

MQSeries Everyplace - Administration tool

User Guide

Version 1.0.2

31 October 2000

Phill van Leersum
MQSeries Development
IBM United Kingdom Laboratories
Hursley Park
Hursley
Hampshire, SO21 2JN
UK

phillvl@uk.ibm.com

Property of IBM

Take Note!

Before using this report be sure to read the general information under "Notices".

Third Edition, October 2000

This edition applies to Version 1.0.2 of *MQSeries Everyplace - Administration tool* and to all subsequent releases and modifications unless otherwise indicated in new editions.

© **Copyright International Business Machines Corporation 2000**. All rights reserved. Note to US Government Users -- Documentation related to restricted rights -- Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule contract with IBM Corp.

Table of Contents

Notices	iii
Preface	v
Description	vi
Prerequisites	1
Installation	1
Running the program	2
Usage	2
Adding QueueManager to Administer	2
Navigation Keys	3
Editing on line	3
Editing off-line	4
Editing keys	4
File Save and Load	4
Scripting	4
Multiple Metamodels	4
Showing the Administering QueueManager	4
Command Line Arguments	5
Making the program	5

Notices

References in this report to IBM products or programs do not imply that IBM intends to make these available in all countries in which IBM operates.

Information contained in this report has not been submitted to any formal IBM test and is distributed “AS-IS”. The use of this information and the implementation of any of the techniques is the responsibility of the reader. Much depends on the ability of the reader to evaluate these data and project the results to their operational environment.

The performance data contained in this report was measured in a controlled environment and results obtained in other environments may vary significantly.

Trademarks and service marks

The following terms, used in this publication, are trademarks of the IBM Corporation in the United States or other countries or both:

- IBM
- MQSeries
- Everyplace

The following terms are trademarks of other companies:

- Java Sun Corporation
- Windows Microsoft Corporation

Summary of Amendments

Date	Changes
17 August 2000	Initial release
23 August 2000	Updated to reflect addition of shell script files to the program directory.
31 October 2000	Updated to reflect code enhancements. Administering QM is now viewable; all resources packaged into the jar files for simple distribution.

Preface

This SupportPac provides a tool that allows administration of all MQSeries Everyplace resources. It is written in pure Java (1.1) and is therefore platform independent. It is built upon a generic metamodel for administration, which allows easy customisation of the administration tool when the customer extends the capabilities of MQSeries Everyplace.

Bibliography

List any supporting publications here otherwise delete this page. Use following format:

- *User Interface Architecture* (es01uia.pdf)
- *MetaModel XML format* (es01xmlformat.pdf)
- *MQSeries Everyplace - Administration tool design* (es01design.pdf)

Description

This SupportPac provides a tool that allows administration of all MQSeries Everyplace resources. It is written in pure Java (1.1) and is therefore platform independent. It is built upon a generic metamodel for administration, which allows easy customisation of the administration tool when the customer extends the capabilities of MQSeries Everyplace.

It has a small footprint. It is designed to allow different renderers to be used on different platforms.

Administration can be performed online or off-line, and work-in-progress can be saved to a human readable xml format.

All administration tasks can be performed simply and naturally, with plenty of visual feedback.

Prerequisites

The SupportPac requires:

- A Java 1.1 compliant JVM
- MQSeries Everyplace

An MQSeries Java client is required for MQe-MQ bridge administration.

Installation

To install the SupportPac:

1. Copy the file *es01.zip* to a temporary directory.
2. Uncompress using an unzip utility. Ensure you specify the appropriate option to re-create stored directories.
3. Four sub-directories will be created; *source*, *MQeExplorer*, *MQeBridgeExplorer*, and *docs*.

The *source* directory contains all the Java source files and an example batch file necessary to build the executables.

The *MQeExplorer* and *MQeBrdgeExplorer* directories contain pre-built versions of the administration tool.

The *docs* directory contains the following documentation in Adobe PDF format:

- *es01.pdf* - The User Guide (This document)
- *es01uia.pdf* - This document describes the architecture of a platform-independent, low footprint, extensible administration tool.

- *es01xmlformat.pdf* - This document describes the format of the XML used as a persistent representation of a UIA MetaModel.
- *es01design.pdf* - This document describes the architecture of a platform-independent, low footprint, extensible administration solution for MQSeries Everyplace.

Running the program

For your convenience the explorer is provided in two forms each started by a batch file (for Windows) or script file (for unix) . There are two important batch files:

- **MQeExplorer.bat (MQeExplorer.sh)**: runs the simple explorer that does not recognise bridge resources (but will manage bridge queue managers). Depends upon Java runtime and MQe.
- **MQeBridgeExplorer.bat (MQeBridgeExplorer.sh)**: The full functionality explorer. Manages bridge and non-bridge resources. Depends upon Java runtime, MQe, and MQ Series Java client.

To execute the program you will need to edit the appropriate batch file before invoking it. Environment variables are set up in the batch files to make this easy. For **MQeExplorer.bat** you will need to ensure that:

- Java classes are on the classpath
- Java executables are on the path
- MQe.jar is on the classpath
- MQeMetaModel.jar, metaModel.jar, and renderer.jar are on the classpath

For **MQeBridgeExplorer.bat** all the above apply and the following as well:

- MQ Series Java Client classes are on the classpath
- MQeBridgeMetaModel.jar is on the classpath

Usage

Adding QueueManager to Administer

The explorer cannot create queue managers. These must be created by MQe. The explorer attaches to a queue manager by creating resources. The instructions assume that you have created and started a queueManager that you wish to administer.

