WebSphere MQ

WebSphere MQ Bridge for HTTP Version 1.1

WebSphere MQ

WebSphere MQ Bridge for HTTP Version 1.1

Note
Before using this information and the product it supports, be sure to read the general information under notices at the back of this book.

Second edition (January 2008)

This edition of the book applies to the following SupportPac:

- IBM WebSphere MQ Bridge for HTTP (SupportPac MA0Y) Version 1.1

and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

Chapter 1. Introduction

This document is about the IBM® WebSphere® MQ Bridge for HTTP SupportPac™ (referred to in this document as the WebSphere MQ Bridge for HTTP). This SupportPac enables client applications to exchange messages with WebSphere MQ from any platform or language with HTTP capabilities without the need for a WebSphere MQ client. The WebSphere MQ Bridge for HTTP is not suitable for use with messages where guaranteed delivery is required.

Benefits

This SupportPac will be of benefit to you if:

- you have environments that you want to connect to WebSphere MQ that are not supported but that can build HTTP requests and handle responses.
- you have environments that you want to connect to WebSphere MQ, but that have insufficient storage space to install a WebSphere MQ client.
- you have multiple systems that you want to connect to WebSphere MQ but it would take too long to install the WebSphere MQ client on all of those systems.
- you want to access a WebSphere MQ infrastructure from a web-based application.
- you want to enhance existing web-based applications, using asynchronous techniques such as AJAX to enhance interactivity.

HTTP support can be used with both point-to-point and publish/subscribe messaging topologies.

How does HTTP support work?

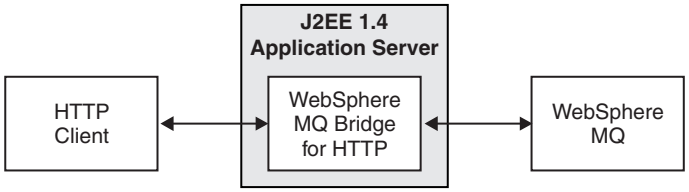

Figure 1. Introduction to WebSphere MQ Bridge for HTTP

As figure one shows, the purpose of the WebSphere MQ Bridge for HTTP is to receive HTTP requests from one or more clients, interact with WebSphere MQ on their behalf and return HTTP responses to them.

The WebSphere MQ Bridge for HTTP consists of a J2EE servlet that is connected to WebSphere MQ via a resource adapter. The HTTP servlet is capable of handling 3 different types of HTTP requests; POST, GET and DELETE.

Table 1. WebSphere MQ Bridge for HTTP verbs

HTTP Request	Result
POST	Puts a message on a queue or topic.

Table 1. WebSphere MQ Bridge for HTTP verbs (continued)

HTTP Request	Result
GET	Browses the first message on a queue. In line with the HTTP protocol this does not delete the message from the queue. This verb cannot be used with publish/subscribe messaging.
DELETE	Browses and deletes a message from a queue or topic.

More information about how to construct HTTP requests and handle HTTP responses from the WebSphere MQ Bridge for HTTP, is given in Chapter 4.

Chapter 2. Preparing to install

Prerequisites

This WebSphere MQ Bridge for HTTP can be used with WebSphere MQ Version 6.0.2.1 or later. It is not necessary for WebSphere MQ to be installed on the same machine as the WebSphere MQ Bridge for HTTP.

In order to make use of the WebSphere MQ Bridge for HTTP you will require some additional software.

Where you install the WebSphere MQ HTTP Bridge you will require:

- a J2EE 1.4 compliant application server.
- a WebSphere MQ JMS provider within your application server.
  - If you are using WebSphere Application Server (WAS) Version 6 or earlier use the WAS Message Listener Port (MLP) to integrate WebSphere MQ as the JMS provider.
  - If you are using an application server other than WAS, use the WebSphere MQ resource adapter. This is included in WebSphere MQ Version 6.0.2.1.

Clients that use the WebSphere MQ Bridge for HTTP must be capable of creating HTTP requests and receiving HTTP responses.

ma0y.install.zip

The WebSphere MQ Bridge for HTTP is delivered as a zip file, ma0y.install.zip and includes:

- this document (ma0y.pdf)
- a readme.txt
- a WMQHTTP.war
- a licenses folder containing product licenses and notices.txt

Security considerations

Client connection authority

Connections between HTTP clients and the application server should be secured at the web container level. It is the responsibility of the administrator of the J2EE application server to secure the WebSphere MQ Bridge for HTTP servlet using standard HTTP server techniques. How you secure the connections will be specific to you application server, refer to your application server's documentation for information.

Resource adapter connection to WebSphere MQ

How you secure the connection between you resource adapter and WebSphere MQ is dependent on your specific resource adapter, refer to your resource adapter's documentation for more information regarding security.

Consideration of what user authorizations are required to initially connect to the WebSphere MQ system from the JMS provider is required.

The resource adapter will connect to WebSphere MQ using a single authorization ID. The user ID used to connect the resource adapter to WebSphere MQ must have the correct WebSphere MQ authorities. In addition, ensure that the user ID used to connect your resource adapter to WebSphere MQ has appropriate authorities for connecting to the relevant queues and topics. Ensure that this user ID does not have unnecessary permissions on the queue manager you are connecting to.

Chapter 3. Installing

To install the WebSphere MQ Bridge for HTTP complete the following steps.

1. Download the SupportPac ma0y.install.zip file from the WebSphere MQ SupportPac website.
2. Before extracting the files from ma0y.install.zip, read the license file in the licenses subdirectory. This license file is supplied in several languages. Ensure that you are prepared to comply with the terms of the license before you continue to install.
3. Extract the files from ma0y.install.zip file into a clean directory of your choice. For example c:\Program Files\IBM\WebSphere MQ HTTP.
4. Optional: The Servlet is pre-configured with a default maximum wait time for HTTP GET and DELETE requests of 35000 milliseconds. For more information about wait times when specified in HTTP GET/DELETEs see “Entity headers” on page 14. Before deploying the Servlet to your application server you can change this value by modifying the maximum_wait_time parameter in the web.xml file inside the Servlet. If you remove this parameter from web.xml, the Servlet will also use a maximum wait time of 5000 milliseconds. If your application server allows, you can set the maximum_wait_time parameter in your deployment plan and override the value set in web.xml. Not all application servers will recognize parameters set in a deployment plan, for more information about your specific application server, refer to your application server’s documentation.
5. Optional: When deploying the Servlet to your application server a context-root forms part of the URI used to send HTTP requests, for more information see “URI Format” on page 11. Depending on your application server this can be configured either in your deployment plan or your application server’s administration console. A sample deployment plan for use with WAS CE is provided in the samples material and includes an example of how to define a context-root. For more information refer to Chapter 6, “WebSphere MQ Bridge for HTTP Samples,” on page 27
6. Deploy the WMQHTTP.war file from the unzipped ma0y.install.zip to your application server. For instructions about how to do this refer to your application server’s documentation.

