-r is the Reference you can add to the generated MQSC file
no -r results in MQDOC.[QMgrName].MQSC
-r CHG001 results in MQDOC.CHG001.[QMgrName].MQSC

-tag is the replacement LeadTagName to replace the default
no -tag results in MQDOC.[QMgrName].MQSC
-tag MYTAG results in MYTAG.[QMgrName].MQSC

-extMQSC is the replacement MQSC exension to replace the default
no -tag results in MQDOC.[QMgrName].MQSC
-tag MYMQSC results in MQDOC.[QMgrName].MYMQSC

-extACL is the replacement ACL exension to replace the default
no -tag results in MQDOC.[QMgrName].ACL
-tag MYACL results in MQDOC.[QMgrName].MYACL

-replace Adds REPLACE parameter to ALL DEFINE MQSC Commands

-defaults compare object settings to the defaults and generate differences

Other Settings

Style Sheet

The default style sheet is `mqdocument.xsl` but an environment variable, `MQD_XSL`, can be set to change the default and/or change the path to the file, e.g.:

```
set MQD_XSL=c:\mystylesheets\mqdocument.xsl      (Windows)
export MQD_XSL=/mystylesheets/my-mqdocument.xsl (UNIX)
```

Licence File

The default licence file is mqdocument.lic but an environment variable, `MQD_LIC`, can be set to change the default and/or change the path to the file, e.g.:

```
set MQD_LIC=c:\mylicenses\mqdocument.lic        (Windows)
export MQD_LIC=/mylicenses/my-mqdocument.lic    (UNIX)
```

Running MQDocument as a WebSphere MQ Service

You can also configure MQDocument to run as a WebSphere MQ Service,
optionally using the -queue command to send the SNAPSHOT.XML files automatically to a central location.

(Windows)

```
DEFINE SERVICE(MQDOCUMENT) +
       DESCR('MQDocument Service Object') +
       STARTCMD('java') +
       STARTARG(' mqdocument -m ALL -queue MQDOC -schedule 0,08:00,12:00,18:00') +
       STDOUT('/MQDocument/Schedule.log') +
       SERVTYPE(SERVER) +
       STOPCMD('tskill') +
       STOPARG('+MQ_SERVER_PID+') +
       REPLACE
```

(UNIX)

```
DEFINE SERVICE(MQDOCUMENT) +
       DESCR('MQDocument Service Object') +
       STARTCMD('java') +
       STARTARG(' mqdocument -m ALL -queue MQDOC -schedule 0,08:00,12:00,18:00') +
       STDOUT('/MQDocument/Schedule.log') +
       SERVTYPE(SERVER) +
       STOPCMD('kill') +
       STOPARG('+MQ_SERVER_PID+') +
       REPLACE
```

In above scenarios it is assumed the MQDOC Queue has been created either as Local Queue or as Remote Queue, also the Queue Manager running the service is assumed to be the default, if it is not the default you need to add the -queuemgr *QMGRName* to the STARTARG attribute.

Trial setup and usage Instructions

In order to get and start using the trial version of MQDocument please follow these instructions carefully!

First: download the following 2 files:
http://www.mqsystems.com/images/mqdocument-trial.zip
http://www.mqsystems.com/images/supportjars.zip (1809KB)

MQDocument Quick start

Put the downloaded files on a machine that has WebSphere MQ installed and Queue managers running.
Create a directory, called **C:\MQDocument**
(**Note:** this is for the *trial only*, with the regular product this is *not needed*)

Unzip mqdocument-trial.zip and supportjars.zip to C:\MQDocument

Open a Command Prompt window and navigate to C:\MQDocument

To see if MQDocument works use: java -cp .;NetRexxR.jar;MQDocument.jar mqdocument ?

You should now receive the message: **Error! file: mqdocument.lic does not exist.**

The software is now installed and you can request a trial license key by sending an e-mail to

MQDocument at MQSystems (e-mail domain .com), don't forget to mention you already downloaded the trial software and installed it on your system!

Wait to receive the mqdocument.lic from MQDocument Support, once you receive the mqdocument.lic file you can put it in C:\MQDocument

Repeat the command: : java -cp .;NetRexxR.jar;MQDocument.jar mqdocument ?
You should now get a screen with all the help options available.

If it works, you can run MQDocument using:
java -cp .;NetRexxR.jar;MQDocument.jar mqdocument -m ALL

(on UNIX systems use **:** as classpath seperator instead of **;**)

It will now create SNAPSHOTs of all queue managers installed and running on this machine,
you will see the messages go by on the screen.

For the Trial Edition, the stylesheet (mqdocument.xsl) is not supplied as it is plain text and can not be protected, so the following **additional step** is only needed for the trial version.

To see the result of MQDocument in a browser or Word Document you need to convert the *[QmgrName]*.SNAPSHOT.XML file to html

A windows command file xml2html.cmd is supplied to make it easier for you to convert the xml files to html:
(**Note:** if you would like to know the actual commands for conversion, look in the command file)

Command format: xml2html option xmlfilename

option values:	mqdoc	using MQDocument	compiled styleheet for IE
	mqdocFF	using MQDocument	compiled styleheet for FireFox

Use: xml2html mqdoc *[QmgrName]*.SNAPSHOT.XML or
xml2html mqdocFF *[QmgrName]*.SNAPSHOT.XML to convert the file to HTML

The HTML files can be opened with the appropriate browser, the Internet Explorer version of the HTML file is also suitable for opening in Word.