NetIQ® AppManager® for WebSphere MQ on UNIX

Management Guide

November 2015

www.netiq.com/documentation

Legal Notice

Contents

About this Book and the Library

The NetIQ AppManager product (AppManager) is a comprehensive solution for managing, diagnosing, and analyzing performance, availability, and health for a broad spectrum of operating environments, applications, services, and server hardware.

AppManager provides system administrators with a central, easy-to-use console to view critical server and application resources across the enterprise. With AppManager, administrative staff can monitor computer and application resources, check for potential problems, initiate responsive actions, automate routine tasks, and gather performance data for real-time and historical reporting and analysis.

Intended Audience

This guide provides information for individuals responsible for installing an AppManager module and monitoring specific applications with AppManager.

Other Information in the Library

The library provides the following information resources:

Installation Guide for AppManager

Provides complete information about AppManager pre-installation requirements and step-by-step installation procedures for all AppManager components.

User Guide for AppManager Control Center

Provides complete information about managing groups of computers, including running jobs, responding to events, creating reports, and working with Control Center. A separate guide is available for the AppManager Operator Console.

Administrator Guide for AppManager

Provides information about maintaining an AppManager management site, managing security, using scripts to handle AppManager tasks, and leveraging advanced configuration options.

Upgrade and Migration Guide for AppManager

Provides complete information about how to upgrade from a previous version of AppManager.

Management guides

Provide information about installing and monitoring specific applications with AppManager.

Help

Provides context-sensitive information and step-by-step guidance for common tasks, as well as definitions for each field on each window.

The AppManager library is available in Adobe Acrobat (PDF) format from the AppManager Documentation page of the NetIQ website.

About NetIQ Corporation

We are a global, enterprise software company, with a focus on the three persistent challenges in your environment: Change, complexity and risk—and how we can help you control them.

Our Viewpoint

Adapting to change and managing complexity and risk are nothing new

In fact, of all the challenges you face, these are perhaps the most prominent variables that deny you the control you need to securely measure, monitor, and manage your physical, virtual, and cloud computing environments.

Enabling critical business services, better and faster

We believe that providing as much control as possible to IT organizations is the only way to enable timelier and cost effective delivery of services. Persistent pressures like change and complexity will only continue to increase as organizations continue to change and the technologies needed to manage them become inherently more complex.

Our Philosophy

Selling intelligent solutions, not just software

In order to provide reliable control, we first make sure we understand the real-world scenarios in which IT organizations like yours operate — day in and day out. That's the only way we can develop practical, intelligent IT solutions that successfully yield proven, measurable results. And that's so much more rewarding than simply selling software.

Driving your success is our passion

We place your success at the heart of how we do business. From product inception to deployment, we understand that you need IT solutions that work well and integrate seamlessly with your existing investments; you need ongoing support and training post-deployment; and you need someone that is truly easy to work with — for a change. Ultimately, when you succeed, we all succeed.

Our Solutions

- Identity & Access Governance
- Access Management
- Security Management
- Systems & Application Management
- Workload Management
- Service Management

Contacting Sales Support

For questions about products, pricing, and capabilities, contact your local partner. If you cannot contact your partner, contact our Sales Support team.

Worldwide:	www.netiq.com/about_netiq/officelocations.asp
United States and Canada:	1-888-323-6768
Email:	info@netiq.com
Website:	www.netiq.com

Contacting Technical Support

For specific product issues, contact our Technical Support team.

Worldwide:	www.netiq.com/support/contactinfo.asp
North and South America:	1-713-418-5555
Europe, Middle East, and Africa:	+353 (0) 91-782 677
Email:	support@netiq.com
Website:	www.netiq.com/support

Contacting Documentation Support

Our goal is to provide documentation that meets your needs. The documentation for this product is available on the NetIQ website in HTML and PDF formats on a page that does not require you to log in. If you have suggestions for documentation improvements, click **comment on this topic** at the bottom of any page in the HTML version of the documentation posted at www.netiq.com/documentation. You can also email Documentation-Feedback@netiq.com. We value your input and look forward to hearing from you.