If you are not using WebSphere Application Sever with the MLP as your WebSphere MQ JMS provider, you can use the WebSphere resource adapter as your WebSphere MQ JMS provider. The WebSphere MQ resource adapter is included in WebSphere MQ Version 6.0.2.1. Refer to your application server’s documentation for information about how to deploy the resource adapter. You must have a JMS connection configured from your application server to WebSphere MQ before you proceed to use the WebSphere MQ Bridge for HTTP.

Chapter 4. Configuring WebSphere MQ Bridge for HTTP

After installing the WebSphere MQ Bridge for HTTP and deploying WMQHTTP.war to your application server, configure the WebSphere MQ Bridge for HTTP to use your connection factory.

To enable the WebSphere MQ Bridge for HTTP to communicate with WebSphere MQ, configure your application server by specifying the connection factory you want to use.

If you are using the WebSphere MQ resource adapter as your WebSphere MQ JMS provider, complete the following steps:

1. Locate and open your deployment plan. The deployment plan you use to deploy your resource adapter will be specific to the application server you are using. If you chose to download the WebSphere MQ Bridge for HTTP samples, a sample deployment plan for use with WebSphere Application Server Community Edition is included in the zip file as an example.
2. Define a resource called jms/WMQHTTPJCAConnectionFactory with a value of the name of your ConnectionFactory object. Ensure that your ConnectionFactory is configured with the name of the WebSphere MQ queue manager that you want to connect to, if a user ID is required to access that queue manager ensure that this is configured on your ConnectionFactory.

If you are using WebSphere Application Server MLP, complete the following step:

1. Using the WebSphere Application Server admin console, create a ConnectionFactory object called jms/WMQHTTPJCAConnectionFactory. For information about how to do this refer to the WebSphere Application Server information center, http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?/topic=/com.ibm.websphere.express.doc/info/exp/ae/tmm_ep.html.

If a connection factory called jms/WMQHTTPJCAConnectionFactory does not exist when the Servlet is invoked for the first time (when it receives the first HTTP request), an MQHTTP00002 error will occur and will be logged in you application server error log.

To confirm that the WebSphere MQ Bridge for HTTP in installed and configured correctly, in a web-browser navigate to http://*hostname*:*port*/*context_root*/msg/queue/*myqueue*, where *myqueue* is an empty WebSphere MQ queue. HTTP error 504 'Message retrieval timed out' will be displayed in the browser window if the WebSphere MQ Bridge for HTTP and JMS resource adapter are configured correctly.

Chapter 5. Constructing HTTP requests and handling HTTP responses

The purpose of the WebSphere MQ Bridge for HTTP is to receive HTTP requests from clients, interact with WebSphere MQ as instructed in those requests and to return HTTP responses to the client. This section of the documentation explains the format of the messages, the information they must contain, and the operations the WebSphere MQ Bridge for HTTP enables you to perform.

Overview of the WebSphere MQ Bridge for HTTP

Some examples of HTTP response and request messages are included in this section to help illustrate the interaction between the HTTP client and WebSphere MQ.

HTTP POST

To put a message to a queue, the client creates an HTTP request. This can be done using an HTTP client tool, for example, AJAX or Java™ HTTP libraries.

The figure below shows an HTTP request to put a message on a queue called `myQueue`. This request contains the HTTP header x-msg-correlId to set the correlation ID of the WebSphere MQ message.

```
POST /msg/queue/myQueue/ HTTP/1.1
Host: www.mqhttpsample.com
Content-Type: text/plain
x-msg-correlID: 1234567890
Content-Length: 50

Here's my message body that will appear on the queue.
```

Figure 2. Simple example of a HTTP POST request to a queue

The figure below shows the response sent back to the client. Note that there is no response content.

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 0
```

Figure 3. Simple example of a HTTP POST response (the POST is to a queue)

HTTP DELETE

To delete a message from a queue the client creates an HTTP request. This can be done using an HTTP client tool, for example, AJAX or Java HTTP libraries.

The figure below shows an HTTP request to delete the next message on queue called `myQueue`. When deleting a message, the message body is sent to the client in the response and in WebSphere MQ terms is a destructive get. This example request contains the HTTP header x-msg-wait to instruct the server how long to

wait for a message to arrive on the queue, and the x-msg-require-headers header to specify that the client wants to receive the message correlation ID in the response.

```
DELETE /msg/queue/myQueue/ HTTP/1.1
Host: www.mqhttpsample.com
x-msg-wait: 10
x-msg-require-headers: correlID
```

Figure 4. Simple example of a HTTP DELETE request to a queue

The figure below shows the response sent back to the client. The correlation ID is returned to the client as requested in the require-headers header of the request.

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 39
Content-Type: text/plain; charset=utf-8
x-msg-correlId: 1234567890

My message the was lying on the queue.
```

Figure 5. Simple example of a HTTP DELETE response (the DELETE is to a queue)

HTTP GET

To get a message from a queue, the client creates an HTTP request. This can be done using an HTTP client tool, for example, AJAX or Java HTTP libraries. After a message has been got from a queue it will remain on the queue, in WebSphere MQ terms this operation is a browse.

The figure below shows an HTTP request for the next message on queue called `myQueue`. This request contains the HTTP headers x-msg-wait to instruct the server how long to wait for a message to arrive on the queue and the x-msg-require-headers to specify that the clients wants to receive the message correlation ID in the response.

```
GET /msg/queue/myQueue/ HTTP/1.1
Host: www.mqhttpsample.com
x-msg-wait: 10
x-msg-require-headers: correlID
```

Figure 6. Simple example of a HTTP GET request to a queue

The figure below shows the response sent back to the client. The correlation ID is returned to the client as requested in the require-headers head of the request.