From the menubar select MQeSystem/Administer QueueManager

The dialogue then allows you to enter the details of the queue manager that you wish to administer. When you have done this, click 'OK'. The defaults in the dialogue assume that you have created a standard 'ExampleQM' (see MQe example code descriptions in MQe documentation) and it is on the same machine on which you are running the explorer).

A queue manager 'proxy' will be added to the navigator pane. The queue manager proxy represents the queue manager that you are administering, it will be in blue, as the proxy is not yet connected to the real queue manager.

A '+' appears to the left of the queue manager proxy, indicating that the queue manager has children. click on the '+' to show the children. Click on it again to hide the children. Currently the only children are two empty categories, one for queues and one for connections.

Right click on the queue manager to bring up a popup menu and choose 'connect - asynchronous'. The queue manager will now attempt to connect to the real queue manager. When this is successful, the queue manager will change colour to black.

From the popup menu select refresh. The qm proxy will enquire the details of the real queue manager and build the navigator display to match.

If you are using a BridgeExplorer and the queue manager is a bridge queue manager, then a folder for 'bridges' will be added.

Navigation Keys

Up/Down keys move the selection bar

Enter/Return toggles expand/collapse

Shift + Enter elicits popup menu.

Editing on line

Now that the queue manager is connected, queues and connections can be created and deleted in real time. Aliases can be added to queues and connections. Store and forward queues can have their queue manager entries added and deleted.

As resources are created and deleted, the progress is indicated by colour changes. These may be too rapid to see (especially when working online).

As a resource is created, it is coloured blue. The explorer will immediately attempt to create the real resource on MQe, and during creation the resource will be represented in cyan. If the creation is successful, then the resource will be coloured black. If the creation is unsuccessful, then the resource will return to blue, and a message will be displayed.

As a resource is deleted, it is coloured red. The explorer will immediately attempt to delete the real resource on MQe, and during deletion the resource will be represented in pink. If the deletion is successful, then the resource will be removed from the explorer. If the creation is unsuccessful, then the resource will return to red, and a message will be displayed.

NB: Please do not delete AdminQueues or AdminReplyQueues as these are used to administer the resources.

Editing off-line

Resources can be edited off-line (while the queue manager is disconnected). The colour changes are more obvious. When all edits have been made to the users satisfaction, the queue manager is connected, and the changes are all committed at once.

Editing keys

Control + X: cut

Control + C: copy

Control + V: paste

These are performed via the clipboard. The internal representation can be saved to file by pasting to a text editor. A saved representation can be copied to the clipboard from a text editor and pasted into the explorer.

The options are also available from the popup menu, or from the 'selected' menu.

File Save and Load

The cut/copy/paste actions use an xml representation of the model objects. This representation can be saved to, and loaded from, file. Note that type checking is performed, and if the type being loaded from file is not appropriate then no action is taken.

Loading xml works just as for pasting, so if online the resources will be created immediately, if off-line the resources can be created later (once connected).

Scripting

Since resources defined in an xml file will be created when the definition is encountered, we have a primitive form of scripting. Obviously simply creating resources is not the full story. Operations can also be declared in the xml file. These operations will be invoked on the owning Mqe resource (if the operation is recognised as one defined in the metamodel). Parameters can be passed to the operation as defined in the metamodel file.

It is recommended that connection should be made in the synchronous mode for this kind of scripting.

No 'control of flow' constructs are available.

Multiple Metamodels

The renderer is designed to allow any metamodel to be loaded, and even to have several loaded at once. The metamodel file defines the classes to be loaded, and these must be on the classpath at startup time.

Showing the Administering QueueManager

In order to perform administration without side effects upon the administered queue managers, the explorer creates a 'hidden' queue manager. It is sometimes helpful to view, or change, the resources available to the hidden queue manager. These actions have been added to the menu bar. Be very careful about editing the resources on this queue manager: the resources that are created automatically are necessary for administration. With this caveat in mind, it is possible to create a very quick demo by starting two explorers and communicating between them. In order to do this it is necessary to understand the command line arguments required by the explorer

(particularly the name and port). To run two explorers as a demo the QueueManager name of one must be changed, and if the two explorers are on the same machine then the port number of one must be changed.

Command Line Arguments

The command line argument give greater flexibility in the use of the explorer. The default values for the arguments should give a working explorer.

/model modelName	The metamodel file to use .Default is mqeMetaModel.xml for MQeExplorer and mqeBridgeMetaModel for MQeBridgeExplorer. The program will seek the file first within the specified resource file and if this fails, will seek it outside the resource file.
/name qmName	The name for the administering qm. Default is AdministeringQM.
/port portnumber	The port to listen on. Default is 8088.
/resource fileName	The name of file containing resources. Default is mqeExplorer.jar for the explorer and mqeBridgeExplorer.jar for the bridge explorer.
-pmenu	Disable popup menus. Default is enabled. Some platforms (eg netbook) do not have popup menus.
-smenu	Disable 'selected' menus Default is enabled. On some platforms the 'selected' menu and the 'popup' menus interact badly.

Making the program

In the code directory is all the source code used in the explorer. There is also a batch file (**make.bat**) that can be used to remake the entire code (on Microsoft Windows or similar). This batch file will need to be edited for different environments, or different platforms.

- End of Document -