Contacting the Online User Community

NetIQ Communities, the NetIQ online community, is a collaborative network connecting you to your peers and NetIQ experts. By providing more immediate information, useful links to helpful resources, and access to NetIQ experts, NetIQ Communities helps ensure you are mastering the knowledge you need to realize the full potential of IT investments upon which you rely. For more information, visit community.netiq.com.

1 Installing AppManager for WebSphere MQ UNIX

This chapter describes requirements for how to install AppManager for WebSphere MQ UNIX.

This chapter assumes you have an AppManager repository, console, management server, and UNIX agent installed. For more information about installing AppManager, see the *Installation Guide for AppManager* and for information about installing the UNIX agent, see the *AppManager for UNIX and Linux Servers Management Guide*, which are available on the AppManager Documentation page.

1.1 System and Account Requirements

For the latest information about specific supported software versions and the availability of module updates, visit the AppManager Supported Products page.

AppManager for WebSphere MQ UNIX has the following system requirements:

Item	Requirement
AppManager repository, management server, and Control Center console	7.0 or later
NetIQ UNIX Agent	7.1 or later
Operating system on agent computers	One of the following: ◆ CentOS ◆ HP-UX ◆ IBM AIX ◆ Oracle Linux ◆ Oracle Solaris ◆ Red Hat Enterprise Linux ◆ SUSE Linux Enterprise Server
IBM WebSphere MQ	Version 6, 7, or 8

If you encounter problems using this module with a later version of your application, contact NetIQ Technical Support.

1.2 Installing the Module

To install the module you must:

- Install the Knowledge Scripts by running the module installer `.msi` on all AppManager repositories that store data for this module.

- Install the Help files by running the module installer `.msi` on all AppManager Control Center and Operator Console computers you will use with this module.
- Ensure that UNIX agent 7.1 or later is installed on the computer you want to monitor.

You can access the `AM70-WebSphereMQUNIX-7.9.x.x.msi` module installer on the AppManager Module Upgrades & Trials page.

The module installer now installs Knowledge Scripts for each module directly into the QDB instead of to the `\AppManager\qdb\kp` folder as in previous releases of AppManager.

To install the module:

1. Ensure you have the UNIX agent installed on the managed computer or computers. For information about how to install the agent, see the *AppManager for UNIX and Linux Servers Management Guide*.
2. Install the Knowledge Scripts into the QDB by running the module installer on the QDB computer:

   2a Select **Install Knowledge Scripts** to install the repository components.

   2b Specify the SQL Server name of the server hosting the QDB, as well as the case-sensitive QDB name.
3. (Conditional) If you use Control Center 7.x, run the module installer for each QDB attached to Control Center.
4. (Conditional) If you use Control Center 8.x, run the module installer only for the primary QDB. Control Center automatically replicates this module to secondary QDBs.
5. Run the module installer on all console computers to install the Help and console extensions.
6. (Conditional) If you have not discovered WebSphere MQ on UNIX resources, run the Discovery_WebSphereMQUNIX Knowledge Script on all agent computers that you want to monitor. For more information about how to run discovery, see Section 1.3, “Discovering WebSphere MQ Resources,” on page 11.
7. To get the updates provided in this release, upgrade any running Knowledge Script jobs. For more information, see Section 1.4, “Upgrading Knowledge Script Jobs,” on page 11.

1.2.1 Silently Installing the Knowledge Scripts

To silently install the module on an AppManager repository, you can use Windows authentication or SQL authentication.

Windows authentication:

```
AM70-WebSphereMQUNIX-7.9.x.x.msi /qn MO_B_QDBINSTALL=1 MO_B_SQLSVR_WINAUTH=1
MO_SQLSVR_NAME=[SQLServerName] MO_QDBNAME=[AMRepositoryName]
```

SQL authentication:

```
AM70-WebSphereMQUNIX-7.9.x.x.msi /qn MO_B_QDBINSTALL=1 MO_B_SQLSVR_WINAUTH=0
MO_SQLSVR_USER=[SQL login] MO_SQLSVR_PWD=[SQLLoginPassword]
MO_SQLSVR_NAME=[SQLServerName] MO_QDBNAME=[AMRepositoryName]
```

1.3 Discovering WebSphere MQ Resources

After you have installed all the NetIQ UNIX Agent on the WebSphere MQ Server, started the UNIX agent, and verified that your WebSphere MQ queues are running, run the WebSphereMQUNIX discovery script to discover WebSphere MQ resources. These resources include WebSphere MQ queues, queue managers, and channels.