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 39
Content-Type: text/plain; charset=utf-8
x-msg-correlId: 1234567890

My message the was lying on the queue.
```

Figure 7. Simple example of a HTTP GET response (the GET is to a queue)

The examples above POST, GET and DELETE messages to and from WebSphere MQ queues. You can also POST and DELETE messages to and from topics by changing the /msg/queue/myQueue/ URI in the request to /msg/topic/myTopic/.

GET is not supported for use with publish/subscribe messaging.

For more information about HTTP verbs see section "WebSphere MQ Bridge for HTTP verbs."

URI Format

The IBM WebSphere MQ Bridge for HTTP is deployed within a J2EE server using a context-root that you define when you deploy the Servlet to your application server. The bridge is configured such that all requests to the following URIs are handled by the bridge.

- For point to point messaging *context_root*/msg/queue where *context_root* is as defined in your deployment plan.
- For pub/sub messaging *context_root*/msg/topic where *context_root* is as defined in your deployment plan.

The URI format supported by WebSphere MQ Bridge for HTTP is as follows:

```
Wmq-http-iri = "http:" "//" connection-name "/" wmq-dest
 connection-name = [ tcp-connection-name ]
 tcp-connection-name = ihost [ ":" port ]
 wmq-dest = queue-dest / topic-dest
 queue-dest = "msg/queue/" wmq-resource [ "@" wmq-qmgr ] "/"
 topic-dest = "msg/topic/" wmq-resource "/"
```

Notes:

1. If a questions mark, (?) character is used in a wmq-dest it must be substituted with '%3f', for example, orange?topic should be specified as orange%3ftopic.

Note: @wmq-qmgr is only valid on a POST.

WebSphere MQ Bridge for HTTP verbs

The 3 verbs that you can use in HTTP requests that are sent to the WebSphere MQ Bridge for HTTP are:

- POST
- GET
- DELETE

Throughout this section there are references to the WebSphere MQ Bridge for HTTP headers, for more information about these see “HTTP headers” on page 14.

HTTP POST

Description

Using the HTTP POST operation you can put a message onto a WebSphere MQ queue or topic. The WebSphere MQ Bridge for HTTP will put messages to queues and topics as WebSphere MQ messages, as a result of this, they do not have a RFH2 folder.

You can use HTTP headers in your HTTP POST request to:

- set the properties of the message that is put to WebSphere MQ.
- you can use the require-headers header to specify what information about the WebSphere MQ message you would like to receive as headers in the response message.

If the HTTP POST request is successful, the entity of the response message will be empty and the content-length of the response will be set to zero. The status code of the HTTP response will be ’200 OK’.

In the case of an unsuccessful request, the response will contain a WebSphere MQ Bridge for HTTP error message and the status code of the HTTP response will be one of those documented in the Chapter 8, “HTTP Return codes,” on page 33 section. The WebSphere MQ message will not be put to the queue or topic.

Supported headers

The following headers can be specified in a HTTP POST request:

- class
- correlId
- encoding
- expiry
- format
- persistence
- priority
- replyTo
- usr

For more information about headers see “HTTP headers” on page 14.

HTTP GET

Description

Using the HTTP GET operation you can browse the next message from a WebSphere MQ queue. HTTP GET is not supported for use with topics. A response message will be sent back to the client. The entity of the response message will contain the data from the WebSphere MQ message. The WebSphere MQ message will remain on the queue.

You can use HTTP headers in your HTTP GET request to:

- you can use the require-headers header to specify what information about the WebSphere MQ message you would like to receive as headers in the response message.
- you can use the correlID header, msgID header or both to determine which message you browse from the queue.
- you can use the wait header to determine how long you will wait for a message to arrive on the queue.
- you can use the x-msg-range header to specify the range of data in the message that should be returned in the response.

If the HTTP GET request is successful, the entity of the response message will contain the data of the message retrieved from the WebSphere MQ queue (see supported message types for more information), and the HTTP content-length headers will be set to the number of bytes in the entity body. The status code of the HTTP response will be '200 OK'. If x-msg-range has been specified to be 0, or 0-0, then the status code of the HTTP response will be '204 No Content'.

In the case of an unsuccessful request, the response will contain a WebSphere MQ Bridge for HTTP error message and the status code of the HTTP response will be one of those documented in the error codes section.

Supported headers

The following headers can be specified in a HTTP GET request:

- correlId
- msgId
- range
- require-headers
- wait

For more information about headers see "HTTP headers" on page 14.

HTTP DELETE

Description

Using the HTTP DELETE operation you can get a message from a WebSphere MQ queue or topic. This message will be removed from the queue or topic (if the message is on a topic is a retained message, it will remain on the topic). A response message will be sent back to the client including information about the message.

You can use HTTP headers in your HTTP DELETE request to:

- you can use the require-headers header to specify what information about the WebSphere MQ message you would like to receive as headers in the response message.
- you can use the correlID header, msgID header, or both to determine which message you get from the queue or topic.
- you can use the wait header to determine how long you will wait for a message to arrive on the queue or topic.
- you can use the x-msg-range header to specify the range of data in the message that should be returned in the response.

If the HTTP DELETE request is successful, the entity of the response message will contain the data of the message retrieved from the WebSphere MQ queue or topic (see “Supported message types” on page 25 for more information) and the HTTP content-length headers will be set to the number of bytes in the entity body. The status code of the HTTP response will be ’200 OK’. If x-msg-range has been specified to be 0, or 0-0, then the status code of the HTTP response will be ’204 No Content’.

In the case of an unsuccessful request, the response will contain a WebSphere MQ Bridge for HTTP error message and the status code of the HTTP response will be one of those documented in the Chapter 8, “HTTP Return codes,” on page 33 section.

Supported headers

The following headers can be specified in a HTTP DELETE request:

- correlId
- msgId
- range
- require-headers
- wait

For more information about headers see section “HTTP headers.”

HTTP headers

The WebSphere MQ Bridge for HTTP supports some custom HTTP headers to use with the WebSphere MQ Bridge for HTTP. HTTP practice is to prefix all custom headers with ’x-’, the WebSphere MQ Bridge for HTTP headers are prefixed with ’x-msg-’. For example, to set the priority header use x-msg-priority.

Limitations exist regarding the use of some HTTP headers, see Chapter 7, “Limitations,” on page 31 for more information.

Note:

- All header value literals are case-sensitive. For example, when using the msgId header, ″NONE″ is recognized as a special case whereas ″none″ would be taken as a normal msgID.
- HTTP practice is to ignore unrecognized headers. Therefore, a misspelled header will not cause an error, the header will simply be ignored.

The custom HTTP headers are grouped into entity headers and request headers.

Some standard HTTP headers are also supported for use with the HTTP Bridge in order to supply more information about the origin of messages.

Entity headers

Entity-header fields define information about the entity-body or, if no body is present, about the resource identified by the request.

Table 2. Headers that are supported in HTTP request messages

HTTP Header	Valid on a POST	Valid on a DELETE/GET
x-msg-class	X	
x-msg-correlId	X	X
x-msg-encoding	X	
x-msg-expiry	X	
x-msg-format	X	
x-msg-msgId		X
x-msg-persistence	X	
x-msg-priority	X	
x-msg-replyTo	X	
x-msg-usr	X	

The following table shows all the headers that can be received as headers in a response message when specified in the require-headers request header.

Table 3. Headers that can be received in response messages

HTTP Header	Valid on a DELETE/GET/POST
x-msg-class	X
x-msg-correlId	X
x-msg-encoding	X
x-msg-expiry	X
x-msg-format	X
x-msg-msgId	X
x-msg-persistence	X
x-msg-priority	X
x-msg-replyTo	X
x-msg-timestamp	X
x-msg-usr	X

class

HTTP header name:	x-msg-class
Description:	• When set on an HTTP POST request, this header specifies the message type of the message that is put to the destination. • When requested in the require-headers request header, this header indicates the message type that was retrieved from the destination.
Allowed values:	**BYTES** **MAP** **STREAM** **TEXT**
Default value:	BYTES

Notes®:	1. Specifying the class header on a GET or DELETE will return a 400 Bad Request with entity body of: MQHTTP40007. 2. If an invalid value is specified for this header a MQHTTP 40005 message will be returned. 3. If the x-msg-class header is not specified and the content-type of the message is application/x-www-form-urlencoded, the data will be assumed to be a map object.

correlId

HTTP header name:	x-msg-correlId
Description:	• When requested in the require-headers request header, this header holds the value of the correlation ID of the message POSTED to or GOT/DELETED from the queue or topic. • When set on an HTTP POST request, this header can be used to specify the correlation ID of the message put to the queue or topic. • When set on an HTTP GET/DELETE request, this header can be used to select the message you wish to receive from the queue or topic. If a message with a matching correlation ID exists on the queue or topic, this message will be retrieved and sent in the HTTP response. If no message exists with the specified correlation ID, an HTTP 504 Gateway Timeout response will be returned. This header can be used in conjunction with msgID to select a message from a queue or topic that matches both selectors.
Allowed values:	**A string value** For example, x-msg-correlId: mycorrelationid Quoted strings are permitted, for example x-msg-correlId: ″my id″ **A hex value prefixed with ″0x:″** For example, x-msg-correlId: 0x:43c1d23a
Default value:	Not applicable
Notes:	Specifying the correlId header without a value on an HTTP GET/DELETE request (e.g. ″x-msg-correlId:″), will return the next message on the queue or topic regardless of its correlation ID.

encoding

HTTP header name:	x-msg-encoding
Description:	• When requested in the require-headers request header, this header holds the value of the encoding of the message POSTED to or GOT/DELETED from the queue or topic. • When set on an HTTP POST request, this header can be used to specify the encoding of the message put to the queue or topic. • When set on an HTTP GET/DELETE request, this header will be ignored.

Allowed values:	A comma separated list of the following values: DECIMAL_NORMAL DECIMAL_REVERSED FLOAT_IEEE_NORMAL FLOAT_IEEE_REVERSED FLOAT_S390 INTEGER_NORMAL INTEGER_REVERSED For example, x-msg-encoding: INTEGER_NORMAL,DECIMAL_NORMAL,FLOAT_IEEE_NORMAL
Default value:	DECIMAL_NORMAL, FLOAT_IEEE_NORMAL, INTEGER_NORMAL
Notes:	1. The value is case-sensitive.

expiry

HTTP header name:	x-msg-expiry
Description:	• When requested in the require-headers request header, this header contains the time before expiry in milliseconds of the message POSTED to, or GOT/DELETED from the queue or topic. • When set on an HTTP POST request, this header specifies a period of time in milliseconds, after which the message becomes eligible to be discarded if it has not been removed from the destination queue or topic. • When set on an HTTP GET/DELETE request, this header will be ignored. This field maps to the Expiry field in the MQMD.
Allowed values:	**UNLIMITED** For example, x-msg-expiry: UNLIMITED **The string representation of a signed integer >0 indicating the period in milliseconds the message is valid for.** For example. x-msg-expiry: 10000
Default value:	UNLIMITED
Notes:	1. ″UNLIMITED″ specifies that the message will never expire. 2. The expiry of a message starts from the time the message arrives on the queue, as a result network latency is ignored. 3. The maximum value is limited by WebSphere MQ to 214748364700 milliseconds. If a value greater than this is specified then the maximum possible expiry time is assumed.

format

HTTP header name:	x-msg-format