NOTE: To successfully discover WebSphere MQ on UNIX, the agent account must belong to the WebSphere MQ Server user group. The default WebSphere MQ Server user group is **mqm**. In the scenario where multiple WebSphere MQ installations are present on the same machine where an agent is present, is not supported by the Knowledge Scripts (although you can discover the installations).

Set the **Values** tab parameters as needed.

Description	How to Set It
Raise event when discovery succeeds?	This Knowledge Script always raises an event when the job fails for any reason. In addition, you can set this parameter to **y** to raise an event when the job succeeds. The default is n.
Path to Queue Managers	Enter the full path to the queue manager on the WebSphere MQ Server you want to discover. The default is: `/var/mqm/qmgrs`.
Event severity when discovery...	Set the event severity level, from 1 to 40, to reflect the importance when the job: ◆ **... succeeds**. If you set this Knowledge Script to raise an event when the job succeeds, set the event severity level for a successful discovery. The default is 25. ◆ **... fails**. The default is 5. ◆ **... is partially done**. This type of failure usually occurs when the root user does not belong to the WebSphere MQ Server user group (by default this is **mqm**) or if there are multiple queues installed in different directories but one of the queues is not running. The default is 11.

1.4 Upgrading Knowledge Script Jobs

This release of AppManager for WebSphere MQ on UNIX contains updated Knowledge Scripts. You can push the changes for updated scripts to running Knowledge Script jobs in one of the following ways:

- Use the AMAdmin_UpgradeJobs Knowledge Script.
- Use the Properties Propagation feature.

1.4.1 Running AMAdmin_UpgradeJobs

The AMAdmin_UpgradeJobs Knowledge Script can push changes to running Knowledge Script jobs. Your AppManager repository (QDB) must be at version 7.0 or later. In addition, the repository computer must have hotfix 72040 installed, or the most recent AppManager Repository hotfix. To download the hotfix, see the AppManager Suite Hotfixes page.

Upgrading jobs to use the most recent script version allows the jobs to take advantage of the latest script logic while maintaining existing parameter values for the job.

For more information, see the **Help** for the AMAdmin_UpgradeJobs Knowledge Script.

1.4.2 Propagating Knowledge Script Changes

You can propagate script changes to jobs that are running and to Knowledge Script Groups, including recommended Knowledge Script Groups and renamed Knowledge Scripts.

Before propagating script changes, verify that the script parameters are set to your specifications. Customized script parameters may have reverted to default parameters during the installation of the module. New parameters may need to be set appropriately for your environment or application.

You can choose to propagate only properties (specified in the Schedule and Values tabs), only the script (which is the logic of the Knowledge Script), or both. Unless you know specifically that changes affect only the script logic, you should propagate the properties and the script.

For more information about propagating Knowledge Script changes, see the *Running Monitoring Jobs* chapter of the *Operator Console User Guide for AppManager*.

Propagating Changes to Ad Hoc Jobs

You can propagate the properties and the logic (script) of a Knowledge Script to ad hoc jobs started by that Knowledge Script. Corresponding jobs are stopped and restarted with the Knowledge Script changes.

To propagate changes to ad hoc Knowledge Script jobs:

1. In the Knowledge Script view, select the Knowledge Script for which you want to propagate changes.
2. Right-click the script and select **Properties propagation** > **Ad Hoc Jobs**.
3. Select the components of the Knowledge Script that you want to propagate to associated ad hoc jobs:

Select	To propagate
Script	The logic of the Knowledge Script.
Properties	Values from the Knowledge Script Schedule and Values tabs, such as schedule, monitoring values, actions, and advanced options.

Propagating Changes to Knowledge Script Groups

You can propagate the properties and logic (script) of a Knowledge Script to corresponding Knowledge Script Group members.

After you propagate script changes to Knowledge Script Group members, you can propagate the updated Knowledge Script Group members to associated running jobs. For more information, see "Propagating Changes to Ad Hoc Jobs" on page 12.