Description:	• When requested in the require-headers request header, this header contains the format of the message POSTED to or GOT/DELETED from the queue or topic. • When set on an HTTP POST request, this header can be used to specify the format of the message put to the queue or topic. • When set on an HTTP GET/DELETE request, this header will be ignored.
Allowed values:	**NONE** For example, x-msg-format: NONE *Any user defined value* For example, x-msg-format: myformat
Default:	Not applicable
Notes:	1. ″NONE″ is case-sensitive, and indicates that the message format is blank. 2. The value of this header will be used even if it contradicts the specified media-type of the HTTP request. For more information about media-types see “Supported message types” on page 25.

msgId

HTTP header name:	x-msg-msgId
Description:	• When requested in the require-headers request header, this header holds the message ID of the message POSTED to or GOT/DELETED from the queue or topic. • When set on an HTTP POST request, this header will be ignored. • When set on an HTTP GET/DELETE request, this header can be used to select the message you wish to receive from the queue or topic. If a message with a matching correlation ID exists on the queue or topic, this message will be retrieved and sent in the HTTP response. If no message exists with the specified correlation ID, an HTTP 504 Gateway Timeout response will be returned. This header can be used in conjunction with correlID to select a message from a queue or topic that matches both selectors.
Allowed values:	**a string value** For example, x-msg-msgId: mymessageid Quoted strings are permitted, for example x-msg-correlId:″my id″ **a hex value prefixed with ″0x:″** For example, x-msg-msgId: 0x:43c1d23a
Default:	Not applicable
Notes:	1. Horizontal whitespace is allowed after the ″0x:″ prefix.

persistence

HTTP header name:	x-msg-persistence

Description:	• When requested in the require-headers request header, this header holds the persistence of the message POSTED to or GOT/DELETED from the queue or topic. • When set on an HTTP POST request this header can be used to specify the persistence of the message put to the queue or topic. • When set on an HTTP GET/DELETE request this header will be ignored.
Allowed values:	**NON_PERSISTENT** The message does not usually survive system failures or queue manager restarts. This applies even if an intact copy of the message is found on auxiliary storage when the queue manager restarts. For example, x-msg-persistence: NON_PERSISTENT **PERSISTENT** The message survives system failures and restarts of the queue manager. . For example, x-msg-persistence: PERSISTENT
Default:	NON_PERSISTENT
Notes:	1. Both literals are case-sensitive.

priority

HTTP header name:	x-msg-priority
Description:	• When requested in the require-headers request header, this header holds the priority of the message POSTED to or GOT/DELETED from the queue or topic. • When set on an HTTP POST request, this header can be used to specify the priority of the message put to the queue or topic. • When set on an HTTP GET/DELETE request, this header will be ignored.
Allowed values:	**LOW** For example, x-msg-priority: LOW **MEDIUM** This priority is equal to a WebSphere MQ priority level of 4. For example, x-msg-priority: MEDIUM **HIGH** For example, x-msg-priority: HIGH *A string representation of an integer between O and 9 (inclusive)* For example, x-msg-priority: 3
Default:	MEDIUM
Notes:	1. 0 represents a low priority, 9 represents a high priority.

replyTo

HTTP header name:	x-msg-replyTo

Description:	• When requested in the require-headers request header, this header holds the replyTo destination of the message POSTED to or GOT/DELETED from the queue or topic. • When set on an HTTP GET/DELETE request, this header will be ignored. • When set on a HTTP POST request this header can be used to specify the replyTo destination of the message put to the queue or topic.
Allowed values:	A point-to-point URI as defined in the URI format section of this document. For example, x-msg-replyTo: /msg/queue/myReplyQueue or x-msg-replyTo: /msg/queue/myReplyQueue@myReplyQueueManager
Default:	Not applicable
Notes:	If replyTo is requested on a HTTP POST using the require-headers header, the URI in the HTTP response can include the name of the queue manager to which the WebSphere MQ Bridge for HTTP is connected.

timestamp

HTTP header name:	x-msg-timestamp
Description:	• When requested in the require-headers request header, this header holds the timestamp of the message POSTED to or GOT/DELETED from the queue or topic. • When set on an HTTP POST request, this header will be ignored. • When set on an HTTP GET/DELETE request, this header will be ignored.
Allowed values:	A date in the format; day, date month year time time-zone (for example, Sun, 06 Nov 1994 08:49:37 GMT), as defined by RFC 822, and updated in RFC 1123.
Default:	Not applicable
Notes:	1. The timestamp header cannot be specified in an HTTP request.

usr

HTTP header name:	x-msg-usr
Description:	The usr header can be used to send and receive user properties. • When set on an HTTP POST request, this header sets the user defined message property and the value of that property. • When requested in the require-headers request header, this header retrieves the value of the specified user defined property or properties.

Allowed values:	The Using user defined properties with the WebSphere MQ Bridge for HTTP section below details the values and properties allowed for usr. For example: x-msg-usr: myCustomPropery;5;i1 x-msg-usr: myCustomProperty1;5;i1, myCustomProperty2;″My String″;string
Default:	Not applicable
Notes	Multiple properties can be set on a message, either by specifying multiple comma separated properties in a single usr header, or by using two or more separate instances of the usr header. To request a specific property to be returned in the response to a GET or DELETE request, specify the name of the property in the require-headers header of the request, using the prefix usr-. For example: x-msg-require-headers: usr-myCustomProperty Alternatively, to request that all user properties are returned in a response, use the ALL-USR constant. For example: x-msg-require-headers: ALL-USR

Using user defined properties with the WebSphere MQ Bridge for HTTP

The usr message entity header can be used to send and receive user defined properties.

Use the user header to set a property as follows:

```
usr-property-value = property-name ";" usr-value ";" usr-type
property-name = quoted-string ; as defined in RFC 822
usr-type = "boolean" / "i1" / "i2" / "i4" / "i8" / "r4" / "r8" / "string"
usr-value = boolean / i1 / i2 / i4 / i8 / r4 / r8 / string
boolean = "TRUE" / "FALSE"
i1 = <in the range -128 to 127 inclusive>
i2 = <in the range -32768 to 32767 inclusive>
i4 = <in the range -2147483648 to 2147483647>
i8 = <in the range -9223372036854775808 to 9223372036854775807>
r4 = <in the range 1.4E-45 to 3.4028235E38 inclusive>
r8 = <in the range 4.9E-324 to 1.7976931348623157E308 inclusive>
string =  quoted-string
```

Request headers

The request header fields allow the client to pass additional information about the request to the server. These fields act as request modifiers.

Table 4. Headers that are supported in HTTP request messages

HTTP Header	Valid on a POST	Valid on a DELETE/GET
x-msg-range		X
x-msg-require-headers	X	X
x-msg-wait		X

Note: The request headers cannot be received as headers in an HTTP response message.

range

HTTP header name:	x-msg-range
Description:	When set on an HTTP GET/DELETE request, this header can be used to specify a range of bytes in the message that you want to be returned in the HTTP response message. The range of bytes is returned in the response message in the the response message in the content-range header.
Allowed values:	Inclusive range in the format n ["-" m] where n and m are signed integer values and n <= m. **n** Returns the first n bytes of the message (inclusive). Where n is a signed integer. **n "-" m** Returns a range of bytes from the message content, from n bytes to m bytes inclusive. Where n and m are signed integer values and n is less than m. **ALL** The whole of the message content is returned in the response message. For example, if x-msg-range: 0-60 is used in a request for a message containing 100 bytes, the content-range header will hold the string '0-60/100'
Default:	ALL
Notes:	• If no data is requested and "x-msg-range: 0" or "x-msg-range: 0-0", the response will be returned as an "HTTP 204 – No Content" response. • If an invalid range is specified, for example, if n is greater than m or the syntax is incorrect, then a 400 Bad Request error will be returned with entity body MQHTTP40005. • If this header is specified on anything but a GET or DELETE request on a queue or topic, then a 400 Bad Request will be returned with entity body MQHTTP40007. • If a valid range is specified on a GET or DELETE request, the response will contain an HTTP 1.1 Content-Range header as specified in the HTTP 1.1 specification.

require-headers

HTTP header name:	x-msg-require-headers
Description:	When set on an HTTP POST/GET/DELETE request, this header can be used to specify which of the entity headers should be included in the HTTP response message.

Allowed values:	A comma separated list of the entity header names supported by this SupportPac. **ALL** **ALL-USR** **class** **content-location** **correlId** **encoding** **expiry** **format** **msgId** **persistence** **priority** **replyTo** **server** **timestamp** **usr-***property name* For example, x-msg-require-headers: msgId or x-msg-require-headers: expiry,correlId,timestamp to request a specific property: x-msg-require-headers: usr-myCustomProperty to request all properties: x-msg-require-headers: ALL-USR
Default:	Not applicable
Notes:	1. The value of this header is case-insensitive, except in the cases of the ALL and ALL-USR constants, and the *property-name* specified in a request for a user property.

wait

HTTP header name:	x-msg-wait
Description:	• When set on an HTTP POST request, this header will be ignored. • When set on an HTTP GET/DELETE request, this header can be used to specify the period of time to wait for a message to arrive on the queue or topic before returning an HTTP 504 Gateway Timeout response.
Allowed values:	**NO_WAIT** For example, x-msg-wait: NO_WAIT **The string representation of a signed integer >=0, indicating the period in milliseconds that the WebSphere MQ Bridge for HTTP should wait for an appropriate message to arrive on the queue or topic.** For example, x-msg-wait: 8
Default value:	NO_WAIT

Notes:	1. ″NO_WAIT″ is case-sensitive. 2. The maximum wait time is 35000 unless the Servlet's maximum_wait_time parameter was changed when the Servlet was deployed, see theChapter 3, “Installing,” on page 5 section for more information. 3. When the x-msg-wait header is set on an HTTP GET or HTTP DELETE request, the header value is used unless it exceeds the Servlet's maximum wait time, in which case the Servlet's maximum_wait_time is used.

Standard HTTP headers

Some standard HTTP headers are supported for use with the WebSphere MQ HTTP Bridge. These headers can be returned in response messages to supply information about the message's origins.

To receive one or more of the three standard HTTP headers in a response message, specify one or more of the headers in the request message. The require-headers header can be used to request the content-location and server headers, but content-range will not be received unless x-msg-range is used.

For more information about the standard HTTP headers refer to the HTTP 1.1 Specification.

Content-Location

HTTP header name:	Content-Location
Description:	When requested in the require-headers header, the content-location header supplies information about the resource location for the entity enclosed in the message when that entity is accessible from a location separate from the requested resource's URI.
Returned value:	*absoluteURI “/” relativeURI*

Content-Range

HTTP header name:	Content-range
Description:	When x-msg-range is specified on a GET or DELETE request, the range of bytes specified in the content-range header are in the response. For example, if x-msg-range: 0-60 is used in a request for a message containing 100 bytes, the content-range header will hold the string ‘0-60/100’

Server

HTTP header name:	server
Description:	When requested in the require-headers header, the server header supplies information about the server and the protocol the client is connected to.
Returned value:	WMQ-HTTP/1.1 JEE-Bridge/1.1

Notes	Where an implementation of the 1.1 WebSphere MQ Bridge for HTTP protocol is deployed to an application server, the HTTP 1.1 specification requires that product details be appended to the server response header in order of significance. For example, the WebSphere MQ Bridge for HTTP deployed to JEE implementation deployed to a WebSphere Application Server CE called Apache-Coyote would give the response “Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1”.

Supported message types

The WebSphere MQ Bridge for HTTP supports the following message types:

- Text
- Bytes
- Stream
- Map

HTTP POST

When using the POST command to send a message to a queue or topic, the message type that arrives at the destination depends on the value of the x-msg-class header and the content-type of the HTTP request.

The following table describes the mappings between these values and the message type put to the queue or topic.

Table 5. Mapping content-type and x-msg-class to message types

x-msg-class	Content-type	Message type on queue/topic
BYTES	• application/octet-stream • application/xml	WebSphere MQ message (MQFMT set to MQC.MQFMT_NONE)
TEXT	• text/*	WebSphere MQ message (MQFMT set to MQC.MQFMT_STRING)
MAP	• application/x-www-form-urlencoded • application/xml (optional)	JMSMap
STREAM	• application/xml (optional)	JMSStream

Note: If the x-msg-class header is set to BYTES or TEXT, and one or more user properties have been set in the POST request using the x-msg-usr header, the message type on the queue or topic will be JMSBytes or JMSText respectively, since JMS RFH2 headers are used to store user properties.

All other content-types will be treated as binary messages, and the messages ill appear on the queue with MQFMT set to MQC.MQFMT_NONE unless the x-msg-format fils has been set in the request.

HTTP GET or DELETE

When using the GET or DELETE command to retrieve a message from a queue or topic, the message type retrieved determines the value of the x-msg-class header and the content-type of the HTTP response.

The following table describes the mappings between these values and the message type retrieved from the queue or topic.

Table 6. Mapping message types to x-msg-class and content-type

Message type on queue/topic	**x-msg-class (if requested)**	**Content-type (always responded)**
WebSphere MQ message (MQFMT set to anything other than MQC.MQFMT_STRING)	BYTES	• application/octet-stream
WebSphere MQ message (MQFMT set to MQC.MQFMT_STRING)	TEXT	• text/plain
JMSBytes	BYTES	• application/octet-stream
JMSText	TEXT	• text/plain
JMSMap	MAP	• application/xml
JMSStream	STREAM	• application/xml

The message body of map and stream messages will be serialized to the client in the HTTP entity as JMS serializes the message on a queue when mapping MQ message to JMS formats. The formats are discussed in *WebSphere MQ Using Java, SC34-6478*.

Map and Stream message formats

Map and Stream message types will be serialized back to the client in the HTTP entity as JMS serializes the message on a queue when mapping MQ messages to JMS formats.

In the Map message format, the XML name/type/value triplets are encoded as:

```
<map>
 <elt name="elementname1" dt="datatype1">value1</elt>
 <elt name="elementname2" dt="datatype2">value2</elt>
 ...
</map>
```

The stream is similar to a map message, but does not have element names:

```
<stream>
 <elt dt="datatype1">value1</elt>
 <elt dt="datatype2">value2</elt>
 ...
</stream>
```

Note: `datatype` is one of the data types listed the Using user defined properties chapter. The default data type is `string`, and so the attribute `dt="string"` is omitted for string elements.

Chapter 6. WebSphere MQ Bridge for HTTP Samples

This chapter documents the WebSphere MQ Bridge for HTTP samples. These samples are available for use on the Windows® operating system only. There are two different types of sample, a web-based sample and Java console samples. These samples demonstrate potential uses of the WebSphere MQ Bridge for HTTP.

The console samples simulate the WebSphere MQ AMQSPUT and AMQSGET sample applications, and illustrate the following functions in a point-to-point messaging environment:

- HTTPPOST - Generates HTTP POST requests in a Java application to put messages to a WebSphere MQ queue, via the WebSphere MQ Bridge for HTTP and handles HTTP responses.
- HTTPDELETE - Generates HTTP DELETE requests in a Java application to get messages from a WebSphere MQ queue, via the WebSphere MQ Bridge for HTTP and handles the HTTP responses containing the WebSphere MQ message.

The web-based sample illustrates the use of the WebSphere MQ Bridge for HTTP to implement a dynamically updating AJAX based web application in a publish/subscribe messaging environment. The web application receives messages asynchronously from WebSphere MQ via the WebSphere MQ Bridge for HTTP. Two web pages are used in this sample:

- The administrator web page is used to send flight status messages to WebSphere MQ.
- The status web page receives flight status messages from the administrator and updates the flight status board accordingly.

Installing the WebSphere MQ Bridge for HTTP samples

To use the console samples, download the ma0y.samples.zip file from the SupportPac web page. Included in the zip file are:

- this document, ma0y.pdf.
- an instance of WebSphere Application Server Community Edition (WAS CE) with a predeployed:
  - WebSphere MQ Bridge for HTTP.
  - WebSphere MQ resource adapter.
  - web-based publish/subscribe sample application.
- the WebSphere MQ Bridge for HTTP application.
- the WebSphere MQ Bridge for HTTP Java console point-to-point samples.
- the WebSphere MQ Bridge for HTTP web-based sample application.
- a licenses folder containing product licenses and notices.txt.

Note: Also included in the ma0y.samples.zip are sample deployment plans for WAS CE. The plans included are for the Servlet (geronimo-web.xml), the resource adapter (geronimo-ra.xml), and the samples (samplesGeronimo-web.xml). All of these objects are already deployed to the instance of WAS CE included in ma0y.samples.zip, it is not necessary to do anything with these deployment plans before running the samples. These deployment samples have been included simply as examples for you to refer to.

To install the WebSphere MQ Bridge for HTTP samples complete the following steps.

1. Download the SupportPac ma0y.samples.zip file from the WebSphere MQ SupportPac website.
2. Before extracting the files from ma0y.samples.zip, read the license file in the licenses subdirectory. The license file is supplied in several languages. Ensure that you are happy with the terms of the licenses before you continue to install.
3. Extract the files from the zip file into a clean directory of your choice. For example c:\Program Files\IBM\WebSphere MQ HTTP.

Running the WebSphere MQ Bridge for HTTP console samples

The pre-deployed WebSphere MQ resource adapter is configured to connect to WebSphere MQ running on the same machine as WebSphere Application Server Community Edition (WAS CE).

To run the WebSphere MQ Bridge for HTTP console samples, complete the following steps:

1. Create and start a WebSphere MQ queue manager called HTTP.QM.
2. Start WAS CE by navigating to *install_dir*/WASCE/bin and run startup.bat. A second command prompt will open. The last line in the information it contains will state 'Server started'.
3. In the original command prompt, navigate to *install_dir*/P2PConsole. Where *install_dir* is the directory that you extracted the contents of ma0y.samples.zip into.
4. Run the HTTP POST sample by typing following in the command prompt: `java -classpath . HTTPPOST [queue-name] [host:port] [context-root]` When the HTTPPOST sample starts, the following will be displayed:

   ```
   HTTP POST Sample start
   Target server is 'host:port'
   Target queue is 'your queue name'
   Target context-root is 'your context-root'
   ```

5. In the command prompt type the text that you want to form your message body.
6. Press enter to post the message to the WebSphere MQ queue.
7. If you want to send another message, input some more text, this will form the message body of a second WebSphere MQ message and press enter.
8. To end the HTTP POST sample, press enter two times. When the HTTPPOST sample ends, the following will be displayed:

   ```
   HTTP POST Sample end
   ```

9. Run the HTTP DELETE sample by typing following in a command prompt.`java -classpath . HTTPDELETE [queue-name] [host:port] [context-root]` . When the HTTPDELETE sample starts, the following will be displayed:

   ```
   HTTP DELETE Sample start
   Target server is 'host:port'
   Target queue is 'your queue name'
   Target context-root is 'your context-root'
   ```

   The sample performs a destructive get of all of the message on the WebSphere MQ queue. Following the text above will be a list of the messages on your queue.

Running the WebSphere MQ Bridge for HTTP web-based sample

The pre-deployed WebSphere MQ resource adapter is configured to connect to WebSphere MQ running on the same machine as WebSphere Application Server Community Edition (WAS CE).

To run the WebSphere MQ Bridge for HTTP web-based sample, complete the following steps:

1. Create and start a WebSphere MQ queue manager called HTTP.QM.
2. Start WAS CE by navigating to *install_dir*/WASCE/bin and run startup.bat.
3. In a web browser navigate to http://*hostname*:*port*/wmqhttp/samples. If your web browser is running on the same machine as WAS CE the URI is http://localhost:8080/wmqhttp/samples
4. Configure a broker on your WebSphere MQ queue manager, HTTP.QM. Instructions regarding how to do this are included in the samples web page.
5. Follow the instructions on the samples web page to use the samples.

Chapter 7. Limitations

The following limitations apply to the WebSphere MQ Bridge for HTTP.

HTTP GET and publish/subscribe

HTTP GET cannot be used with publish/subscribe messaging and will return an error code.

Using correlation ID with WebSphere MQ Publish/Subscribe messaging

The correlation ID in the MQMD is reserved for use by the WebSphere MQ publish/subscribe engine to identify subscribers. When putting a message to a topic using an HTTP POST request, if the message type is a non-JMS message (see message types section), the correlId header value will be ignored by WebSphere MQ as the publish/subscribe engine will specify its own MQMD correlation ID.