To propagate Knowledge Script changes to Knowledge Script Groups:

1. In the Knowledge Script view, select the Knowledge Script Group for which you want to propagate changes.
2. Right-click the Knowledge Script Group and select **Properties propagation** > **Ad Hoc Jobs**.
3. (Conditional) If you want to exclude a Knowledge Script member from properties propagation, deselect that member from the list in the Properties Propagation dialog box.
4. Select the components of the Knowledge Script that you want to propagate to associated Knowledge Script Groups:

Select	To propagate
Script	The logic of the Knowledge Script.
Properties	Values from the Knowledge Script Schedule and Values tabs, including the schedule, actions, and Advanced properties.

5. Click **OK**. Any monitoring jobs started by a Knowledge Script Group member are restarted with the job properties of the Knowledge Script Group member.

2 WebSphere MQ UNIX Knowledge Scripts

AppManager provides the following Knowledge Scripts for monitoring IBM WebSphere MQ Servers running on UNIX operating systems.

From the AppManager consoles, you can select a Knowledge Script and press **F1** for complete details.

Knowledge Script	What It Does
ADMINClearLocalQueue	Clears local queue messages.
ADMINQueueMgrStartStop	Sends a command to the WebSphere MQ Server to start or stop a queue manager.
ChannelStatus	Monitors the status of WebSphere MQ channels.
DynamicLocalQueueDepth	Monitors the depth of local queues, including dynamically created queues.
LocalQueueDepth	Monitors the total number of messages in discovered local queues.
PingQueueManager	Pings a local queue manager and restarts the command server or queue manager if either is detected down.
ServerDown	Monitors processes that belong to a specified WebSphere MQ Server user group.
TestQueueManager	Monitors the connection between the WebSphere MQ Server and a local queue manager.
WebSphereMQErrorLog	Monitors the WebSphere MQ error log file for specific strings and messages logged since the last monitoring interval.

2.1 ADMINClearLocalQueue

Use this Knowledge Script to clear local queue messages. This Knowledge Script clears the local queue of any messages when the local queue depth exceeds the specified threshold.

Alternatively, you can configure this Knowledge Script to clear the local queue each time the Knowledge Script job runs, regardless of how many messages are in the local queue.

AppManager raises an event if the local queue cannot be cleared.

This Knowledge Script makes WebSphere MQ Server easier to administer because it can be used to clear several local queues at once.

2.1.1 Resource Object

MQSeries queue.

2.1.2 Default Schedule

The default interval for this Knowledge Script is **Run once**.

2.1.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Local queue message depth threshold	Enter a threshold for the maximum number of messages, from 0 to 6400, in a local queue. If the threshold is exceeded, the local queue is cleared. The default is 5.
Clear all messages regardless of queue depth?	Set to **y** to ensure that a threshold cannot be set and all messages in the local queue are cleared regardless of queue depth. The default is n.
Event severity if queue manager or command server is down	Set the event severity level, from 1 to 40, to indicate the importance of the event. The default is 5.

2.2 ADMINQueueMgrStartStop

Use this Knowledge Script to send a command to the WebSphere MQ Server to start or stop a queue manager. This Knowledge Script verifies that the command completed successfully.

This Knowledge Script makes WebSphere MQ Server easier to administer, because it can start or stop several queue managers at once.

2.2.1 Resource Object

MQSeries queue manager.

2.2.2 Default Schedule

The default interval for this Knowledge Script is **Run once**.

2.2.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Start queue manager? (y/n)	Set to **y** to start a queue manager. The default is y.
Stop queue manager? (y/n)	Set to **y** to stop a queue manager. The default is n.

Description	How to Set It
Time to wait before checking command results	Enter the number of seconds, from 0 to 9999, for the Knowledge Script to wait before attempting to confirm that a start or stop command completed successfully. When specifying this value, consider the following: ◆ The system resources available to WebSphere MQ Server. ◆ It takes longer to stop a queue manager than to start a queue manager. If the amount of time you specify is insufficient for the queue manager to start or stop, the status of the queue manager is not confirmed and AppManager raises an event. The default is 30 seconds.
Event severity if command...	Set the event severity level, from 1 to 40, to indicate the importance if the command: ... fails. The default is 5. ... succeeds. The default is 12.

2.3 ChannelStatus

Use this Knowledge Script to monitor the status of WebSphere MQ channels. AppManager raises an event when the channel is inactive, stopped, paused, or running.

If AppManager cannot retrieve the status of a channel because the channel is not initialized correctly, AppManager raises an event with an event message that includes a WebSphere MQ internal code describing the problem. You can use this information to locate the origin of the problem.

2.3.1 Resource Object

MQSeries channels.

2.3.2 Default Schedule

The default interval for this Knowledge Script is **Every 5 minutes**.

2.3.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Event if channel is...	Set to **y** to raise an event if the channel status is: ... inactive. The default is y. ... stopped. The default is y. ... paused. The default is y. ... running. The default is n. ... binding. The default is n. ... starting. The default is n. ... stopping. The default is n. ... retrying. The default is n. ... requesting. The default is n. ... initializing. The default is n. ... not found. The default is n.
Collect Data?	Set to **y** to collect data for graphs and reports. The default is **n**. If set to **y**, the script returns a value of: ◆ **100** if the channel is running. ◆ **50** if the channel is binding, starting, stopping, retrying, requesting, or initializing. ◆ **25** if the channel is inactive, stopped, or paused. ◆ **0** if the channel status cannot be retrieved.

Description	How to Set It
Event severity if channel...	Set the event severity level, from 1 to 40, to reflect the importance when the channel status is: ... inactive. The default is 5. ... stopped. The default is 5. ... paused. The default is 5. ... running. The default is 25. ... status cannot be retrieved. The default is 25. ... binding. The default is 25. ... starting. The default is 25. ... stopping. The default is 25. ... retrying. The default is 25. ... requesting. The default is 25. ... initializing. The default is 25. ... not found. The default is 25.

2.4 DynamicLocalQueueDepth

Use this Knowledge Script to monitor the total number of messages in any specified local queue. AppManager raises an event if the number of messages in a queue exceeds the threshold you specify or the queue threshold specified in WebSphere MQ Explorer.

This Knowledge Script monitors an implementation of WebSphere MQ in which local queues are created and deleted dynamically. This Knowledge Script specifies queue names, does not raise events for queues that have already been deleted, and can monitor queues created since AppManager discovery was last run. For these reasons, the DynamicLocalQueueDepth Knowledge Script functions more efficiently than LocalQueueDepth for a dynamic implementation of WebSphere MQ.

If necessary, adjust the schedule to reflect the rate at which local queues are created and deleted.

2.4.1 Resource Object

MQSeries queue manager.

2.4.2 Default Schedule

The default interval is **Every 5 minutes**.

2.4.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Event? (y/n)	Set to **y** to raise events. The default is y.
Queue name search criteria	Enter the queue name you want to monitor. The asterisk can be used as a wildcard. For example, the default (SYS*) monitors all system local queues. Queue names are case-sensitive.
Use queue's own threshold? (y/n)	Set to **y** to use the queue threshold defined in the queue's local attributes. AppManager raises an event if the threshold is met or exceeded. The default is n.
Local queue message depth threshold	If you do not want to use the queue's own threshold, enter a threshold for the maximum number of messages in the local queues being monitored. AppManager raises an event if the threshold is met or exceeded. The default is 5.
Event severity if queue manager or command server is down	Set the event severity level, from 1 to 40, to indicate the importance of the event. The default is 5.

2.5 LocalQueueDepth

Use this Knowledge Script to monitor the total number of messages in the local queues. AppManager raises an event if the number of queue messages exceeds the threshold. You can define the threshold by the following:

- The number of messages set by the user.
- The queue's own threshold.
- The percentage of the queue's maximum depth represented by the number of messages in the queue (also called the percentage high).

This script monitors static implementations of WebSphere MQ in which local queues remain fixed. For the monitoring of dynamic implementations of WebSphere MQ, use DynamicLocalQueueDepth (which specifies queue names), because that script does not raise events for queues that were discovered by AppManager but have already been deleted.

2.5.1 Resource Object

MQSeries queue.

2.5.2 Default Schedule

The default interval is **Every 5 minutes**.