Using persistence with WebSphere MQ publish/subscribe messaging

The persistence of a message without RFH2 properties that is put to a topic in WebSphere MQ is not maintained between the point of publishing the message and the point of subscribing. When putting a message to a topic using an HTTP POST request, if the message type is a non-JMS message (see message types section), the persistence header value will be ignored by WebSphere MQ.

Using format with WebSphere MQ publish/subscribe messaging

The format of a message without RFH2 properties that is put to a topic in WebSphere MQ is not maintained between the point of publishing the message and the point of subscribing. When putting a message to a topic using an HTTP POST request, if the message type is a non-JMS message (see message types section), the format header value will be ignored by WebSphere MQ.

Requesting the encoding value of a message

When requesting the encoding value of a message by specifying the encoding header in the require-headers header of an HTTP request, a value will only be returned in the response if an encoding value was explicitly set on the WebSphere MQ message in the HTTP request. If no encoding value was set on the WebSphere MQ message, the encoding cannot be derived and an encoding header value will not be returned.

Requesting the correlation ID of a message

When requesting the correlation ID of a message by specifying the correlId header in the require-headers request header of an HTTP request, the value returned will be in hexadecimal form ("0x:" prefix followed by a hexadecimal value) if the message retrieved was an WebSphere MQ (that is, a non JMS) message.

Content-Location

When requesting the content location of a message by specifying content-location in the require-headers request header of a HTTP request, the value will only be returned if the message is a JMS message, since the original location of a WebSphere MQ message (a non-JMS message) cannot be derived from the message itself.

Chapter 8. HTTP Return codes

Servlet errors - Logged but not returned to the client

Where these errors are logged is specific to your application server, refer to your application server's documentation for information.

MQHTTP0001
No Connection Factory specified in the Servlet context.

MQHTTP0002
Could not get Connection Manager for {queueOrTopic} using the jndi Name of {jndiNameTried}

Successful operations

If operations are successful a return code of 200xx will be returned.

Table 7.

Header	Header
200 OK	**MQHTTP200** This class of status code indicated that the client's request was successfully received, understood and accepted.
204 No Content	**MQHTTP204** Sent when a successful GET/DELETE happens and x-msg-range: 0 was sent in the request.

Client errors

HTTP status code	MQHTTP errors
400 Bad Request	**MQHTTP40001** Text: Invalid message header specified: '{header_name}'='{header_value}'. Cause: A header has been sent across in the request that is not valid for this action. Action: The client should remove the header in question. **MQHTTP40002** URI '{uri}' is not valid for MQ HTTP. **MQHTTP40003** URI '{uri}' is not valid. {@qmgr} is only valid on a POST. **MQHTTP40004** Invalid content-type specified in the request: '{content_type}' **MQHTTP40005** Bad message header value: '{header_name}'='{header_value}'. **MQHTTP40006** '{header_name}' is not a valid request header. **MQHTTP40007** '{header_name}' is only valid on '{HTTP_operation_types}'. **MQHTTP40008** '{header_name}' maximum length is '{maximum length is '{max_length in decimal}'. **MQHTTP40009** Header field '{header_field}' is not valid for this messaging provider. Explanation: The implementation of the WebSphere MQ protocol does not support the header sent in the request. The request will not be carried out. Action: Remove the unsupported header. **MQHTTP40010** Message with content-type '{content_type}' could not be parsed. Explanation: This is thrown when a message POSTed with a request content-type failed to parse correctly. This could be caused by a badly formed form-urlencoded message (MAP). Action: The client should check the format of the entity body.
401 Unauthorized	None

HTTP status code	MQHTTP errors
403 Forbidden	**MQHTTP40301** You are forbidden from accessing the destination '{destination_name}'. **MQHTTP40302** You are forbidden from accessing the queue manager.
404 Not found	**MQHTTP40401** The destination '{destination_name}' could not be found.
405 Method not allowed	**MQHTTP40501** Method '{method_name}' not allowed.
413 Request entity too large	**MQHTTP41301** The message being posted was too large for the destination: '{destination}'.
415 Unsupported media type	**MQHTTP41501** The media type character set '{charset}' us unsupported. **MQHTTP41502** Media-type '{media-type}' is not supported for the operation {operation VERB e.g. POST} on resource type {resource_type e.g. queue message, topic message}. **MQHTTP41503** Media-type '{media-type}' is not valid with class '{x-msg-class}.
417 Expectation failed	**MQHTTP41701** The HTTP header 'expect' is not supported.

Server errors

HTTP status code	MQHTTP errors
500 Internal server error	**MQHTTP50001** There has been an unexpected problem in the MQ HTTP server. Please contact your system administrator.
502 Bad Gateway	**MQHTTP50201** There has been an unexpected problem in the connection to MQ. Please contact your system administrator.
504 Gateway timeout	**MQHTTP50401** Message retrieval timed out.
505 HTTP version not supported	**MQHTTP50501** HTTP 1.1 and upwards is supported.

When a server error occurs, a complete stack trace will be outputted to application server error logs and will also be returned to the HTTP client in the HTTP response. This trace can then be recognized and handled by the client application or used by the application server administrator to resolve the cause of the issue.

If the stack trace contains resource adapter errors, refer to you resource adapter's documentation for more information.

Notices

This information was developed for products and services offered in the United States. IBM may not offer the products, services, or features discussed in this information in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this information. The furnishing of this information does not give you any license to these patents. You can send license inquiries, in writing, to:

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

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation,
Licensing,
2-31 Roppongi 3-chome, Minato-k,u
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the information. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this information at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM United Kingdom Laboratories,
Mail Point 151,
Hursley Park,
Winchester,
Hampshire,
England
SO21 2JN.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

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 and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

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

IBM SupportPac WebSphere

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

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

Printed in USA