2.5.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Event? (y/n)	Set to **y** to raise events. The default is y.
Collect data? (y/n)	Set to **y** to collect data for graphs and reports. The default is n. If set to **y**, the script returns the number of messages in the local queue.
Event if message queue is full?	Select y to create an even if the message queue is full. The default is n.
Event if message queue is over threshold (-1 disables)	Specify a number of messages in the queue to raise an event when the number of messages reach that threshold. If you do not want AppManager to raise an event for this criteria, enter -1. The default is 5.
Event if message queue is over percentage utilization (-1 disables)	Specify a percent of the queue's maximum depth to raise an event when the depth reaches that percentage. If you do not want AppManager to raise an event based on this criteria, enter -1. The default is 90%.
Event severity if queue manager or command server is down	Set the event severity level, from 1 to 40, to indicate the importance of the event. The default is 5.

2.6 PingQueueManager

Use this Knowledge Script to ping a local queue manager. AppManager raises an event when the queue manager or the command server is detected down, and the Knowledge Script then attempts to restart the command server or queue manager.

2.6.1 Resource Object

MQSeries queue manager.

2.6.2 Default Schedule

The default interval is **Every 5 minutes**.

2.6.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Event? (y/n)	Set to **y** to raise events. The default is y.
Collect data? (y/n)	Set to **y** to collect data for graphs and reports. The default is n. If set to **y**, the script returns a value of 100 if the queue manager is up, or a value of 0 if it is down.

Description	How to Set It
Restart queue manager if down? (y/n)	Set to **y** to automatically restart the queue manager if it is detected down. The command server will also automatically be restarted if this option is set to **y**. The default is y.
Event Severity if queue manager or command server...	Set the event severity level, from 1 to 40, to reflect the importance when the queue manager or command server: ... is down. The default is 5. ... restarted successfully. The default is 12.

2.7 ServerDown

Use this Knowledge Script to monitor processes that belong to a specified WebSphere MQ Server user group. AppManager raises an event when there are no processes that belong to the specified WebSphere MQ Server user group.

2.7.1 Resource Object

WebSphereMQUNIX Server.

2.7.2 Default Schedule

The default interval for this script is **Every 30 minutes**.

2.7.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Event if over the threshold? (y/n)	Set to **y** to raise an event if the server is down or the number of processes that belong to the specified user group is 0. The default is y.
Collect data? (y/n)	Set to **y** to collect data for charts and reports. If set to **y**, this script returns the number of processes belonging to the specified user group running on the server. The default is n.
WebSphereMQ server user group	Enter the name of the WebSphere MQ Server user group. The default is `mqm`.
Event severity level	Set the event severity level, from 1 to 40, to indicate the importance of the event. The default is 5.

2.8 TestQueueManager

Use this Knowledge Script to monitor the connection between the WebSphere MQ Server and a local queue manager. AppManager raises an event if the WebSphere MQ Server fails to connect to the local queue manager.

This Knowledge Script only establishes that there is a connection between a queue manager and the WebSphere MQ Server. It does not monitor the status of the command server or send any messages to queues. If you need to monitor the command server, or restart either the command server or the local queue manager, use the PingQueueManager Knowledge Script instead.

2.8.1 Resource Object

MQSeries queue manager.

2.8.2 Default Schedule

The default interval is **Every 10 seconds**.

2.8.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Event? (y/n)	Set to **y** to raise an event if the connection between a local queue manager and the WebSphere MQ Server fails. The default is y.
Collect data? (y/n)	Set to **y** to collect data for reports and graphs. The default is n. If set to **y**, the script returns a value of 100 if the queue manager connection is up, or a value of 0 if it is down. The default is n.
Event severity if queue manager or command server is down	Set the event severity level, from 1 to 40, to indicate the importance of the event. The default is 5.

2.9 WebSphereMQErrorLog

Use this Knowledge Script to monitor the WebSphere MQ error log file for specific strings and messages logged since the last monitoring interval. This Knowledge Script allows you to specify the filename and a regular expression to identify the string to look for or to exclude. The Knowledge Script then scans the ASCII file and reports the matching entries found since the last monitoring period and checks for changes to the text file that match the expression you enter. The Knowledge Script does not re-scan the entire file at each interval.

In the first interval, this Knowledge Script reads the file and inserts a marker at the end of the file. In subsequent intervals, the script checks the file for changes that match the search criteria you specified. If the file is recreated between intervals and the file size is smaller than the previous version of the file, the script treats the file as a new file and searches the entire file from the beginning. AppManager raises an event when the number of lines matching your search criteria exceeds the threshold you set.

Use Perl regular expressions to specify the include and exclude patterns.

2.9.1 Resource Object

WebSphereMQUNIX Server.

2.9.2 Default Schedule

The default interval for this script is **Every 30 minutes**.

2.9.3 Setting Parameter Values

Set the following parameters as needed:

Description	How to Set It
Event? (y/n)	Set to **y** to raise events. The default is y.
Collect data? (y/n)	Set to **y** to collect data for graphs and reports. If set to **y**, the script returns the number of lines containing matching strings. The default is n.
WebSphere MQ error log file (full path)	Enter the full path to the WebSphere MQ error log file. The default path is `/var/mqm/errors/AMQERR01.LOG` You can only specify one file name for any job instance. To monitor multiple logs or files, create separate Knowledge Script jobs.
Regular expression specifying the include filter	Type a regular expression, in Perl, to identify the pattern you want to look for in the text file being monitored. Strings matching the include filter pattern are returned. The default expression matches all strings.
Modifier for the regular expression include filter	Optional modifiers can be used to change the behavior of the regular expression. For example, specifying "i" for this parameter makes the include filter case-insensitive.
Regular expression specifying the exclude filter	Type a regular expression, in Perl, to identify the pattern you want to exclude from matching in the text file being monitored. Strings matching the include filter pattern are returned.
Modifier for the regular expression exclude filter	Optional modifiers can be used to change the behavior of the regular expression. For example, specifying "i" for this parameter makes the exclude filter case-insensitive.
Threshold for matching lines	Enter the number of times to detect a match before raising an event. The default is 0, the first instance exceeds the threshold and raises an event.
Event severity level	Set the event severity level, from 1 to 40, to indicate the importance of the event. Adjust the severity based on which log or type of event you are monitoring. The default is 5.

2.9.4 Creating Filters with Regular Expressions

Some Knowledge Scripts enable you to use regular expressions to define include and exclude filters for pattern-matching against the text being evaluated. Depending on the Knowledge Script you are working with, you may be able to use regular expression include and exclude filters when you are setting job properties, or you may be able to maintain your search criteria independently of the Knowledge Script parameters in a separate filter file. You may also be able to use regular expression modifiers to further refine your filtering.

For example, if your **include filter** looks like this `replic.*` and you specify the modifier `i` to make the search case-insensitive, the regular expression contains the wildcard (`.`) and repeat (`*`) special characters, indicating you want to find strings that start with `replic` followed by any string of characters. Messages containing either `replication` or `replicated` are captured.

The format is the same for the exclude filter. For example, to find log entries that do not start with the string `success`, the exclude filter might look like this:

```
^success.*
```

If you are only searching for included strings, you can leave the exclude filter blank. If you want to retrieve all messages in the log in a given interval, you can specify `.*` for the include filter and leave the exclude filter blank.

2.9.5 Using Special Characters

The following special characters can be used in regular expressions:

Character	Description
.	Wildcard for any one character
*	Repeat zero or more occurrences
^	Beginning of the line
\$	End of the line
\	Escape the next meta-character
|	Alternate matches
[ ]	Any character in the class set. You can specify individual characters or ranges.
( )	Grouping characters. For example, you can specify (a|b|c) to indicate a match with a, or b, or c.
+	Quantifier indicating one or more occurrences
?	Quantifier indicating zero or one occurrence
{*n*}	Quantifier indicating exactly *n* occurrences
\w	A word character (alphanumeric plus _)
\s	A white-space character
\d	A digit character

2.9.6 Using Regular Expression Modifiers

In addition to the special characters you can use in creating the regular expression, there are a number of modifiers that can be used to modify how pattern-matching is handled. Valid modifiers include:

Modifier	Description
`c`	Complements the search list
`g`	Matches globally as many times as possible
`i`	Makes the search case-insensitive
`m`	Treats the string as multiple lines
`o`	Interpolates variables only once
`s`	Treats the regular expression string as a single long line
`x`	Allows for regular expression extensions

For additional information about writing regular expressions, see your Perl documentation or other regular expression resources.